Rev. 4.2.0.0115-e02 (2026/6/18)

Machine Learning Quality Management Guideline

4th Edition

(Revision 4.2.0)

June 18, 2026
(Japanese: December 12, 2023)

Technical Report IPRI-TR-2026-03
Intelligent Platform Research Division

Technical Report CPSEC-TR-2026003
Cyber Physical Security Research Division

Technical Report
Artificial Intelligence Research Center

National Institute of Advanced Industrial Science and Technology (AIST)

© 2026 National Institute of Advanced Industrial Science and Technology

Foreword and disclaimer

This English version of the Machine Learning Quality Management Guideline is a translation of 「機械学習品質マネジメントガイドライン 第 4 版」(Machine Learning Quality Management Guideline 4th edition), published by AIST on December 12, 2023 in Japanese.

The Guideline was prepared by the Committee for Machine Learning Quality Management in the National Institute of Advanced Industrial Science and Technology (AIST).

The Guideline is non-binding in relation to laws and regulations/official guidelines. Provisions described as normative in the Guideline have normative meaning only if the Guideline is adopted voluntarily.

This document is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

This document is developed with support from the New Energy and Industrial Technology Development Organization (NEDO).

Copyright and licensing



Copyright © 2026 by the National Institute of Advanced Industrial Science and Technology. This document is licensed under the Creative Commons CC BY 4.0 License. To view a copy of the license, visit https://creativecommons.org/licenses/by/4.0/.

k.konishi@aist.go.jp
デジタル署名者 : k.konishi@aist.go.jp
日付 : 2026.06.23 16:25:59 +09'00'

Table of Contents

1. Aims, goals, and structure of the Guideline 1
1.1 Purpose and background 1
1.2 Intended targets of the Guideline 1
1.3 Challenges in machine learning quality management 3
1.3.1 Importance of environmental analysis 3
1.3.2 Need for risk assessment throughout the lifecycle 4
1.3.3 Quality assurance based on data 4
1.4 Basic concept of quality management 5
1.5 Structure of the Guideline 9
2. Overview 11
2.1 Scope of the Guideline 11
2.1.1 Products and systems subject to the Guideline 11
2.1.2 Target of quality management 11
2.1.3 Scope of quality management 12
2.2 Relationship between the Guideline and other standards for system quality 13
2.2.1 ISO/IEC 15408 security standard 13
2.2.2 ISO/IEC 25000 series software quality model 13
2.3 Definitions of terms 14
2.3.1 Terms related to machine-learning-based system structure 14
2.3.2 Terms related to development stakeholders and their roles 17
2.3.3 Terms related to quality 17
2.3.4 Terms related to the development process 18
2.3.5 Terms related to the use environment 19
2.3.6 Terms related to data used for machine learning 20
2.3.7 Other terms 21
3. How to apply the Guideline 23
3.1 Business risk analysis 23
3.1.1 Preliminary assessment of the need for safety functions and identification of applicable standards 23
3.1.2 Identifying system purpose, goals, and functional requirements 23
3.1.3 Examining risk scenarios related to system use 23
3.2 Identifying quality-in-use characteristics 24
3.3 High-level system design 24
3.3.1 Identifying functions to be provided by machine learning components 24
3.4 Determining external quality levels 25
3.5 Selection of verification methods for each internal quality 25
3.6 Quality control activities 25
3.7 Quality control during operation 26

3.8 Overall lifecycle process 26
3.8.1 Relationship between development through iterative training and the quality management lifecycle 26
3.8.2 Notes on differential development (informative) 29
3.8.3 Relationship with development processes involving multiple stakeholders 31
3.9 Work sharing and quality control in AI development *not built from scratch* 32
3.9.1 Outsourcing contracts 32
3.9.2 Use of foundation models 36
3.9.3 Use of AutoML services 38
4. External qualities addressed by this Guideline 39
4.1 External quality characteristics to be achieved 39
4.2 Safety and risk avoidance 39
4.3 AI performance 40
4.4 Fairness 41
4.4.1 Definitions of ethics and fairness in the Guideline 41
4.4.2 Considerations regarding sensitive attributes 42
4.4.3 Basic concept of fairness assurance 42
4.5 Privacy 43
4.5.1 Privacy as an external quality 43
4.5.2 Privacy quality management 44
4.6 AI security 45
4.6.1 AI security as an external quality 45
4.6.2 Quality management for AI security 46
4.7 Relationships among external qualities 48
4.8 Relationships with other social and technical quality requirements 48
4.8.1 Explainability of AI 49
4.8.2 Social aspects such as ethical considerations 50
4.8.3 Limits of handling complex external environments 51
5. Levels of external quality characteristics 52
5.1 Safety or risk avoidance 52
5.2 AI performance 54
5.3 Fairness 54
5.4 Privacy 55
5.5 AI security 55
6. Internal quality characteristics subject to quality management 56
6.1 A-0: Sufficiency of preliminary problem analysis 58
6.2 A-1: Sufficiency of problem domain analysis 59
6.3 A-2: Sufficiency of data design 61
6.4 B-1: Coverage of datasets 62
6.5 B-2: Uniformity of datasets 63
6.6 B-3: Adequacy of data 65

6.7 B-4: External-quality-dependent adequacy of datasets 66
6.8 C-1: Correctness of trained models 66
6.9 C-2: Stability of trained models 66
6.10 C-3: External-quality-dependent adequacy of trained models 66
6.11 D-1: Reliability of underlying software systems 67
6.12 D-2: Additional aspects of program adequacy 67
6.13 E-0: Continuous monitoring and recording during operation 67
6.14 E-1: Maintainability of quality characteristics during operation 67
7. Requirements for quality assurance 68
7.1 A-0: Sufficiency of preliminary problem analysis 68
7.1.1 A-0s: Safety or risk avoidance 68
7.1.2 A-0p: AI performance 68
7.1.3 A-0f: Preliminary analysis of fairness requirements 68
7.1.4 A-0pr: Preliminary analysis of privacy requirements 71
7.1.5 A-0se: Preliminary analysis of security requirements 73
7.2 A-1: Sufficiency of problem domain analysis 77
7.2.1 General 77
7.2.2 Approaches 78
7.2.3 Requirements for quality levels 80
7.2.4 Sufficiency of fairness requirements analysis 81
7.3 A-2: Sufficiency of data design 82
7.3.1 General 82
7.3.2 Approaches 83
7.3.3 Requirements for quality levels 83
7.4 B-1: Coverage of datasets 84
7.4.1 General 84
7.4.2 Approaches 85
7.4.3 Requirements for quality levels 85
7.5 B-2: Uniformity of datasets 86
7.5.1 General 86
7.5.2 Approaches 87
7.5.3 Requirements for quality levels 87
7.6 B-3: Adequacy of data 88
7.6.1 General 88
7.6.2 Approaches 89
7.6.3 Requirements for quality levels 92
7.7 B-4: External-quality-dependent adequacy of datasets 94
7.7.1 B-4f: Adequacy of datasets in terms of fairness 94
7.7.2 B-4pr: Adequacy of datasets in terms of privacy 96
7.8 C-1: Correctness of trained models 97
7.8.1 General 97

7.8.2 Approaches 97
7.8.3 Requirements for quality levels 98
7.9 C-2: Stability of trained models 99
7.9.1 General 99
7.9.2 Approaches 99
7.9.3 Requirements for quality levels 100
7.10 C-3: External-quality-dependent adequacy of trained models 101
7.10.1 C-3f: Adequacy of trained models in terms of fairness 101
7.10.2 C-3pr: Adequacy of trained models in terms of privacy 103
7.10.3 C-3se: Adequacy of trained models in terms of security 104
7.11 D-1: Reliability of underlying software systems 108
7.11.1 General 108
7.11.2 Approaches 109
7.11.3 Requirements for quality levels 110
7.12 D-2: Additional aspects of program adequacy 111
7.12.1 D-2se: Adequacy of programs in terms of security 111
8. Preparation and quality management during system operation 113
8.1 E-0: Continuous monitoring and recording during operation 113
8.1.1 General 113
8.1.2 Approaches 113
8.2 E-1: Maintainability of quality characteristics during operation 115
8.2.1 General 115
8.2.2 Approaches 117
8.2.3 Requirements for quality levels 119
9. Technologies for quality management (reference information) 121
9.1 A-0: Sufficiency of preliminary problem analysis 121
9.2 A-1: Sufficiency of problem domain analysis 121
9.2.1 Initial hints 121
9.2.2 Modeling of risk factors in input space 122
9.2.3 Designing problem structures as data characteristics 123
9.3 A-2: Sufficiency of data design 124
9.3.1 General 124
9.4 B-1: Coverage of datasets 124
9.4.1 Data acquisition plan 125
9.4.2 Pre-flight tests in the data review stage 125
9.4.3 Additional tests in the testing stage 125
9.5 B-2: Uniformity of datasets 126
9.6 B-3: Adequacy of data 126
9.6.1 Quality control cycle from the perspective of data 126
9.6.2 Technical support for organizing outliers and corner cases 127
9.7 B-4: External-quality-dependent adequacy of datasets 127

9.7.1 B-4pr: Adequacy of datasets in terms of privacy ... 127
9.8 C-1/C-2: Correctness and stability of trained models ... 130
9.8.1 Testing machine learning components ... 130
9.8.2 Technologies for addressing stability issues ... 134
9.9 C-3: External-quality-dependent adequacy of trained models ... 138
9.9.1 C-3pr: Adequacy of trained models in terms of privacy ... 138
9.9.2 C-3se: Adequacy of trained models in terms of security ... 141
9.10 D-1: Reliability of underlying software systems ... 141
9.10.1 General ... 141
9.10.2 Quality management of open-source software ... 141
9.10.3 Configuration management and tracking of bug information ... 142
9.10.4 Feasibility of specific checks through testing ... 142
9.10.5 Software update and possible adverse effects on performance and operation. 142
9.11 D-2: Additional aspects of program adequacy ... 143
9.11.1 D-2se: Adequacy of programs in terms of security ... 143
9.12 E-0: Continuous monitoring and recording during operation ... 143
9.13 E-1: Maintainability of quality characteristics during operation ... 143
9.13.1 Monitoring ... 143
9.13.2 Concept drift detection methods ... 145
9.13.3 Retraining ... 145
9.13.4 Creation of additional training data ... 146
10. Explanations of specific issues for each external quality ... 147
10.1 Quality management for fairness ... 147
10.1.1 Background ... 147
10.1.2 Challenges in fairness ... 149
10.1.3 Framework for process management to ensure fairness ... 155
10.1.4 Refining fairness requirements ... 159
10.1.5 Extended topics in fairness implementation ... 162
10.1.6 Development infrastructure and tools supporting fairness assurance ... 163
10.2 Privacy ... 165
10.2.1 Privacy protection ... 165
10.2.2 Machine learning and personal data protection ... 172
10.2.3 Privacy quality levels defined for various artifacts ... 178
10.3 AI security ... 180
10.3.1 Overview ... 181
10.3.2 Damage to machine-learning-based systems ... 183
10.3.3 Attacks on and damage to machine-learning-based systems ... 187
10.3.4 Threats, vulnerabilities, and control measures specific to machine learning ... 194
10.3.5 Summary and supplementary notes on quality management for AI security ... 204
10.3.6 Related documents ... 209
10.3.7 Security considerations (informative) ... 211

11. Information on related documents (informative) 244
11.1 Relationships with other documents and rules 244
11.1.1 Social Principles on Human-centric AI 244
11.1.2 AI-related rules and guidelines of foreign governments and international organizations concerning AI technology 244
11.1.3 Contract guidelines for AI of the Ministry of Economy, Trade and Industry 245
11.1.4 Relationships with guidelines for quality assurance of machine-learning-based artificial intelligence (QA4AI) 245
11.2 Relationships with international initiatives for AI quality 248
11.2.1 Quality and safety 248
11.2.2 Transparency 250
11.2.3 Fairness (bias) 250
11.2.4 Other perspectives on machine learning quality management 251
12. Reference models for development processes (informative) 252
12.1 PoC phase 252
12.1.1 Handling the PoC phase, including trial operation 252
12.2 Main development phase 253
12.2.1 Machine learning model building phase 254
12.2.2 System building and integration test phase 259
12.3 Quality monitoring and operation phase 260
13. Bibliography 261
13.1 International standards 261
13.2 Documents and regulations issued by countries and international organizations 262
13.3 Official standards and forum standards 264
13.4 Research articles 266
13.5 Miscellaneous 280
14. Revision history 282
14.1 English version of the fourth edition (June 2026) 282
14.2 Japanese version of the fourth edition (December 2023) 282
14.3 English version of the third edition (January 2023) 282
14.4 Japanese version of the third edition (August 2022) 282
14.5 English version of the second edition (February 2022) 283
14.6 Japanese version of the second edition (July 2021) 283
Editors and authors 284
The Committee for Machine Learning Quality Management (FY2023) 284
List of authors of the Machine Learning Quality Management Guideline 285
Members of the Guideline Taskforce (FY2023) 286

1. Aims, goals, and structure of the Guideline

This chapter is informative. Any normative content included in this chapter is restated in later chapters.

1.1 Purpose and background

Artificial intelligence (AI), especially machine learning technology, has demonstrated its effectiveness across a wide range of application fields, including manufacturing, automated driving, robotics, healthcare, finance, and retail, and is increasingly being deployed in real-world settings. At the same time, when incidents occur, it remains difficult to identify their causes or to demonstrate that the benefits of AI-based products justify the investment. This is because reliable methods for measuring and demonstrating the quality of AI-based products and services have not yet been well established. These difficulties have delayed the development of institutional frameworks for the wider acceptance of AI in society and have also hindered the expansion of AI development businesses.

The Guideline establishes a basis for setting quality goals for machine-learning-based products and services and provides procedural guidance for realizing those goals through development-process management and system evaluation. It aims to help providers evaluate and improve the quality of their systems, thereby reducing accidents and other losses caused by AI malfunctions in real-world use. The Guideline also enables stakeholders to communicate product quality using common norms, for both commercial purposes, such as pricing AI-based products, and social purposes, such as demonstrating accountability to society.

1.2 Intended targets of the Guideline

The primary expected users of the Guideline are providers of products and services developed using machine learning (hereinafter referred to as *service providers*[1]) and system developers that actually implement products and services as software. For each product, the service provider and the system developer may be either a single entity that develops products or services and provides them to end users (referred to as *self-development entity*) or two separate entities depending on the sharing of responsibilities based on contract, sub-contracting, or quasi-entrustment. Moreover, a service developer may sell implementation of machine learning components as a separate product. We primarily expect the Guideline to be used as a reference

[1] In cases where software developers implement systems in advance and sell these as packaged or customized products, such developers are treated as service providers in the Guideline.

for these entities to share clear goals on required quality in accordance with situations in which products or services are used and to realize said quality throughout the system development process.

Furthermore, as a secondary usage, we expect the quality levels set by the Guideline can be referred to by end users for judging whether a particular system is safe to use. Also, the Guideline is expected to serve as a technical starting point for specifications and evaluation or certification of the quality.

The Guideline provides a general description that can be applied to a broad range of use cases of machine learning technology. For each application area, one can pick the required parts of the Guideline to make a specialized guideline. In addition, we expect each developer to create and use their specific version of development guidelines, based on the document. The research project currently plans to publish some examples of such document as reference [251].

The following list shows some possible use cases of the Guideline as examples.

- When establishing a cooperative relationship for AI-based system development by contract, etc.,
  - An entity that requests the development of a machine-learning-based system or a machine learning component which constitutes that system (referred to as *development entruster*[2]) concretizes the Guideline in line with an application and designates it for an entity to which development is entrusted (referred to as *development entrustee*) as specifications that serve as contract requirements.
  - An entity that is to be a development entrustee refers to the Guideline as a standard for process management to guarantee the quality for a development entruster and uses them as a ground or reference to calculate man-hours and the price for order.
- Design and development
  - An entity that designs and develops a machine-learning-based system or a machine learning component (development entrustee or self-development entity) uses the Guideline as a standard for process design, system design and quality management.
- Use in society
  - A self-development entity or development entruster bases its self-compatibility declaration on the Guideline with respect to quality requirements as social norms when a system is provided to the society or used for its own business.
  - The Guideline serves as a standard for social consensus on the quality of machine-learning-based systems in the future.
  - The Guideline serves as a standard for the design and operation of a third-party

[2] The "Guideline for Contract on Use of AI and Data" compiled by the Ministry of Economy, Trade and Industry of Japan defines *development entruster* in the Guideline as "user", and *development entrustee* as "vendor" or "system integrator", as it is written mainly for business-to-business development agreements. The Guideline, however, uses the term user only for end users of products and services, who are finally affected by the quality of the products.

assessment system on the quality of machine-learning-based system.

The Guideline is currently intended to apply to machine-learning-based systems implemented with supervised learning. Although the basic concept can apply to other implementation methods such as unsupervised learning, semi-supervised learning and reinforcement learning, a specific way of treating them will be added in future revisions.

1.3 Challenges in machine learning quality management

In simple terms, the implementation of artificial intelligence by machine learning is a type of software. However, a conventional concept of software quality management is not technically feasible to improve and maintain the quality of machine-learning-based systems. This section sorts out challenges related to *differences between AIs and conventional software* from several viewpoints.

1.3.1 Importance of environmental analysis

Machine-learning-based products and services are often used under environmental conditions whose complexity is difficult for humans to understand. Compared to regular programs for which humans have to analyze and identify the complexity of all environmental conditions and implement every program code as rules for judgment, there are high expectations on machine learning technology capable of automatically establishing judgment rules based on data without having to identify detailed environmental conditions through analytical works by humans as an efficient means for implementing systems that operate under highly-complex environmental conditions.

On the other hand, from the viewpoint of ideal quality management, it is desirable to analyze environmental conditions as precisely as possible and understand risks at the early stage of system design especially from the viewpoint of quality management of systems that require compatibility with rare environmental conditions. If machine learning technology is used for these systems, analyses on environmental conditions that used to be carried out when implementing program codes in the past would be omitted. This is the reason why environment analysis at the early stage is important.

Therefore, it is important to consider how deeply the status of use and environmental conditions are analyzed to balance between two characteristics (efficiency of implementation and environmental compatibility) at an early stage of system design, including differences from the development of conventional software.

1.3.2 Need for risk assessment throughout the lifecycle

Generally speaking, a system that operates in the real world and constitutes so-called cyber-physical systems (hereinafter referred to as *CPS*) has risks originating from external environmental changes in addition to risks that exist in regular software systems. Regular software implementation can logically analyze unfamiliar situations, simulate them, and take countermeasures to some extent. In contrast, a machine-learning-based system established based on real data, while on the one hand is expected to show some generalization capability, on the other hand cannot deal with significant changes in data trends originating from changes in our surroundings.

Moreover, a machine-learning-based system may be designed to attain practical quality only after both the initial training phase based on past data and additional learning using real data at the operation stage.

From this viewpoint, it is often necessary to introduce a continuous process lifecycle that integrates development and operation so that it can respond to situational changes after the system's initial operation. From the early stages of the development planning, we need to plan a lifecycle that includes operation-time updates and embed the necessary in-operations steps into the operational plan and contracts between stakeholders.

However, ensuring a certain level of quality at the beginning of operation is often necessary, even if quality management will be carried out continuously. Particularly for systems with safety risks, it is essential to ensure a certain level of quality in the early stages of their operation. From this viewpoint, the Guideline focuses on quality testing conducted in stages from the implementation to the beginning of operation. Moreover, the effects of risks and the required quality level could be changed by varying the form of system operation at the early and full-blown stages, for example, different levels of possibility of risk avoidance due to the presence or lack of monitoring or intervention of the system by humans. In this case, it is required to synchronize the time when the form of operation is changed and the time when quality management goals are changed.

Furthermore, if the system implementation is updated during operation, there is a risk that sometimes the quality deteriorates, called regression in software engineering, even if that update aims to improve the quality. Therefore, we need a certain form of quality monitoring and countermeasures during operation, but they vary depending on the forms of development and operation. Section 12.3 sorts out some possibilities for such operation models.

1.3.3 Quality assurance based on data

A request for the *quality of data* used for building machine learning is often mentioned in data-

oriented development. The Guideline assumes that the quality of data itself is not the ultimate goal of quality management but rather the cause of quality deterioration or means for ensuring quality. Of course, it is almost indispensable to ensure a certain level of data quality to ensure machine learning quality through the lifecycle process management such as the Guideline. In addition, when a machine-learning-based system is developed by sharing works, learning data may be sold, bought or shared for development. In such a case, there is room for discussion by treating *data quality* as an independent character. Moreover, it has been pointed out at recent academic meetings that there are security risks, such as poisoning of learning results due to the intentional injection of improper data. Simple numerical evaluations cannot detect the effects of such improper data. Such detection requires checks on data sources from broader perspectives.

The Guideline is supportive of ensuring the quality utilizing numerical evaluations to the extent possible, but occasionally, we need to secure the quality through qualitative data quality management as well.

1.4 Basic concept of quality management

Note: The Guideline defines *quality in use* as the quality to be provided by the whole system to the end users. Meanwhile, *external quality* and *internal quality* are defined as characteristics to be examined at the level of components that are combined into a single system[3].

The *external quality* of a component is its objectively observable characteristics that the component should have in order to be a part of the system. It includes, for example, security, reliability, and consistency. This external quality can be either qualitative or quantitative and cannot always be expressed by a single measurable indicator.

On the other hand, the *internal quality* of a component refers to its inherent characteristics that are concretely measured during its implementation or evaluated throughout its development phases, including design.

Based on those concepts, the Guideline presents a hierarchical quality model in which the external quality of each component is realized through its improved internal qualities and is necessary to achieve the internal quality of the element one layer outside (Figure 1). The *external quality* of the whole system is realized and provided by the provider of the products/services to achieve the *quality in use* observed by the end users.

[3] The term *internal quality* roughly coincides with the "internal measure of the quality" in ISO/IEC 9126 and its successors ISO 25000 series. The concept behind the *external quality* is not directly associated with "external measure of the quality", because the Guideline assumes that external quality is not generally satisfactorily measurable as a numeric quality indicator. Instead, the Guideline ensures satisfaction of an external quality level through assessments of internal qualities.

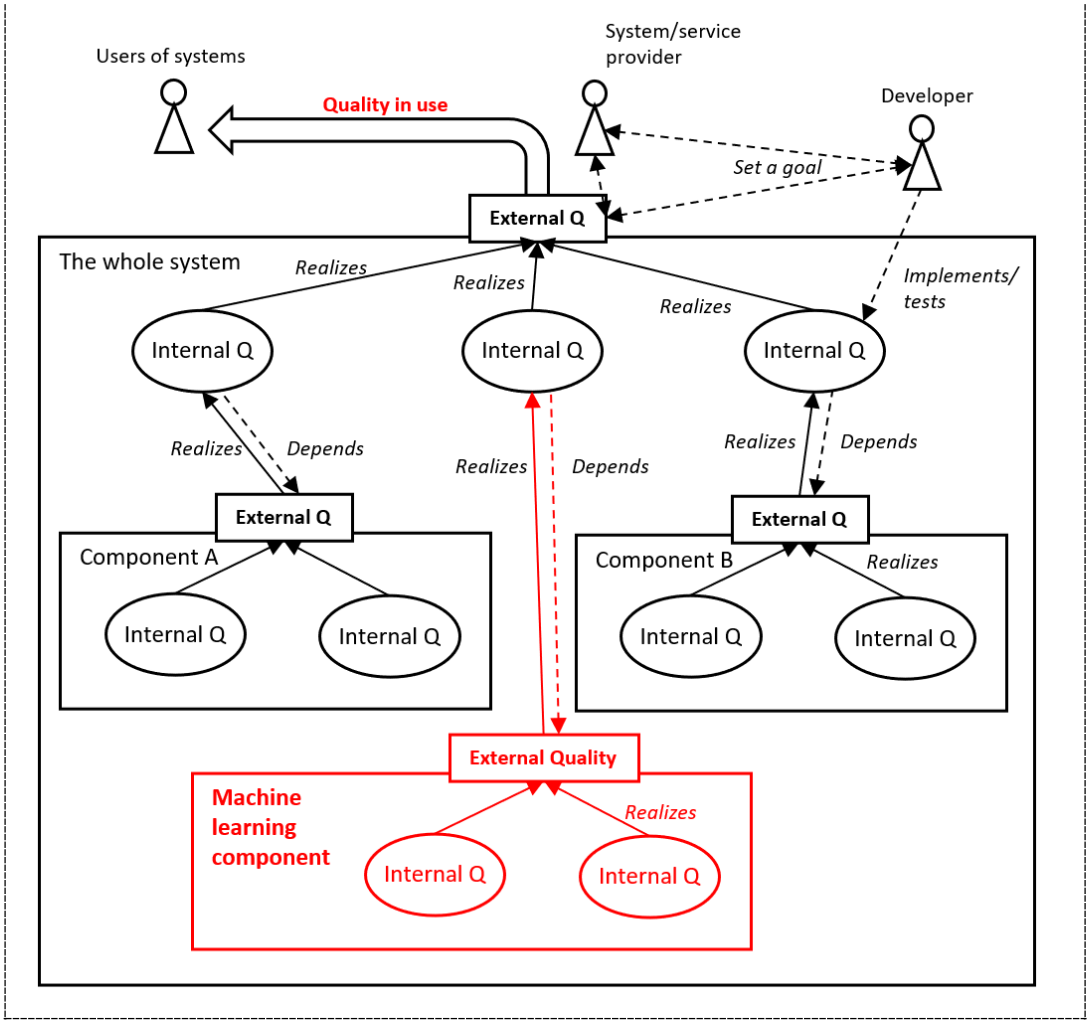


Figure 1: Hierarchical quality model

The Guideline asserts that the *quality* of a machine-learning-based system consists of the following three characteristics:

- *quality in use* to be satisfied by the whole system in use
- *external quality* to be satisfied by the system components built by machine learning
- *internal quality* that the system components built by machine learning uniquely have

Improving the *internal quality* of machine-learning-based components achieves the required level of their external quality and eventually realizes *quality in use* of the system (Figure 2).

Four viewpoints listed in Section 4.1 and an overarching viewpoint, AI security, were selected as external qualities of machine learning components, i.e., the targets of quality management. As internal qualities to achieve the external qualities, we focused viewpoints unique to machine learning components and extracted 14 viewpoints in 5 groups listed in Chapter 6 at this stage. The basic concept of quality management in the Guideline is as follows. First, the quality goal level is chosen for each of the first four of the external quality viewpoints. Then the level specifies the performance goals for each of the internal quality viewpoints. Finally, the performance goals

are achieved through various technologies and development process management.

Since quality-in-use characteristics of the whole system to be realized ultimately have different focuses depending on its application, it is not specified specifically in the Guideline (See the following supplement).

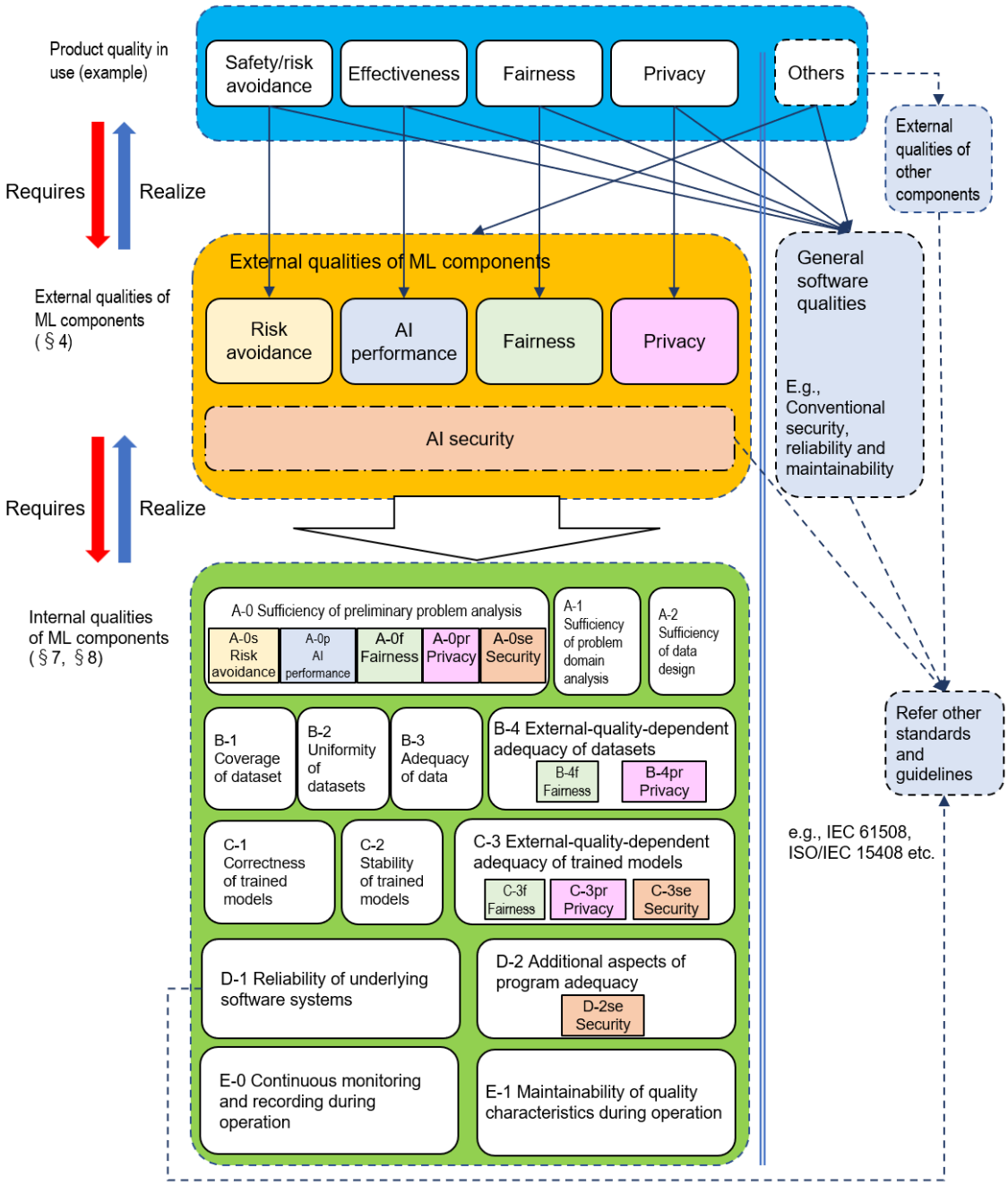


Figure 2: Overall structure for realizing product quality

Example 1:

First, consider a module to recognize objects based on front images installed on a self-driving car. One of the quality-in-use characteristics of automobiles is the *safety for avoiding collisions with obstacles under all environmental conditions under which automobiles can be driven*. In order to realize this, the object recognition module must have a characteristic that it *can correctly recognize objects in all possible climate conditions, times, etc.* as an external quality. Internal qualities required to realize that include the completeness of the set of conditions learned by the model, which in turn is realized during the process of, e.g., preparing machine learning training data.

Example 2:

Next, examine AI to estimate stock prices included in automated stock trading services. One of the quality-in-use characteristics of the whole service is the *maximization of profits*. A module to estimate stock prices must have a characteristic that *minimizes errors in stock price predictions and maximizes the sum of envisioned trading results* as an external quality. The internal qualities required to realize this include, for example, the precision of reasoning of machine learning. This is then realized by optimizing the machine learning training parameters.

(Supplement) When a product or a service is developed using one of the strict processes to evaluate the safety and reliability of the whole industrial system, such as IEC 61508 [13] or IEC 62278 [17], the process of designing the whole conventional system naturally specifies the required external quality of the machine learning components.

On the other hand, products or services that have a particularly high demand for machine learning technology, such as IT services, often have a lower presumed risk level than industrial systems. Therefore, applying a strict risk management process to the whole system development may not be necessary.

Moreover, when we expect machine learning components to “make a wise decision” at all in a use of the AI technology in general, their external quality viewpoints listed in this section may often have somewhat direct similarity to the external qualities and quality-in-use characteristics of the whole system. The above two examples illustrate such cases.

Based on such observations, Figure 2 is drawn so that quality-in-use characteristics of the whole system and external qualities of machine learning components are paralleled. Moreover, five quality-in-use characteristics, *safety, effectiveness, fairness, privacy,* and *AI security*, are exemplified.

1.5 Structure of the Guideline

The structure of the remaining parts of the Guideline is as follows.

- Chapter 2 clarifies the scope of the Guideline and its relationship to existing standards.
- Chapter 3 mentions specific methods of operating and applying the Guideline.
- Chapter 4 outlines 5 types of external quality characteristics. The interactions between external quality characteristics and their relationship to other quality requirements are also mentioned.
- Chapter 5 explains the details about quality-in-use characteristics mentioned in Chapter 4 including decisions on leveling.
- Chapter 6 details the internal quality characteristics.
- Chapter 7 enumerates the items to be addressed mainly during the development phase among the internal quality characteristics described in Chapter 6 and shows points to consider about them and possible methods for their realization.
- Chapter 8 shows examples of specific measures for items among the internal quality characteristics that should be continuously addressed after starting operations.
- Chapter 9 provides a list of examples of specific technologies and measures that can be used to achieve internal qualities as reference information.
- Chapter 10 presents explanations of the specific background and assumptions for each of the three external qualities: fairness, privacy, and AI security.
- Chapter 11 shows reference information, such as the relationship to other guidelines.
- Chapter 12 shows reference models about the development processes whose outline is explained in Section 3.8.

(Note)

The Guideline can be read in the following ways (other than reading in order from the beginning).

- Understand the overview of the approach to apply the Guideline in Chapter 3.
- Supplement prerequisite information as needed in Chapter 10.
- Read Chapter 5 to determine the quality level referred to in the development process.
- Refer to Chapters 6, 7, and 8, and identify the checkpoints required at each level and understand the specific approach of each.
- Refer to each section of Chapter 9 as needed for technology that is applicable to the above checkpoints.

2. Overview

2.1 Scope of the Guideline

2.1.1 Products and systems subject to the Guideline

The products and services subject to the quality management method described in the Guideline are generally information processing systems, including industrial equipment, consumer products, and information services, and particularly the systems containing components built with the use of the machine learning technology (hereinafter referred to as machine-learning-based systems).

The Guideline is currently applicable to machine-learning-based systems implemented using *supervised learning*. However, the basic approach of the Guideline can also be applied to some of the systems implemented using other machine learning methods, such as unsupervised learning, semi-supervised learning, and reinforcement learning, especially when test datasets that are at least partially complete with ground truth labels are available for the development of the systems (for example, when a system is developed using reinforcement learning to optimize machine operations and some of the operations are obviously to be prohibited). Future revisions of the Guideline will consider specific methods of handling them.

On the other hand, for applications of methods such as reinforcement learning, where the developer cannot present the correct answer (for example, AI that plays a game better than the developer), a different approach is obviously required, including the method of identifying functional requirements. Quality management for such applications is outside the scope of the current version of the Guideline and will be discussed in the future based on analysis of case studies.

2.1.2 Target of quality management

The quality characteristics that the Guideline aims to set goals for, manage, and guarantee are the external quality of software components built with machine learning technology (hereinafter referred to as *machine learning components*) that are incorporated in the machine-learning-based system. The external quality corresponds to the effect that such components have on the system in which they are embedded.

On the other hand, the quality management described in the Guideline focuses directly on the internal qualities of machine learning components.

The quality in use of a system is realized by the synergy of the external qualities of all the components of the system. Moreover, the external quality of each component depends on the

quality of its internal subcomponents and its internal quality characteristics, i.e., the internal quality of the component itself. The quality management of a machine-learning-based system is realized by identifying the requirements for its machine learning components as internal quality characteristics and by managing the process to achieve them (Figure 3). Note that the Guideline addresses the software and hardware used to implement the system components other than the machine learning components only when necessary, for example, to explain their relevance to the quality management of the system.

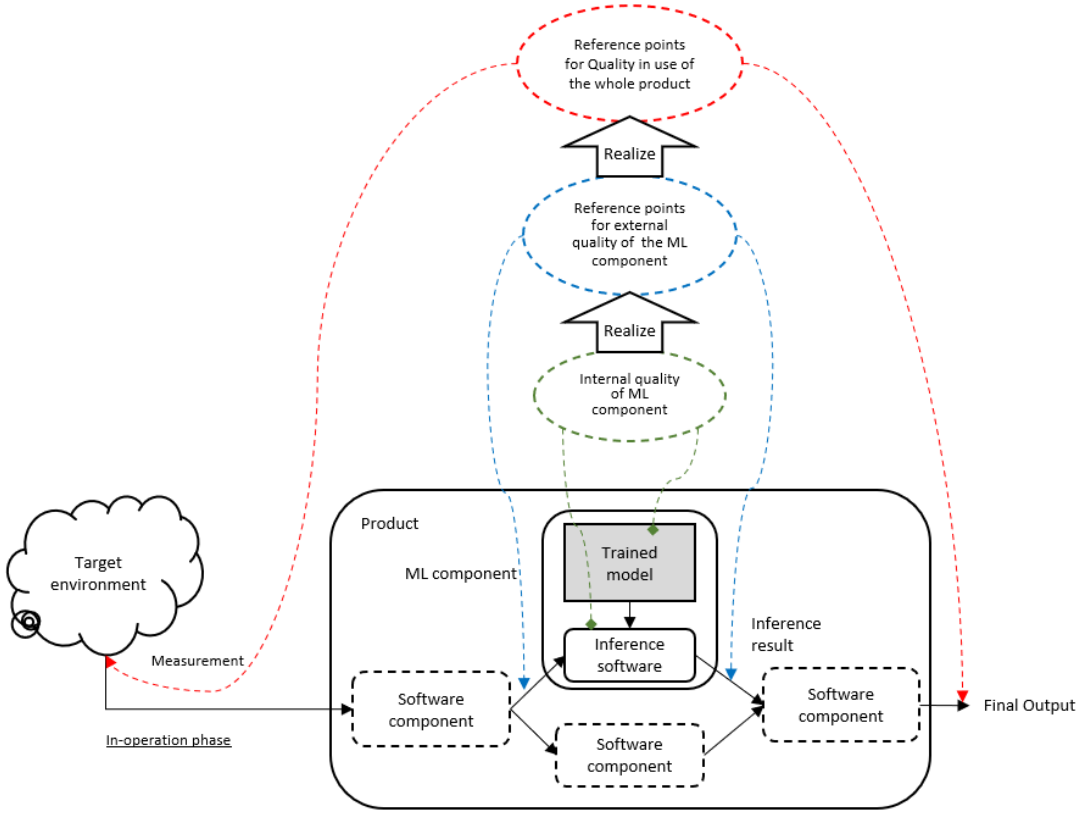


Figure 3: Quality requirements for an entire AI-based system and means of achieving them

2.1.3 Scope of quality management

In the Guideline, *quality management* refers to a process of quality activities across the lifecycle of machine-learning-based systems, from the planning stage to the operational stage. It includes such meanings as quality goal setting, planning, confirmation, quality assurance, and management. Although the scope of this *quality management* is broader than that of quality assurance or management of general software, it does not include organizational planning and

management of responsibilities or resources, which are included in quality management as defined in documents such as ISO9001.

2.2 Relationship between the Guideline and other standards for system quality

This section presents the relationship between the Guideline and the existing international standards, primarily those focused on information system quality. Also see Section 11.1 for the relationship between this Guideline and the guidelines concerning superordinate concepts such as sociality.

The Guideline presents quality management methods for a variety of machine learning applications. For the systems that require safety, in particular, it serves as a complement or extension of some of the conventional functional safety standards (e.g., IEC 61508-3 [15], IEC61508-4 [16]).

- For systems that require the highest levels of functional safety, IEC 61508-1 [13] or other equivalent standards are the primary standards to be applied.
- Additionally, the Guideline lists the safety issues specific to machine learning and the methods to address them in order to achieve the required safety level of the machine learning component in the targeted system. It then proposes a methodology for complementing or replacing the methods introduced in IEC 61508-3 for conventional software.

2.2.1 ISO/IEC 15408 security standard

The Guideline and information system security standards such as ISO/IEC 15408 [3] are independent, and they should be applied simultaneously when needed. Information security is a mandatory requirement for systems that require integrity and availability to realize safety discussed in the Guideline. The basic countermeasures defined in those standards are also applicable to machine-learning-based systems.

2.2.2 ISO/IEC 25000 series software quality model

The ISO/IEC 25000 series which define software quality models, especially ISO/IEC 25010 [7], can be partially applicable to machine-learning-based systems.

Those standards analyze the quality of software-based products, assuming that they rely on conventional software. While the quality properties listed in those standards can sometimes be

achieved also in machine learning components, they do not necessarily correspond to the properties identified in the Guideline as a result of the analysis and decomposition of the machine learning process.

Although there is a proposal discussed at an international level to extend the above standards to cover machine learning and AI, we continue to discuss this matter in view of future standardization.

2.3 Definitions of terms

This section shows definitions of terms used in the Guideline. Some of the terms on artificial intelligence and machine learning are subjects of discussions in ISO/IEC 22989 [4]. Definitions of those terms in the Guideline are to be aligned after the conclusion of discussions.

2.3.1 Terms related to machine-learning-based system structure

1. Machine-learning-based systems/systems using machine learning technology
   A system containing software components (machine learning components, 2.3.1.2) implemented by applying machine learning technology.
   (Note) The common terminology “machine learning systems” refers to machine-learning-based systems or machine learning components (2.3.1.2) depending on the context.

2. Machine learning component
   A software component implemented by applying machine learning technology. A machine learning component realizes the functions of trained machine learning models (2.3.1.5) as software. Generally, this component consists of a prediction/inference software component (2.3.1.6) implemented as software and a trained machine learning model (2.3.1.5) incorporated as a fixed input.

   In addition, pre-processing and post-processing whose content and implementation depend on the parameters or hyperparameters of the machine learning model may also be considered part of the machine learning component, assuming that they play an integral part in the quality control of the component. On the other hand, pre-processing and post-processing that contribute to quality control based solely on the problem definition regardless of the model, are considered external components that encompass machine learning components.

3. Machine learning algorithm
   An algorithm that provides the calculation method for prediction/inference of machine learning and the calculation method for obtaining the former calculation method through training. The former calculation method corresponds to prediction/inference software

component (2.3.1.6), and the latter corresponds to the trained machine learning model (2.3.1.7).

There are a lot of types of machine learning algorithms, such as neural networks, support vector machines, and decision trees, and so on. Appropriate algorithms are selected based on the type and purpose of the knowledge to be obtained by the machine learning components.

4. Hyperparameter
   A value given to the training software component (2.3.1.7) to configure the execution of machine learning training. It may need to be adjusted as the training progresses. In some cases, hyperparameters may not exist, hyperparameter adjustment may be intentionally omitted, or a technique for automatic adjustment may be used.
   (Note) As regards the diversity of mathematical structures that can be subject to machine learning, there is arbitrariness in choosing which scope is fixed as *machine learning algorithms* prior to the start of training and which scope is set/adjusted during training as *hyperparameters*. The choice depends on the developer's decision about the software implementation method and the training policy. In some cases, the structure embodying the calculation formula itself is treated as data and subject to automatic adjustment for optimization as a part of hyperparameters.

5. Trained model/trained machine learning model/knowledge base/trained parameter
   Information that defines the functional operations of the machine learning component and is produced as a result of training
   (Note 1) *Machine learning model* in the Guideline refers to a trained machine learning model or the data processed during the training to produce such a trained model.
   (Note 2) What is simply called *parameter* in the context of machine learning technology often refers to this trained machine learning model or the numerical data included therein.
   (Note 3) Mathematical structures such as neural networks and Bayesian networks are sometimes called *graphical models* or *statistical models*. Accordingly, the terms such as *machine learning model selection*, *model design* and *untrained model* may be used for the selection of a machine learning algorithm and the setting of their parameters.
   (Note 4) In the literature, the term *trained model* is sometimes used to refer to *a result of model training*, and sometimes to specific software that implements a trained machine learning model. The latter is used in contrast to *trained parameters*. In the Guideline, the term *trained model* is used for the former meaning while the latter is referred to as *machine learning component*, to clearly distinguish a result of training itself and pre-implemented prediction/inference software component.
   (Note 5) When the context is clearly about machine learning or AI, it may be no problem to simply call it a trained model.

6. Prediction/inference software component

The part of a machine learning component that is used in operation and is the unchanging software implementation of machine learning models. This component receives a trained machine learning model (2.3.1.5) as static or semi-static input and data obtained from real environments as dynamic input.

7. Training software component
A component to generate trained machine learning models (2.3.1.5) using training datasets. This component implicitly corresponds to the prediction/inference software component (2.3.1.6). These components are a pair; they are the implementations of the different aspects of the same machine learning algorithms and are usually used together.

Although a training software component is usually not part of the machine learning component used in operations, in some cases it is (e.g., a system that performs online learning or reinforcement learning in operations).

(Reference) The relationships between the terms used in this section are shown in Figure 4.

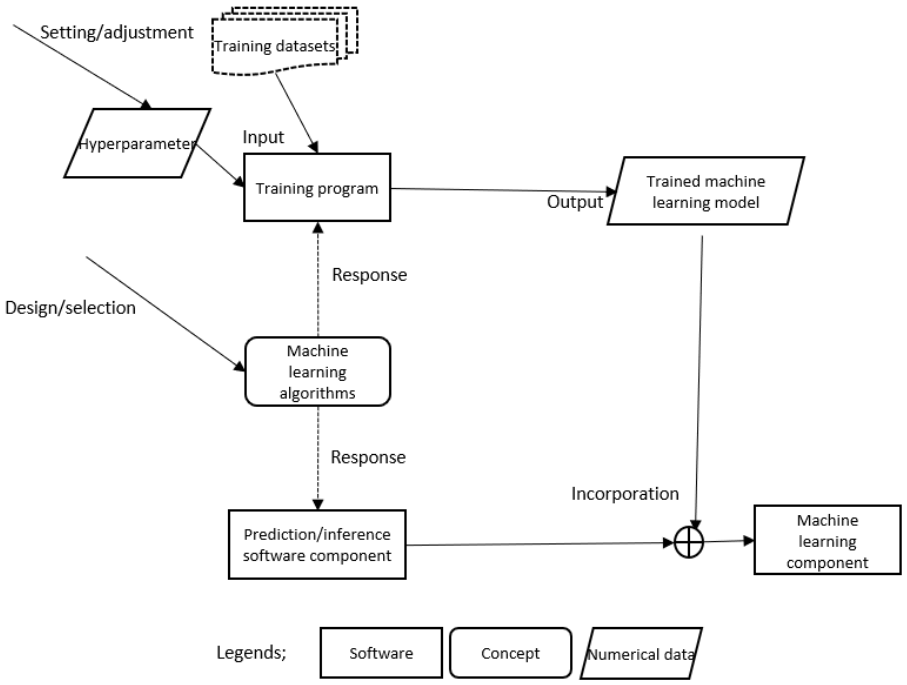


Figure 4: Relationships among terms concerning the structure of machine-learning-based systems

2.3.2 Terms related to development stakeholders and their roles

1. Service provider
   An entity that uses machine-learning-based systems for its own purposes, sells them, or provides services based on them to customers.
   Moreover, as regards a business structure in which software providers, etc., plan and develop systems in advance in anticipation of services provided or used by others and sell them as packaged or customized systems, those entities that engage in the planning and development are treated as a type of service provider.

2. Self-development entity
   A product development entity that designs and implements the machine learning components to be used in its products.

3. Development entruster
   A product development entity that asks others to implement machine learning components, regardless of the type of contract (whether labor supply or task completion).

4. Development entrustee
   An entity that implements machine learning components upon request from the development entruster.

5. (Development) stakeholders
   All entities that engage in development and operation of machine-learning-based systems, including product operators, self-development entities, development entrusters and development entrustees.

2.3.3 Terms related to quality

1. Harm avoidance (risk avoidance / safety)
   A quality characteristic concerning the avoidance of negative effects, such as human harm, economic loss, and opportunity loss, on operators or users of products or third parties caused by undesirable judgements of machine learning components. The improvement of *harm avoidance* corresponds to the concept of *risk reduction* in the safety field.
   [Defined in Sections 4.2 and 5.1 in the Guideline]

2. AI performance
   This characteristic concerns the ability of machine learning components to produce outputs expected by machine-learning-based systems and their users that are, on average in the

long run, maximally good in terms of correctness or probability. AI performance is evaluated based on overall performance rather than whether individual outputs are acceptable or not. **[Defined in Sections 4.3 and 5.2 in the Guideline]**

3. Fairness
   This characteristic means that the outputs of a machine learning component or their distribution are not affected by differences in some of the attributes of the people on whom the machine learning component is making judgements, or that such an effect is kept sufficiently small.
   (See Sections 4.4 and 5.3 of the Guideline)

4. Attack resistance
   This characteristic is concerned with preventing machine learning components or machine-learning-based systems from behaving contrary to the operator's expectations, but in accordance with an attacker's intention, in response to input data or external environments intentionally created by the attacker.

5. Ethical considerations
   This refers to the extent to which the behavior of machine-learning-based systems is appropriate in a human-centered society.

6. Robustness
   This characteristic indicates the ability of systems to maintain their level of performance under all circumstances.

2.3.4 Terms related to the development process

1. System lifecycle process
   A process management model looking down the flow from the planning of systems to the end of operation and disposal.
   [Source of definition: ISO/IEC/IEEE 15288[2]]

2. Agile development process
   A collective term for agile and value-driven development approaches.
   (Note) This term originates from the Agile Manifest announced in 2001 and can be used to refer to various forms such as scrum and XP.

3. Proof of concept
   Preliminary development activities carried out with the aim of validating feasibility of ideas

to be realized and prospective solution for the problem, rather than their realization as products.

4. Systems engineering
   An approach spanning across several fields to deploying balanced system solutions that address the needs of diverse stakeholders. It applies both management and technology processes and strikes a balance between them to reduce risks to project success

5. RAMS (Specification and demonstration of Reliability, Availability, Maintainability and Safety)
   In the Guideline, RAMS refers to the concept of the comprehensive system lifecycle process specified in the IEC 62278 [17] (EN 50126) standard on the reliability management process in the field of railway service.
   (Note) *The RAMS standards* may refer generally to IEC 62278 itself.

2.3.5 Terms related to the use environment

1. Environment
   In the Guideline, this term is used in three contexts: 1) external environment, 2) computing environments, and 3) operational environment.

2. External environment
   The physical environment or cyberspace that lies outside of a machine-learning-based system and sometimes becomes a source of input to the system, interfering with the system and vice versa.

3. Open environment
   An external environment in which the system operation is greatly affected by the condition of the people other than the users or the things such as the nature.

4. Environmental condition
   Characteristics that denote the differences in the state of the external environment. In the context of machine learning quality management, characteristics that cause changes in hazard states and risks are particularly noteworthy.

5. Computing environment
   A software execution environment in which programs such as machine learning components, prediction or inference software components, and training software components run. Depending on the situation, it may include the hardware environment that supports the above, in addition to the operating systems and middleware.

6. Operational environment
   An operational environment includes a computing environment and the human operational structure to work with it.

7. In-operation environment
   The operational environment in which machine-learning-based systems are run for production use.

8. Runtime (computing) environment
   A computing environment consisting of computers, clouds, and IoT devices, used as an in-operation environment to run machine-learning-based systems, including machine learning components.

9. Development environment
   A computing environment in which machine learning components are built and modified, where such modifications do not directly affect production use, and an operational environment that includes the computing environment.

2.3.6 Terms related to data used for machine learning

1. Training dataset
   A set of data used to train machine learning models in the iterative training phase.

2. Training data (instance)
   Data included in training datasets.
   (Note) Generally, the term *data* is used for both a single instance and a set of instances. In the Guideline, the term *data* is used only for the former, while the term *dataset* is used for the latter, to avoid this ambiguity. That is, the term *dataset* is used when the subject is regarded as a single entity and its distribution and total size are to be discussed. The term *data* is used to denote individual data points or discrete data points prior to their incorporation into a set. The relationship between these terms is described by expressions of the relationship between a set and its elements, such as *added to the dataset* and *included in the dataset*.

3. Validation dataset
   A set of data used to evaluate the convergence of machine learning models in the iterative training phase.

4. Test dataset

A set of data that is used as input to tests as one of the means to assess whether the model has the desired quality and performance in the quality control and assurance phase.

5. Adversarial example(s)
   Data that is intentionally designed to be input into a machine learning component and cause it to make inference that contradict human assumptions or intuition.

6. Overfitting
   A phenomenon in the training of a machine learning model in which the model is excessively adapted to the data in the training dataset and it does not produce appropriate outputs in response to input data that are not part of the training dataset.

7. Attribute
   Characteristics derived from the analysis and classification of elements constituting the environmental condition during the problem domain analysis for machine learning.

8. Attribute value
   A state with respect to one of the attributes of the environment identified during the problem domain analysis for machine learning.

9. Label
   An identifier attached to each data point in datasets used in supervised learning to solve classification problems, indicating the correct class to which the data belongs.

2.3.7 **Other terms**

1. (Software) regression
   In software engineering, this term means that, when software is improved, it stops behaving as expected in response to inputs that did not cause problems before.
   (Note) In machine learning and statistical analysis, the term regression is often used to refer to regression analysis. This Guideline does not do so because it can be confusing.

2. Safety integrity level (SIL)
   The functional safety levels of products, specified in IEC 61508. Used to show the goals and achievement of functional safety of products.
   [Definition source: IEC 61508-1]

3. Key performance indicator (KPI)
   An indicator that quantifies the achievement of functional requirements enabled by the

outputs of machine learning components that are part of a machine-learning-based system.

4. Continuous learning
   A form of operation that involves data collection and additional machine learning training to update the trained machine learning model at appropriate times.
   (Note) This is a concept including not only *online learning* described below but also *additional offline learning*.

5. Online learning
   A form of system implementation in which additional training is carried out and the results are incorporated into the in-operation environment without going through a validation process in the development environment.
   (Note) The Guideline mentions another form of system implementation as *additional offline learning*, where additional learning is performed in the development environment and the results are used, after going through the assurance process, to update the model parameters in the in-operation environment by means such as firmware updates.

3. How to apply the Guideline

In this chapter, we overview the process of applying the Guideline.

Although this chapter focuses on the development of machine learning components, it includes processes, such as whole system analysis, that are likely to have been commonly performed in the past to the extent necessary for explanation. Therefore, in cases where the development team has built a specific development process to conform to international standards and its adequacy can be reasonably explained, it is permissible to adopt that existing process with modifications to align it with the process shown in this chapter, such as adding any necessary stages listed here.

3.1 Business risk analysis

3.1.1 Preliminary assessment of the need for safety functions and identification of applicable standards

- Evaluate in advance whether the machine-learning-based system requires safety functions against risks of a certain level or higher, i.e., roughly higher than SIL 1, where non-negligible personal injury is likely.
- In cases where safety functions are required, select an applicable standard, either IEC 61508 or one of the standards for the application field and follow that process.

3.1.2 Identifying system purpose, goals, and functional requirements

- Roughly identify the purpose, the expected diversity of the environment, and the KPIs to be achieved in using the system.

3.1.3 Examining risk scenarios related to system use

- Examine the possibility that the functional requirements of the system will not be met, or that the system will become incompatible with the purpose of use or societal requirements, and list the damages and other problems that may result. This step is equivalent to a risk assessment.

3.2 Identifying quality-in-use characteristics

- Using a specific quality metric for the system, identify the quality characteristics required when the system is in use.
  - The following is an example of what could be done if there is no suitable option: Provisionally take the five external quality characteristics listed in Chapter 4 as quality-in-use characteristics to see if any of the problems examined in the previous section are relevant to them. Assess the severity of these problems using the criteria listed in the sections of Chapter 5 to determine the target quality levels.
  - Determine the maximum level for each of the five characteristics. Use this as the level of quality in use that the system should achieve.
- However, in cases where an existing safety standard is adopted in Section 3.1.1, the safety level concerning physical damage shall be determined according to the quality indicators specified in the existing standard, such as functional safety, etc.

3.3 High-level system design

3.3.1 Identifying functions to be provided by machine learning components

This section can be skipped in cases where the functions which machine learning components should perform within the system, e.g. those for safety, have been fully identified in a conventional system development process, e.g. IEC 61508 [13].

In cases where quality-in-use characteristics of the entire machine learning system have not been identified, quality-in-use characteristics and external qualities of machine learning components are identified through the following process.

3.3.1.1 Design of the components of the system and identification of the components that contribute to the achievement of the quality-in-use characteristics of the system

- Design a system as a combination of components and distinguish functions implemented by machine learning components from functions implemented by mechanical components or conventional software components.
- Analyze which of the components are essential to achieving the quality-in-use characteristics.

3.4 Determining external quality levels

Identify the external qualities and their levels that machine learning components should achieve. As a clue, consider the functions to be implemented by the machine learning components identified in the previous section 3.3.1. For each of the five external qualities listed in Chapter 4, use the following steps to examine the level to be met.

- Use the following procedures to determine the level of external quality that the machine learning component should achieve in order to contribute to quality-in-use characteristics.
- With respect to risk avoidance, particularly risks of physical or human harm, if you choose to apply a conventional functional safety standard, you must prioritize an analysis based on that standard, and you must adopt the functional safety level specified by the standard as the risk avoidance level.
- In other cases, do as follows:
  - Basically, use the quality in use levels of the overall system as a substitute for the target external quality levels of the machine learning component.
  - In cases where undesirable output from a machine learning component is monitored and corrected, i.e., overwritten by a software component that works in serial or parallel with the machine learning component (Figure 3), and the quality of the software component is likely to be realized sufficiently by a conventional method, set the target quality level of the machine learning component to one level lower than the overall system level.
  - If a machine learning component neither contributes to nor interfere with the achievement of a particular quality of the whole system, then do not set a target level for that quality for the machine learning components (or set it to Lv 0).

3.5 Selection of verification methods for each internal quality

- For each item of the internal quality characteristics listed in the sections of Chapter 6, derive a target level from the target level of the external quality characteristics determined in the previous section by applying the correspondence shown in Chapter 6.

3.6 Quality control activities

- Find a method to achieve the target level of the internal quality characteristics selected in the previous section by referring to Chapter 7.
- Ensure that each characteristic is achieved using the technologies and processes listed in each section of the chapter, and other technologies deemed equivalent.

3.7 Quality control during operation

- Referring to Chapter 8, determine the methods to be implemented during the operational phase from the methods selected in Section 3.5 to achieve the target levels of internal quality characteristics.
- Implement the selected technologies and measures and realize the internal quality characteristics during the operational phase.

3.8 Overall lifecycle process

3.8.1 Relationship between development through iterative training and the quality management lifecycle

Machine-learning-based systems are sometimes developed using the traditional waterfall process, but they are also often developed using more adaptive and flexible development processes, such as the introduction of a preliminary experimentation phase called PoC and agile development. In addition, when developing a machine-learning-based system, it may also be beneficial to apply continuous development, which achieves better quality or adapts the system to environmental changes during the operational phase using data obtained from the real operations.

Therefore, the Guideline assumes a system lifecycle process that integrates the development and operational phases, which consists not only of early and late development phases in which software is actually built but also of the phases of formulating system specifications and post-deployment operations. In particular, the Guideline considers the following two processes to be essential to the system development and treats them as quality management subjects: finding quality objectives in the problem domain analysis phase prior to system design, and monitoring behavior, obtaining additional data, and performing additional learning during the operational phase.

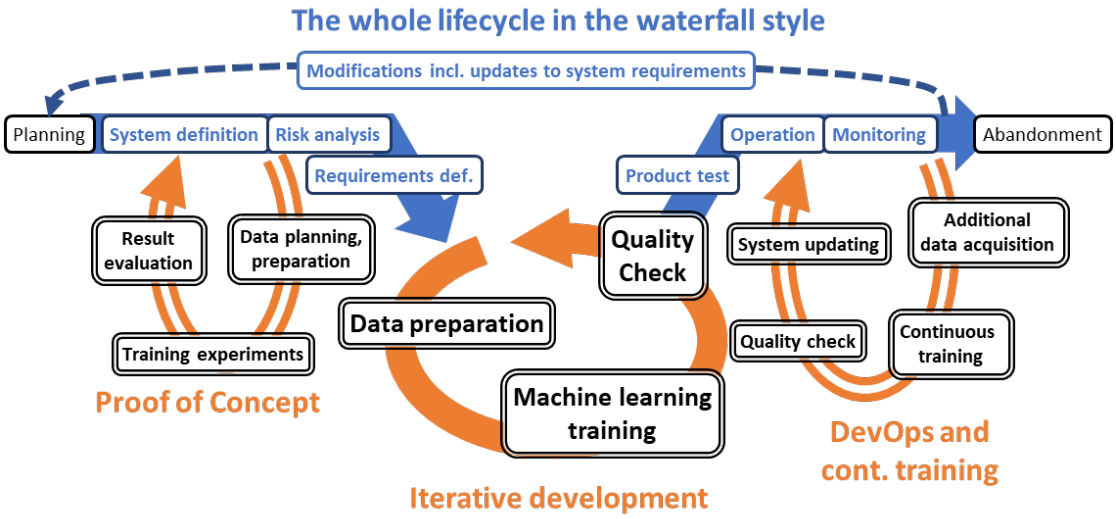


Figure 5: Conceptual diagram of a mixed machine learning lifecycle process

In the Guideline, the entire lifecycle is modeled as a hybrid process integrating agile or iterative development process, in which the progress is managed based on quality inspections or tests, and the whole top-down V-shaped development process, which focuses on process management (Figure 5).

Specifically, the requirements analysis of the overall system quality is considered as a linear process similar to a top-down process and is assumed to conduct detailed analysis as an independent process before the start of the actual system development and data management. Rework and iterative work in this phase should ensure that the same level of consistency is maintained in the final results as when work is restarted from the middle of the development process. In addition, processes during operations, such as obtaining additional data and monitoring, are handled as part of the iterative quality management process based on a concept such as DevOps, and should also be mapped, where necessary, to the concept of the top-down operations phase, such as in the RAMS specifications.

The so-called PoC development phase, which is often practiced in the AI development, should be considered as an important quality management phase, part of the process of the early analysis phase leading to the definition of requirements. The idea is that re-identifying and categorizing quality requirements when the knowledge and data obtained from the PoC are about to be used ensures that quality management in the main development phase and free exploratory development in the PoC phase are harmonized. Of course, in the actual development process, full quality management as in the main development phase can be partially introduced also into the PoC phase to reduce the burden of re-identification.

The lifecycle process described in this section is only a *model* that represents a plausible case and is not intended to show the only development process that developers should follow. This model is intended to help the reader understand the correspondence between the various

phases of the development process for different needs at hand and the sections of this Guideline, and to provide guidance on which of the quality management methods described in the chapters of the Guideline are suitable for which phases of the reader's development process.

(Note) The development process of AI has often been categorized as a non-waterfall iterative process. In iterative training, various works are performed, such as adding newly collected data, changing data selection, and adjusting parameters, in addition to changing algorithm implementation codes. Sometimes, these changes are reflected in the new implementation policy or the new specification, according to which new training is performed, accuracy is improved, and the goal is achieved. Such practices are quite common. On the other hand, if the product requires particularly high quality, then, even if an iterative development process has been adopted, some *quality gate* is often implemented in the form of pre-production decision or acceptance tests of the deliverables.

A model shown in Figure 6, based on the development process of AI understood as an iterative system, illustrates its relationship to the mixed process shown in Figure 5, which is the basic assumption of the Guideline.

Specifically, the development trials performed until the implementation policy (or sometimes called implementation specification) is concretized are mapped to several iterations in the PoC stage to identify the quality to be achieved. Then, one or more trials immediately preceding to the operational phase, i.e., trials consisting of training, quality testing, and release, performed based on the final specifications, are considered as the main development phase in the mixed process.

Occasionally, the specifications need to be changed again after the main development phase. This is quite common. From the perspective of the waterfall process, this is a step back from the implementation phase to the requirements analysis phase, but from the perspective of the iterative process, it just means that the product was still in the PoC stage, and we do not need to consider it as a serious step back. The PoC development stage has an implementation forecasting

aspect, and the knowledge gained in this stage is implicitly reflected in the main development stage. Therefore, even if this process is considered as a waterfall type, the implementation efforts in the PoC development phase are not wasted.

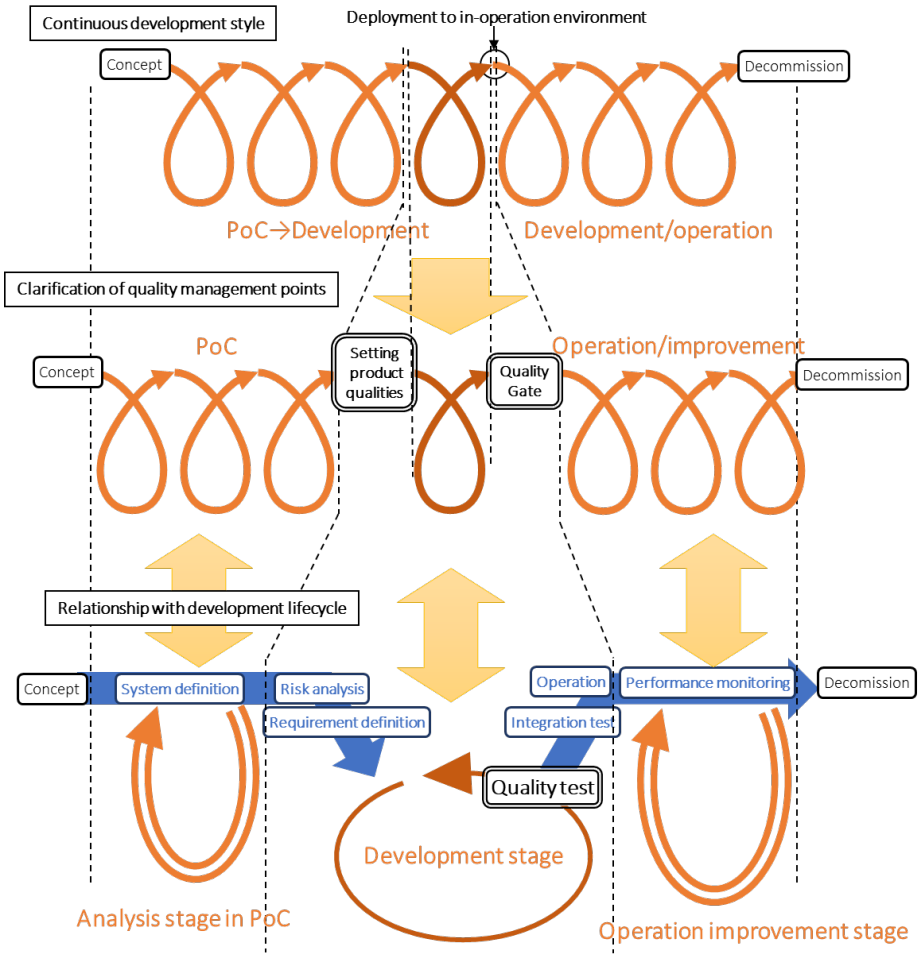


Figure 6: Relationship between circular process modeling and the model used in the Guideline (example)

(Note 2) As an application example, Section 12.1.1 (Page 252) describes the concept of the operation with changes in operational situations and multiple gates.

3.8.2 Notes on differential development (informative)

The development of systems and services that use software components, including systems

based on machine learning, often reuses existing software components. Such reuse in machine-learning-based systems poses difficulties for their quality management, especially when models that have been partially trained on some datasets are reused with adaptation through additional training.

A fundamental principle of the quality management of the systems that reuse software components is that the quality of the system should be achieved in the context and situation of its new use, whether in conventional functional safety or in the quality management of the machine learning covered by the Guideline. This principle should be consistently maintained from the analysis of the conditions of use, through the definition and implementation of functional requirements, to testing. Such consistency can be achieved, for example, by the following methods. The choice should be made according to the level of quality that the system should achieve and the situation in which the components have been provided.

1. Establish a new quality management process in the context of the new system, including detailed checks of the datasets that have been used to build the reused machine learning components, and perform quality management independently of the original components.
2. If the machine learning component to be reused has been subject to quality management activities equivalent to those specified in the Guideline and more stringent than those required for the new system, in particular, with respect to the *sufficiency of the problem domain analysis*, if the intended use situations of the new system are a subset of those of the reused component, consult the records of the original quality management and, if necessary, verify that the quality is not degraded by additional learning.

   As this method assumes that the quality management of the components to be reused has been sufficiently performed and recorded, this method may not be applicable if the components to be reused have not been designed with quality in mind from the start. In addition, it may not be possible to assess the inclusiveness of the expected situations, as the analysis of use situations implicitly assumes a certain context of use.

   Furthermore, a developer who provides machine learning components as general-purpose parts can improve their reusability by performing quality management activities in advance in anticipation of this type of reuse, and by clarifying their intended quality levels.
3. If the application requires a relatively low level of quality and the existing datasets are treated as black boxes, quality management activities may be planned with a focus on the testing phase. For example,
   - For Section 7.2 *Sufficiency of problem domain analysis* and Section 7.3 *Sufficiency of data design*, perform a new analysis for the new system to set new quality targets.
   - For Section 7.4 *Coverage of datasets* and Section 7.5 *Uniformity of datasets*, prepare new test datasets based on the new requirements analysis results, and check during the test phase that these characteristics are achieved to a certain degree.

- For Section 7.8 *Correctness of trained models* and Section 7.9 *Stability of trained models*, if additional learning is performed, evaluate the model quality in the validation and test phases using the performance indicators obtained during additional learning and using training datasets added based on new requirements analysis. If no additional learning is performed, perform a validation based on new requirements analysis result in the test phase, in the entire system integration test phase, or later.

However, given the current state of the art, the current descriptions in Chapters 7, 8, and 9 still assume that it is difficult to carry out sufficient quality management activities by using only spot checks of training results without analyzing the datasets used for training. In practice, it is advisable at this stage to supplement the current method with a white box approach as described in the previous item, after obtaining detailed information on the original datasets.

3.8.3 Relationship with development processes involving multiple stakeholders

In the actual development of AI, the work may be divided in various forms. For example, a service provider may do all the planning, development, and operation itself. In another case, it may outsource only the training stage. Yet in another case, it may entrust stages from system design to model development, training, and integration into the system. In the quality management process defined in the Guideline, a development entruster responsible for product planning plays a central role and quality management activities are carried out by everyone, including developers, etc. The management of each stage of the process is shared by agreement between the workers. As a result, the responsibility for identifying, from the purpose of the application system, the quality-in-use characteristics and the external qualities of the system may sometimes fall on the development entruster and at other times on the development entrustee, depending on the work sharing scheme[4]. In any case, it is important to reach an agreement to jointly ensure sufficient quality for the end user, and to clearly reflect the cost share in the contract, so as not to disrupt activities in the quality management process that continue until after the operation starts.

[4] In particular, consider cases where one of the acceptance requirements is that certain performance indicators (accuracy, etc.) are achieved with the data provided by the development entruster. This is a common form of contractual requirement for the development of AI. It should be noted that in such a case, the development entruster is responsible for ensuring in advance the so-called data quality, i.e. that the data is suitable for learning to fulfill the purpose of the products, as explained in Sections 6.1 to 6.7 of the Guideline. In cases where the development entruster is not able to validate or ensure the data quality, it will probably be necessary to explicitly outsource some of this work as "design support". It will also be necessary to consider the possibility of system development failing during training due to lack of data quality and the process going back to the development entruster.

3.9 Work sharing and quality control in AI development *not built from scratch*

The content of this section is informative.

The variety of in-house AI development *not built from scratch* is increasing. In addition to the outsourced development based on outsourcing contracts described in the previous section, the use of *foundation models*, which are mostly generative AI, and the use of *AutoML* services, which offer the advantage of no-code and low-code AI development, are becoming more popular.

This section mainly explains the case of ordinary outsourcing, in particular the role-sharing model for AI quality control, and other points to consider. We will also briefly discuss, from a process perspective, quality control in the case of AI development using foundation models and AutoML services, where there is mostly one-way provision and acceptance, as opposed to the case of an outsourcing contract based on two-way consensus building.

3.9.1 Outsourcing contracts

This section explains the points that development entrusters and entrustees need to consider when working together, including the points for dealing with AI-specific aspects. For contractual issues such as ownership of intellectual property and the sharing of damages, please refer to Section 11.1.3 "Contract guidelines for AI of the Ministry of Economy, Trade and Industry".

3.9.1.1 Exploratory approach

Outsourced development of machine learning components often requires an exploratory approach, as it is often difficult for the development entruster to provide clear KPIs and acceptance criteria to the entrustee at the start of such development. In such cases, enterprise agile practices are likely to be beneficial, as the difficulties of the above approach are similar to those we encounter when applying so-called agile development to the development process of the product requiring strict cost management and quality.

For example, the following process could be adopted. The PoC phase described in Chapter 12 can be considered as an Inception phase as described in Disciplined Agile [5] (DA). The development entruster and the development entrustee agree on the purpose of the PoC phase, i.e. the PoC exit criteria, and the deliverables to the next phase. They can do this by referring to the deliverables recommended in the *Inception* phase such as the testing strategy and the selection of the base architecture. Then they review the criteria and deliverables together at the

[5] A summary of best practices for businesses to adopt agile development. It was acquired by PMI (Project Management Institute) in 2019. https://disciplinedagiledelivery.com/

stage gate. It is highly important to set goals for the PoC stage gate.

Using this approach to AI development, the activities in the PoC phase can be organized as follows. The PoC phase should cover the following two aspects: *requirement analysis* and *feasibility study* (see Figure 7). For both activities, two perspectives should be considered: *what results should be required by the PoC exit criteria,* and *which of the PoC results should be delivered to the entruster*. In this way, all important issues can be addressed, and the subsequent phases can be carried out more smoothly.

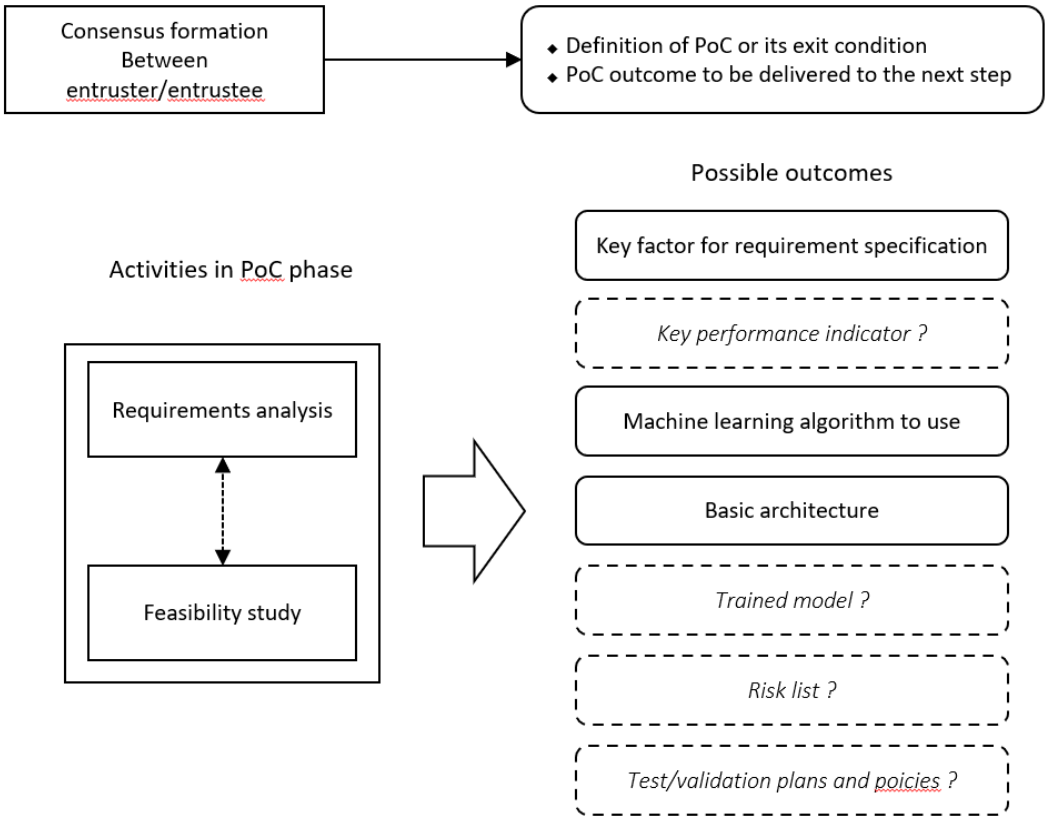


Figure 7: Activities and deliverables in the PoC phase

3.9.1.2 Clarifying the roles of the entruster and the entrustee

Clarifying the roles of the development entruster and the development entrustee to be played at each phase of the development process will reduce risk on both sides and prevent oversight of necessary tasks, leading to quality assurance of machine learning components.

An example of developing tailor-made AI from scratch is shown below.

Table 1: Example of role division

Step	Development entruster	Development entrustee
1. Select the safety standards to use	Examine primarily the applicability of existing standards.	Confirm the examined result.
2. Identify the functional requirements of the system	Describe the requirements in natural language, based on the requirements of XXX, and then provide it to the entrustee. Include qualitative expressions of quality-in-use characteristics.	Review the provided document to clarify the purpose and goals.
3. Identify the risk scenarios	Obtain a risk list for the similar product YYY. Present the constraints.	Analyze risk scenarios based on the established method ZZZ then present the result. Update as necessary.
4. Consider the quality-in-use characteristics	Present the priorities and constraints of the quality characteristics (e.g., which are essential).	Present the quality levels to be achieved for the four quality characteristics.
5. Design the system components	Review and confirm.	Design the system. Provide the materials AAA to the entruster for review.
6. Determine the external quality levels of the machine learning components	Review and confirm.	Select the target levels and propose them to the entruster.
7. Determine the internal quality levels of the machine learning components	Review and confirm.	Study the methods for ensuring internal quality characteristics, such as testing, and then select and propose target quality levels.
8. Realize the internal quality targets of the machine learning components	Provide the dataset sources. Review the test reports.	Perform the methods selected in 7. and report the results.

The details in the table would differ depending on the project, and the contents once decided may change, when the work is repeated.

(Example 1)

In some cases, neither a specific KPI nor a quality target can be presented by the entruster, and in such a case, a practical measure may be to make *the use of particular training dataset* the only clear requirement (the condition defined in the contract). In this case, additional training data are likely to be required as a result of conducting an internal quality analysis on data described in Section 6. It is preferable for both sides to recognize such a risk and agree on the necessary arrangements (their roles) in advance, considering item 7 in Table 1.

(Example 2)
When the development entruster *lacks understanding or has ambiguity* about the quality to be achieved by the machine learning components in the early stages, activities such as “Present the priority and constraints” in Table 1 can hardly be performed. In this case, the development entrustee (in a bottom-up manner, with a sufficient understanding of the outline of system design) can present an initial idea of the quality-in-use characteristics in the first PoC phase.
In the subsequent full-scale development phase, the role table should be revised to show that the entruster provides to the entrustee.

In the PoC phase, the content of each step in Table 1 may change with each iteration, but no step can be omitted. On the other hand, in the full-scale development phase, looping within Step 8 is usually sufficient if the goals for the PoC stage gate have been appropriately set and achieved.

However, depending on the project, the goals for the PoC stage gate should be dropped or compromised. This means that there may be no clear separation between the PoC phase and the main development phase. In this case, the development entruster should take the risk of the uncertainty of the quality management outcome and should decide whether to accept it and what to do with the work division to mitigate the risk.

3.9.1.3 Points to note when determining the detailed roles

When concretely defining the work and agreeing on the division of roles in the development lifecycle as described in the previous section, there are several points to consider. Here are the results of extracting and examining the AI relevant points to be considered from the IPA/SEC

"Guide to Quality Assurance in Connected World" [242].

1) The organization and agreements under which it operates to achieve accountability

In AI development, quality evidence can often only be provided in the form of process characteristics. In such cases, their definition and who can approve them should be clearly specified. In addition, it is necessary to consider the underlying risks in the process. For example:

- Regarding the training dataset preparation, it is not enough to specify that the preparation of raw data should be done by the entruster. It should also be specified who will perform preprocessing (e.g., data cleansing), and who and how will record evidence that this preparation has been appropriately achieved.

In any case, it is important to consider and specify *compromises and logics that both sides can agree on* in advance, given the time and cost constraints.

2) Testing

Even if the model training is left entirely to the entrustee, the entruster usually needs to have a deep understanding of the content of the *quality review and assurance* phase. To that end, it is desirable to have agreements that allow both sides to be as satisfied as possible at the time of placing the order, e.g., defining not only the evidence to be collected, but also the method and scope of testing, and what to give up in terms of effectiveness and efficiency. It is important that both sides cooperate for the purpose of *final quality improvement*. It goes without saying that *any action intended solely to gain approval during verification*, such as using test datasets provided by the entruster for training, should be strictly avoided.

3) Quality management during operation

Regardless of whether the model is updated in real time or not, the machine learning component, by its very nature, cannot do without continuous quality management even after it is released, as described in Section 12.3 “Quality monitoring and operation phase”. Therefore, the scope of work should include the design and implementation of operational quality management methods that optimally meet the functional requirements of the system.

For example, both sides can discuss and agree on the identification of *data to be collected,* such as performance evaluation data and environmental data that cannot be sufficiently obtained during development, and also the system implementation for obtaining and storing those data during operation.

3.9.2 Use of foundation models

For the safe and efficient use of generative AI, of which ChatGPT is the most notable example, various studies are underway, not only by the IT industry but also by the government. The user

perspective was considered first (for example, Ministry of Education, Culture, Sports, Science and Technology [47], Chiba Prefecture [46]). From the developer's perspective, it is also planned to revise the guidelines for developers issued by the Ministry of Economy, Trade and Industry [44] to include views on generative AI. Furthermore, at the 3rd AI Strategy Meeting [45] of the Cabinet Office, a policy was presented to integrate the guidelines created by each ministry and agency for each field within the year. Currently, we believe that the two most important issues in this area are as follows.

Issue 1: Division of responsibilities in the supply chain that has become more complicated than before.

Issue 2: Method for evaluating the output of generative AI, such as highly expressive text and images.

The following are the two important actors specific to the foundation model.

Foundation model developer	Train the foundation model
Service developer	Develop services for the intended use by *tuning* (fine-tuning, etc.) the foundation model and other means.

Unlike the case of the development entrusters (client) and the development entrustees (contractor) described in the previous section, there is no outsourcing contract between these two parties. Therefore, although the service developer uses the foundation model as material for their model, it is difficult for the service developer to assign to the model developer the roles necessary for each specific AI case.

Therefore, for the time being, the service developer needs to examine the quality information provided by the foundation model developer and implement individual measures to close the gap with the quality target to be provided. In doing so, the following two points should be kept in mind, and the necessary measures should be determined while adjusting the balance between the potential and risk of generative AI according to the objectives.

■ The following three points to be kept in mind regarding the division of roles in outsourcing, as described in the previous section, are basically the responsibility of the service developer.

1) The organization and agreements under which it operates to achieve accountability
2) Testing
3) Quality management during operation

■ As described in (Issue 2), there are many aspects of evaluation methods for generative AI that are still under consideration, and the quality information from the developers of the foundation models is also based on their own indicators.

3.9.3 Use of AutoML services

The features provided by AutoML services vary by vendor, but in general they aim to support a wide range of machine learning processes, as shown below.

- Preprocessing of the training dataset.
- Machine learning models (algorithms) selection
- Development of machine learning models (implementation of structural parts)
- Adjustment of various hyperparameters during training of machine learning models and execution of the training
- Deployment to the operational environment (exposing functionality via APIs)

When using an AutoML service, while the training dataset preparation process can be subject to the data or dataset quality control approach described in Chapter 7, the post-training processes may be limited by the following constraints.

1. Constraints on rationale and explainability
2. Constraints on quality improvement methods
3. Constraints on the form of results delivery

- Regarding the selection of machine learning algorithms, rationale or algorithm details are not provided in most cases, which can lead to an increased black box nature of the target AI.
- Since the model building and training processes are automated, the in-process quality improvement methods described later (various methods that could be used during the training phase) are not applicable. Also, only the metrics provided by each service can be evaluated during training.
- The trained machine learning models provided as deliverables may be in a generic format file that can be deployed anywhere, or the models may be primarily intended to be deployed on a specific platform and provide functionality via an API, allowing for comprehensive coverage up to the operational monitoring process.

In this way, the use of AutoML is subject to some constraints in exchange for the benefits of automation. Also, as with the use of foundation models, it can be difficult to assign the necessary roles for each specific AI project to the AutoML service provider.

However, a major difference from the use of foundation models is that the entire AI development, including the dataset for training, is *in the palm of your hand*, even though constraints impose uncertainty and inconvenience on some parts of the process. In addition, some vendors provide metrics that are relevant from an XAI perspective, and more progress can be expected in the future in resolving the aspects of constraints described in this section.

4. External qualities addressed by this Guideline

4.1 External quality characteristics to be achieved

First, the Guideline lists the four characteristics, namely risk avoidance, AI performance, fairness, and privacy, as axes of the external quality of machine learning components, and quality levels are specified for each of them. The characteristics are explained in Sections from 4.2 to 4.5.

It can be difficult to achieve all these characteristics in a machine learning model and to achieve optimal results for each simultaneously. For example, training a model with lots of data representing risk-prone cases to strengthen its risk avoidance capability can result in poor performance for other cases, particularly in terms of accuracy. It is also known that measures to improve a model's privacy or fairness can reduce its accuracy. When developing machine-learning-based systems, it is often necessary to prioritize these characteristics and strike a balance to maintain them at sufficiently high levels.

Next, the Guideline identifies *AI security* as a characteristic that influences the other four. This is a concern about malicious operations originating from outside the system affecting the level of achievement in the other four characteristics. While measures to address AI security may also reduce the general performance in the other four characteristics, they can be regarded as a means to maintain the other important characteristics, even in a hostile environment, especially when machine-learning-based systems are used in the real world. Section 4.6 explains the scope of the problems that fall within the AI security category.

4.2 Safety and risk avoidance

The *safety*, or risk avoidance, of a machine learning component is a property that concerns the prevention of undesirable judgements or behaviors by the component itself, as well as the reduction of the risk of physical or economic harm to users or the system provider caused by the system comprising it.

The following are specific examples:

- An object recognition system for automated driving fails to detect or identify an object.
- A contaminant detection system in a food factory's production line fails to detect foreign objects.
- A trading system places an order for a quantity of securities that exceeds an acceptable limit due to illegal input, such as spoofing.

We have defined seven levels of safety, ranging from one that affects many human lives and

business continuity to one that results in minor loss of profit opportunities (Section 5.1). As the safety characteristics are closely related to the properties specified in the safety specifications for conventional systems, the four most stringent levels (Levels 4 to 1) have been defined as equivalent to the levels defined in existing specifications (e.g. the Safety Integrity Level (SIL) of IEC 61508-1[13]). This compatibility makes them easier to use alongside the existing specification. In addition, three more levels have been added to represent the less stringent quality requirements typically associated with machine learning applications for IT services and smart devices. Such applications usually cause only minor damage, if any, which is not considered a risk in conventional industrial products. Their quality requirements are categorized as *out of scope* in the existing safety specifications. The new levels should make the level set more useful in practice.

In general, it is not practical to guarantee that *machine learning systems will always operate safely.* The machine learning components incorporated into such a system may not be of a sufficiently high quality to enable the system to exhibit the required quality-in-use. Current system development practices often rely on the surrounding software rather than the machine learning components themselves to ensure safety, much like a *safety valve* that prevents a boiler from bursting. As with the conventional safety specifications, the Guideline distinguishes between the requirements for quality-in-use characteristics of the entire system, and the requirements for external qualities of the machine learning components. Target levels for these external qualities should be chosen based on risk assessments of the entire system and its structural design (Section 3.3.1.1).

4.3 AI performance

As the second axis of quality characteristics, we focus on the application areas where the benefit brought by the machine learning function is most important, and call the quality required in such areas *AI performance*. Typical application scenarios where AI performance is more important than safety include a scenario where higher average performance is required than avoiding negative effects caused by individual misjudgments, such as retail demand forecasting and economic forecasting for investment decisions.

Of course, in some cases, both *AI performance* and *safety* are required. For example, if a demand forecasting application predicts very high demand for a product, this could lead to excessive purchasing. Such a purchase could exceed the assumptions in the operational environment, resulting in indirect negative effects and much larger economic losses than the difference in average profits would suggest. In this case, the requirement should specify an optimal balance between two quality characteristics, *AI performance* and *risk avoidance*.

Because the specific AI performance targets to be achieved depend on what is selected as the application's Key Performance Indicator (KPI) and vary from one application to another, we

focused on the degree to which these targets should be achieved and created three levels within it.

4.4 Fairness

In recent years, there has been extensive discussion about ethical considerations and fairness issues of AI and the norms in society of AI. While the movements of social demands and social principles such as the AI Social Principles and AI Governance are behind this trend, there are various difficulties regarding fairness, such as diversity of requirements and socially embedded unfairness. (See Section 10.1 for details)

In this Guideline, AI fairness is addressed within the scope described in the next section, Section 4.4 outlines the fundamental processes and key measures for ensuring fairness. Furthermore, specific measures for each internal quality are explained in Chapters 7 and 8. In addition, Section 10.1 explains the background to and difficulties of fairness, process details to ensure fairness, and supplementary information such as various tools.

4.4.1 Definitions of ethics and fairness in the Guideline

At present, there seems to be no clear and universally accepted distinction between terms such as “ethics” and “fairness” across social and technical domains. In the Guideline, ethics is defined—drawing from the field of sociology—as acting in accordance with societal norms, i.e., demonstrating “well-behaved” conduct. This concept of ethics extends beyond fairness; it may also include principles for avoiding certain types of decisions, ensuring specific forms of safety, and addressing dilemmatic situations.

In general, ethical requirements for systems are often implicitly shaped by societal expectations. They are derived from social norms that are not always explicitly stated and typically serve as constraints or considerations during the planning and design phases of a system, particularly in defining system requirements.

On the other hand, fairness in the Guideline is defined more narrowly: it refers to avoiding differential treatment based on variations in input conditions that are not explicitly defined as part of the system’s requirements. From a broader social perspective, this notion of fairness can be understood as a component of ethics.

Furthermore, the fairness of machine learning components—specifically addressed in the Guideline—is treated as an engineering-level concern. It is defined as the process of identifying fairness-related requirements that arise from ethical or functional considerations during the requirements definition phase, and then translating them into concrete system requirements by the system provider.

4.4.2 Considerations regarding sensitive attributes

Before outlining the fundamentals of quality assurance methods, this section highlights key points regarding *sensitive attributes*, an important perspective when designing appropriate fairness approaches.

In general, fairness requirements focus on preventing discrimination based on attributes such as race or gender, which are recognized as *sensitive attributes*—that is, characteristics that could lead to unfair treatment. It is required that no unjust disparity arises between groups differing in the values of these sensitive attributes.

It is important to note that, as illustrated in the examples below, simply removing all sensitive information from the training dataset does not necessarily achieve the intended fairness goals.

- In some cases, real-world data may reflect existing societal inequities. To prevent machine learning models from faithfully reproducing such biases, it may be necessary to *deliberately* utilize sensitive information in a controlled manner.
- In other cases, sensitive attributes may be indirectly represented through other correlated features, leading to unintended bias.

For the former, analysis grounded in the fairness requirements is essential. For the latter, it is necessary to correctly identify correlated attributes and carefully balance the potential disadvantages—such as reduced predictive performance—that may arise from removing or mitigating these related features. To conduct such analysis effectively, a top-down approach rooted in high-level fairness requirements is beneficial, along with proper documentation to ensure transparency and explainability.

The above summarizes the key points regarding sensitive attributes. To gain a deeper understanding of the fairness assurance process described in the following section, it is recommended to also read the details in Section 10.1.

4.4.3 Basic concept of fairness assurance

This section provides a brief overview of the basic model of fairness assurance, which is described in detail in Section 10.1.3.

In general, fairness requirements (or goals) are initially expressed in conceptual and qualitative terms, such as “users must not be treated unfairly.” In contrast, fairness metrics used in the development of machine learning AI—such as those applied to training datasets or model outputs—are quantitative indicators that measure degrees of disparity with respect to specific attributes. Therefore, a critical aspect of ensuring fairness is to appropriately bridge qualitative goals and quantitative metrics throughout the development process.

In the process recommended in this Guideline (Figure 15, Chapter 10), fairness is first

discussed at the level of external quality provided in Section 4.4.1 , focusing on ensuring fairness across features and attributes. In the internal quality assessment phase, qualitative fairness requirements are translated into measurable, quantitative criteria, taking into account actual data distributions. During the development and testing phases, these quantitative metrics—along with any additional indicators deemed necessary—are used to guide efforts to achieve fairness.

To implement the above model, the concept of “three types of fairness interventions”[155][247][176] is essential.

Pre-processing approach	Adjust the *training dataset* to reduce bias.
In-processing approach	Embed fairness into the model training process.
Post-processing approach	Modify model outputs to meet the goals.

Applicable approaches are determined by the scope within which developers can intervene in the model development process.

4.5 Privacy

Privacy, along with fairness, is one of the external qualities called for from ethical AI [37].

4.5.1 Privacy as an external quality

Generally, the right to privacy is understood to be one of the fundamental rights of natural persons. The right to privacy has evolved from legal principles discussed in Britain and U.S.A. in the 19th century based on the "Refusal to intervene from personal matters (the right to be left alone)”, and later "information privacy (the right to control information about oneself)”, which was established in the middle of the 20th century [254].

Services such as micro-targeting, recommendations, and profiling have become more prevalent with the advancement of AI. However, these are facing legal initiatives to prohibit or restrict them due to the potential risk of increasing intrusion into personal matters [29]. From the perspective of information system development, the problem is in determining whether the requirements for the functions and services provided are related to privacy. In other words, the concerns are related to the discussions on system requirements, which is beyond the scope of machine learning quality management.

The information privacy includes the basic notion that a piece of information about oneself should not be used without the consent of the data subject, who is a natural person associated with the personal information, and encompasses rights such as the "Right to be forgotten".

Aiming to balance the promotion of utilizing shared and distributed digital data with the protection of the right of the data subject, based on the 8 principles [27] of OECD Privacy, regulations are being discussed in various countries and regions. Laws related to the protection of personal information, such as GDPR[30] in Europe and CCPA [43] of California in the US are being developed [255]. The privacy concerns require compliance with the relevant laws by technical methods together with measures across the organizations from development to operations. The matters are being discussed along with the data governance[33][29].

Privacy as an external quality involves the appropriate management of the information privacy, providing suitable actions to violations of the information privacy and mitigating privacy risks throughout the system lifecycle, from development to operations, with the aim of complying with relevant regulatory laws. Furthermore, while compliance with the regulatory laws of the countries and regions in which the system is marketed is mandatory, technical measures for the protection of personal data are anticipated due to ethical requirements on the system.

4.5.2 **Privacy quality management**

From the perspective of machine learning quality management, privacy as an external quality should be considered when the training data contains personal data and refers to sensitive information. These include analyzing whether the training data contains personal data and applying appropriate technical measures such as removal of sensitive attributes, pseudonymization, or data protection processing so as to reduce the threat of re-identification. In addition, if the sensitive information to be protected is related to a system in an application domain or application sector that is defined by law, appropriate handling of sensitive attributes is legally required.

For simplicity, a trained model can be considered as a piece of derived information, because it is the result of data training. However, the re-identification (such as membership inference attacks) of training data from the trained model is known to be technically not impossible. Therefore, there is a risk of a vulnerability such that training data, not subject to protection processing, are leaked from the trained model. These vulnerabilities may sometimes be reduced by methods such as adjusting the distribution of training data so as to remove outliers, and employing privacy-preserving machine learning methods so as to make the re-identification of training data difficult. Namely, it is sometimes necessary to consider whether these techniques are effective as technical measures for protection of privacy in trained models.

When establishing a quality management system with regard to privacy, it is necessary to analyze, in advance, the relationship with the other external qualities, taking into account the characteristics of the system under development. Privacy is, indeed, related to fairness because of handling sensitive information of data. Furthermore, the introduction of cybersecurity measures will reduce the risk of information leakage in general, and as a result, improve the

protection of personal data as well. On the other hand, depending on the privacy-preserving machine learning methods employed, enhancing privacy may lead to a decrease in AI performance or a deterioration in fairness. Therefore, it is important to examine the adoption of an appropriate method for the enhancement of privacy in view of the trade-off relationships.

Furthermore, when it is difficult to achieve the required level of privacy with data protection processing or privacy-preserving machine learning methods, it is necessary to implement cybersecurity technologies or stringent organizational measures aimed at ensuring compliance with the regulations. Alternatively, by adopting suitable privacy protection measures and cybersecurity technologies, legal issues can be avoided and thus the costs associated with organizational measures can be reduced. In other words, the quality management related to privacy requires the consideration of a balance between reducing the burden of organizational measures and increasing the cost of adopting advanced technologies.

4.6 AI security

4.6.1 AI security as an external quality

As with other information systems, we should anticipate various malicious uses of machine-learning-based systems. Even when the above four external quality characteristics are achieved as expected in normal, natural use, if a person with malicious intent modifies the environment or input data, they can intentionally change the behavior of the machine learning model to prevent the achievement of these external qualities. Vulnerability to such malicious input has generally been identified as a weakness of machine learning. Especially in systems that are intended to be used by external users or in open environments, it is difficult to detect or prevent such malicious input data, and such vulnerability needs to be addressed from the design and development of machine-learning-based systems. We define *AI security* as an external quality to address security measures against such risks specific to machine learning.

In this Guideline, the external quality *AI security* refers to maintaining the confidentiality, integrity, and availability of assets to be protected, such as data, models, and systems, against "risks arising via trained models" [135]. For example, malicious data inputs (evasion attacks) to machine-learning-based systems such as those described above are subject to the external quality *AI security*, because a trained model's malfunction can cause system malfunctions. For another example, tampering with training data (data poisoning attacks) by exploiting the characteristics of machine learning algorithm is subject to the external quality *AI security*, because the contamination of a trained model can cause damage such as system malfunctions.

We remark that "risks that did not arise via trained models" are not addressed by the external quality *AI security*, because the security controls for conventional information systems address

them. For example, attacks that infiltrate the development environment of machine learning models to tamper with training data is not addressed by the external quality *AI security* because they are handled only by conventional information security measures. Attacks that tamper with input data to the system to cause software malfunctions such as buffer overruns are also outside the scope of the external quality *AI security*. These should be managed by the security management process as before and, if necessary, by conventional standards, such as the Evaluation Assurance Level (EAL) of ISO/IEC 15408 [3].

On the other hand, unlike conventional software, machine learning models trained with data can often malfunction with certain input data, even if the training data are within the scope of the specifications (e.g., image data from a real environment that have not been altered after being captured by the camera). Furthermore, when using publicly available (e.g., open-source) data or machine learning models for system development, it can be difficult to detect malicious modifications made to them. For these problems, conventional approaches such as pre-checking of software security do not work, and they should be addressed concurrently with other quality perspectives during the design and development process of machine-learning-based systems.

4.6.2 Quality management for AI security

Quality management for AI security consists of security risk assessment (Section 4.6.2.1) and security controls for vulnerabilities specific to machine learning (Section4.6.2.2).

4.6.2.1 Security risk assessment

A security risk assessment should be performed for the entire system lifecycle when examining the security measures of a machine-learning-based system, as with the assessment of a conventional information system (Section 7.1.5). The first step in risk assessment is to identify the stakeholders and assets to be protected that appear in the system lifecycle. Possible adversaries and attack surfaces are then listed, and possible threats and vulnerabilities are identified using attack tree analysis and other methods[6], to assess the impact on external qualities and quality-in-use characteristics.

For the purpose of risk assessment, this Guideline lists assets and stakeholders for systems in general that use learning models obtained through supervised learning, enumerates possible adversaries and attack surfaces, comprehensively enumerates and categorizes known threats and vulnerabilities specific to machine learning, and introduces security controls (Section 10.3).

[6] Definitions of assets, stakeholders, threats, vulnerabilities, and control measures are in accordance with ISO/IEC 27000.

The classification of threats, vulnerabilities, and control measures is based on the literature [135].

A risk assessment for machine-learning-based systems needs to identify the threats and vulnerabilities specific to machine learning, and also those for conventional information systems in the entire system lifecycle. However, identifying all threats and vulnerabilities and implementing sufficient security measures for all possible attacks may not be possible. For this reason, we need to prioritize the threats and vulnerabilities to be handled in the security risk assessment, and should implement control measures in the priority order.

In addition, considering that the system and environment can change during operation, security risk assessment should be conducted periodically not only during system design and development, but also during system operation.

4.6.2.2 Security control measure

Developers and operators of machine-learning-based systems should consider and implement control measures for vulnerabilities of each asset based on the aforementioned security risk assessment. Vulnerabilities to threats specific to machine learning can be broadly classified into (i) insufficient risk assessment based on the confirmation of credibility, (ii) insufficient measures to prevent and mitigate attacks, (iii) insufficient risk assessment using attack detection techniques, and (iv) insufficient measures to prevent and mitigate damage. The full picture and details of control measures for these vulnerabilities are detailed in Section 10.3.5.

Developers of machine-learning-based systems should examine and implement controls for vulnerabilities in the source of training data, the training datasets, the pre-trained models, the learning mechanisms (development software and development environment, such as training programs and testing programs), the trained models, and the other programs in the system design and development phases.

Specifically, the controls for vulnerabilities of the source of training data and the training datasets consists mainly of techniques to evaluate and improve the internal quality "B-3: Adequacy of data" (Section 7.6). In addition, the controls for vulnerabilities of pre-trained models, learning mechanisms, and trained models mainly consist of techniques to evaluate and improve the internal qualities "C-2: Stability of trained models" (Section 7.9), "C-3: External-quality-dependent adequacy of trained models" (Section 7.10), and "D-2se: Adequacy of programs in terms of security" (Section 7.12.1). Control measures for vulnerabilities of programs associated with machine learning components (programs for access control, pre-processing, post-processing, and risk monitoring and response), include techniques (Section 7.12.1) to evaluate and improve the internal qualities "C-3: External-quality-dependent adequacy of trained models" (Section 7.10) and "D-2se: Adequacy of programs in terms of security".

On the other hand, operators of machine-learning-based systems should implement security control measures in the system operation phase based on the aforementioned security risk assessment. Controls during operation are explained in the internal quality "E-0: Continuous monitoring and recording during operation" (Section 8.1).

4.7 Relationships among external qualities

Ideally, all five external qualities described earlier should be achieved in machine learning AI systems, but they are often in trade-off relationships with each other. The external qualities which should be prioritized depend, first of all, on the business requirements of the system. Since social demands are generally prioritized over business requirements, the external qualities that satisfy social demands usually take precedence over the external qualities that satisfy business requirements.

AI performance is directly linked to the original purpose of machine learning AI systems. In this sense, it is the most important external quality. However, other external qualities may need to be prioritized for various reasons, as described below.

Risk avoidance, fairness, and privacy are often not the immediate objectives of machine learning AI systems, and the achievement of these quality characteristics alone will not guarantee the system's objectives. In that sense, these are the secondary qualities. In that sense, these are the secondary qualities. However, if these external qualities are not achieved, it could be detrimental to various stakeholders, including the system's users and subjects of its judgment. For this reason, these quality characteristics are often considered social demands that take precedence over the business requirements of private companies. Thus, these quality characteristics should be achieved in a machine learning AI system, even at the cost of deteriorating AI performance. Deterioration in AI performance in this context means compromising on a level of AI performance that falls short of the highest level achievable technically but still satisfies business requirements.

AI security prevents other external qualities from being compromised by attacks, and can be said to support other external qualities. However, AI security measures designed to protect certain external qualities may compromise others. In addition, there is often a social demand for AI security due to regulations, etc. For this reason, there may be cases where AI security should be realized even if the realization level of other external qualities is reduced.

4.8 Relationships with other social and technical quality requirements

The Guideline set forth that the quality is managed with a focus on five viewpoints (safety, AI performance, fairness, privacy, and AI security) as described above. Generally, various

viewpoints are under discussion concerning the *AI quality*. In this section, we summarize our thoughts on the relationship with the Guideline for some perspectives other than the five we have organized so far.

4.8.1 Explainability of AI

The *explainability* of AI systems is sometimes emphasized as one of the components of trustworthiness. In fact, there are several aspects to considering explainability as a quality. First of all, from the *perspective of explainability of being able to use the product with confidence*, or in other words, explainability of quality, the Guideline considers this as a property required for quality management rather than quality itself. Throughout the Guideline, quality management is viewed as a series of activities that enable quality management activities to be explained and accepted afterwards with conviction, and explainability from this perspective is the very purpose of the Guideline.

Next, from the standpoint of the explainability of the behavior, explainability in this sense can be considered as one of the means to achieve quality explainability, which is the objective of the Guideline. Indeed, if the behavior of a machine learning system is fully explained, it is highly likely that its behavior can be logically analyzed and its quality explained in the same way as an ordinary program. And even if the behavior cannot be perfectly explained, machine learning elements with a simple structure may be expected to be easier to explain the content of the behavior, and at the same time have lower quality concerns that may lead to different behavior than expected. On the other hand, however, when the environment is complex as described in Section 1.3.1, even ordinary software programs that would logically be perfectly explainable often do not necessarily lead to reliability, and the explainability of behavior and the explainability of quality are not always considered to be the same thing.

4.8.1.1 Explainable AI

The idea of *explainable AI* is often referred to as a form of achieving explanatory behavior. Once the technology for explainable AI is well established, where the behavior of AI can be logically re-described and organized at a sufficiently understandable level, it is expected that the achievement of various qualities can be logically explained from the standpoint of quality management. However, at least at present, such technology is still under research and has not reached the stage where it can be adopted stably and universally as an all-purpose means to achieve quality. Of course, if the technology can be applied, it can be a powerful means of achieving quality concerning the Guideline. In particular, it could be a strong means to obtain explanations of internal quality specific to fairness (Section 4.4, Section 10.1) and stability of

machine learning models (Section 7.9, Internal Quality C-2). As mentioned above, it should also be noted, that even if the algorithmic computational processing of machine learning elements can be perfectly explained, it does not always explain their quality in the real world.

4.8.1.2 Transparency

For the same reason, we consider transparency to be one of the means of achievement for quality management. At least in the field of machine learning, transparency of behavior is a rather similar property to explainability of behavior, and may be useful in the analysis of fairness and stability. Also, in applications where the resulting machine learning model is not used as it is, but is rather designed logically based on the parameters inside the obtained model and implemented as a rule base or as an ordinary program not considered as a machine learning model in the Guideline, transparency and explainability as an ordinary program may be considered.

4.8.2 Social aspects such as ethical considerations

As qualities required for machine-learning-based systems, ethical considerations and social validity of results and judgments made by machine learning components may be included. For example, in fields closely related to personal information (e.g., prior scorings for hiring, crime forecast, etc.), it may be required to guarantee that differences in gender or races do not affect judgments legally or socially. Moreover, in fields that are likely to cause human and economic damages, there may be a case where the validity of possible judgments with different risks is called into question (e.g., a judgment made by a self-driving system under a situation where some human damage is inevitable).

The Guideline does not directly examine *what kind of judgment machine* learning components should make to have social validity in those cases, since it should be sorted out by humans in advance as a part of required definitions at the beginning of system development. Then, a machine-learning-based system draws out processing results with a high probability to the extent possible toward correct output sorted out by humans, and this is considered as a request for quality-in-use characteristics.

As such, *fairness and privacy*, which have not been treated as a direct external quality in conventional software engineering in particular, is listed in Sections 4.4, 4.5, 10.1 and 10.2 as one of the external quality axes. As for ethical aspects, we summarize some international efforts in Section 11.2 for reference.

4.8.3 Limits of handling complex external environments

As a general problem of AI, discussions on the limit of complexity of environments where AI can be utilized can draw attention. Machine-learning-based systems are often incorporated into a part of so-called cyber-physical systems in open environments such as streets and public spaces. Therefore, it is impossible for AI to make expected judgments under all environmental conditions if ultimate conditions are included. This issue is inherently common not only to machine learning but also to devices and software that operate in open environments. Conventional specifications for reliability engineering such as IEC 62998 [20] seem to intend to tackle this issue somehow.

The Guideline follows the overall concept of past specifications. The adequacy of risk analysis and system design required for realizing the quality in complex external environments becomes subject to examination of quality management methods with regard only to differences in characteristics from conventional software and points to remember for unique analysis and design originating from those differences.

5. Levels of external quality characteristics

As described in Section 4.1 of the Guideline, the Guideline defines five aspects of external qualities for machine learning components contained in machine-learning-based systems. In this section, we set target levels of quality for each aspect. The procedures for determining target levels in development will be provided in Section 3.4.

5.1 Safety or risk avoidance

For the safety aspects, quality levels are classified into seven *AI safety levels* (AISL 4, AISL 3, AISL 2, AISL 1, AISL 0.2, AISL 0.1, AISL 0). Throughout the Guideline, causes of the safety risks include both physical hazards such as injury of humans and economic hazards as well. The required AISL should be determined using Table 2 and Table 3, depending on the nature of corresponding hazards[7].

If the cell containing the "*" in Table 2 is referred, the functional safety standard IEC 61508-1 [13] or any similar standard for specified application fields is consulted first, safety function is assigned for the components in the system, and the SIL 4–1 assigned to the machine learning component in question or any equivalent assignments in other related standards is translated to AISL 4–1 respectively. Optionally, even in the other cases, stakeholders can apply risk analysis based on IEC 61508-1 or others and assign AISL based on the SIL evaluation result. For those with different levels of corresponding SILs, etc. depending on the frequency of occurrence, and the applicable application, the appropriate level may be set based on risk assessment.

In the meantime, the Guideline does not expect the direct assignment of AISL 4 or 3, which is a high level of reliability, to any machine learning component. In that case, the overall system design should be reconsidered to reduce risks caused by the machine learning components.

Table 2: Estimation of AI safety levels for human-related risks

Expected severity of hazards/possibility of avoidance	Unavoidable	Avoidable via human oversight	Human check or manual operation always needed
Simultaneous deaths of multiple people	*	*	*

[7] Each AISL is described to correspond roughly to safety integrity levels 4∼1 of the standard IEC 61508 of functional safety. Furthermore, to precisely recognize safety requirement strength to be categorized as No SIL applicable, the corresponding AISLs are divided into three classes as 0.2, 0.1 and 0.

Death or injury of one person	*	*	*
Injury causing permanent disability	*	AISL 2	AISL 1
Serious injury	*	AISL 1	AISL 0.2
Minor injury	AISL 1	AISL 0.2	AISL 0.1
Minor injury (where possible victims can avoid the hazard easily)	AISL 0.2	AISL 0.2	AISL 0.1
No damage is expected	AISL 0	AISL 0	AISL 0

Table 3: Estimation of AI safety levels for economic risks

Impact/possibility of avoidance	Unavoidable	Avoidable via human oversight	Human check or manual operation always needed
Damages affecting continuity of business entities	(AISL 4)	(AISL 3)	AISL 2
Serious damage that undermines business operations	(AISL 3)	AISL 2	AISL 1
Considerable/specific damage	AISL 2	AISL 1	AISL 0.2
Minor loss of profit	AISL 1	AISL 0.2	AISL 0.1
No damage is expected	AISL 0.1	AISL 0	AISL 0

(Note)

AISL 0.1, 0.2, 1, 2, 3, 4 may be considered to correspond to Sensor Performance Classes A to F in IEC 62998 [20] respectively. AISL 0.1 to 3 may also be considered to correspond to Performance Levels PLa to PLe in ISO 13849 [1] respectively.

5.2 AI performance

For the AI performance aspects, quality levels of components are assigned by the agreement between stakeholders based on the following criteria.

- AIPL 2 (mandatory requirements):
  - When it is indispensable or strongly expected that the product or service satisfies certain performance indicators (e.g. accuracy, precision or recall) for the system's operation.
  - When the fulfillment of the certain performance indicators is clearly stated as requirement in contracts.
- AIPL 1 (best-effort requirements):
  - When certain performance indicators are identified as a factor for satisfying purpose of the product, but AIPL 2 is not applicable.

    For example, when shorter development period is beneficial, or when gradual improvement of performance to a satisfactory level during in-operation phase is acceptable.
- AIPL 0
  - When performance indicators are not specified at the beginning of development, and discovery of a performance indicator itself is the purpose of development.
  - When the development completes at the PoC stage.

5.3 Fairness

Quality levels of components for fairness aspects are determined based on the following criteria.

- AIFL 2 (mandatory requirements)
  - When a certain level of fair treatment is required according to the laws, regulation or equivalent de-facto social guidelines.
  - When the product deals with personal data and its output directly affects right of individuals.
- AIFL 1 (best-effort requirements)
  - When it is possible to define the requirements that there is no bias into the product or the services.
  - When there is possible demand for the explanation of the fair treatment provided by machine learning component (or AI in general) that might affect social acceptability or operation of the product.
- AIFL 0
  - When there are no identifiable requirements for fairness of the product or service.
  - When the product or service would be affirmatively accepted even if their outputs

are unfair or non-uniform, in consideration with other factors such as performance.

- When requirement for fairness is understood as high performance regardless of input class. In this case, it will be interpreted as a requirement for diversity support of input classes, related to other properties (safety or performance).
  (For example, in an application to detect some mechanical defects, if a detection rate for some specific defect class is relatively high, there is no need to decrease the detection rate to match with those for other unfavored classes).

5.4 Privacy

Quality levels of components for privacy aspects are determined based on the following criteria.

- AIPrL 2
  - When the product or service represents an explicit and potential threat to the protection of data subjects and requires personal data protection processing and organizational measures and application of technical measures to reduce threat of re-identification as well, aimed at compliance with applicable laws and regulations.
- AIPrL 1
  - When the product or service represents an explicit and potential threat to the protection of data subjects and requires application of personal data protection processing and organizational measures aimed at compliance with applicable laws and regulations.
- AIPrL 0
  - When there is no data subject protection requirement for the product or service in question.
  - When the product or service in question represents an explicit and potential threat to the protection of data subjects and requires application of organizational measures aimed at compliance with the applicable laws and regulations that stipulate the protection.

5.5 AI security

We define no levels specific to the external quality *AI security* in this version of this document. The basic approach is to conduct risk assessments for the situations of development and operation according to the strength of the requirements of the above four quality characteristics, and to implement countermeasures when necessary.

6. Internal quality characteristics subject to quality management

In the Guideline, the following 14 characteristics, in five categories, were extracted as characteristic axes of quality management, for the achievement of five external qualities described above[8, 9].

A: Designing quality structures and datasets
- A-0: Sufficiency of preliminary problem analysis
- A-1: Sufficiency of problem domain analysis
- A-2: Sufficiency of data design

B: Dataset quality
- B-1: Coverage of datasets
- B-2: Uniformity of datasets
- B-3: Adequacy of data
- B-4: External-quality-dependent adequacy of datasets

C: Quality of machine learning models
- C-1: Correctness of trained models
- C-2: Stability of trained models
- C-3: External-quality-dependent adequacy of trained models

D: Quality of software implementation
- D-1: Reliability of underlying software systems
- D-2: Additional aspects of program adequacy

E: Operational quality
- E-0: Continuous monitoring and recording during operation
- E-1: Maintainability of quality characteristics during operation

[8] When creating the 2nd edition, one characteristic was split from the 1st edition and the structure of the field was added. The previous internal quality characteristics 1-8 correspond to A-1, A-2, B-1, B-2, (split into B-3 and C-1), C-2, D-1 and E-1. Section 12.1 describes the outline of analyses leading to this extraction.

[9] During the preparation of the 4th edition, the structure of the internal quality characteristics was significantly revised. For A, the prior analysis was added as A-0. For B, C, and D, the required adequacy for each external quality was added as B-4, C-3, and D-2. For E, continuous monitoring was added as E-0.

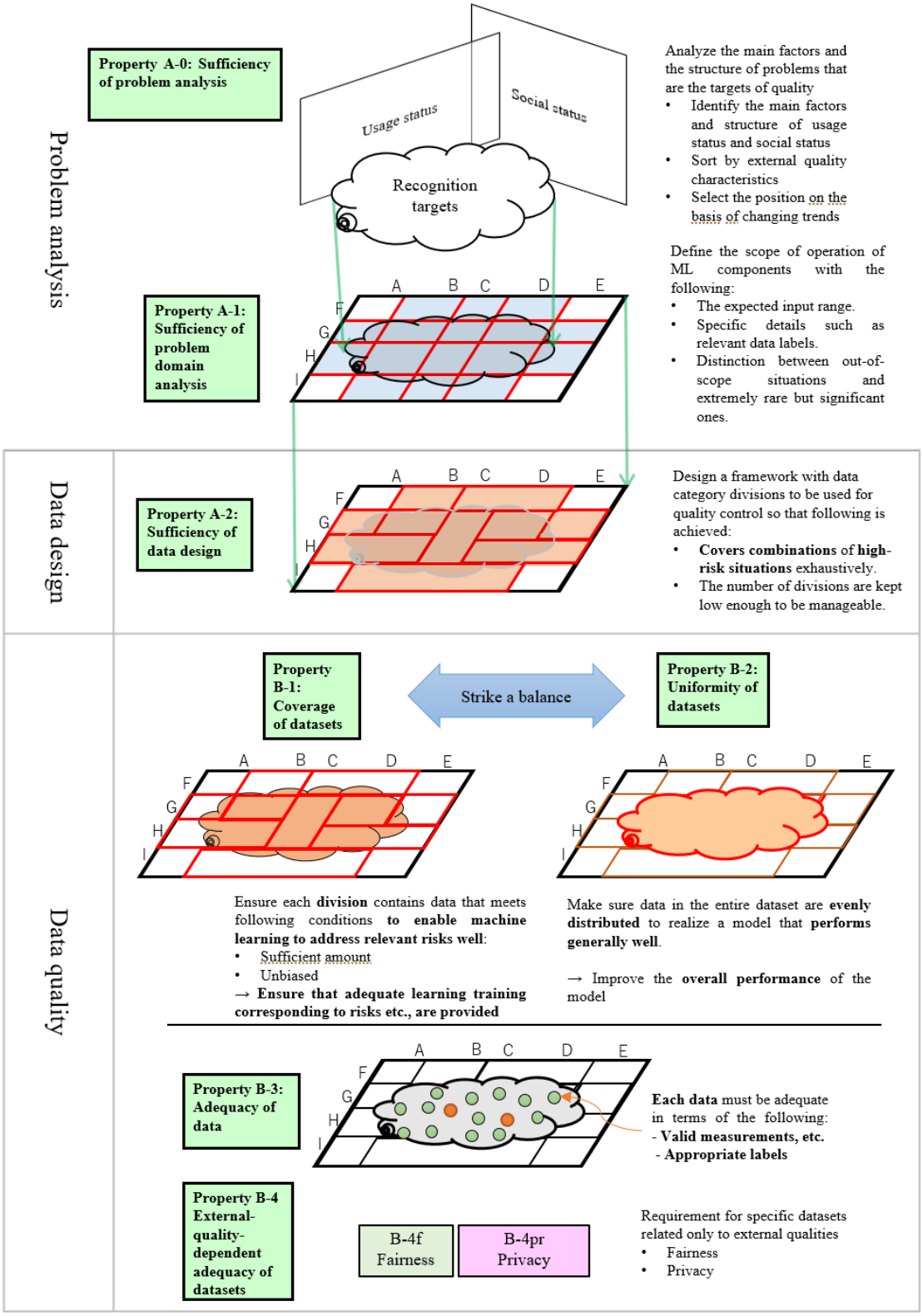


Figure 8: Internal quality characteristics addressed (1)

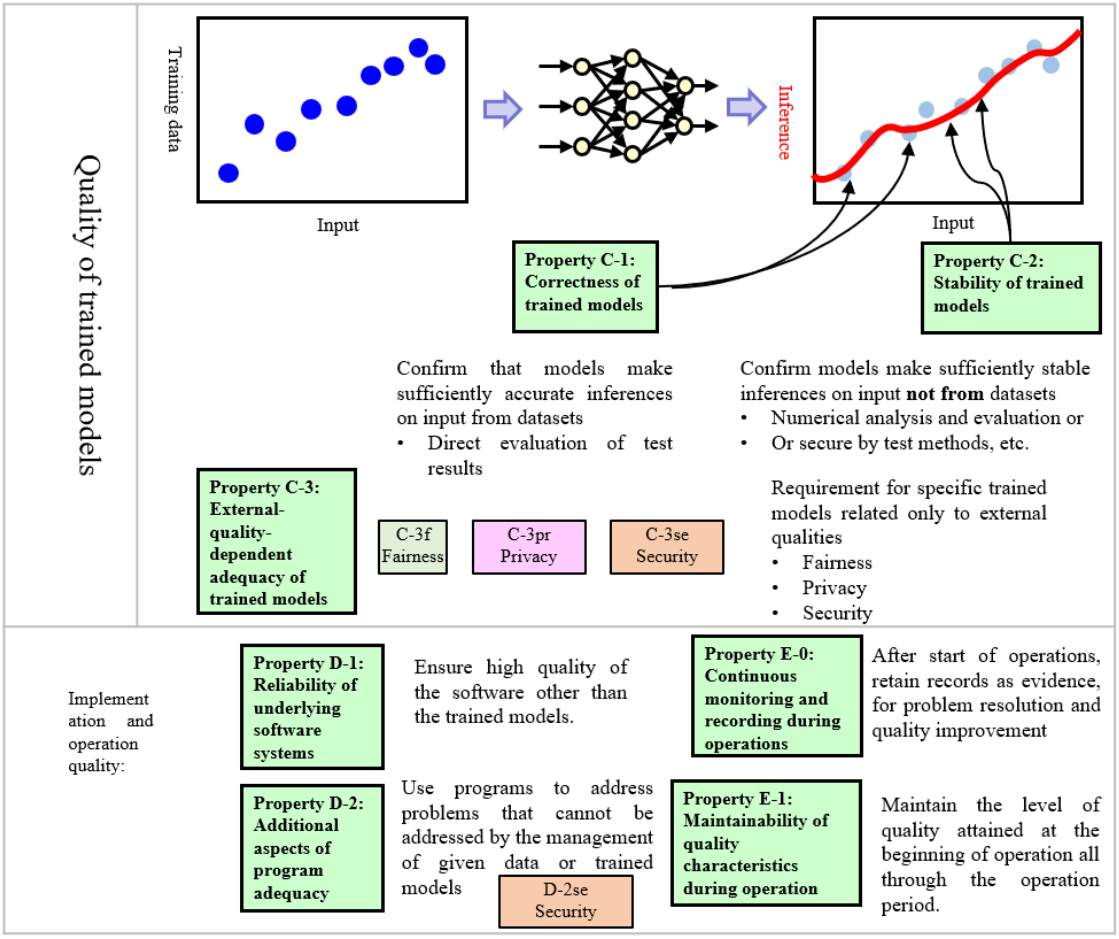


Figure 9: Internal quality characteristics addressed (2)

6.1 A-0: Sufficiency of preliminary problem analysis

To develop a machine learning system that requires a certain level of quality, it is necessary to analyze data directly required for training and learning work, and also pay sufficient attention to the factors and structures of problems subject to quality. For example, in issues related to safety (risk avoidance), external factors such as weather and other usage conditions may significantly impact quality. Alternatively, in issues involving fairness, the risk to fairness cannot be adequately mitigated without an understanding of the social conditions that may cause unfairness and the complex structures behind them.

Specific analysis of the structure, etc., behind such problems, etc., is identified in these Guidelines as the first step in ensuring internal quality and is referred to as "A-0 Sufficiency of preliminary problem analysis". In practice, it is organized by breaking down into five sub-items corresponding to the five external quality characteristics.

In particular, for external qualities derived from social demands, such as fairness and privacy, the correct approach may not be fixed or may change over the years. Therefore, it is necessary

to investigate the latest social conditions and social norms, and then select which approach to take.

The approach is well established in the case of external qualities that have been considered for relatively long time, such as risk avoidance and AI security. However, how much risk to take and how to handle it is something that the entity facing the risk needs to determine each time. Even for AI performance, which is the most technical requirement, we should know the kind of value we want to create and what should be done to achieve that.

6.2 A-1: Sufficiency of problem domain analysis

Next, we define *sufficiency of problem domain analysis* as the fact that the analysis of the nature of the actual data, expected to be input to the machine learning component during operation, corresponds to the real-world usage of the machine learning application system, and that the results of the analysis cover all expected usage situations.

With respect to the operation domains of the target system identified during the preliminary requirement analysis phase, the goal is to define *the axis* for analyzing quality based on the data by writing down the analyzed requirements in specific terms to a level where they can be linked to specific individual data for use in organizing training data and confirming the availability of necessary inspection data at a later stage.

While there are several methods to determine *the axis* for such specific data analysis, the basic approach of this Guideline is based on the traditional feature tree concept in software product line engineering or a simplified version of it. This involves classifying and organizing several independent conditions as a list, as shown in the example below, and understanding specific usage scenarios as a combination of these conditions. Moreover, we conduct both top-down analysis from the request side, such as risk analysis, failure mode analysis and bottom-up analysis consisting of preliminary data analysis in the Proof of Concept (PoC) stage to sufficiently sort out possible situations in which external qualities change or deteriorate.

(Example 1)

We can write down the following situations which a self-driving vehicle running outdoor might encounter in machine learning components that recognize traffic lights based on images. (Please be reminded that this example is not sufficiently comprehensive for actual application)

Significance of display: Green, yellow, red

Time zones: Day, night

Weather conditions: Sunny, cloudy, light rain, heavy rain, snow, fog, etc.

Others:

For example, the situation in which "the traffic light is **red** on a **clear night**" is one *combination of situations*.

(Example 2)

We can write down how a machine learning component that predicts the sales trend of a retail store is used as follows (this example is not comprehensive, either).

Day: Monday, middle day of week, Friday, Saturday, Holidays

Weather conditions: Sunny, cloudy, light rain, heavy rain, snow

Time zones: Morning, before noon, noon, afternoon, evening, night, midnight

Season: Spring, summer, fall, winter

Neighboring events: Yes/No

In this example, one combination could be "**there is a** neighboring event on a **holiday morning** under **winter clear sky**".

The sufficiency of this requirement analysis deals with analysis of risk factors in conventional software and test designs to include those risk factors when a black-box test is conducted. It is also an important characteristic that establishes a unit to grasp and check the quality. On the other hand, when machine learning is implemented, minor characteristics above a certain level may be left to learning processing in the training stage, and a person who implements the system may not give detailed instructions on how to judge individual conditions. We can say that these

are the biggest benefits. Moreover, in some cases, the implementation by machine learning shows better performance than that by humans. From this viewpoint, the configuration of *details to be included in requirement analysis* becomes a very important point in examining an implementation strategy of machine learning components taking the quality into consideration.

Furthermore, when this type of analysis is made, we may recognize a situation which we do not encounter in real life and an extremely rare situation in which it is not practical or it is not required to respond to guarantee operation (e.g. snowfall in summer) or a rare situation that does not appear in training data to be collected but it is required to respond to (e.g. snowfall in spring in the Tokyo area). Specifically, it is very important to distinguish those two situations in any system for which safety is required and the distinction is directly connected to design of safety/robustness of the whole system. At this stage of considering the *sufficiency of problem domain analysis*, it is also important to identify situations that cannot happen in real life and eliminate them from examinations in later stages, in the process of identifying rare situations or cases difficult to be found based only on data collected from real life that require response and designing corresponding systems.

A specific concept of configuring these situations will be explained in Section 7.1 in more detail.

6.3 A-2: Sufficiency of data design

Assuming sufficiency of problem domain analysis, sufficient examinations of data design in response to various situations to which systems must respond, for collecting and organizing sufficient training data and test data, are required as *coverage for distinguished problem cases*.

In an extremely simple system, it is enough to mention that corresponding data to all *combinations of situations* identified in the problem domain analysis described in the previous section included in training datasets and test datasets. However, if a situation in which a system is envisioned to be used is complex, the number of possible combinations becomes enormous. Therefore, it is not practical to cover all combinations with datasets. For example, only in the simple case mentioned in the above Example 2, the total number of combinations is 1400. In a real case, this number is envisioned to reach 10,000. In this case, it is required to check completeness with a rough granularity level that covers several situations to sufficiently deal with combinations of detailed situations in which any danger or reduction in performance is likely to occur. There is a concept of *coverage criteria* applied to test design in the field of software engineering which aims to achieve practical and sufficient operational completeness by selecting appropriate means for each application.

6.4 B-1: Coverage of datasets

Next, we require that enough data, especially, test data, are given to each combination of situations to be supported designed in the previous section without any missing situation, to be called *coverage of datasets*. Although the property mainly concerns datasets for testing and validation for the purpose of quality management, to achieve targeted levels of qualities, it matters also on datasets for training.

When a regular software is developed, the details of all features in real world which software operation depends on is grasped in any point from problem domain analysis to implementation and reflected ultimately as conditional branching or calculation formula in programs. However, when machine learning components are built, more minute situations than a certain degree are not grasped explicitly as *feature quantity or ground truth labels* of training datasets and are only included implicitly in training datasets. They will be reflected in final operations throughout the training stage of machine learning. The purpose of configurating this axis of characteristic is to guarantee that the shortage of learning due to the shortage of data or any oversight of learning in specific conditions due to biased data does not occur in any situation or case identified in requirement analysis or data design.

(Example 1)

In the case of recognition of images of traffic lights mentioned earlier, this characteristic means that all forms of traffic lights in each prefecture, their height and distance between them, and road conditions around them are included as data without any bias and that training is not carried out based only on limited data of a specific city.

(Example 2)

When image recognition AI that aims to recognize *cats* from small animals around town is to be built and individual characteristics of cats such as breed and size are excluded from recognition, this characteristic means that sufficiently diverse images of *cats and other small animals such as dogs* are made available as data and that training does not only target specific breeds (e.g. calico cats) in environments where the product is expected to be used.

6.5 B-2: Uniformity of datasets

A concept contrary to *coverage* mentioned earlier is *uniformity of datasets* in relation to the overall assumed input data. When each situation or case in datasets is extracted in accordance with the frequency of its occurrence in whole data to be input, data are considered as uniform (Figure 10). Balance between this uniformity and coverage sometimes needs proper attention and evaluation. The prediction accuracy of machine learning technologies is generally supposed to improve by using samples extracted uniformly in relation to input environments as training datasets. However, the coverage of situations explained in the previous property may be emphasized depending on actual application and quality characteristics required therefor. It is necessary to consider priority or compromise between coverage and overall uniformity carefully.

When safety is strongly requested, sufficient training data are required for high-risk situations that must be avoided by making correct judgments. If such a rare case occurs and you intend to train data by ensuring enough data in relation to that rare case and maintaining the uniformity to all other situations, the necessary amount of data might be enormous. In this case, priority may be given to training of rare, high-risk situations.

On the other hand, when overall performance, AI performance is requested, certainty of inference results in other cases might deteriorate by giving more priority to the training of rare cases than the actual probability of occurrence, thereby resulting in the deteriorated average performance as a whole. In this case, standards for coverage for each situation mentioned in the previous section are not appropriate.

Moreover, when fairness is strongly requested, *what kind of fairness* is requested may change our decision. For example, you may choose to give the cases artificially-equal training by artificially processing data (selection or addition of data) or let the cases learn at random in line with the distribution of extracted training data.

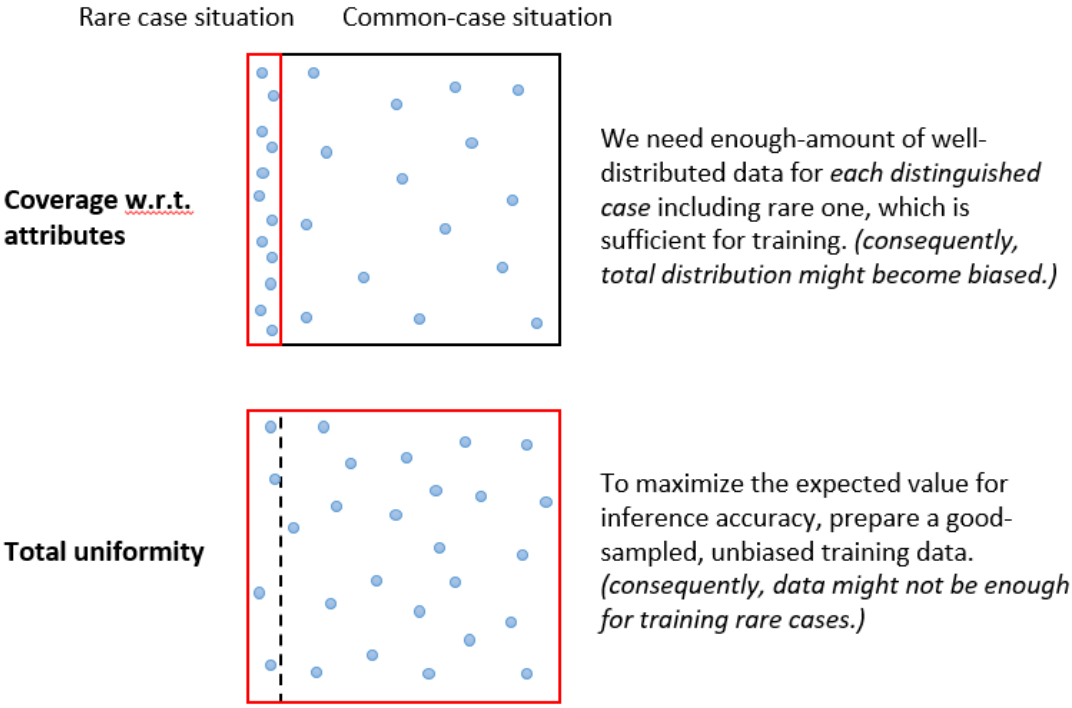


Figure 10: Relationship between coverage and uniformity

Since the two viewpoints mentioned in Section 6.4 and this Section may be compatible or incompatible, you may need to adjust appropriate training data to strike a balance at an adequate level. Moreover, different characteristics may be sought in the training stage and the quality test stage.

(Example 1)

For example, if snowfall may affect the performance of signal image recognition in a self-driving vehicle, the necessary training or verification data must be prepared to suppress the effects of false recognition even in snowfall conditions that are expected to occur only one or two days a year in Tokyo, which is set as the operational area. For this reason, the proportion of snow images in the dataset may be increased compared to the actual probability of snow events.

In this case, it may be necessary to prioritize the coverage of the dataset over the *uniformity of the dataset*.

(Example 2)

On the other hand, if one or two snowy days a year are strongly trained in sales predictions of a retail store in Tokyo, the prediction performance in other climate conditions may deteriorate and the average profit cannot be maximized. In this case, we might need to give priority to *uniformity of datasets*. Of course, what kind of data is actually prepared as training datasets will finally be examined in the implementation process taking the balance of both internal quality characteristics into consideration.

6.6 B-3: Adequacy of data

In contrast to the properties of the distribution of the dataset in B-1 and B-2, the *adequacy of data* refers to the fact that each item of data in the dataset is appropriate for the purpose of training. Adequacy includes not only the absence of errors in the values, but also the absence of data that should not be used for training even if the values themselves are accurate as consistency, the absence of inappropriate modifications to the data as authenticity, and the fact that the data are sufficiently new as currentness. In the context of supervised machine learning, two perspectives are also included: *adequacy of data selection*, which is the adequacy of the measurements to be trained where values that correspond to the input side when the machine learning component is viewed as a function, and *adequacy of labeling*, which is the adequacy of the correct answers added for training where values that correspond to the output side.

The adequacy of the data is primarily determined according to the requirements definition for the implementation of the machine learning components. In general, the quality needs to be reevaluated when the requirements definition is updated, while there are some aspects that can be judged as general-purpose data (e.g. removal of obviously invalid data, authenticity and traceability of the dataset as a whole). Even in the development process of a single system, when requirements definitions and labeling policies are updated due to trials in the PoC phase or regressions in the production phase, the data need to be rearranged to accommodate the new requirements and policies, so it is important to carefully consider the man-hours and procedures in the development process.

Although the Guideline aims to evaluate validity through inspection of data as much as possible, there are some aspects that are particularly difficult to check from the data itself, such as credibility and traceability, and some aspects, such as uniformity of labeling policies, that should be achieved through process management.

6.7 B-4: External-quality-dependent adequacy of datasets

Some of the requirements for datasets relate only to specific external qualities. Here, they are collectively referred to as *external-quality-dependent adequacy of datasets*. The specific details for each external quality are described in Section 7.7.

6.8 C-1: Correctness of trained models

The term *correctness of trained models* represents that a machine learning component functions as intended upon the input from the training dataset (whose quality is ensured to a certain extent in B-3) consisting of training data, validation data, and test data.

Normally, the training dataset does not provide sufficient information to reflect all input given from the in-operation environments. Therefore, a trained model might show high performance with respect only to training data, but not to the data other than the training data. In other words, a trained model may overfit to the training data and may not generalize well to unseen data. When a machine learning component is evaluated using only the validation datasets and test datasets, it is not possible to check if the machine learning component functions as intended when given data input from the general environment. Hence it is important to evaluate not only the correctness but also the stability, which will be explained in the next section.

6.9 C-2: Stability of trained models

The term *stability of trained models* represents that a machine learning component functions as intended upon an input not included in the training dataset. When a trained model does not satisfy stability, it may not function correctly upon the input data not included in the training, validation, or test datasets. For example, a trained model may function incorrectly when the input is perturbed by natural or adversarial noise.

6.10 C-3: External-quality-dependent adequacy of trained models

Some of the requirements for machine learning model relate only to specific external qualities. Here, they are collectively referred to as *external-quality-dependent adequacy of trained models*. The specific details for each external quality are described in Section 7.10.

6.11 D-1: Reliability of underlying software systems

The *property reliability of underlying software systems* represents that the underlying conventional software (e.g., training programs and prediction or inference programs) functions correctly and reliably. This concept includes various software quality requirements such as correctness of algorithms, memory resource or time constraints, and software security.

6.12 D-2: Additional aspects of program adequacy

Even if a program meets the quality requirements as general software, it may not be possible to realize and maintain the reliability of the machine learning components. The ability of a program to successfully address problems that cannot be addressed by the management of given data or trained models is considered here as *other reliability*. The specific details are described in Section 7.12.

6.13 E-0: Continuous monitoring and recording during operation

A basic requirement during operations of a machine learning AI system is the need to continuously obtain information about the operational status and to retain the records of the information. For a system to gain the trust of stakeholders, it is necessary to present records as evidence and explain the system's status after it becomes operational, in addition to the quality control process during development. Furthermore, records are necessary for review and analysis to improve quality and restore quality when some problems occur.

6.14 E-1: Maintainability of quality characteristics during operation

The term *maintainability of quality characteristics during operation* means that internal quality characteristics fulfilled at the time when the operation started is maintained throughout the operation period. The maintainability implies the following three aspects; internal qualities sufficiently adapt to changes in external environments, they do so also to changes in the system feeding input to the AI, and they do not deteriorate due to changes made in trained machine learning models to make such adaptations.

A specific method to realize quality maintenance depends on forms of operation, especially on how to carry out additional learning or iterative training. This point will be described in Section 8.2.1.

7. Requirements for quality assurance

In this chapter, the following internal qualities are set as the quality management characteristics to specifically manage the quality of machine learning components.

7.1 A-0: Sufficiency of preliminary problem analysis

7.1.1 A-0s: Safety or risk avoidance

Analysis of risks and hazards has always been performed in the domains of safety or risk avoidance. Considering the technical maturity and social impact of conventional methods, it is recommended that risk factors be identified using conventional methods for functional safety, such as those recommended in IEC 61508-1 or ISO 26262, in domains relevant to risk avoidance. We also advise expanding and applying conventional thinking in mechanical engineering and other fields to economic risks.

7.1.2 A-0p: AI performance

The external quality of AI performance is generally considered to correspond directly to analytical works performed in system requirements analysis and the KPIs (Key Performance Indicators) obtained from it.

7.1.3 A-0f: Preliminary analysis of fairness requirements

If the required fairness level is assumed to be AIFL 1 or higher, the following points should be considered during the requirements definition phase. In cases where multiple sensitive attributes are relevant to ensuring fairness in the target system, the level should, in principle, be determined based on the most stringent one.

7.1.3.1 Clarification of fairness goals

First, the goal of ensuring fairness for the target system should be clarified as follows.

(1) Analyze relevant sensitive attributes for fairness considerations

a. Clarify the information, such as sensitive attributes, that needs to be considered for fairness from the target function's point of view. This information is not limited to that

which is explicitly identified as an attribute of the actual input data. For example, gender, age, race, and ancestry could be identified as sensitive attributes even though not included in the data.

b. If possible, clarify the observable information (e.g., written test scores, department of choice, etc.) that can be used for judgment, with sufficient consideration of the influence of the sensitive attributes clarified above.
c. Furthermore, if possible, articulate the essential characteristics—such as potential academic ability or likelihood of graduation—that underlie the above observable attributes. These characteristics serve as key factors in the AI's judgment, but may not be directly observable.

Here, it should be noted that the scope of fairness to be considered by the target system (or the broader social activity in which it is embedded), as well as any assumptions about indicators that are deemed "fair" or "unfair" as a precondition, may influence the above considerations and subsequent actions.

(Example)

For example, consider a college admission examination system, and the scoring and pass/fail judgment process of the system is our target of discussion.

From the standpoint that having all examinees solve the same prepared test questions at the same time provides a fair opportunity, the validity of their answers to the questions becomes the essential abstract criterion for the final judgment of pass or fail, and it is important to ensure that the scoring results serve as an appropriate observable numerical index corresponding to that criterion..

On the other hand, as in the case of college entrance exams discussed in the fairness related study [95], the examination system itself may have fairness problems. From their point of view, potential academic ability and the possibility of graduation may be essential ideal criteria for the judgment. In addition, the score result would be affected by the disadvantages derived from the poor past educational opportunities, which might be strongly correlated with racial and income data. They may argue that some corrections considering those aspects might be necessary for a fair pass/fail decision close to the ideal criteria.

(2) Clarify the basic handling policy for sensitive attributes.

a. Does the sensitive data require that the system not be treated in a discriminatory manner because of the information (equal treatment), or does it require that the

results follow a predefined distribution with specific numerical targets (equal outcome)?

Are there any legal requirements regarding handling the information in question, such as those in equal employment opportunity, equal race, etc.?

b. If discrimination related to sensitive data already exists in the real world, how should the AI, built based on the data collected from the real world, be handled? Should such discrimination be removed as much as possible, or can a certain amount of residual discrimination be tolerated, or, depending on the system's requirements, should it be reproduced to reflect real-world conditions?
c. Clarify the constraints on the available measures to remove residual discriminatory judgments in machine learning results. Is it permissible to introduce intentional corrections to the training data or to intentionally adjust the trained model? It should also be clarified how the decision to stop development will be made if the available means cannot correct the problem.

After making the above decisions, the fairness requirement level (AIFL) is checked again.

7.1.3.2 Modeling dependencies and causal relationships among data attributes

Since data acquired from the real world have complex structures and correlations, machine learning without properly understanding the dependencies and causal relationships between attributes may lead to inaccurate results.

As we plan to implement the system using machine learning with data, rather than logical specification through program description, the dependencies and causal relationships among attributes are not completely known (if they were known, the non-AI program could have been written), and it is difficult to grasp them completely even after exhaustive analysis. However, it is very important to investigate the following aspects as early as possible.

(1) Information paths that may cause unfairness

In the process by which a sensitive attribute unfairly affects the output of a system, the sensitive attribute may directly affect the result, or there may also be contributions from other attributes that are affected by the sensitive attribute or even further indirect influence through those attributes. When these series of causal paths are not understood, it is often the case that the final fairness requirement cannot be met nor explained, even removing the direct impact.

In addition, when there are attributes that are correlated with both those that should contribute to system decisions and sensitive attributes that should not, typical pre-processing—such as removing bias related to sensitive attributes from the dataset—may inadvertently degrade the system's decision-making performance. Therefore, such steps require careful consideration.

(2) Simpson’s Paradox.

Simpson's paradox refers to the phenomenon that when analyzing to find a relationship between two data attributes, ignoring the third factor involved in both of them, a correlation that does not exist in reality or a correlation that is the opposite of reality may be shown [184]. This paradox was proposed over 50 years ago in statistics, but it applies to machine learning today. Preventing such a phenomenon requires understanding the relevant third attribute and preparing learning data with case separation for the third attribute. The specific examples are described in Section 10.1.4.1.

(3) Necessity of using sensitive attributes

There can be cases where fairness cannot be achieved without daring to use sensitive attributes. In other words, fairness through unawareness may sometimes be insufficient for the objective.

See Section 10.1.4.1 for specific examples of methods that are beneficial for analysis such as mentioned above.

7.1.4 A-0pr: Preliminary analysis of privacy requirements

7.1.4.1 Overview of implementation items

Privacy quality management should be based on the privacy-by-design principle. The activities are described below in two phases: domain analysis and system design (Table 4).

Table 4: Overview of privacy quality management activities

<table>
<tr><td rowspan="2">Domain analysis</td><td colspan="2">Protected data</td><td>Use of artifacts</td></tr>
<tr><td colspan="2">• Compliance with applicable law
• Identification of personal data</td><td>• Plan of artifacts reuse
• Confirmation of users' consent</td></tr>
<tr><td rowspan="4">System design</td><td>Pre-stage</td><td>In-stage</td><td>Post-stage</td></tr>
<tr><td>• Training data quality
• Protection processing
• Data distribution (outliers)</td><td>• Generalization performance
• PPML (Differential privacy)</td><td>• Safeguards (Processing of output values)</td></tr>
<tr><td colspan="3">Trade-off analysis</td></tr>
<tr><td colspan="3">• Data protection measures vs. Usefulness
• Protection strength vs. Fairness</td></tr>
</table>

7.1.4.2 Domain analysis phase

For the machine learning components under development, it is necessary to conduct requirements analysis from the perspective of privacy protection and personal data protection. This analysis includes examining whether the component may create privacy risks that are socially unacceptable. The domain analysis phase focuses on two main aspects: the analysis of the quality of training data and handling of artifacts.

7.1.4.2.1 Protected data

If the target machine learning system is subject to privacy-related quality management, the types and scope of personal data contained in the training data should be identified. In this process, compliance with applicable laws and regulations is essential. Specifically, it should be confirmed whether the system handles categories of personal data that are legally required to be protected, and the items to be protected should be extracted from the data provided by users or data subjects for service use.

In addition, when regulations governing activity data are applicable, compliance with the relevant regulations should be verified. Based on this, the functions of the service under development, its usage scenarios, and the data acquisition flow should be analyzed to identify the personal data generated or collected during service use. This enables identification of service-specific activity data in addition to the data collected at the time of registration.

7.1.4.2.2 Handling of artifacts

It should be clarified whether the development artifact is a machine learning system to be operated, a training dataset that may be provided to others, or a trained machine learning model.

First, consider the case of training datasets. In machine learning software, adaptive maintenance is often performed through retraining. The dataset used for the initial training should be maintained over the expected adaptive maintenance period. Retraining is performed within the scope permitted by the terms of consent given by the data subject.

When a training dataset is to be provided to third parties through distribution or publication, it is necessary to confirm that such provision is consistent with the terms of consent given by the data subjects. The recipient should handle the dataset in accordance with those terms.

Next, consider the provision of trained models, particularly pre-trained models. Transfer learning and knowledge distillation are machine learning methods to reuse pre-trained models. The resulting models are used as components in new machine learning systems. Transfer learning may be employed to conduct adaptive maintenance, including functionality enhancement. Such enhancements should not conflict with the purposes and retention periods specified in the data subjects' consent. For example, consider a case in which we have a training

dataset consisting of facial image data provided with the data subject's consent to develop age prediction components. Then, conducting transfer learning based on the pre-trained model to build a new machine learning component for inferring race would be likely constitute use for a purpose outside the original consent. When knowledge distillation is used, the rights of the data subjects should be respected in the same manner as in transfer learning.

In addition, when pre-trained models, such as foundation models, are distributed or made public, information derived from the training dataset may be indirectly made available to third parties. It is necessary to confirm that such distribution or publication is consistent with the terms of consent given by the data subject.

7.1.4.3 System design phase

In the system design phase, technical measures for privacy protection are examined. Table 4 organizes these measures into three steps. The three steps are a Pre-stage regarding the preparation of the training dataset, an In-stage to devise the machine learning algorithm or the learning mechanism adopted in the training process, and a Post-stage to process the output of the trained model when input data are entered. Details of these measures are summarized as privacy-related quality assurance perspectives in Chapter 7. These various measures will be combined appropriately by considering multifaceted trade-off relationships. After the development, it is recommended to conduct the data protection impact assessment preferably with the assistance of tools.

7.1.5 A-0se: Preliminary analysis of security requirements

7.1.5.1 Process of security analyses and measures

As with the security analysis and response required for general system product development (general information security), the management of the external quality *AI security* follows the analysis and response framework provided in the ISO/IEC 27000 series. Specifically, we list assets[10] and stakeholders throughout the system lifecycle, enumerate attack scenarios for the assets, identify threats and vulnerabilities, assign each threat and vulnerability a response priority, and implement control measures[11] for each threat and vulnerability, starting with

[10] General information security defines assets as information assets. In contrast, AI security targets attacks that intentionally exploit the characteristics of trained models. We define assets in Section 10.3.3.2.

[11] Technical requirements should also be taken into account. After system operations begin, analyses and

those with the highest priority (risk-based approach).

Managing AI security involves analyzing and responding to risks arising through trained models. We target attack scenarios where a risk arises through trained models. Then we identify threats and vulnerabilities, and design control measures. Security management of machine-learning-based systems also requires general information security analysis and response (Section 10.3.5.3).

The analysis of security based on ISO/IEC 27005 [11] (security risk assessment) is conducted according to the following steps[12]

a) BIA (Business Impact Analysis): Stage to identify business risks (see SP800-34 [52] and FIPS 199 [48])
b) Requirement definition: Stage to incorporate BIA results into technical requirements (see ISO/IEC/IEEE 29148 [13])
c) Security risk assessment 1: Stage to determine external specifications to be provided to the design and development department
d) Security risk assessment 2: Stage to review the artifacts from the design and development department

This Guideline assumes that a) and b) have been completed and describes c) and d), which are implemented in the AI security risk assessment conducted after that.

7.1.5.2 AI security risk assessment

(1) Security Risk Assessment 1 (Stage to determine external specifications to be provided to the design and development department)

Based on the requirement definition, create assessment items that conform with the design and development targets based on the asset, stakeholder, threats, vulnerabilities, and control measure models described in Section 10.3. Implement a security risk assessment once the external specifications of the system have been created based on the requirements. Provide the design and development department with items that require responses in the design stage and/or in the operational stage.

(2) Security Risk Assessment 2 (Stage to review the products provided by the design and development department)

Implement a security risk assessment once the design verification results have been created and the developed product is validated based on external specifications. Thereafter, confirm that

responses measures should be carried out regularly. Therefore, the scale of implementation should be appropriately adjusted, with the response priority values assigned to threats and vulnerabilities used for the adjustment.

[12] Steps a) to d) are applicable to both AI security and general information security.

the implementation of control measures in the design stage have been completed, and that items required to be implemented in the operational stage have been prepared.

(3) Additional security risk assessment

Implement a security risk assessment when the design content is reviewed.

Considering reducing rework in design and development, it is recommended that control measures against threats and vulnerabilities be designed and developed in parallel with the design and development of internal quality relevant to the control measures. Therefore, an AI security risk assessment should be completed before the internal quality design work is completed.

Here, we use the security risk assessment framework based on ISO/IEC 27005 [11]. Figure 11 provides an overview of each task in security risk assessment and the dependencies between the tasks.

Figure 11: Flow of security risk assessment

See Chapters 7 and 9 for the correspondence between internal quality characteristics and control measures, and Chapter 10.3 for general information on AI security.

[Precautions]

- In identifying assets, select assets deeply related to machine learning in the target system, concerning Section 10.3.3.2.
- In identifying attack scenarios, select scenarios appropriate for the target system and how it is used. Technically possible attackers and attack interfaces are described in Sections 10.3.3.3 and 10.3.3.4, from which attack scenarios can be selected. However, other attack scenarios should be added depending on the target system and how it is used.
- In identifying threats and vulnerabilities, select threats and vulnerabilities specific to machine-learning-based on the attack scenarios. Based on the literature [135], Section 10.3

provides a comprehensive list of known machine learning-specific threats and vulnerabilities for systems in general that use learning models obtained through supervised learning and systematically describes vulnerability control measures per asset. However, developers should not only consider the threats, vulnerabilities, and control measures presented in this Guideline but also investigate the security threats and vulnerabilities specific to the system being designed and developed and consider whether to add or reduce control measures.

- Some controls to protect against attacks that cause degradation of an external quality or a quality-in-use may degrade another external quality or quality-in-use. For example, a control for strengthening the robustness against adversarial examples may increase the risks of privacy attacks (Section 10.3.4.5.2). For another example, a control against privacy attacks may be incompatible with fairness (Section 4.5.2). Therefore, we may need to analyze the trade-offs between different external and internal qualities.
- At present, the threats, vulnerabilities, and controls for machine-learning-based systems are being studied actively. Therefore, system developers and operators should study the latest information on the security of machine learning technology, considering the specific system they work with, its development and operational environments. Then, they should take a risk-based approach to identify new types of threats and vulnerabilities in the system.
- To deter or mitigate attacks, developers should also consider the risks associated with disclosing the following information right from the design stage of the entire machine-learning-based system.
  (1) Specification information on the system and models (including hyperparameters and architecture)
  (2) Models used in the system
  (3) Datasets used to train the models
  (4) Information related to these models
  (5) Information on system function during operation

  Keeping the above specification information and other related information confidential may be useful in limiting the attacker's prior knowledge and deterring attacks by preventing information collection during the initial reconnaissance stage of attacks (Section 10.3.3.4). However, this does not guarantee the prevention of attacks, and doing so may cause transparency and accountability to deteriorate. In actual development, models are often trained on publicly available datasets, and thus, keeping the dataset information confidential may not be possible. In such cases, the developers should note that the risk of attack is higher and implement other control measures.
- Responses to trends in international standards and regulations are described in Section 10.3.6.1.

7.2 A-1: Sufficiency of problem domain analysis

7.2.1 General

Sufficiency of problem domain analysis means that usage of a target machine-learning-based system is sufficiently analyzed and that all requirements are properly captured as described in Section 6.2.

Problem domain analysis is particularly important in the early stages of development of conventional software used for a safety application. The main purposes of requirements analysis for developing machine-learning-based systems in the Guideline are as follows.

(1) Sufficient identification of cases in which risk management is needed, mainly in applications requiring safety.
(2) Sufficient identification of attributes of objects for which inequality is not allowed, mainly in applications requiring fairness.
(3) Sufficient analysis of real-world conditions in order to verify that training datasets and test datasets are comprehensive and appropriate extractions of the real world, to all requirements including AI performance.

Sufficiency of problem domain analysis is always required not only for machine-learning-based systems but also equipment and services controlled by software. However, if any situation where machine-learning-based systems are used is overlooked at this stage, there are few opportunities for taking note of an error from the data collection stage to the actual training and test stages, so that it may cause malfunction that appear for the first time during final system testing in real environment or during actual operation.

On the other hand, if the details of all possible situations where machine-learning-based systems are used are analyzed down to minute differences, the analysis results can be implemented as regular programs, and there is no merit in using a machine learning technology in the first place. This in turn means that there is a demand for systems to acquire knowledge through machine learning in any way because the systems are not built to perform such comprehensive and thorough analysis in advance.

From these two points of view, it is extremely important to appropriately set the level of detail for requirements analysis in machine-learning-based systems from the aspects of both quality assurance and feasibility and efficiency of implementation. This is believed to correspond to the judgment of what should be analyzed by humans and what should be acquired by machine learning. Maintaining an appropriate balance between the two is an important goal of requirements analysis in machine learning quality control.

7.2.2 Approaches

The two goals of this internal quality aspects described in this paragraph are

- Clarifying what is required for machine learning components; and
- Clarifying the limited scope which machine learning components have to deal with.

7.2.2.1 Extracting and listing attributes (characteristic viewpoints) and attribute values (specific features)

First, inputs in real world which can be extracted as specific characteristics are organized from the aspects listed below. Each identified viewpoint will be referred to as attribute, while the specific features belonging to the viewpoint as an attribute value.

The following examples can be given as attributes and their viewpoints.

1. Attributes that characterize the differences in output required for machine learning components as functional requirements.
   In supervised machine learning, supervision labels are different.
   For example, in the recognition of numeric characters, 10 distinctions from 0 to 9. In anomaly detection, whether there is any anomality is detected.
2. Attributes whose output is the same but require machine learning components to explicitly respond at a functional specification level.
   One example is a case where, in character recognition, the specifications are set that each of l and ℓ and l and / should be recognized correctly.
   Another example is that, when the specification says that various kinds of obstructions such as pedestrians, bicycles and automobiles with crossing movements are to be avoided in automated driving, each kind of object should be identified as a separate attribute value.
   Or, gender and age groups in the scoring for recruitment that requires fairness apply to this paragraph.
3. Attributes for which deterioration risks of performance expected for machine learning components are likely to be different significantly in real operation
   Some examples include climate, distinction of day and night, backlight and front light and movement speed in the case of outdoor object recognition.
4. Elements whose expected output is the same due to the characteristics of machine learning, but it is difficult to find a common ground by the model after learning or they can be recognized as different internally.
   For example, in the case of outdoor object recognition, different characteristics of objects may be captured in the daytime and nighttime, or another example is the distinction of the horizontal and vertical installation of traffic lights.

Another example is that different forms of handwritten numbers such as 1 and / are categorized into several characteristics which differ greatly depending on countries of origin. In order to recognize all different types, it is necessary to intentionally include them at least in training datasets and test datasets as different general populations, even if they are not clarified in the functional specifications.

5. Characteristics which make it difficult to specify the method of evenly collecting learning data as a process when data are collected.
6. Differences which humans can easily capture, though there are no differences as described above, e.g. kinds of animals, skin color, etc.
7. Object which humans cannot recognize or explain their characteristics with word, but it is possible to extract enough samples without bias. For example, it is difficult to analyze all possible ways of describing the number 8 in advance. However, if people do not intentionally write various styles of the number 8 and we have a sufficient number of randomly extracted samples, they are expected to be unbiased samples.
8. Cases where it is extremely easy to mechanically cover and complement differences. For example, once required specifications are established, it is possible to mechanically generate samples of parallel shift of images, color changes and expansion of rotations within a certain range, and intentionally synthesize training data and test data.

We make up a list for candidates of possible attributes from requirements analyzed on the overall system, and then we should determine whether each attribute is picked up for explicit target of management by the developers or is left to machine learning without picked up. To determine that, the level of risks related to values of each attribute and the type of the application using machine learning should be considered. Results of PoC trials may be useful as a reference. In general, the characteristics shown earlier in the above list 1 to 3 are highly likely to be extracted as attributes. In contrast, it is often reasonable to leave the characteristics shown later 7 to 8 for machine learning without extracting them as attributes. However, the diversity of data types corresponding to attributes left to machine learning will be examined in the third internal quality coverage of datasets, so that it is necessary to take this point into consideration in the analysis, too.

7.2.2.2 Combinations of attributes that should be excluded

Impossible combinations of attribute values should also be examined. The purpose of examination is to distinguish cases which should be excluded as functional requirements such as snow in summer in Tokyo from cases which should be handled as functional requirements such as snow coverage in winter which occurs very rarely. It is important to make this distinction, because it will become impossible to determine whether lack of data is acceptable and to explain the quality of quality management method itself in subsequent quality management.

To be more specific, after attributes and attribute values corresponding thereto are listed, impossible combinations of attribute values are selected based on system requirements.

7.2.3 Requirements for quality levels

For sufficiency of problem domain analysis, it is required to handle the following three levels in accordance with the level of external quality.

The correspondence between external and internal qualities regarding the required level is as follows.

Safety

AISL 0.1: Lv 1 or above
AISL 0.2: Lv 2 or above
AISL 1: Lv 3
AISL 2~4: Requirements to be added to Lv 3 will be considered.

AI performance

AIPL 1: Lv 1 or above
AIPL 2: Lv 2 or above

Fairness

AIFL 1: Lv 1 or above
AIFL 2: Lv 2

The requirements for each level of internal qualities are as follows.

- Lv1
  - Examine and record the major cause of possible deterioration of quality.
  - Based on the examination results, design data and reflect it in necessary attributes.
- Lv2
  - Analyze risks of deterioration of quality in use in overall system and their impact with a certain level of engineering coverage and record the results in documents.
  - Analyze if any measure is required for each of those risks, and analyze attributes related to the risk which are contained in an input to machine learning components.
  - Analyze and record the application-specific characteristics of environments which will generate machine learning input, with regards to the difficulty for machine learning and other aspects.
  - Examine sets of attributes and attribute values, based on the results of those analyses and record the background of such decisions.
- Lv3
  - The following activities are carried out in addition to those listed in Lv 2.

- Investigate documents on own past examination results and those of others with regard to elements to be extracted as characteristics of system environment and record the background of examinations leading to the extraction of necessary subsets.
- Investigate past examination results in line with application fields of systems with regard to deterioration risks of quality-in-use characteristics of overall systems and record the examination results including the background of selection.
- Moreover, extract deterioration risks of quality-in-use characteristics of overall systems using engineering analysis such as Fault Tree Analysis and record their results.

7.2.4 **Sufficiency of fairness requirements analysis**

By following the guide presented in the preliminary preparation (Section 7.1.3), the following aspects can be investigated.

- Dependency analysis between attributes
- Clarification of requirements for training dataset distribution
- Metrics to be checked (fairness metrics)

In addition, it is necessary to thoroughly review the following aspects in light of the overall characteristics of the system under development.

- Consistency with legal and social requirements

Since the development team may not have sufficient knowledge of legal and social requirements, it is advisable to consult experts in the field.

The required actions for each target AIFL level are as follows in the preliminary preparation phase.

- AIFL 1
  - Define the fairness requirements and record them, including the history.
  - Based on this requirement, requirements for data preparation such as fairness metrics for training datasets should be defined.
  - Based on this requirement, fairness metrics for model outputs should be defined.
- AIFL 2
  - In addition to those listed in AIFL 1, the following actions should be taken
  - To model dependencies and causal relationships between data attributes.
  - The results of modeling should be reflected in the fairness metrics of the training dataset.

The *fairness metrics* for the training dataset are basically from the same perspective as those for the model output. For example, if the fairness requirement is *no difference in pass or fail judgments based on gender* for the output, the training dataset should also be free of bias from

that perspective. However, even if the training dataset is biased—reflecting existing societal imbalances—developers may, depending on the scope of their feasible interventions, be unable to pursue fairness metrics related to the dataset itself, and instead aim only to meet fairness metrics at the level of model outputs. Examples of metrics are described in Section 7.10.1.3.

7.3 A-2: Sufficiency of data design

7.3.1 General

Assuming sufficiency of problem domain analysis described in the previous paragraph, sufficient examinations of data design in response to various situations to which systems must respond, for securing sufficient training data and test data, are required as *coverage for distinguished problem cases*. More specifically, the number and details of combinations of attribute values focused in the stage from training data preparation to testing process is considered at this stage.

In an ideal situation, for example, if there are sufficient data corresponding to all combinations of attribute values (direct products of attributes) for combinations of attributes that are supposed sufficient as described in the previous paragraph, it can be said that it covers all possible situations in the real world. However, the number of attributes may reach 10 or more in actual system development so that the number of combinations of attribute values may often reach thousands or millions. In this case, it is crucial to consider appropriate coverage and design for quality management in this paragraph.

It is important to focus on two viewpoints in actual quality management. First, combinations of attributes that may cause malfunction or misjudgment need to be reliability addressed at the training or testing stage. Second, at the same time, it is necessary to cover as much as possible all situations which the machine-learning-based system to be implemented may encounter during operation from the viewpoint of both the quality of training and the quality of artifacts.

Such a problem has been addressed as a task of test design in developing conventional software. However, in machine learning where not only the testing process but also the implementation process is performed based on datasets, it is unique that the same task is required in the implementation process.

Some practical solutions to the excessive number of combinations for test design have been already presented in conventional software engineering. The Guideline aims to provide practical and sufficient training and quality testing by applying such existing knowledge to machine learning applications.

7.3.2 Approaches

Firstly, if the number of attributes to be recognized is very small and the total number of all attributes where the impossible cases described in the previous paragraph are deducted is only 10 to 20, their combinations are named as cases, and they are checked if all cases are included in test datasets and training datasets in later stages.

On the other hand, if there are too many combinations when the total of those attributes are used or sufficient data cannot be acquired especially in rare cases, combinations of some attribute values should be extracted under certain standards and set as cases to cover all those combinations. For example, in the example of traffic lights listed in Example 1 in Sections 6.3, 6.2, such combinations as daytime green, daytime yellow, nighttime red, daytime rainfall, nighttime rainfall and red signal in rain are extracted and called cases to make sure that data applicable to these combinations is included in test datasets. Even if all attributes cannot be covered completely, it is possible to intend to achieve a certain degree of coverage by avoiding cases where high-risk situations such as nighttime rainfall are not included at all in datasets or eliminating cases where red signal is not included in training at all. When considering such simplified coverage, the data of red signal and daytime rainfall are included in the several cases such as daytime rainfall, daytime red signal and red signal and rainfall at the same time.

Furthermore, completeness should be guaranteed more specifically in applications that require high quality by introducing mathematical coverage criteria to such extraction works. In the case of black box tests in software engineering, methods such as the sampling rate of random sampling, the orthogonal table and pair-wise test and the coverage criteria based on them are known, we should select appropriate means for each application.

7.3.3 Requirements for quality levels

For sufficiency of problem domain analysis, it is required to handle the following three levels in accordance with the level of external quality.

The correspondence between external and internal qualities regarding the required level is as follows.

Safety

AISL 0.1: Lv 1 or above
AISL 0.2: Lv 2 or above
AISL 1: Lv 3
AISL 2~4: Requirements to be added to Lv 3 will be considered.

AI performance

AIPL 1: Lv 1 or above
AIPL 2: Lv 2 or above

Fairness

AIFL 1: Lv 1 or above
AIFL 2: Lv 2

The requirements for each level of internal qualities are as follows.

- Lv1
  - Set cases for each of attributes corresponding to major risk factors.
  - Moreover, set cases corresponding to combinations of composite risk factors.
  - Furthermore, extract attributes of differences in particularly-important environmental factors and prepare cases corresponding to combinations with serious risk factors.
- Lv2
  - Satisfy all requirements listed in Lv 1.
  - Particularly-important risk factors should satisfy, in principle, the standards for pair-wise coverage. To be more specific, a case of combining an attribute value of combination of those factors and individual attribute values included in all attributes other than those to which the attribute value belongs should be included.
- Lv3
  - Based on engineering consideration, set standards for coverage of attributes and establish sets of combinations of attribute values that satisfied standards for coverage as cases.
  - The level of strictness of the standards for coverage such as pair-wise coverage, triple-wise coverage, etc. should be set taking into account system usage and risk severity. Standards can be set individually for each risk where necessary.

7.4 B-1: Coverage of datasets

7.4.1 General

The term coverage of datasets means that enough data are given to each of the cases covered by establishing the standards as described in the previous paragraph without omissions for the input possibilities corresponding to each of the cases.

When conventional software is developed, the details of all characteristics in real world which software operations depend on are captured at any stage from the machine learning requirements analysis phase to the implementation phase and reflected in software components in the form of conditional branching or calculation formula. On the other hand, when a machine

learning component is built, differences in details exceeding a certain level are not captured as attributes of ML requirements analysis or labels at the time of training but are reflected in ultimate operations through the training phase of machine learning as a distribution of datasets used for learning, as described in Section 7.1. The purpose of establishing this characteristic axis is to guarantee that no inappropriate learning behavior occurs due to lack of data with regard to characteristics of those unidentified details as attribute values.

7.4.2 **Approaches**

The main purposes of this quality aspects are to achieve sufficient level of quantity and coverage over input situation. However, since there are characteristics whose minor differences have not been identified as attributes, the latter coverage often has no choice but to depend much on the process of collecting and processing data. On the other hand, when the standards for coverage are introduced in Section 7.2.4, it would be possible to test if attributes which are not included in cases are distributed without bias.

Moreover, as regard the former quantity, it is important to appropriately define granularity of how to set the cases described in the previous section, but it is possible that sufficient data for specific cases cannot be obtained, for example, in rare cases where the frequency of occurrence is low. In those cases, it may become necessary to take measures in the whole development process for evaluating the response status by means of tests on the whole system by partially abandoning coverage of datasets for cases that lack specific data and after covering looser coverage standard.

7.4.3 **Requirements for quality levels**

The correspondence between external and internal qualities regarding the required level is as follows.

Safety

AISL 0.1: Lv 1 or above
AISL 0.2: Lv 2 or above
AISL 1: Lv 3
AISL 2~4: Requirements to be added to Lv 3 will be considered.

AI performance

AIPL 1: Lv 1 or above
AIPL 2: Lv 2 or above

Fairness

AIFL 1: Lv 2 or above

AIFL 2: Lv 3

Note: We consider that this quality characteristic is important in dealing with data bias, which is an important factor that impairs fairness.

The requirements for each level of internal qualities are as follows.

- Lv1
  - Consider the source and method of acquiring test datasets to ensure that no bias is found in application situations.
  - Extract samples without bias from original data for each case to ensure that no bias is found.
  - Record activities carried out to prevent bias from entering.
  - Check that there are sufficient training data and test data for each analyzed case in the training phase, validation phase, and so on.
  - When sufficient training data cannot be acquired for any case, review and loosen the coverage standards and record what should be checked individually by system integration tests in line with the original standards.
- Lv2
  - The following activities are carried out in addition to those listed in Lv 1.
  - Grasp an approximate probability of occurrence for each attribute value or each case.
  - Check if acquired data are not deviated from the distribution.
  - Positive check other than acquisition methods should be made regarding the coverage of the data included in each case.
  - For example, in each case, when there is any attribute not included in that case, extract the distribution related to attribute and check if there is no significant bias.
- Lv3
  - Acquire certain indicators for coverage of data included in each case in addition to those listed in Lv 2.
  - For example, check if there is no correlation between data other than attribute values included in combinations of cases using feature extraction or any other technique.
  - Or consider an expected distribution of attributes not included in each case, and analyze and record differences.

7.5 B-2: Uniformity of datasets

7.5.1 General

Evaluating only the coverage of each case in the previous paragraph does not always mean that all datasets are good sampling of the overall environment expressed by input data. When

the probability of occurrence differs significantly from case to case, simply preparing samples for each case will generate datasets with a large bias as a whole, which may significantly impair performance, especially in terms of AI performance. On the other hand, when performance is required for rare cases that probability of occurrence is very low, it cannot be generally coexist the preparation of the practical amount of uniform data without bias for all input and the preparation of the sufficient amount of data for rare cases. For example, when 100 training data are required for an event that occurs at a frequency of one millionth, it is not usually acceptable that the number of cases of all unbiased data is 100 million. From this viewpoint, it is considered that the uniformity in this paragraph needs to make an appropriate compromise in some cases, contrary to the coverage in the previous paragraph.

In general, it is required to prepare sufficient training data for combinations of attribute values with risks which should be avoided by making correct judgments when safety is strongly sought. When such a risk occurs on rare occasions, an enormous amount of data might be required for training all other cases with sufficient amount of data. In such case, it is quite conceivable to focus on training of particularly rare risky cases.

On the other hand, when overall performance (AI performance) is required, by focusing on training rare cases more than the actual probability of occurrence, the inference accuracy deteriorates in other cases, and it may also worsen average performance. In such case, it is not always appropriate to deeply explore the coverage of detailed cases in the previous section.

Moreover, when fairness is strongly required, it may change whether the training should be artificially equivalent between cases or should be randomly trained according to the distribution of extracted training datasets, depending on a type of fairness required.

7.5.2 Approaches

The basic concept itself is to focus on cases of overall in the topic of coverage in Section 7.4.2. While taking care not to bias in the process of acquiring whole datasets, it is necessary to monitor the frequency of occurrence of each attribute value as appropriate.

Rather, as stated in general, in this section, it is important to consider how to coexist with the coverage in the previous section and data design.

7.5.3 Requirements for quality levels

The correspondence between external and internal qualities regarding the required level is as follows.

Safety

AISL 0.1: Lv S1 or above

AISL 0.2・1 → Lv S2 or above

AISL 2~4: Requirements to be added to Lv 2 will be examined.

AI performance

AIPL 1: Lv E1 or above

AIPL 2: Lv E2 or above

Fairness

AIFL 1・2 → Lv E2 or above

The requirements for each level of internal qualities are as follows.

- Lv E1
  - Same as Lv 1 in Coverage of datasets in the previous section.
- Lv E2
  - Same as Lv 2 in Coverage of datasets in the previous section. However, assumed probabilities of occurrence are compared with the whole sets of assumed events.
- Lv S1
  - Regarding the amount of data for each case considered in L1 of the previous section, explicitly check if there are sufficient amount of data for risk cases.
  - When data of rare cases is insufficient for training, comparing the amount of the whole sets of training data with a probability of occurrence of rare cases, consider focusing on learning of rare cases. However, especially when Lv E2 is required, with prioritized, the impact of reduced training of other cases on whole system quality should be considered.
- Lv S2
  - In addition to what is listed in Lv S1, estimate and design in advance the amount of data of each case, based on the estimated probability of occurrence for each risk event/case.

7.6 B-3: Adequacy of data

7.6.1 General

Adequacy of data means that the data used in the machine learning training and testing process are free of errors and inappropriate data, and include a variety of internal perspectives.

The data quality defined by SQuaRE (ISO/IEC 25012) [8] can be classified in terms of data-specific and system-dependent aspects used as follows. Of these, the underlined terms are considered to be particularly important as the source data for building machine learning.

Data-specific quality characteristics

- (1) Accuracy
- (2) Completeness
- (3) Consistency
- (4) Credibility
- (5) Currentness

Data quality characteristics from both data-specific and system-dependent perspectives

- Accessibility
- Compliance
- Confidentiality
- Efficiency
- (6) Precision
- (7) Traceability
- Understandability

Data quality characteristics from the system-dependent perspectives

- Availability
- Portability
- Recoverability

Also, in terms of the machine learning process, there are two distinct factors:

A) Data Selection Appropriateness:
Existence of each data point in the dataset is appropriate for the policy fixed by the coverage and uniformity of the dataset (B-1, B-2)
- For example, each data point does not contain measurement errors or is not outliers that should be removed.

[(1) Accuracy, (2) Completeness, (4) Credibility, (5) Currentness, (6) Precision, (7) Traceability]

B) Adequacy of labeling:
For each data point in the dataset, the information added in the dataset preparation stage is appropriate.
[(2) Completeness, (3) Consistency, (4) Credibility]

These aspects are generally considered to correspond to the properties of SQuaRE as shown in square brackets.

Authenticity, which is often pointed out as data quality in the field of artificial intelligence, and is often important from the perspective of security, can be considered to correspond to (4) Credibility and (7) Traceability in the above quality characteristics.

7.6.2 **Approaches**

Although the adequacy of data is ultimately aimed at obtaining good machine learning

components, from the perspective of quality management, it is basically considered to be evaluated against the requirements for the data expected by the system, which are defined by the internal quality characteristics from A-1 to B-2.

In the specific evaluation of this quality characteristic, it is necessary to evaluate various viewpoints in the inspection of the data itself as well as in the management of the construction process.

7.6.2.1 Unification and scrutiny of labeling policies

Although the labels to be assigned as teacher data themselves are specified in A-1 Sufficiency of problem domain analysis, there can be various fluctuations, confusions, and ambiguities in the actual data. For example, in image feature detection, the size and distance of objects to be extracted as labels, and the handling of blocking conditions of overlapping objects need to be clarified in terms of functional requirements. If these points of view are not consistent among workers, fluctuations in the labels may lead to reduced accuracy in the training process and inaccurate inspections in the testing process.

In addition, when the requirements are extended due to repeated trials in the PoC phase or rework due to insufficient accuracy, the labeling itself needs to be extended or modified. However, relabeling work is generally expensive, and even here, there may be inconsistencies in labeling among workers. Furthermore, when acquiring additional data, the environmental conditions of the data itself may not be consistent.

From these perspectives, it is important to conduct a thorough study as early as possible in the PoC phase, to solidify the labeling policy in as much detail as possible, and to record it in writing, in order to ensure traceability of quality control.

7.6.2.2 Consistency checking and rechecking of datasets

In the construction of a machine learning system, it may be possible to use existing pre-measured datasets or labeled datasets. However, since the adequacy of data is determined by its consistency with the requirements, the validity of existing data should be re-evaluated whenever the functional requirements or the assumption of the operating environment is changed. In addition, when data collection or labeling work is outsourced, it is necessary to conduct inspections for acceptance from the viewpoint of quality control. How such inspections are conducted needs to be thoroughly discussed in advance, even before the process is established.

7.6.2.3 Handling the long tail and determining mismeasurement and outliers

It is possible to automatically screen for a certain amount of data being out of tendency from other data by using statistical analysis or other methods. However, whether such rare data should be taken as a meaningful training target, such as the long tail, or should be dismissed as outliers/mismeasured values may depend on the nature of the problem and the content of the individual data and may also depend on the priorities of external quality of safety and AI performance. Without a clear policy or decision-making process in this regard, the data selection and labeling process will produce ambiguities.

7.6.2.4 Addressing data poisoning (security and authenticity)

When training and test data can contain intentional errors or biases, or when there are malicious modifications (e.g., interference with sensors) to the data source (e.g. measurement environment), the functionality of the final system may be severely affected. Such errors are not detected in the testing process if there is data contamination, and in general cannot be prevented sufficiently by system testing.

From this point of view, it is necessary to prevent manipulation of test data through general information security measures in order to prevent and reduce data contamination (data poisoning). Also, to avoid data collection from contaminated data sources as far as possible, the physical security and diversity of the data acquisition environment should be ensured from a process management perspective. It is also important to keep records of such quality assurance activities at each data preparation stage and store them as audit trails.

Data poisoning detection techniques could also be used to identify and remove data that degrade the model's performance. However, currently there is no general-purpose method to detect data contamination before training, especially in the case of procuring data from external sources. From this point of view, a system that requires a certain level of quality may have to rely on the credibility and traceability of the dataset itself or the provider to prevent data contamination. For example, it is useful to utilize a framework that provides reliable datasets.

It is also useful to consider the possibility of synthesizing and processing datasets to prevent or mitigate data poisoning. For example, increasing the volume of reliable training data through data augmentation can reduce the impact of data poisoning. If the dataset includes privacy information, trade secrets, or other sensitive data violating laws, regulations and contracts, then identification and removal these data or information and add other data is preferred.

Details are discussed in Section 10.3.4.1 as a control measure in response to data poisoning attacks.

7.6.2.5 Currentness

In machine learning applications, performance often degrades as time passes since the acquisition of training data. Managing the currentness of the training data is important to prevent such quality deterioration. On the other hand, the requirement for currentness often conflicts with the amount of data available for training, and there can be a trade-off between currentness and completeness, especially when rare cases need to be handled. From this point of view, it is necessary either to consider the policy of currentness in advance, or to identify a reasonable requirement level at the PoC stage.

7.6.2.6 Processes, organizations, and systems

As mentioned above, in ensuring the adequacy of data, there are many factors that depend not only on the specific handling of individual data but also on the overall construction process and the system, including auditing. In addition, in the actual construction of a machine learning system, handling concrete data in the process of judging the validity of data often leads to more detailed requirements definition, which may have a cascading effect on the internal quality characteristic A-1 in the previous stage. In the Guideline, this kind of work is classified as trial and error in the PoC stage. However, when this kind of circulatory work occurs, it is especially important to verify whether the final outcome is consistent. On the other hand, it is not practical to redo all data inspections from scratch in every trial. From this perspective, it is considered that quality control requires the creation of a system and structure to properly manage the process, including the management of changes in policies during development.

7.6.3 Requirements for quality levels

The correspondence between external and internal qualities regarding the required level is as follows.

Safety

AISL 0.1: Lv 1 or above
AISL 0.2: Lv 2 or above
AISL 1: Lv 3
AISL 2~4: Requirements to be added to Lv 3 will be considered.

AI performance

AIPL 1: Lv 1 or above
AIPL 2: Lv 2 or above

Fairness

AIFL 1: Lv 1 or above

AIFL 2: Lv 2

The requirements for each level of internal qualities are as follows.

- Lv1
  - General:
    - ✧ Properly examine and confirm whether the source of the data is appropriate for the problem.
    - ✧ Organize the labeling policy.
    - ✧ Review and summarize the criteria for labeling and outlier removal in advance.
    - ✧ Determine whether the criteria are appropriate in the context of the given data, and if necessary, review and recheck the criteria.
    - ✧ When labeled data are used, the validity of existing labels should be examined in advance and confirmed by pre-testing, etc. if necessary.
  - Fluctuation of labels:
    - ✧ A consistent standard should be established among workers to judge labels, or a double check is to be performed.
  - Data contamination:
    - ✧ Consider the impact and likelihood of contamination of data sources.
  - Currentness:
    - ✧ Consider in advance whether the dataset contains data from an inappropriate time period, according to the characteristics of the problem.
- Lv2
  - Take the following actions in addition to Lv1.
  - General:
    - ✧ Incorporate the data preparation stage into the quality control process and manage it properly.
    - ✧ If data are procured externally, incorporate data preparation methods, processing methods, quality control processes, and security controls into the requirements.
  - Labeling Policy:
    - ✧ Establish a control process to eliminate variations in labeling by workers.
    - ✧ Establish a process to manage label changes when data attribute definitions are changed.
  - Fluctuation of labels:
    - ✧ Review and document in advance the range of acceptable label variability.
    - ✧ Record labeling decisions regarding fluctuations that occur during production.
  - Data contamination:
    - ✧ As far as possible, consider using technologies to detect contamination of

training data and sources.
    - ✧ Consider designs that detect contamination (poisoning) of data sources or datasets.
    - ✧ Consider the likelihood of data synthesis and processing.
    - ✧ For more information, see section 10.3.4.1 of external quality AI security.
  - ➢ Currentness:
    - ✧ Incorporate the data preparation stage into the quality control process and manage it properly.
  - ➢ A posteriori inspection:
    - ✧ If possible, consider applying input impact analysis, neuron firing status, and other internal information analysis, and manually eliminate obvious errors to the extent possible.
- Lv3
  - ➢ Take the following actions in addition to Lv2.
  - ➢ Labeling Policy:
    - ✧ Conduct and record a risk analysis of the impact of label design.
  - ➢ Confirmation of label data removal:
    - ✧ Double-check at receiving inspection at the time of outsourcing, or set up and inspect the audit process in advance.
  - ➢ Data contamination:
    - ✧ Conduct and document a risk analysis of data contamination.

7.7 B-4: External-quality-dependent adequacy of datasets

This section describes only the requirements of datasets concerning the external qualities of fairness and privacy. Requirements for external quality AI security are discussed in B-3: Adequacy of data (Section 7.6) and B-4pr: Adequacy of datasets in terms of privacy (Section 7.7.2).

7.7.1 B-4f: Adequacy of datasets in terms of fairness

In the data preparation stage, in addition to general internal quality measures for *adequacy of data*, *dataset coverage*, and *dataset uniformity*, pre-processing techniques are implemented to ensure fairness target. What is commonly referred to as data fairness is addressed at this stage, encompassing both measures taken during data collection and those applied to the collected data.

7.7.1.1 Training data collection process without introducing bias

From the perspective of preventing disparate treatment, first of all, we should avoid bias in data collection operation, such as sample size disparity and tainted examples [13], which makes datasets become biased against the real world. Techniques such as weighting the collected data by groups (Reweighing [132]) are used for the purpose.

These measures are essentially a process of combining the two aspects of confirming whether there are enough data for the attribute combinations of interest, or whether the data are close to the distribution in the real world, from a fairness point of view. This is achieved by a unique perspective, namely sensitive attributes.

7.7.1.2 Data adjustment, enhancement, and synthesis

If we want to prevent *disparate impact*, even if the collected dataset is not biased against reality, we may need to take further measures against the distortions and discriminatory factors inherited from the real world.

Examples of major techniques include the following

- **Disparate Impact Remover**

To remove the bias caused by the so-called proxy, the attribute value that may be correlated to the sensitive attribute is modified to reduce the possibility of indirectly inferring the sensitive attribute from that proxy attribute.

- **Optimized Pre-Processing**

A framework for pipelined transformations to pre-process training data. Consideration is given to both the removal of bias and the usefulness of the data-should not be too far from the real data [86].

7.7.1.3 Required actions for each quality level in data preparation

Required actions for each target AIFL level are as follows in dataset preparation phase.

- AIFL 1
  - ➢ Measure and record fairness metrics for datasets.

[13] Sample size disparity: The number of data points varies greatly depending on the value of the attribute requiring consideration.
Tainted Sample: Bias caused by human labeling

  - When the dataset preparation is within the work scope, if the metrics deviate from the target, at least different treatment measures described in Section 7.7.1.1 should be taken to improve the metrics.
- AIFL 2
  - In addition to those listed in AIFL 1, the following actions should be taken
  - When the dataset preparation is within the work scope, if the metrics deviate from the target, different impact measures described in Section 7.7.1.2 should be taken.
  - When dataset preparation is outside the work scope, consider measures to minimize the impact of the dataset metrics deviation from the target on learned model output later in the process. In some cases, collateral outside of the machine learning component should be considered in advance.

7.7.1.4 Considerations for the expansion of generative AI

With the expansion of generative AI, concerns have begun to be raised about the impact of its outputs being reused in subsequent training [203]. In general, generative AI tends to prioritize majority data, and repeated AI learning involving many generative AI outputs can lead to reduced diversity and sharper distribution than original (real world) dataset distribution. This may amplify the bias inherent in real datasets, and "pre-processing" methods discussed in this section. Looking ahead, especially from the perspective of fairness, greater attention should be paid to the provenance of training datasets, and it may be preferable to rely more heavily on real-world data than on generated data.

7.7.2 B-4pr: Adequacy of datasets in terms of privacy

Implement the technical measures described in the Pre-stage of the Overview of quality management implementation (Table 4). Studying data in relation to the privacy protection is conducted in view of both the handling of individual training data and the data distribution of the training dataset.

It sometimes happens that the training data may be revised and retraining may be required after the system begins operation. Even in this case, check the revised training data to ensure that the quality requirements in this section are met.

7.7.2.1 Handling of training data

The quality to be satisfied by training data are relevant to B-3 in general. In particular, the conformance with privacy-related legal regulations implies fulfilling the compliance of the

quality characteristic. Moreover, the quality characteristics that the training data should satisfy may vary depending on the regulatory laws. For example, GDPR specifies accuracy and currentness in Article 5. In addition, there is a method to study the GDPE-regulated quality aspects, by mapping them to those of the SQuaRE data quality model [188], [170].

In the process of preparing raw data into training data, appropriate protection processing should be applied, and measures should be implemented to prevent the leakage of personal information. Especially, consider possibilities that the technical measures of the protection processing may hinder the achievement of the system requirements.

7.7.2.2 Control of data distribution

To lower threats of privacy leakage (training data identification), the data distribution of the training dataset should be controlled so that the trained model does not introduce unintended training data memorization. Specifically, the measures start with detection of outliers using a method specific to the application problem at hand [68], [245]. Next, the data distribution is adjusted by methods such as removing outliers or adding training data near outliers, according to the requirements of the target system.

7.8 C-1: Correctness of trained models

7.8.1 General

The term correctness of trained models represents that a machine learning component functions as expected upon the input from the training dataset consisting of training data, test data, and validation data.

7.8.2 Approaches

The correctness of trained models is usually evaluated in terms of quantitative measures, such as accuracy, precision, recall, or F-value. These measures may overfit the training dataset; that is, they may perform well for the training data, but badly for the data that are not included in the training dataset.

Therefore, in the test phase, the correctness needs to be evaluated using a test dataset that is disjoint from the training dataset, or more generally, by applying cross validation.

Data scientists should select concrete techniques suited to the application and need to explain their selection. In Section 9.7 we will explain some examples of techniques that can be used to

test the correctness of a trained model.

7.8.3 Requirements for quality levels

The correspondence between external and internal qualities regarding the required level is as follows.

Safety

AISL 0.1: Lv 1 or above
AISL 0.2: Lv 2 or above
AISL 1: Lv 3
AISL 2~4: Requirements to be added to Lv 3 will be considered.

AI performance

AIPL 1: Lv 1 or above
AIPL 2: Lv 2 or above

Fairness

AIFL 1: Lv 1 or above
AIFL 2: Lv 2

The requirements for each level of internal qualities are as follows.

- Lv1
  - Prepare a test dataset by taking into account the coverage of data and the past experiences, e.g., obtained in the PoC stage.
  - Prepare a training dataset in an analogous way to the test dataset. Note that the training dataset may not necessarily follow the same distribution as the test dataset.
  - Decide and record how to deal with the incorrect behavior of a trained model (e.g., false negative or false positive in the test) before the validation phase.
  - If fairness is required, establish a means to compare fairness in advance. If fairness is based on the results of a control test, establish the criteria for passing the test in advance.
- Lv2
  - Take the following actions in addition to Lv1.
  - Decide and explain methods and criteria to validate the trained model (e.g., accuracy and its threshold) before the validation phase.
  - Test the trained model using the given test dataset and additional test data (e.g., generated by data augmentation techniques).
  - If possible, consider applying input impact analysis, neuron firing status, and other internal information analysis, and manually eliminate obvious errors to the extent possible.

- Lv3
  - In addition to all the requirements listed in Lv2, examine the means of internal verification of learning maturity status in advance.
  - Clarify the correspondence between the verification plan for the entire system after integration tests and the test plan for machine learning components.
  - Focusing on particularly risky cases, reflect the correspondence with the requirements of machine learning components during system-level testing in the test plan, and monitor and confirm their coverage status.

7.9 C-2: Stability of trained models

7.9.1 General

The stability of trained models indicates that the machine learning components respond as expected to inputs that are not part of the datasets. Insufficient stability leads to low prediction performance for unknown inputs (low AI performance) and increases risk due to potential serious misjudgments (low safety). Therefore, it is important to evaluate stability, especially when safety is required. The following two issues are well known regarding stability.

- The trained model may infer completely incorrect predictions when the input is not contained in the training data. Typically, such incorrect predictions are often caused when the model is overfit to the training dataset.
- The trained model may behave significantly differently when a small amount of noise is added to the input to the model. Such input noise can be either random noise in nature (e.g. dirty camera lens) or adversarial perturbation caused by malicious attacks. For example, those attacks are triggered by data poisoning in the training dataset, and by certain tricks on physical objects (e.g. attachment of tiny stickers to road signs).

7.9.2 Approaches

Stability can be evaluated and improved mainly in the following three phases in the machine learning lifecycle. Some useful techniques will be explained in Section 9.8.2.

- Training phase: Overfitting to the training dataset can be avoided by various techniques for improving generalization ability. In addition, noise robustness can be improved by evaluating the degradation due to small perturbations during training.
- Evaluation phase: Noise robustness can be evaluated by correct rates (i.e. accuracy) for

test datasets, where random or adversarial perturbations are added to input data. In addition, generalization errors (i.e., the expected incorrect rates for any inputs) can be also used for evaluating stability for unseen data not included in datasets.

- Operation phase: By monitoring inputs to trained models during operation, some of adversarial inputs may be detected and removed.

7.9.3 Requirements for quality levels

It is required to record the techniques applied to improve the stability of a trained model and the evaluation results of the stability. The following items should be recorded at each level, where "neighborhood data" represent data generated by adding small perturbation to original data. Examples of techniques will be explained in Section 9.8.2.

The relationships between the required levels AISLs for external qualities and the required levels Lvs for the internal quality "the stability of a trained model" are explained as follows.

- Safety
  - AISL 0.1: Lv 1 or above
  - AISL 0.2: Lv 2 or above
  - AISL 1: Lv 3
  - AISL 2∼4: Requirements to be added to Lv 3 will be considered.
- AI performance
  - AIPL 1: Lv 1 or above
  - AIPL 2: Lv 2 or above
- Fairness
  - AIFL 1: Lv 1 or above
  - AIFL 2: Lv 2

The requirements for the above required levels Lvs for the internal quality are defined as follows.

Lv 1: Record the techniques applied to improve the stability and the robustness for small noise.

- For stability at Lv 1, we recommend applying some widely accepted techniques for avoiding overtraining and overfitting, such as cross validation, regularization, early stopping, dropout, and ensemble learning, and also applying techniques for evaluating noise robustness by adding random perturbations to test data.

Lv 2: Record the evaluation results of robustness for adversarial perturbations.

- For stability at Lv 2, we recommend applying techniques for improving and evaluating the robustness for adversarial examples, which are very similar to but are intentionally generated from original data for causing misclassifications. Such

techniques include adversarial training to prevent the attacks by adversarial examples, robustness evaluation for adversarial examples, and detection of adversarial examples during operation. Tools for applying such techniques are published as, for example, Adversarial Robustness Toolbox [174] and Foolbox [189].

Lv 3: Provide some formal guarantee to the stability for data not included in datasets.

- For the stability at Lv 3, we require to guarantee a certain level of stability for unseen data not included in datasets. For example, there are techniques for certifying that no adversarial example exists in neighborhood of some data or statistically guaranteeing generalization error bounds. Such techniques have been studied recently and might be used in system development in the future.

7.10 C-3: External-quality-dependent adequacy of trained models

7.10.1 C-3f: Adequacy of trained models in terms of fairness

When, developing, training, and testing models using datasets prepared as described in Section 7.7.1, in addition to internal quality measures for general model correctness and stability, in-processing and/or post-processing measures are implemented to ensure fairness goals.

7.10.1.1 Measures in model development and learning phase (in-processing)

In-processing measures are those that aim to embed fairness in the model and can mainly be implemented by modifying the objective function with *additional constraints* or *adding model elements*. Examples are:

- **Prejudice Remover (additional constraint type)**

The resultant probability distribution is adjusted by adding a constraint to remove indirect prejudice[14] that remains even if the sensitive attributes are not used directly. For this purpose, the constraints are implemented as a regularizer using the sensitive attributes and added to the objective function. This method is highly explanatory, although its applicability is limited to classification type models[133].

- **Adversarial Debiasing (additional model example)**

This technique adds an adversarial model that uses the inference results from the target model

[14] One source of bias. The statistical dependence of a target variable or a non-sensitive attribute on a sensitive attribute.
Gender

as input to infer the sensitive attribute. The target model should be trained so that the inference of that adversarial model fails as much as possible. Adversarial debiasing allows for a wide variety of target model algorithm and may handle the problem of insufficient proxy analysis in data preparation phase [236].

7.10.1.2 Adjustment to trained models (post-processing)

When pre- or in-processing measures are insufficient to satisfy the required fairness target metrics, adjustments [186] to trained models, adjustments can be applied to trained models as post-processing techniques such as Equalized Odds framework and calibration methods.

Moreover, in products and services where fairness is critical—particularly those deployed in public or open environments—there is a higher likelihood of issues such as concept drift. It is essential to give sufficient consideration to the possibility that changes in model quality during deployment may lead to broad social consequences.

Some post-processing methods assume that bias remains in the trained model, and aim to mitigate it by applying corrections at the inference stage. One such method is introduced below as an example. Further adjustments may also be implemented at the system level, including components outside the machine learning model itself, to suit the operational context.

- **Reject Option-based Classification (ROC)**

This technique is based on the hypothesis that discrimination tends to occur when the model's confidence is low. ROC either rejects or modifies such uncertain predictions to prevent discriminatory outcomes. Specifically, if the favorable outcome of the favored group or the unfavorable outcome of the non-favored group falls below a predefined confidence threshold, the result may be replaced or adjusted [131].

Fairness metrics used in the post-processing methods described in this section should also be quantified during training—namely, when applying the in-processing methods discussed above. Defining these metrics appropriately and measuring them rigorously is essential. The next section presents several examples of fairness metrics.

7.10.1.3 Example of target metrics

The metrics used in the measures described so far should be selected based on the specific fairness requirements—such as whether the focus is on treatment or outcomes. Representative examples of fairness metrics are listed below. For others, please refer to the literature [155].

Metric	Explanation (Typical implementation)
Equalized odds	The proportion of *correct decisions* (e.g., TP / (TP + FN) and TN /

	(FP + TN)) is equal across sensitive attributes.
Predictive parity	The positive predictive value (e.g., TP / (TP + FP)) is equal across sensitive attributes.
Demographic parity	The proportion of *favorable outcomes* (e.g., (TP + FP) / (TP + FP + TN + FN)) is equal across sensitive attribute.

7.10.1.4 Required actions for each quality level in data preparation

Required actions for each target AIFL level are as follows in model training phase.

- AIFL 1
  - Measure and record the model output metrics.
  - If there is a deviation from the target, at least one in/post-processing technique should be used to make improvements, and if there is still a deviation, record it.
- AIFL 2
  - In addition to those listed in AIFL 1, the following actions should be taken
  - If there is a deviation from the target, try multiple in/post methods to make improvement as much as possible.
  - If there is still a discrepancy that cannot be resolved, consider combining additional measures in actual operation, sometimes extending to outside the machine learning module.

7.10.2 C-3pr: Adequacy of trained models in terms of privacy

Implement the technical measures described in the In-stage of the Overview of quality management implementation (Table 4). While using an appropriate learning mechanism is important, advanced training algorithms with a theoretical foundation, such as privacy-preserving machine learning methods, may be used as well.

Adequacy of trained models in terms of privacy should be checked during the initial development and continuous training after the system is put in-operation.

7.10.2.1 Basic training mechanism

Training methods that prevent overfitting the training data should be used to avoid the training data memorization as much as possible. To enhance the generalization performance, methods such as regulation and dropout have been devised so far in standard training mechanisms for deep neural networks. The latest techniques are to be appropriately used.

7.10.2.2 Privacy-preserving machine learning

In some cases, privacy-preserving machine learning (PPML) might be used. However, achieving expected levels of data protection performance with the current PPML technology is difficult. Using PPML improperly may bring about the trade-off issues with AI performance and/or fairness.

7.10.2.3 Measures for trained models

Implement the technical measures described in the Post-stage of the Overview of quality management implementation (Table 4). Basically, the output of the trained model is processed to prevent outputting information that would facilitate re-identification of training data.

In addition, some membership inference methods use meta-classifiers that apply machine learning techniques. Adding adversarial perturbation to the prediction results is sometimes an effective safeguard, which can reduce the threats of the membership inference by leading to mispredictions.

Moreover, the safeguards introduce a change in the external interface of the trained model, and thus it needs to examine, with regard to the system requirements, whether the introduction of such safeguards is allowed.

7.10.3 C-3se: Adequacy of trained models in terms of security

This section describes the control measures that should be implemented for pre-trained models (Section 7.10.3.1), control measures that should be implemented in the training of machine learning models (Section 7.10.3.2), control measures that should be implemented for trained models (Section 7.10.3.3), and control measures to be taken by preprocessing and postprocessing programs for machine learning components (Sections 7.10.3.4 and 7.10.3.5) from the quality management implementation for AI security. The specific techniques for these control measures are described in Section 10.3.

7.10.3.1 Control measures to obtain pre-trained models

If a pre-trained model is contaminated by a malicious attack, the model trained using that pre-trained model may behave unintentionally, exhaust computing resources, or leak sensitive information that is embedded in the pre-trained model.

Therefore, developers should evaluate the credibility of pre-trained models and take

measures to prevent or mitigate poisoning of the pre-trained models used to train the model.

- **Authenticity of the pre-trained models:** Check the authenticity of the pre-trained models. For example, by using digital signatures or frameworks for providing trusted pre-trained models, check that the pre-trained model has not been manipulated.
- **Credibility of the provider of pre-trained models:** Check the credibility of the provider of the pre-trained models. For example, check information to determine the social credibility of the provider.
- **Reliability of the training process of pre-trained models:** Check the reliability of the training of the pre-trained models. For example, check with the model provider regarding the process of model training and the development environment. In certain situations, sign a contract for the provision of pre-trained models, and the process management.

On the other hand, model poisoning attacks may be carried out by exploiting vulnerabilities in the development software or development environment (Section 10.3.3.3.2.1). Hence, the developers should implement conventional information security measures for the development of software and development environment.

The developers should take measures to prevent or mitigate model poisoning attacks when they assume a possibility of model poisoning, e.g., when they cannot sufficiently trust the model training/provision process of the pre-trained model.

- **Detection of pre-trained model poisoning:** To prevent model poisoning attacks, use model poisoning detection techniques (Section 10.3.4.2.2) to detect poisoning in the pre-trained models.
- **Processing of pre-trained models:** To mitigate damage due to model poisoning attacks, process the pre-trained models (Section 10.3.4.2.2).

7.10.3.2 Control measures for training the model

If the possibility of attacks is assumed, such as when the training data or pre-trained models cannot be trusted sufficiently, control measures for learning mechanisms (development software, such as training programs and test programs, and development environment) to mitigate the damage from attacks. To this end, developers should consider measures to prevent or mitigate damage that may be caused during system operation during the process of model training, depending on the situation.

- **Learning mechanisms that can reduce the impact of data poisoning:** If the possibility of data poisoning attacks is assumed, such as when the process of providing training data cannot be trusted sufficiently, use learning mechanisms that can remove or mitigate the impact of data poisoning. Details are described in Section 10.3.4.1.2.
- **Learning mechanisms that can remove or reduce model poisoning:** If the possibility of pre-trained model poisoning is assumed, such as when the process of

training/provision of pre-trained models cannot be trusted sufficiently, use learning mechanisms that can remove or mitigate the pre-trained model poisoning. Details are described in Section 10.3.4.2.2.

- **Learning mechanisms that can reduce malfunction and information leakage:** To reduce the degree and frequency of model malfunction or information leakage from the model, consider the possibility of using multiple different models or systems together, depending on the situation. For example, by considering the output results of multiple different models, such as in *ensemble learning*, there may be cases where attacks using adversarial example(s) can be mitigated. However, configuring successful attacks against models of multiple architectures may be possible. Furthermore, note that using multiple models and systems increases the cost of training and system operation.

Other learning mechanisms for mitigating attacks are addressed under other internal qualities as described below.

- The learning mechanisms to train robust models against evasion attacks are described in Section 9.8.2 as techniques for evaluating and improving the internal quality C-2: Stability of trained models.
- The learning mechanisms to prevent or mitigate the leakage of sensitive information of training data from the trained model are described in Section 7.10.2 and Section 9.9.1 as techniques for evaluating and improving the internal quality C-3pr: Adequacy of trained model in terms of privacy.

7.10.3.3 Control measures for trained models

A trained model that is contaminated by a malicious attack may behave in an unintended manner, exhaust computing resources, or leak sensitive information that is embedded in the pre-trained model. Therefore, to prevent or mitigate contamination of the trained model (model poisoning) in the development environment, the developers should implement general information security measures to prevent tampering of the trained model in the development environment.

Developers should also assess the risks associated with malicious attacks on the trained model.

- **Risk assessment of model extraction attacks:** To protect information about the attributes and functions of trained models, use tools to evaluate model extraction attacks (Section 10.3.4.4). A well-known tool is ML-Doctor [148] (see section 10.2.2.4.2). Moreover, new libraries and benchmarks for evaluating trained machine learning models are being continuously developed. So it is recommended to find and use those that can cope with the latest attack methods.
- **Risk assessment of sponge attacks:** Risk assessment and response of the model to sponge attacks (Section 10.3.4.6) may be considered according to future technological developments. At present, measures against sponge attacks depend on monitoring and

limiting the maximum consumption of computing resources during system operation.

Risk assessments of other attacks are addressed under other internal qualities as described below.

- Risk assessment and response of the model to evasion attacks (Section 10.3.4.5) are described in Section 9.8.2 as techniques for evaluating and improving the internal quality C-2: Stability of trained models.
- Risk assessment and response to attacks for information leakage of training data (Section 10.3.4.7) are described in Section 7.10.2 and Section 9.9.1 as techniques for evaluating and improving the internal quality C-3pr: Adequacy of trained model in terms of privacy.

If the possibility of model poisoning is assumed, such as when the learning mechanisms (development software such as training programs and test programs and development environment) cannot be trusted sufficiently, the developers should take measures to prevent or mitigate poisoning attacks.

- **Detection of trained model poisoning:** To prevent model poisoning attacks, use model poisoning detection techniques (Section 10.3.4.2.2) to examine the possibility of poisoning in the trained models.
- **Processing of trained models:** To mitigate model poisoning attacks, consider the possibility of processing trained models (Section 10.3.4.2.2).

7.10.3.4 Control measures with preprocessing programs for machine learning components

The developers should consider control measures to detect, process, and limit malicious input in the preprocessing program (a program that processes the input to the trained model during system operation) when designing and developing machine learning components. The detection techniques may not be able to detect attacks, and attack that evade the detection may be performed. Therefore, other control measures should be implemented to prevent or mitigate attacks, and detection techniques should be used only as secondary measures.

- **Detection, processing, and limit of inputs that exploit model poisoning:** Use techniques to detect, process, and limit attempts to exploit the poisoned model during system operation (e.g., input of malicious input to trigger backdoor attacks) (Section 10.3.4.3.2) in the preprocessing program.
- **Detection, processing, and limit of model extraction:** To prevent or mitigate model extraction attacks during system operation, use techniques to detect, process, and limit input that attempt model extraction during operation (Section 10.3.4.4.2) in the preprocessing program. Note that prevention or mitigation of model extraction attacks may also mitigate evasion attacks and attacks for information leakage of training data

(Section 10.3.4.4.1).

- **Detection, processing, and limit of adversarial examples:** To prevent or mitigate evasion attacks during system operation, use techniques to detect, process, and limit input of adversarial examples during system operation (Sections 9.8.2.8 and 10.3.4.5.2) in the preprocessing program.

If the system operators prepare the data input to the system during operation, they may not require control measures using a preprocessing program, because they can directly apply attack detection techniques and data processing techniques to the input data.

Access control for input during system operation is described in Section 7.12.1.1.

7.10.3.5 Control measures with postprocessing programs for machine learning components

The developers should consider control measures to limit the observation of trained model output/internal information (such as confidence scores and information from the model's interpretation function) in a postprocessing program (a program that processes the output of the trained model) and interpretation functions when designing and developing machine learning components.

- **Limit on the quantity/quality of output and internal information:** To prevent or mitigate attacks, limit the quantity and quality of the output and internal information of the model by adding perturbations (Sections 7.10.2.3 and 10.2).

Although this measure may reduce the possibility and probability of successful attacks, they do not guarantee the prevention of attacks. Furthermore, measures that restrict the observation of the model's output and internal information may reduce the system's transparency and explainability.

7.11 D-1: Reliability of underlying software systems

7.11.1 General

The term *reliability of underlying software* systems represents that the underlying conventional software (e.g., training programs and prediction/inference programs), excluding trained machine learning model, functions correctly. This concept includes various software quality requirements such as correctness of algorithms, memory resource or time constraints, and software security.

When electronic hardware such as MPU is developed, this concept also includes the hardware

reliability.

In many applications of machine learning systems, open-source software is used to train machine learning models and to develop conventional software systems. Although the quality of open-source software may not be guaranteed, the developers of machine learning systems should be responsible for ensuring sufficient quality even when using open-source software.

Furthermore, the correctness of software should usually be tested under the same environment as actual operation environments. When the test environment is different from actual operation environments, it is necessary to evaluate the differences between them. In the development of machine learning systems, however, there are often significant differences between an environment used for training (e.g., computing environment with cloud and GPU) and an actual operational environment (e.g., built-in computer). Even the behaviors of numerical calculations (e.g., the accuracy of floating-point computation) can change between them. In some cases, numerical processing itself may be changed, for example, by model compression. When implementing a machine learning system, the developer needs to cope with these environmental differences that may affect the AI performance of the system.

From the perspective of AI security, developers should use measures to manage vulnerabilities in the development software and development environment.

Learning mechanism vulnerabilities: It is necessary to implement security measures for conventional information systems to prevent or mitigate attacks exploiting vulnerabilities in the development software and the development environment (Section 10.3.3.3.2.1). It is necessary to collect the latest information on vulnerabilities in the development software and the development environment, such as machine learning frameworks.

Credibility assessment of learning mechanism: Evaluate the credibility of the development software and development environment. Confirm the credibility of the provider of the developed software, and the reliability of the software provision process.

7.11.2 **Approaches**

Quality management methods for conventional software systems can be applied in order to ensure the reliability of the underlying software system.

As regard the use of open-source implementations, appropriate quality assurance measures such as those listed below should be taken to achieve the safety and reliability of the system.

Moreover, when the in-operation environment is different from the environment in the training process (e.g., when performing model compression), the Guideline recommends to test the software that reproduces the same calculations as the in-operation environment in the test phase. If this is difficult, the developer should test the whole system to check if the quality deterioration falls within the acceptable range.

7.11.3 Requirements for quality levels

Safety

AISL 0.1: Lv 1 or above

AISL 0.2: Lv 2 or above

AISL 1: Lv 3

AISL 2~4: Requirements to be added to Lv 3 will be considered.

AI performance

AIPL 1: Lv 1 or above

AIPL 2: Lv 2 or above

Fairness

AIFL 1: Lv 1 or above

AIFL 2: Lv 2

- Lv1
  - Select reliable software used for the machine learning system, and record the process of this selection.
  - Monitor the system's operation to check and update the selected software.
  - Examine in advance the impact of differences between the environment in the training/test phases and the environment in the actual operation phase.
- Lv2
  - Evaluate the reliability of the software used for the system by testing.
  - If possible, use software whose reliability is SIL 1 or equivalent.
  - Prepare for the maintenance of software during its operation.
  - In the validation and test phases, conduct validation tests in an environment that simulates the environment used in the actual operation phase. Alternatively, validate the consistency of operations of the trained model between in the test phase and the actual operation phase.
- Lv3
  - Check the quality of software for SIL 1 (or a higher SIL level when required by the system).
  - Perform testing or formal verification of the behaviors of the trained model in an actual environment.
  - Check the consistency of those models and operations in an actual environment in any stage after integration tests.

7.12 D-2: Additional aspects of program adequacy

7.12.1 D-2se: Adequacy of programs in terms of security

This section describes the security measures using a program around machine learning components (Sections 7.12.1.1 and 7.12.1.2) in the implementation of AI security quality management.

7.12.1.1 Control measures with access control programs

In machine learning-specific attacks on systems in operation, attackers often repeatedly input malicious data into the system or machine learning components in order to understand the operation of the machine learning components. Hence, the developers should implement an access control program (a program that stops or limits malicious input during system operation) when designing and developing machine learning components.

Specifically, it is necessary to examine the restriction of access rights to the system and the number or frequency of accesses to the system to prevent or reduce malicious input to the machine learning components during in-operation.

Input limit during operation: When input data during system operation is obtained from an external environment (e.g., image data obtained by a camera), examine the restriction of the number and frequency of data input to the machine learning components.

Limit access: Examine the restriction or suspension of access rights and limit and stop services to users who query suspicious inputs into the system depending on the situation.

If the system operator prepares the data input to the system during operation, the above-mentioned control measures with an access control program are not required.

7.12.1.2 Control measures with risk monitoring and response programs

To address the risks that arise via trained models, it is useful to investigate the attacks on trained models during system operation and the damage caused by malfunction, resource wastage, and information leakage in trained models. Therefore, the developers should consider implementing a risk monitoring and response program (a program that monitors system operation and addresses risks that arise via trained models) when designing and developing machine learning components.

Means of detecting attacks: To prevent or mitigate attacks, consider the possibility of implementing a means of detecting attacks on trained models during system operation in the

risk monitoring and response program.

Means of detecting damage: To prevent or mitigate attacks, consider the possibility of implementing a means of understanding the damage caused by malfunction, resource wastage, and information leakage in trained models in the risk monitoring and response program. For example, consider applying model explainability techniques to investigate malfunction in trained models. In addition, if it is necessary to implement measures against sponge attacks (Section 10.3.4.6), consider monitoring and limiting the maximum consumption of computing resources to suppress the consumption of power during system operation.

Recording risk monitoring and response: Records of attacks on the system and damage caused should be maintained so that the operational organization can understand the status of risk monitoring and response.

8. Preparation and quality management during system operation

8.1 E-0: Continuous monitoring and recording during operation

Continuously monitoring quality indicators during the operation of a machine learning model, and retaining the records, is indispensable to explain the quality that has been realized and as evidence, and to maintain and improve the quality.

8.1.1 General

The goal of quality management for machine learning AI systems is to realize a reliable AI system. Therefore, it is necessary to realize the required quality and present evidence that the quality has been achieved in the machine learning model. Quality is important during the development period, at the start of operation and after the machine learning system has started operating. In this sense, it is important to continuously collect information related to the quality of the machine learning system during development and after operation starts, and to be prepared to present the information to the appropriate person at the appropriate timing.

8.1.2 Approaches

The first candidates for quality indicators that can be observed objectively are the evaluation indicators that are in line with each internal quality explained in chapters 6 and 7, and the following are considered to be the main targets for observation during operation.

- Operation status of the system
- Modification of system
- Change of external environment

8.1.2.1 Operation status of the system

Only important items such as input, output, and internal status should be acquired and recorded to indicate the daily operation status of the machine learning system.

In addition to retaining raw data, it is important to calculate and record the distribution of the input dataset corresponding to internal quality, and evaluation indicators such as accuracy, robustness, and fairness of output of the system. Furthermore, it is important to retain the records that examine how these factors impact the internal qualities of the system.

8.1.2.1.1 Relationship with privacy protection

Privacy has a common aspect with information security since it addresses the problems of information leakage. Information security addresses the problem of ensuring that the data handled by the system is managed appropriately according to the access privileges. Especially, confidentiality addresses the issue of direct information leakage of the data managed by the system.

On the other hand, privacy addresses the issue of indirect leakage of information related to the data subject from the data output in accordance with legitimately granted access privileges. The indirect leakage of information involves the utilization of auxiliary information or background knowledge that exists externally or in the world. For example, it is recognized that even when protected data is output, re-identification of information related to the data subject is possible by leveraging unexpected auxiliary information.

Those involved in the development and operational management cannot foresee when and how such unintended re-identification may be implemented. Therefore, logs of information related to input and output are collected during in-operations, and are utilized for failure situation analysis if an occurrence of indirect leakage is indicated later.

8.1.2.1.2 Monitoring during operations for AI security

The operator or organization managing the system should monitor the behavior of the model or system during in-operation and examine the strategies to respond to risks. For example, examining the following is preferable.

- Examine using technology to detect and grasp the malfunctions of the model and system during in-operation.
- When the operator collects and processes input data to the system during operations, instead of a preprocessing program, they should consider (i) evaluation of trustworthiness, (ii) Implement measures to prevent or reduce modifications and (iii) use modification detection technology with respect to in-operation input data and data source.

On the other hand, it should be noted that these in-operation monitoring methods may not be capable of detecting and preventing attacks or damage, and the attacks may be carried out by evading these in-operation monitoring methods. In addition, since the cases where the operator or organization managing the system can directly monitor and respond to the risk are limited, it is necessary to consider implementing the "risk monitoring and response program" described in Section 7.12.1.2 and analyzing risk monitoring and response records.

If the system is subject to a security attack, provide incident support based on a framework such as SP800-61, in the same way as general information security.

8.1.2.2 Modification of system

Retain records of the modifications made to the system. Possible cases include when performing additional learning of the machine learning components using new data, discussed in the next section (Section 8.2) and when installing security patches to the system. System modification of can be considered as a partial modification of development, since it will impact various internal qualities that have to be considered during development. Therefore, examining the impact and retaining the records is important.

When performing additional learning of a model, it is necessary to confirm risks related to additional learning data and learning processes, and examine and implement security control measures. In addition, if it is necessary to reduce the opportunities for inputting incorrect data into the system during in-operation, measures such as changing the installation environment of the system should preferably be examined to prevent the system from being used in unexpected environments.

8.1.2.3 Change of external environment

Even if no modifications are made, continuously observe the external environment related to the usage conditions of the system, and record any changes as soon as they are noticed. If there are changes in the source of input data or collection conditions, changes in the usage conditions of the output, and changes in operation conditions such as processing frequency, operation duration and timing, or changes in the business requirement and social demand even if the system does not change, these may be some of the cases where there are changes in the situation to be detected and the criteria for judgment. These will first impact internal qualities A-0, A-1, A-2, and then there is a risk of impacting other internal qualities. If the impact is significant, modification of the system is required. Even if it is not, examining the scope of impact and its contents and retaining the records of the results is important. Even with respect to AI security, it is important to implement security risk assessment in response to changes in the external environment and examine whether the security control measures need to be updated.

8.2 E-1: Maintainability of quality characteristics during operation

8.2.1 General

The term maintainability of quality during operation means that internal qualities satisfied at the beginning of operation are maintained throughout operation. This concept means that

internal qualities can fully respond to changes in operational environments outside the system and that any change in trained machine learning models do not cause unnecessary deterioration of quality.

A specific method to realize quality maintenance depends on forms of operation, especially on how to carry out additional learning or iterative training. The Guideline envisions the following two patterns of additional learning and retraining.

(a) Cases where the results of additional learning are reflected in an operation environment for the first time after going through additional learning and quality tests outside (developmental environment) of actual systems in service (operational environment) through explicit updates of software. Solutions such as automatic driving envisioned now belong to this pattern.

(b) Cases where trained models are updated in real-time during operation and updates are completed without any human tests in an operational environment. This pattern includes online learning used to process streaming data, language recognition and application of conversations in projects in which development and operation are integrated.

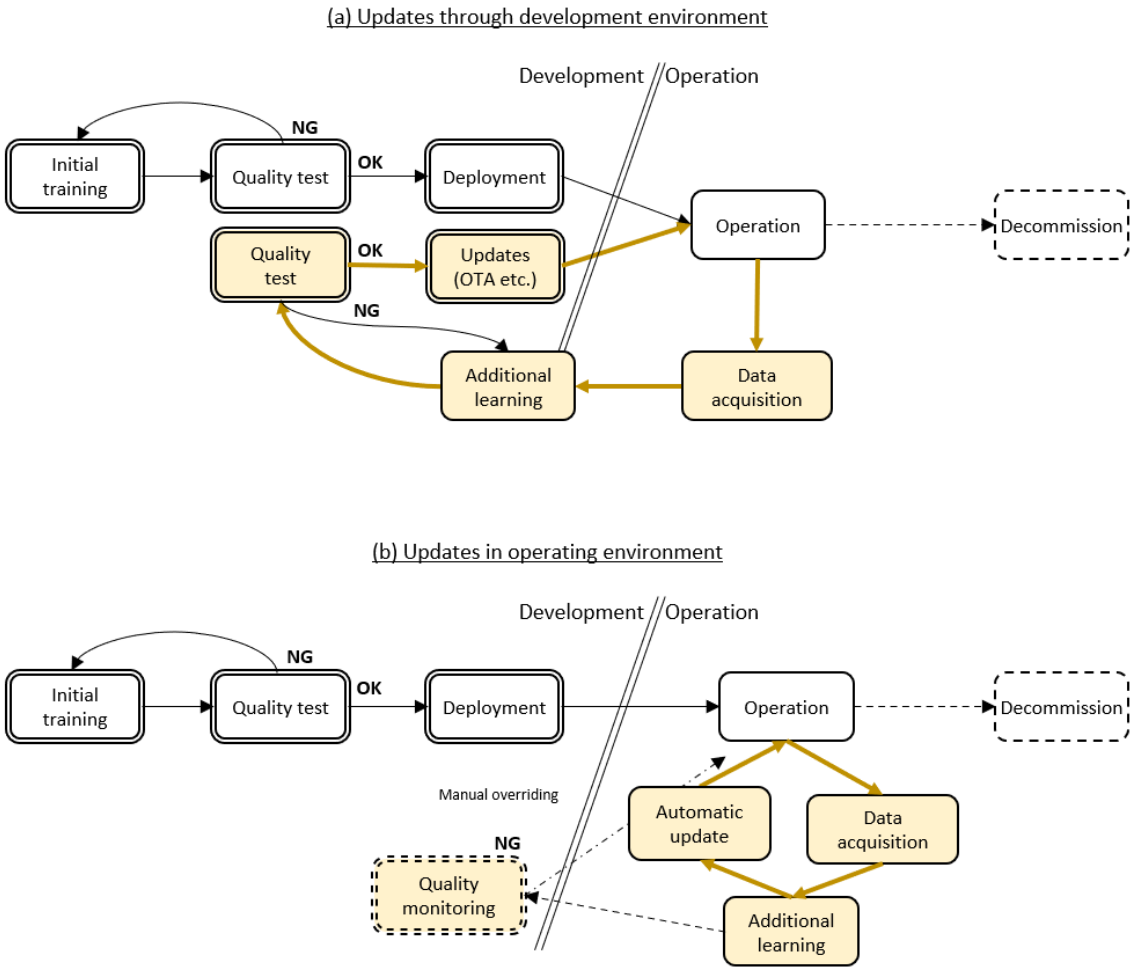


Figure 12: Update patterns for machine learning components during operation

As described in the Figure 12, quality testing is always carried out before updates in the development-environment update pattern described in (a). If the content of tests is the same as those conducted in the test phase of initial development, a certain level of quality can be maintained. On the other hand, in the operational-environment update pattern described in (b), there is a higher risk that a model whose quality has deteriorated is reflected in operation, because updates are fully automatic. In this case, it is important to incorporate a quality monitoring mechanism and a mechanism dealing with deterioration in an operational system.

Moreover, regarding the maintainability of quality characteristics during operation, in addition to the deterioration of overall performance, it may be necessary to pay attention to a problem that a machine learning component makes a misjudgment on specific input after its update[15], although it used to draw out correct answers before the update. We might think that it is enough if the external qualities described in Section 4.1 improve or are maintained as a whole, but in actual industrial fields, it may be difficult to accept that any component implemented and used to operate correctly stops operating properly at a later stage. On the other hand, additional learning inevitably produces different output values from past input due to the characteristics of machine-learning-based systems. Therefore, it is necessary to examine in advance how to handle additional learning in the form of operational guidelines.

Moreover, though the Guideline does not cover directly this, consideration should be given to legal positions of contracts and privacy of data in real environment used for additional learning, when operation is examined. From the viewpoint of quality management, it is desirable to save all data used for training including additional learning and use them to validate and monitor quality and verify performance deterioration when a software component used to give correct answers. However, some restrictions on handling of data may be imposed from the viewpoint of privacy in actual applications. When data, which are prerequisites for quality management when designing a machine-learning-based system, is unavailable during its operation, the overall logic of system quality assurance may collapse. This is why identifying available and reservable data is an important factor to examine an operational system.

8.2.2 **Approaches**

Updates are handled according to the two patterns described above, depending on whether developers or other responsible parties conduct quality testing when the system is updated.

(a) Updates preceded by a quality inspection process

[15] In software engineering, this problem is often called “regression”. The content mentioned in this section is particularly regarded as a “regression test”. However, “regression” is used in totally different contexts in machine learning. Therefore, we the using these terms in the Guideline.

- Examine in advance the frequency of updates or the criteria for determining whether updates are necessary.
- Analyze the update process required during operation in the design stage, envision its outline, and design specific procedures before operation begins. At minimum, the following points should be considered.
  - Methods for collecting available data from the operational environment and deploying such data to the development environment for updates.
  - Methods for preprocessing data for additional training, including filtering and labeling.
  - The range of data used for additional training and model updates, including how to exclude old data, especially when environmental changes are expected over time.
- Before operation begins, examine quality testing methods for updates, especially acceptance criteria for updates or decision-making procedures.
- Depending on the application, determine in advance how to handle cases in which quality deteriorates for particular examples with known correct answers.

(b) Updates made without undergoing a quality inspection process in the development environment

- Consider in advance the possibility of significant quality deterioration caused by additional learning and the range of its impact on the system if such deterioration occurs.
- If the impact may be unacceptable, examine technical or operational mechanisms for keeping risks to the overall system within the acceptable range, and incorporate these mechanisms into operation. Examples include the following.
  - Technically limit the range of output values. Use surrounding system components, such as software, to enforce the expected normal range and prevent deviations from it.
  - Roll back to a previous model version, or stop or suspend operation, based on performance monitoring from outside the operational environment.
- If quality can be monitored within the operational environment or the operational system, such monitoring is useful for quality management (see Figure 13).

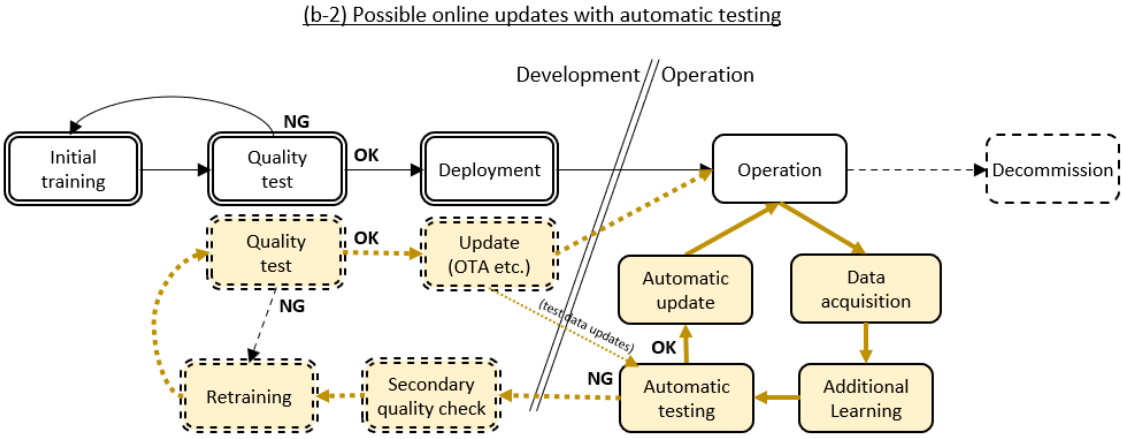


Figure 13: Possible automatic quality monitoring in the operational environment

8.2.3 Requirements for quality levels

The correspondence between external and internal qualities regarding the required level is as follows.

Safety

AISL 0.1: Lv 1 or above
AISL 0.2: Lv 2 or above
AISL 1: Lv 3
AISL 2~4: Requirements to be added to Lv 3 will be considered.

AI performance

AIPL 1: Lv 1 or above
AIPL 2: Lv 2 or above

Fairness

AIFL 1: Lv 2 or above
AIFL 2: Lv 3

The requirements for each level of internal qualities are as follows.

- Lv1
  - ➢ Examine in advance how to respond to notable system quality deterioration caused by changes in external environment.
  - ➢ In the case where online learning is given, examine in advance the impact of unexpected quality deterioration and take measures from the system side such as the limitation of operation range if necessary.
  - ➢ Quality control in line with regular learning should be performed when additional learning is given offline.

- Lv2
  - Monitor system quality deterioration and misjudgments by comparing it with operation results within the range permitted by system use. It is necessary to sufficiently examine factors other than product quality, such as privacy, during monitoring.
  - When online learning is performed, regularly monitor additional learning results by any method. When monitoring reveals any deviation from the performance requirements, taking immediate action should be possible.
  - When additional learning is given offline, conduct regression tests on quality deterioration with test datasets used in the system development stage to check if the quality has been maintained before updates. Update test datasets using the same method used in the system development stage where necessary.
- Lv3
  - Make sure to establish measures for monitoring system quality, including an operational system, compatible to privacy.
  - When online learning is given, before the results of additional learning are reflected on systems, implement a mechanism to check quality to some extent within those systems so that updates are suspended if it becomes impossible to ignore unexpected quality deterioration. Make sure to ensure measures for making updates and modifications offline.
  - When additional learning is provided offline, the quality should be managed using data collected from operations, test datasets used for the initial system building and test datasets updated regularly using the same method.

9. Technologies for quality management (reference information)

In this chapter, individual methods, techniques and tools that may be used for the management of internal qualities are listed as reference information.

9.1 A-0: Sufficiency of preliminary problem analysis

9.2 A-1: Sufficiency of problem domain analysis

A problem domain analysis for machine learning called ML problem domain analysis in the Guideline is a step in the development process. Diversity of data input to machine learning components is analyzed from various viewpoints in regard to the real-world situations, in which machine learning systems or services are used. In particular, the analysis aims to concretize the identified risks or divergence in system requirements as attributes of data employed for training and testing of the machine learning products. The analysis activities are basically similar to the risk analysis, hazard analysis, or problem domain analysis, all of which are subjects in Systems Engineering. These existing technologies, or the body of knowledge, can be helpful in conducting the ML problem domain analysis.

The ML problem domain analysis, viewed from Software Engineering, is a part of initial stages of software development processes, in which software engineers often rely on trial-and-error styles of the development. The steps in the Guideline are basically meant to establish information necessary to achieve the required quality levels of machine learning products. These require activities such as what data scientists conduct in developing PoC systems with their informal know-how concerning the quality aspects of machine learning products.

9.2.1 Initial hints

Recent standard textbooks on Software Engineering (e.g., [171]) may present an overview of the analysis methods/steps employed in software development processes. Because safety analysis, or safety, is one of the primary foci in the system analysis steps, several modeling notations or methodologies have been proposed and studied. They include Causal Loop Diagram and STAMP/STAP [241]. Furthermore, the methodologies have detailed notations such as Fault Tree Analysis, Fault Mode and Effects Analysis, Loop Diagram, and Feature Tree. In the Guideline, such results are concretized as data characteristics used to build, in particular, machine learning components. Therefore, the Guideline recommends that Feature Tree is suitable as the tool for the final artifacts obtained using the other modeling notations.

There might be two major problems if the analyses follow the mentioned approach: (a) modeling of input incidents, especially of risk factors, and (b) design of the problem structure to be concretized as data characteristics. This section mainly refers to concepts specific to machine-learning-based systems and machine learning components from these two viewpoints.

9.2.2 Modeling of risk factors in input space

Identifying risk factors mostly involves trial-and-error activities without any systematic means, and thus, it may often be conducted through brainstorming sessions by various stakeholders. It is, indeed, an analysis process regarding Known –Unknowns. How well the risk identification is done is difficult to exhibit with quantitative measures, but it may be confirmed only through the inspection by human experts.

As an example of such brainstorming activities, the NASA Hazard Analysis Process, published by the National Aeronautics and Space Administration (NASA) [100], presents an outlined list of what should be considered in the early stage of the hazard analysis.

(1) Standard Hazard List
(2) Historical experience/documentation from legacy systems
(3) Engineering training and experience

This literature also presents the NASA Generic Hazards List comprising 26 items as the initial (1) Standard hazard list. Although these items include some hazard causes specific to particular applications, not causing any problem in the others, this list can be general purposed as the causes of hazard are associated with mechanical operations of systems operating in outdoor environments. The list serves as a starting point for analyzing other systems.

Moreover, (2) is similar to what data scientists conducted in the past in order to build machine-learning-based systems. Since findings on similar systems in the past or early versions of the same systems are valuable, it is desirable to be utilized actively. In addition, a PoC system is recommended to act as a testbed, helping the PoC developer understand the characteristics of final machine learning products. Finally, (3) can be considered a form of utilizing domain knowledge. It would be desirable to incorporate into the body of knowledge for the users and the vendors involved in the machine learning software business. The NASA publication promotes conducting the analysis activities based on these viewpoints and concludes that hazard analysis requires rationale to justify hazard classification.

The Guideline recommends that developers or engineers model risk factors using the above three viewpoints, together with findings collected from the PoC stage and record precisely the modeling process and modeling results. Specifically, the followings are compiled into an integrated list of identified risk factors.

- (1) Utilization of basic knowledge on the existing functional safety design, the existing

hazard lists in each application domain, or brainstorming results based on the above NASA Hazard List;

- (2) Utilization of analysis cases on prior systems and similar machine-learning-based systems;
- (3) Introduction of domain knowledge by brainstorming with users (e.g. entrusters in contracted developments); and
- Knowledge about data employed preliminarily and exceptional cases identified in the trials of the training in the PoC stage.

9.2.3 Designing problem structures as data characteristics

This stage of the development process produces, as an output artifact, a piece of information on the attachment of attributes to data used for machine learning. It is affected inevitably by the characteristics of the next process, machine learning and training. Considering the training process that follows, the main activity of the process is to identify attributes which are handled distinctively throughout the machine learning process, which consists of the following two aspects.

(4) Problematic if data are not recognized as distinct with respect to the differences in the output result values or risks; and

(5) Problematic if data, resulting in the same output results, are likely to be understood as different due to their input values and characteristics of machine learning models.

(4) is drawn mainly from the system specifications and the hazard analysis. A difference in output values is a difference in labels of training data, and risk incidents with different levels can basically be enumerated once the cause of hazard is identified.

On the other hand, (5) is dependent on application domains, for example, depending on differences in background images or character fonts for image recognition tasks. Moreover, (5) may be affected by preprocessing or architecture network of machine learning models. It is necessary to forecast a final implementation form to some extent in order to conduct such an analysis. This is called Implementation Forecast in software engineering because the information relating to the final implementation is indeed necessary to conduct the analysis at the early stage of the development. Specifically, the activities are concerned with the implementation consideration made in the PoC development stage, and the analysis of PoC behavior using data.

In order to extract viewpoints with which the forecast is actually carried out, the three points listed in the previous section are considered applicable in view of *a risk that AI built makes different judgments in similar situations*. In the future, it would be preferable to combine (2) and (3) in the previous section and (5) in this section to accumulate system analysis results for each application area to some extent and compile them as detailed standard hazard lists in each area.

Even as the author of the Guideline, I want to actively consider the development of related documents from this perspective in the future.

9.3 A-2: Sufficiency of data design

9.3.1 General

The internal quality in this section has two purposes, which follow the analyses described in in the previous section. First, it prepares data needed at the time of training and testing to avoid an unknown situation in which a machine-learning-based system has not been learned or tested. Second, it either decimates or integrates systematically an enormous number of combinations of prepared attributes so that building machine learning software of a required quality level is practical.

If a system intends to solve a simple problem and the number of combinations of attributes is a few dozen, it is possible to consider the amount of data that achieves a certain recognition level for each attribute. (e.g., Only 10 characters x 2 fonts need to be recognized in number recognition and there is no need to take into account differences in paper quality. Then, there are 20 combinations.) If each attribute has such volume of data, sufficient and comprehensive training can be practiced. However, since the number of combinations of attributes often exceeds several tens of thousands if calculated simply, it is not practical to prepare data representing all combinations for testing. In a certain case, training data or testing data may be a few or even null due to breaking up attribute combinations. It would then be impossible to rely on the Law of large numbers to make a judgment such as the convergence of training.

Combination testing is a method developed in conventional software engineering to mitigate the complexity problem. The method employs the notion of test coverage and establishes the degree to which combinations of attributes are detailed and the degree to which combinations of test conditions are covered, yet devises them separately. The method is often used to evaluate test data in conventional software testing and is also expected to be effective in the case of machine learning because training is data-centric.

For example, [75] presents examples of applying the combination test technology (t-way) to evaluate datasets for machine learning.

9.4 B-1: Coverage of datasets

Establishing coverage of datasets is an extremely difficult but important problem in the quality management of machine learning. A dataset, if a certain anomaly is associated with it, may derive trained machine learning models to exhibit unstable or unfavorable functional behavior in a

certain specific envisioned situation, lowering fairness levels.

Since the differences in "language-enabled" situations has already been covered in the "Sufficiency of Requirements Analysis", the minor differences in other situations should not be missed when constructing the data here.

The discussions in this section are mostly related to the preliminary stages of data preparation or data evaluation in the development lifecycle. Auxiliary checks may sometimes be conducted in the software testing stage.

9.4.1 Data acquisition plan

A data acquisition plan in the data preparation stage is a basis to study the issues relating to the coverage of dataset.

In order to construct a dataset that is unbiased considering the circumstances during operations, the plan should refer to the range, period and size of data collection. Specific plans should be considered for each application following outcomes of the brainstorming sessions, which may be those activities, in the stage of analyzing domain analysis problem, to utilize prior knowledge about the similar applications so as to examine in advance the degree of diversity.

9.4.2 Pre-flight tests in the data review stage

Auxiliary checking of data distribution belonging to certain attributes is desirable in some cases. These attributes are what, although identified as candidates in the early stages of the analysis, may not be adopted, or what may become latent, because they are composed to be a new attribute. The composed attributes are accompanied with explicit specifications, and in some cases those checks may be conducted mechanically, because data labels are given.

This is not perfect, because attributes overlooked in the brainstorming sessions cannot be recovered. It is worth considering.

Moreover, for some data, what is called data specific feature may be identified in the preprocessing stage, which makes it possible to check the distribution.

9.4.3 Additional tests in the testing stage

In a case where the data distribution has some problems, what can be re-checked in the test stage is limited. For instance, latent correlations may exist between the adopted features and overlooked unidentified ones. In such cases, analyses of correlation or influence between the input values and their resultant counterparts' output from the trained machine learning model may be helpful to see whether the model behaves as expected. The model may make inferences

based on features different from what is assumed. It is worth considering this analysis when extreme safety is a major concern.

9.5 B-2: Uniformity of datasets

The uniformity of datasets is equivalent to considering the coverage of datasets, described in the previous Sectio, as a single case of all the input data. The measures listed in each section in Section 9.4are deemed applicable here as well.

9.6 B-3: Adequacy of data

9.6.1 Quality control cycle from the perspective of data

The Guideline mainly examines the lifecycle process from the perspective of the development of machine learning models, but from the perspective of data quality, the lifecycle process of establishing consistency between data collection policies and actual data is also important. From the standpoint of data, there may be a recursive process iterating the following steps.

(1) Determination of data collection policy
(2) Collection and selection of data in accordance with the policy
(3) Validation of both the policy and data

In (1) determining the data collection policy, the following are necessary:

- Formulation of a data collection policy that is appropriate for the quality of the target machine learning system
- Advance consideration of the acceptable range of data validity
- Determination of the data validity evaluation method in advance

In the process of (2) data collection and selection, we perform the following according to the given policy:

- Determination of the appropriateness of the selection and collection method,
- Process management of the actual selection and collection methods
- Collection and consolidation of evidence on selection and collection methods
- Management of meta-information about the source and traceability of data

In (3) validation, we perform the following:

- Verification of data in accordance with the policy
- Review of the explanatory nature of the policy itself
- Feedback to the policy based on the collected data

And the process continues back to (1) until a sufficiently satisfactory and valid policy and data pair are obtained. Furthermore, this entire process is subject to quality management and assurance, such as:

- Evaluation of the completion of the process itself
- Evaluation of the skills of the people in charge of executing the process and their appropriate assignment
- Evaluation of the mechanism for managing the entire process and securing of evidence

Regarding the development lifecycle described in Section 3.8, such a cycle of data policy improvement is abstractly organized as a repetition of trial and error in the PoC process, However, in reality, sometimes the improvement cycle described above is repeated even in actual development process, leading to agile developments. In such a case, it is particularly important to assess whether the requirements definition has been updated together with the data collection policy, or whether it is necessary to reconfirm the contents identified in the previous stages, such as internal quality A-1 to B-2, etc., in accordance with such updates, to ensure the overall quality. It is also extremely important to ensure that such policy and requirement updates are subject to change management to reconfirm and verify quality later on.

9.6.2 Technical support for organizing outliers and corner cases

There are several techniques that can be applied to efficiently extract and organize data that require data validity judgments either as outliers or as rare cases to be addressed.

First, DeepXplore [185] uses the Neuron Coverage technique for corner case extraction. Surprise Adequacy [136] provides a measure of the surprise of the inputs to a machine learning model, which also allows corner cases to be extracted from the input values. Tinghui et. al [179] improved on Distance-based Surprise Adequacy to perform corner case analysis. Whether the data extracted in this way should be removed as outliers or treated as important rare cases is to be determined on a case-by-case basis in conjunction with policies.

Section 5.3 in Zhang [238] and Section 2.3 in Dong [103] are also helpful.

9.7 B-4: External-quality-dependent adequacy of datasets

9.7.1 B-4pr: Adequacy of datasets in terms of privacy

Technical methods, such as protection processing of individual training data, control of data distribution in training dataset, and protection processing of the entire dataset, are to be combined.

9.7.1.1 Data protection pre-processing

First, is a method of examining attributes (components of multidimensional vectors) and removing sensitive attributes that can be associated with data subjects. In general, multiple data are processed so that they are mapped to the same value of quasi-identifier; a quasi-identifier is introduced from the sensitive attributes. Alternatively, data with different attribute values can be pre-processed to be indistinguishable by replacing them with abstract values through generalization, aggregation, etc., considering the meaning of the attributes. For example, the attribute of birthday is pre-processed to have the year and month information only, discarding the day.

Last, K-anonymity is a systematic approach to obtain K or more similar data [195] and [212]. However, machine learning training data are multidimensional and sparse, which makes it difficult to protect data with K-anonymity [67]. Therefore, problem-specific measures should be taken according to the characteristics of the data to be protected.

9.7.1.2 Detection and handling of outliers

The training process does not achieve removal of specificity sufficiently, and thus trained models are not anonymous. As a result, re-identification attempts of the training data from the trained model are sometimes successful. The reason for this is "Memorization" of the label of each individual training data in the trained model [237] and [87].

The memorization problem can be summarized as follows. Compare the certainties of the predicted labels obtained when the input x is given to the two trained models M and M' where M is derived from dataset S containing data points <x, y> consisting of data x and label y, and M' is derived from another training dataset S' with this data point removed (S' = S\{<x, y>}). In the case of label memorization of training data, the result of the former M(x) is more certain than M'(x).

The “memorization” phenomenon appears as a result of the training mechanism leading to poor generalization performance or overfitting and also appears in those trained models derived from the inadequate data distribution in the training dataset [149]. In addition, it is known that outliers are easy to be memorized. Therefore, it is important to detect outliers in a given dataset and to examine how to handle them. Outliers are defined based on the assumption of data distribution (statistical data model), but expressing the data distribution concisely is difficult for training dataset. It needs to devise methods specific to individual cases for the detection [68].

9.7.1.3 Privacy-preserving synthetic data

When the training data is image data, there are methods to generate privacy-preserving data or Privacy-Preserving Synthetic Data (PPSD). These are used to synthesize a dataset protected by differential privacy from an original training dataset [232]. Such a method uses a generative model trained on the input dataset to synthesize new data that follow the same data distribution as the input dataset [191]. Basically, the generative model training employs privacy-preserving machine learning methods (DP-SGD) with machine learning architectures such as GAN and VAE. After obtaining the dataset of PPSD, the original input dataset is to be discarded. The generated model and protected training data are to be maintained.

Since the privacy-preserving synthetic data method usually employs privacy-preserving machine learning, the reproducibility of the data distribution of the original dataset may not be good compared to regular GAN or VAE. In order to overcome the disadvantage, it needs to be dependent on some empirical techniques considering the characteristics of the target image. Furthermore, the resultant generative model itself is known to have the possibility of unexpected "Memorization" of the training data. Therefore, appropriate security measures should be implemented as well to prevent the stealing of the generative models when they are maintained afterwards.

9.7.1.4 Note on Data Quality in Large Language Models

Large language models (LLM) are neural language models based on the idea of foundation models (Section 3.9.2). The problem of privacy will appear in a slightly different form than what was discussed up to now. LLM is a pre-trained model derived using a large corpus of natural language sentences as training data. LLM will become the foundation to construct various natural language processing application systems as downstream tasks, following the transfer learning method (fine-tuning). In addition, advances in generative LLM technology have reached a practical stage where chatbots are used in practice, taking the form of a type of Q&A system that outputs appropriate responses to input text fragments (prompts).

When the corpus used as the source material contains contents that are contrary to social justice or relate to personal information of data subjects, it is known that the output of the chatbot may raise fairness issues [163] or privacy issues [83][163]. One of the reasons for this is the "Memorization" of the training data in the LLM [87]. Therefore, research on alignment is extensively conducted to adjust LLM so that it does not generate inappropriate information. However, the current alignment method is not satisfactory in view of human requirements. It needs research from various perspectives, including software engineering practices to properly configure the final application system incorporating LLM as a black-box reusable software

component.

Since LLM reflects corpus information obtained from various external sources, LLM can be a platform for implementing various natural language applications. One of the problems with LLM is *unintentional memorization* [88] In general, eliminating the threat to privacy completely is not possible, unless a corpus that does not contain sensitive information about any data subjects is used [83]. It can be said that the level of threats to privacy issues is determined at the stage when the corpus, is chosen, that is, when the training data to be used is determined. In practice, a large corpus is used for training LLM, and it is nearly impossible to make sure whether its contents are appropriate afterwards. Thus, a corpus with clear evidence of its provenance should be chosen.

9.8 C-1/C-2: Correctness and stability of trained models

Checking the correctness and stability of machine learning models is a task corresponding to unit testing in conventional software testing. It is because the check is focused on each machine learning component. In this section, we will introduce the following topics.

- Basics of software testing techniques [71], the aspects to be taken into account depending on the characteristics of machine learning components, and recent research works on machine learning component testing
- Potential of technology for testing of stability directly

9.8.1 Testing machine learning components

9.8.1.1 Characteristics of test target

First, there are two distinctive test targets to consider when discussing the quality of machine learning component results: training and learning programs that input training datasets to obtain trained models, and prediction and inference programs of which the resulting trained models specify functional behavior. Second, there is no clear criterion for deciding whether a computation result is correct, which is called test Oracle problems [76]. For a training and learning program, it is difficult to know in advance the correct value of its computation result, that is, a trained model. In addition, the answers derived by prediction and inference programs are not definite but accompanied by probabilities referring to certainty degrees, which illustrates difficulties to define absolute criteria for correct answers [106]. In general, machine learning programs are classified as non-testable programs [229].

9.8.1.2 Test oracle problems

Software testing technology consist of test input generation methods and test execution result confirmation methods. The first is how to generate test inputs that efficiently achieve the testing purpose. There are some methods available to obtain new test inputs based on a piece of information that quantitatively represents the part of program to be tested, which makes use of a test coverage index. As will be introduced later, there are many research on software testing methods for machine learning components.

The second is concerned with methods for comparing the results of program execution with known correct answers to a given test inputs and is collectively referred to as test oracle techniques. A non-testable program is one in which the correct answer to a given test input is not known, and derived oracles, pseudo-oracles, or partial oracles have been used in its software testing so far.

Metamorphic Testing (MT) [93] is a practical method of partial oracles. It was introduced as a testing method for programs in which it is difficult to know in advance the correct value of the computation result for an input, such as numerical computation or translators including compilers, and was later applied to the testing of operating systems and security systems to use implicit oracles [94]. It is now a standard testing methodology for machine learning components [166]. For some testing purposes, a combination of partial oracles and statistical testing can be useful [168].

9.8.1.3 Two perspectives on testing

In general, when checking a program, positive testing and negative testing are conducted.

Positive testing is to confirm that the test target exhibits the expected functional behavior. For the sake of explanation, let's assume that a program has pre-conditions and post-conditions. The relation is established such that when the pre-condition is satisfied, the post-condition is satisfied in the state after the program body is executed. When the above relationship holds for test input data that satisfies the pre-conditions, the program under test is considered to pass the test.

Negative testing checks whether a program will not fail catastrophically under unexpected conditions. In general, a program is expected to behave reasonably even in response to input that does not satisfy its pre-conditions. For example, the program should not run out of control and terminate abnormally. Checking these cases does not need an application-specific correctness criterion. Such test oracles are called implicit oracles, meaning that there is no explicit test Oracle at all.

Fuzz Testing or fuzzing [157], which uses randomly generated data, while satisfying some

constraints, as test input, has been known effective for integration testing of open system software such as operating systems or security systems. This method is sometimes used in negative testing of programs. An appropriate fuzz, a test input data, is generated to meet each test objective.

In testing for machine learning components, positive testing is, in a sense, related to checking the accuracy. We can use data that follow the same statistical characteristics as the training dataset. On the other hand, negative testing includes checking the accuracy in regard to the generalization performance and checking behavior against unexpected inputs in the case of checking the stability or the model robustness. Stability testing includes the case where the input is either corrupted data or adversarial examples. As described below, various fuzzing techniques have been proposed for testing machine learning components, depending on the purpose and perspective of the test.

9.8.1.4 Test coverage metrics

In machine learning component testing, it is important to know an extent to which a target trained model becomes activated for a given test input. The idea corresponds to the test coverage criteria in conventional software testing methods. Neuron Coverage (NC) [185] is, an early proposal for such a structural metric, defined by a frequency of activated neurons. Later, in order to use more detailed information than NC, several coverage metrics are proposed [151]; those are based on structural patterns over activated neurons, such as correlations between activated neurons in different layers. If these structural coverage values are large, it can be assumed that a wide range of the learning model is activated, and thus is checked as well. The test coverage is considered sufficient and thus we may conclude that such test inputs are useful.

In general, it is difficult to define the test inputs used for negative testing which do not satisfy pre-conditions, because the statistical characteristics of training data, which might be regarded as such pre-conditions, are not explicitly stated for machine learning components. Surprise Adequacy [136] provides ways to check how test inputs deviate from the statistical characteristics of the training data. Although the adequacy metrics are originally introduced for evaluating quality of test inputs, they are also considered to be a kind of indicators referring to degrees of being outliers. Note that these indicators refer to the internal activated states of trained models, and thus that measuring these requires monitoring of execution states of trained models.

9.8.1.5 Automatic generation of test inputs

In general, it is important to establish a method on generating appropriate test input data for

each test target. In addition, such an appropriateness depends on the testing purpose, either positive or negative. When the test target is a training and learning program, the input is a collection of data (a dataset), and the notion of dataset diversity [164] provides a guideline for test input generation methodologies for both positive and negative testing.

In testing of prediction and inference programs, various types of data that were not considered during training stages are expected to input for checking prediction results. The method is sometimes called Test-time Augmentation. For generating test data, there have been various approaches including those to adopt machine learning techniques. They include conventional Data Augmentation methods for image data [140], or methods using Generative Adversarial Networks (GANs) [116]. Furthermore, some methods make use of the adversarial example generation method [194] [240]. In the case of generating a large number of test data, the aforementioned dataset diversity is also useful as a conceptual guideline.

9.8.1.6 Test generation and coverage metrics

In generating test inputs, it is desirable to find useful data that can be used to conduct testing efficiently depending on test purposes. As in conventional software testing techniques, we apply some metrics to control the process of generating data that, for example, improves the structural coverage criteria. This method is collectively called Coverage-Guided Test Generations. There are some research works that use the NC when machine learning components are test targets. The NC is combined with the classical data augmentation method [215], or with the GAN-based data generation method [239].

Some recent works report that the structural coverage may not be useful depending on the purpose and perspective of the test [119]. For example, the correlation between NC and adversary robustness is known to be small [224]. The NC values are more highly correlated with model capacity than with defects in trained models[245, Chapter 4]. For example, experimental results show that the accuracy can be high even if the NC is small, and conversely that the NC value can be large even when the accuracy is poor. In addition, it is known in conventional software testing techniques that the structural test coverage indices have a small correlation with the effectiveness of test suits or the efficiency of detecting faults [124]. This empirical report results seem to be applicable to machine learning components as well [165]. As alternatives to the structural coverage criteria, a new method is proposed that is based on an error function to be used as metrics[245] and [168].

Although identifying data for executing corner cases (edge cases) is desirable for effective testing, the relationship between inputs and corner cases is not clear in machine learning components. Therefore, it is necessary to consider a metric that replaces the structural coverage criterion. For example, the method using the error function is used for evaluation metrics [224].

9.8.1.7 Prioritization for Preparing Training Data

In conventional software testing, regression testing involves a large number of test cases. Selecting test cases that are useful for detecting faults early can improve the overall efficiency of test activities. Therefore, when a large number of test cases are available, prioritization is performed [193]. Choosing metrics is essential so as to find useful subset of the test cases.

For machine learning, the prioritization ordering techniques can be utilized from the perspective of improving efficiency of task for preparing training data, especially of labeling task[85]. We consider here a task of augmenting an existing training dataset when a lot of data without any tag is available. First, candidate data are picked up in a certain way so as to be put into a data pool. Second, labeling data, selected from the pool, with a tag is done manually, which is time-consuming to require a lot of human work if a number of target data is large. Therefore, if we select data that have characteristics different from the existing training data and label these data only, we can obtain useful training data while reducing the amount of work.

Technically, the method is essentially deciding whether data is useful or not, and it comes down to another problem of devising appropriate indices to be used for the decision. Recent research works propose a number of methods using either external metrics [111] such as Confidence and Gini Impurity, or internal metrics [85]. Choosing an appropriate metric should take into account what different features are desirable for data in regard to the existing training data.

9.8.2 Technologies for addressing stability issues

The technologies on stability issues are broadly divided into the evaluation and improvement. Figure 14 shows how each technique is related to the level described in Section 7.9.3. The techniques are explained in the technical report[246] (an annex of this guideline). Especially, about adversarial training, randomized smoothing, adversarial attack, and adversarial verification, read Chapter 7 in the report. About generalization error and noise robustness, read Chapter 8. About adversarial example detection, read Chapter 9.

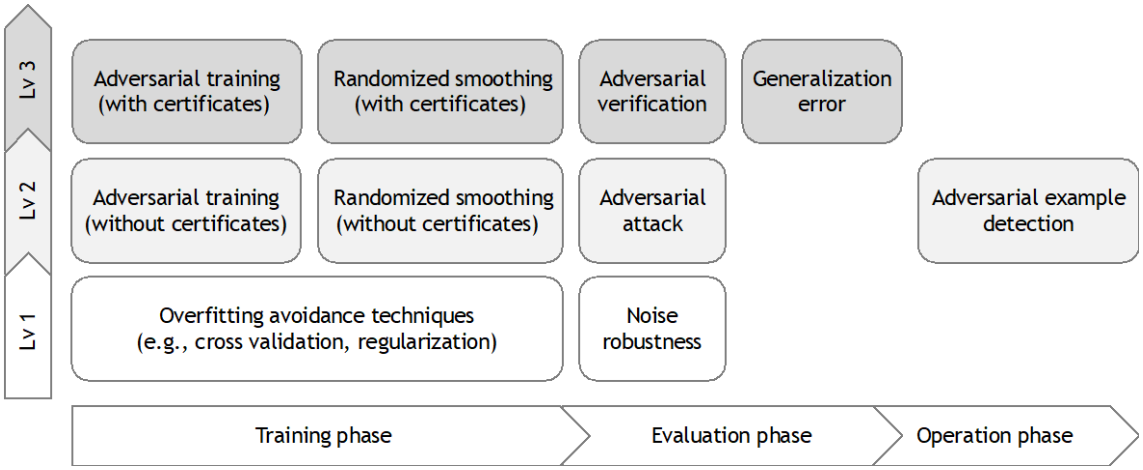


Figure 14: Technologies for evaluating and improving stability, by phase and level of application

9.8.2.1 Overfitting avoidance techniques

Various techniques for avoiding overfitting have been presented and are recommended to be applied at Lv 1 or more. Some well-known techniques are introduced below.

- Cross validation is a classical technique to mitigate overfitting by dividing a whole dataset into training, validation, and testing datasets [139]. For example, in K-division cross validation, a whole dataset is divided by K, and $1/K$ of the dataset is used for validation and the remaining $((K-1)/K)$ datasets for training. The roles of datasets, either training or validation are interchanged during the training process.
- Regularization is a technique to mitigate overfitting by suppressing the absolute values of learnt weight parameters not to become excessively large [175]. For example, loss functions may include a regularization term which is the sum of the square value of every weight parameter.
- Ensemble learning is a technique for considering (e.g., average or majority) of outputs from multiple trained models [128]. The ensemble can improve the accuracy even though the accuracy of some models is low.
- Dropout is a technique for randomly excluding some neurons from the training target during the training process [209]. Dropout can give similar effects to ensemble learning where multiple models are trained simultaneously.
- Early stopping is a technique for stopping the training process before overfitting begins by monitoring the states of the training model. For example, overfitting can be avoided by stopping the training process when the value of loss function for a validation dataset stops decreasing and begins increasing.

9.8.2.2 Adversarial training

Adversarial training is a technique for learning a dataset and adversarial examples generated from the dataset. For example, the robustness for adversarial examples can be improved by detecting data which may cause misclassification in the training process and also learning such data [153]. In addition, some adversarial training techniques can certify that there is no adversarial example near some training data samples in the trained models [230]. The adversarial training is recommended to be applied at Lv 2 or more, while the adversarial training with certification is expected to be applied at Lv 3 in the future because it needs expensive computation cost and is still in the research phase.

9.8.2.3 Randomized smoothing

Randomized smoothing is a technique for smoothing decision boundaries by adding noise to trained models and computing the expected values (means) of outputs with respect to the noise distribution as the final outputs. The technique is reported to improve the robustness for adversarial attacks [147]. In addition, conditions of noise-addition have been also presented for guaranteeing that adversarial attacks do not affect [98][142]. Although multiple inferences for each input are necessary for computing the means of outputs and there is often a tradeoff between accuracy and smoothing decision boundaries, if high stability is required then randomized smoothing is recommended to be applied at Lv 2 or more while randomized smoothing with certification is recommended to be applied at Lv 3.

9.8.2.4 Noise robustness

Noise robustness is an ability that trained models keep correct output even when noise (disturbance) is added to inputs (or weight parameters). For example, a noise robustness can be evaluated by the accuracy of a model for a test dataset to which Gaussian noise with various standard deviations is added [219]. The noise robustness is also useful for deciding the level of noise when the randomized smoothing explained in the item 9.8.2.3 is applied. The noise can be applied not only to inputs but also to weight parameters for evaluating their sensitivity. The (random) noise robustness is easily measured and can evaluate the stability of trained models at a different perspective from the adversarial attack explained in the next item 9.8.2.5 [219]. It is recommended to evaluate (random) noise robustness at Lv 1 or more.

9.8.2.5 Adversarial attack

Adversarial attack is a technique for generating adversarial examples and can be used for evaluating the defense-ability of trained models. For example, robustness for adversarial attack can be measured by adversarial perturbation instead of random noise [219]. In general, such adversarial attack can more explicitly evaluate the stability because adversarial examples can significantly degrade accuracy than random noise. In addition, the minimum distance (called *minimum adversarial distortion* or *maximum safe radius*) between an original data sample and the *nearest* adversarial example generated from the original one can be used as a measure of robustness because the large maximum safe radius means that adversarial examples cannot be generated by small perturbations. Some techniques (e.g. [89]) for searching an almost-nearest adversarial example have been presented even if it is not certified that the adversarial example is nearest. Adversarial attack is recommended to be applied at Lv 2 or more.

9.8.2.6 Adversarial verification

In adversarial verification, it is required to estimate a (ideally maximum) safe radius, such that no adversarial example exists if the perturbation is less than the safe radius. Although techniques (e.g. [134][216]) have been presented for computing the maximum safe radius with the help of SMT solvers, the computation cost of the maximum safe radius is very expensive and therefore it is difficult to verify large neural networks. Then, approximative techniques (e.g. [228][80][231]) have been also presented for computing a safe radius, which is a little smaller than the maximum safe radius. In addition, probabilistic techniques (e.g. [227]) have been also presented for computing safe radius with a confidence (e.g. 99%) less than 100%. Although adversarial verification is still in the research phase, it can give certification to the evaluation and therefore it is expected to be applied at Lv 3 in the future.

9.8.2.7 Generalization error

The generalization error is the expected value of the error rate for any input data randomly selected according to the input data distribution. In general, it is difficult to measure the error rate because the innumerable many input data exist. Then, techniques (e.g. [219]) have been presented for estimating the generalization error bound e such that the generalization error is less than $e\%$ with the confidence $p\%$. As explained in Chapter 8 of the annex[246], by simultaneously estimating generalization errors and noise robustness, the stability can be evaluated at a different perspective from evaluation by test datasets. Generalization error

bounds can statistically guarantee the ability of trained models for unseen data, and therefore are recommended to be estimated at Lv 3.

9.8.2.8 Adversarial example detection

Techniques (e.g. [152][233]) have been presented for detecting adversarial examples from input data with high confidence during operation. Even though adversarial training explained in the item 9.8.2.2 is applied, it is difficult (almost impossible) to remove all the adversarial examples from training models, and therefore adversarial example detection during operation is useful for avoiding misclassifications. If high stability is required, then adversarial example detection is recommended to be applied at Lv 2.

9.9 C-3: External-quality-dependent adequacy of trained models

9.9.1 C-3pr: Adequacy of trained models in terms of privacy

9.9.1.1 Learning methods to mitigate overfitting

Membership inference for machine learning components is the problem of checking whether a data point $\langle x,y\rangle$ was included in the training dataset S ($\langle x, y\rangle \in S$) from a piece of information obtained by executing a trained model M. There are black box methods based on the classification probability vector Px of the execution results of M(x), and white box methods that use information from loss function $\ell(Y(W;\ x),y)$ computed in the course of executing M(x). Intuitively, the method of membership inference is based on the observation that the distribution of Px or $\ell(Y(W;\ x),y)$ differs depending on whether or not a data point is included in the training dataset.

The reason for such differences in the distribution of indicators is “memorization”. Additionally, it is known that overfitting to the training data significantly affects the basic prediction performance (correctness and stability) of machine learning components and thus makes membership inference possible. Therefore, learning methods that utilize techniques such as regularization and dropout, which have been studied in the past, are used for mitigating the problem of overfitting.

9.9.1.2 Privacy-preserving machine learning

Training of deep neural networks is formulated as a numerical search method for nonconvex

optimization problems. A standard method is the stochastic gradient descent (SGD) algorithm, which searches, in an iterative manner, for appropriate values of learning parameters.

DP-SGD is a privacy-preserving algorithm to combine SGD with (ε,δ)-differential privacy. When updating the learning parameter values in the search process, Gaussian noise determined from the given protection strength ε is added [64]. The training parameters or trained model obtained by DP-SGD are protected by the strength represented by the accumulated protection strength calculated in the iterations. As a result, the threat of training data identification can be reduced.

9.9.1.3 Combination with Federated learning

Federated learning is a method of applying client-server distributed computing architecture to machine learning. A large training dataset is segmented into multiple datasets, and for each segmented dataset, training is conducted on the client using the segmented dataset as input. The intermediate training results obtained from this client-side computation process are aggregated at the server. The client-server collaborative process is repeated to obtain the final trained model [130], [79].

When applied to privacy-preserving machine learning, a client is prepared for each training data related to a particular data subject, and a machine learning method with differential privacy, such as DP-SGD, is adopted for the training mechanism at the client side. Since only the client processes the personal data associated with the data subject, it can be expected that the data subject's information will not be leaked to the outside such as the server. In addition to the client-server type of federated learning shown above (horizontal federated learning), there are also vertical federated learning and federated transfer learning, for which privacy-preserving is considered.

9.9.1.4 Caution on using privacy-preserving machine learning

The privacy-preserving machine learning method is an application of (ε,δ)-differential privacy. In general, the strength of protection depends on the value of ε. Therefore, whether the protection is achieved as expected depends on the ε value chosen. In other words, just using privacy protection learning methods does not necessarily mean that adequate protection is achieved. In addition, increasing the protection strength (by decreasing the ε value) will worsen the prediction performance of the trained model that is obtained, lowering its usefulness. There is a trade-off relationship between protection strength and usefulness. Furthermore, establishing a general way to determine the appropriate ε value is an unresolved issue. Decisions are often made experimentally through trial and error.

If a training dataset is biased in the first place and the trained classifier obtained from it produces prediction results that are unbalanced of large variability, then using DP-SGD together with this biased training dataset will further increase the variability [74]; the bias in the prediction results may be amplified. For example, if DP-SGD is used due to the importance of privacy protection for a particular system, the privacy protected trained model leads to a situation where group fairness is adversely affected and more unbalanced than the case without DP. It has also been reported that trained models derived by fairness-conscious learning methods increase the threat of membership inference for unprivileged subgroup [91]. Fairness and privacy are two quality perspectives of ethical AI, but achieving both satisfactorily is difficult with current technologies

Privacy-preserving machine learning methods can theoretically guarantee data protection. On the other hand, evidence through empirical experiments is required as to whether effective data protection is achieved when adopting known training data identification methods. Furthermore, if the worst-case privacy protection estimation method based on the existing differential privacy theory is used, it is difficult to completely avoid training data identification with a protection strength of reasonable ε value [126]. The application of differential privacy has become dominant in privacy-preserving machine learning, and further advances in differential privacy theory that improve the estimated worst-case value are needed (e.g., [159]).

9.9.1.5 Post processing for privacy protection

The basic method for training data membership inference is to use information on predicted probabilities output from the trained models. To give a concrete explanation, let us consider a machine learning task to classify the data into C categories. The trained model output result for input x is a C-dimensional vector P^x, whose j-component $P^x[j]$ represents the probability that the input is classified as category j. When the component j*, which is the maximum of $P^x[j]$s, matches the tag y, the classification result is interpreted to be correct. The membership inference uses the information in this C-dimensional vector.

The idea here is to filter out some of the output information by means of safeguards placed at the exit of the trained model. Some methods include those that the output is to be limited (1) to the predictive label information (the index j* of the component with the largest probability value) only, (2) to the top 1 (<j*, $P^x[j^*]$>) of the predictive probability, (3) to the top 2, or (4) to reduce numerical precision for P^x, etc. [169].

In addition, since membership inference methods are dependent on machine learning technology as well, the measure of outputting data by adding adversarial perturbation to output data as a safeguard is effective [112]. This may be able to reduce the threats of the membership inference by causing mispredictions.

On the other hand, the safeguards introduce a change in the external interface of the trained model, and whether the introduction of such safeguards is appropriate needs to be considered in view of the system requirements of the target application to be developed.

9.9.2 C-3se: Adequacy of trained models in terms of security

Details of the specific techniques for quality control of AI security are described in Section 10.3.

9.10 D-1: Reliability of underlying software systems

9.10.1 General

The reliability of underlying software system is an important item even in conventional software. Quality management is expected to be difficult especially in implementation of machine learning AI which uses a large number of libraries including open-source libraries. Open-source software does not usually count on guarantee. Therefore, in relation to end users, open-source users (e.g., developers and operators) are responsible not only for differences in malfunction but also for discovering and monitoring potential errors and, in some cases, making modifications.

Moreover, when a machine learning model is built, it has been revealed that a training process incorporates bugs (e.g., program error) so that the impact of bugs does not appear in tests, causing quality deterioration [167].

On the other hand, in relation to conventional software, configuration management and quality monitoring of libraries and fundamental software are emphasized and infrastructure and services for configuration management and quality monitoring are improving. Some data included in those services contain information such as libraries specific to machine learning, although they are somewhat limited. These existing infrastructures are deemed worth being utilized in applications of machine learning.

9.10.2 Quality management of open-source software

It is desirable for each business operator to think about how much open-source libraries can be trusted and their quality maintained by itself or outsourced.

It can be expected that supported libraries whose quality is assured and software that went through the quality inspection process are used where necessary in relation to a necessary

quality level.

9.10.3 Configuration management and tracking of bug information

For example, Common Platform Enumeration (CPE) [56], [243] is used for configuration management of software components as a common ID to list system constituent components, in the security field. There are commercial products that manage versions of software components in use and extract information on their updates based on CPE. A list of vulnerability information related thereto called Common Vulnerability Enumeration (CVE) [57], [244] does not include bugs that are not directly related to security vulnerability. However, at least tools related to CPE are very likely to be beneficial to track the latest version of libraries and infrastructure software.

9.10.4 Feasibility of specific checks through testing

Moreover, when constituting software components have any bug, that bug does not always have a direct effect on actual ML results in machine learning components different from conventional software. If software with bugs is placed in a feedback loop of learning or training, a trained model memorizes behaviors of said bug and training seems to be superficially successful in some cases. In these cases, it is reported that potential quality deterioration caused by software bugs can be found by analyzing statistical behaviors using any of the metamorphic test technologies listed in Section 9.8.1.

9.10.5 Software update and possible adverse effects on performance and operation

On the other hand, actual software whose configuration is complex often operates unintentionally even if a developer of software libraries updates it with the intention of improving performance and operation. Depending on how a machine-learning-based system is configured, behaviors of bugs are sometimes adapted or learnt in the training process. Therefore, when any software constituent component is updated, its operation and performance should be reviewed. In some cases, it is important, to make a judgment to start training over or keep using an older version taking into account vulnerability.

The Guideline cannot recommend any option, because a specific judgment depends on each situation. However, it is important to record the background of each judgment for accountability reasons in order to claim the proper quality management.

9.11 D-2: Additional aspects of program adequacy

9.11.1 D-2se: Adequacy of programs in terms of security

Details of the specific techniques for quality control of AI security are described in Section 10.3.

9.12 E-0: Continuous monitoring and recording during operation

Many of the technologies developed for previous systems will be useful for the continuous close observation of operational status and preparing the documents.

9.13 E-1: Maintainability of quality characteristics during operation

This section describes technologies to maintain internal qualities satisfied at the commencement of operation throughout the operation period. To maintain internal qualities realized at the beginning of operation despite changes in external environments that may arise during operation, a machine learning component needs to respond to those changes. As described in Section 6.8.1, there are two operational patterns therefor. One is to make batch-processing updates by returning to the developmental environment and deploying it again. Another is automatically updating the software component when needed or at a high frequency in the operational environment. In the former pattern, monitoring to determine the timing of update and update processing are the main elements, while in the latter pattern, automated update processing is the main element. To realize this, online learning [199] is adopted. Even if updates are made automatically, it is necessary to monitor if automatic updates operate properly and process updates to respond to cases where there is any deviation from normal operation conditions.

Some of monitoring technologies necessary for both patterns are presented here. We focus especially on technologies to detect changes in data distribution over time called concept drift [113]. Moreover, technologies to retrain machine learning models which play the core role of updating trained machine learning models and technologies to create additional training data used therein will be presented.

9.13.1 Monitoring

The relation between newly acquired input data and output (e.g., inference) results may have

changed during operation from the relation between them at the time of training due to various factors including changes in external environments. When a machine learning model trained in the design or development stage using such data whose input and output relations change is kept using during operation, its performance (e.g., accuracy) may deteriorate and result in serious damage. Therefore, it is required to continuously monitor behaviors of machine-learning-based systems and machine learning components for the purpose of checking if the quality fulfilled at the beginning of operation is maintained throughout the operation period.

There are the following four tasks of monitoring during operation.

- Accuracy monitoring
- KPI monitoring
- Model output monitoring
- Input data monitoring

Accuracy monitoring directly measures the accuracy of trained machine learning models. This monitoring is divided into some patterns in accordance with the method of collecting correct answers compared with inference results of trained machine learning models required for calculating the accuracy. That is, after making an inference, (1) cases where correct answers are acquired automatically after a certain period, (2) cases where correct answers cannot be acquired automatically so that it is necessary to manually put labels and (3) cases where manual labeling is beyond budget or it is impossible to put labels. It is necessary to select an appropriate monitoring method in accordance with the above grouping of cases where correct answers are collected in order to appropriately monitor the accuracy. Moreover, some applications emphasize not only monitoring the accuracies of models but also monitoring from the viewpoint of KPI, KPI monitoring, in line with a conversion rate and the benefit of users. In this case, it is required to monitor the consistency between the accuracies of the models and the KPIs of the applications.

Model output monitoring and input data monitoring refer to the monitoring of results of inferences made by a trained machine learning model and the monitoring of its input data, respectively. The monitoring methods are divided into human monitoring where 100% sampling, automated monitoring when alert conditions are known and filtering when conditions with higher possibilities of alert are known. Model output monitoring is further categorized into a case where each output inference is checked by experts as in the case of medical diagnostic and a case where all inferences are checked altogether after a certain period of time. In the same way, various conditions may be imposed in the case of monitoring by filtering (For example, false positive is acceptable but false negative is unacceptable). In addition, when input data are monitored, whether an alert is issued in the case of monitoring by filtering or input is disregarded depends on an application.

9.13.2 Concept drift detection methods

Concept drift is one of major causes of the deterioration of the accuracy of trained machine learning models during operation. A variety of monitoring or detection methods have been proposed recently. Concept drift detection methods are categorized as shown in Table 5 according to whether correct-answer labels on data acquired during operation are used [197].

Table 5: Classification and characteristics of concept drift detection methods

Use of label	Method	Characteristics
Labeled detection (Supervised detection)	Sequential analysis	Monitoring of absolute values such as accuracy
	Statistical Process Control	Monitoring of the increase or decrease of error rate (Early detection by finding indicator)
	Window based distribution monitoring	Monitoring of distribution differences during training and during operations
Unlabeled detection (Unsupervised detection)	Novelty detection/clustering method	Simple detection by clustering
	Multivariate distribution monitoring	Detection by applying statistical hypothesis testing to data distribution
	Model dependent monitoring	Output dependent on models. Usage of confidence score

Studies of unlabeled detection methods have been carried out actively in recent years. For example, a literature [110] proposes an unknown class detection algorithm (MINAS) in multi-class problem based on clustering methods such as k-means. Moreover, another literature [190] proposes an algorithm of incremental KS test which improved the amount of calculation by developing incremental Kolmogorov-Smirnov test to judge whether samples are generated from the same distribution. Moreover, a literature by Lindstrom et al. [145] proposes a method of detecting data changes by using confidence scores associated with output from trained machine learning models without using labels (CDBD).

9.13.3 Retraining

When any change in data distribution or deterioration in the accuracy of a trained model is detected as a result of the monitoring describe above, it is necessary to retrain the machine

learning models using datasets which add recent data or replaced recent data with existing training data. Many researches have been made about this retraining. For example, a literature by Tian et al. [214] provides a platform design method to automatically determine the appropriate timing of model update in relation to existing machine learning frameworks. Moreover, it has also been proposed a method of applying balanced parameters of both existing models and models that learnt new tasks to retrain models in order to reduce forgetting of old tasks caused by retraining of a neural network called catastrophic forgetting [143].

9.13.4 **Creation of additional training data**

There are many cases where labels cannot be collected automatically. In this case, it takes much cost to manually place labels to new data acquired during operation. A study on reduced costs of retraining by reducing the number of data points to which labels are attached has been proposed through a method of reducing labeling work using software equipped with GUI (Graphical User Interface) [221] and active learning have been proposed to solve this problem [198].

10. Explanations of specific issues for each external quality

This chapter describes the background, unique challenges, and general measures regarding fairness, privacy, and AI security, among external qualities.

10.1 Quality management for fairness

In this section, we analyze the social background and problems of quality management related to fairness, examine the technical challenges specific to fairness, and summarize key considerations to be addressed as internal quality characteristics.

10.1.1 Background

10.1.1.1 Social demands and social principles

10.1.1.1.1 AI social principles and other norms

In recent years, there has been a lot of discussion about ethical considerations and fairness issues of AI and the norms in society of AI. Correspondingly, all the documents on AI social principles introduced in Section 11.1 explicitly address requirements for fairness and ethics as important topics. The Council for Comprehensive Innovation Strategy's "Principles for a Human-Centered AI Society" [24] states in point 6, Principles of fairness, accountability and transparency, that under the design concept of AI, all people shall be treated fairly without unfair discrimination based on their race, gender, nationality, age, political beliefs, religion, or other diverse backgrounds, and requests the sufficient consideration regarding fairness from the design phase of an AI system. The OECD AI Principles [26] require that AI systems be designed with human rights and diversity in mind and include appropriate safeguard mechanisms against fairness-related issues. Furthermore, based on the OECD AI Principles, AI fairness subjects have been mentioned in EU AI White Paper [32] and the European AI High-Level Expert Group's AI Transparency Guidelines [37] and are considered to be gaining consensus at the principle level [107], [40].

10.1.1.2 AI governance

According to a report by the Ministry of Economy, Trade and Industry (METI) [40], AI governance is the design and operation of technical, organizational, and social systems by

stakeholders with the aim of maximizing the positive impact of AI while managing the risks arising from its use at a level acceptable to stakeholders. The principles exemplified in the previous section are generally conceptual, technology-neutral, and cross-cutting statements of goals, while AI governance aims at achieving these goals through activities of the developers, supervisors, and regulatory enforcers, based on various grounds such as legally binding rules, non-legally binding guidelines, and international standards.

10.1.1.2.1 Legally binding rules

Fairness has a long history in the U.S. legal system and has been the basis for several concepts. More than half a century ago, the Civil Rights Act of 1964 was enacted to prohibit discrimination by race, religion, etc. Title VII of this act addresses discrimination in employment. Subsequently, laws prohibiting discrimination based on attributes such as race were enacted in various fields, such as the Fair Housing Act of 1968 regarding housing. Through the administration of these laws, two important legal concepts, Disparate impact and Disparate treatment, have been established. However, these old laws do not address issues specific to current AI system technology. For example, Facebook's 2019 lawsuit was [16] also based on technology-neutral laws that are not limited to AI.

Recently, in April 2021, the EU announced an AI-specific policy package, including a regulatory framework draft on AI [29]. There has been a lively debate among experts that legal regulations that apply to a wide range of AI fields may be a significant impediment to innovation. In terms of the severity of the restrictions, the EU's draft regulation takes a pragmatic, risk-based approach to classifying AI systems according to their intended use. For example, all AI systems those may contravene EU's values, for instance by violating fundamental rights will be classified as unacceptable risk AI and prohibited in principle. AI systems that create a high risk to the health and safety or fundamental rights of natural persons are classified as high-risk AI and subject to compliance with certain mandatory requirements and an ex-ante conformity assessment. As with the case for GDPR, when the regulation is formally passed after further deliberations, it is expected to apply not only to AI system providers within the EU, but also to AI system providers outside the EU if used within the EU.

10.1.1.2.2 Non-legally binding guidelines

In general, it can be assumed that non-legally binding guidelines serve two purposes.

1) In cases where legal regulations exist, to specify the means to fulfill the requirements (as detailed checklists, etc.),

[16] In March 2019, the U.S. Department of Housing and Urban Development filed a lawsuit against Facebook for violating the Fair Housing Act when its targeting ads excluded certain demographics from ad delivery based on race and other characteristics.

2) In cases where legal restrictions do not apply, such as low-risk classification cases, provide the basis for ensuring the quality of the AI products and services.

For example, "The Assessment List For Trustworthy Artificial Intelligence (ALTAI)" [38] by the European High Level AI Expert Group addresses fairness as a Diversity, Non-discrimination, and Fairness requirement, and lists the following checklist items as the purpose of avoiding unfair bias:

- Did you establish a strategy or a set of procedures to avoid creating or reinforcing unfair bias in the AI system both regarding the use of input data as well as the algorithm design?
- Did you consider diversity and representativeness of end-users and/or subjects in the data?
- Is your definition of fairness[17] commonly used and implemented in any phase of the process of setting up the AI system? (Did you consider other definitions of fairness before choosing this one? etc.)

This guideline likewise holds both positions described in 1) and 2) above.

10.1.1.2.3 International standard

ISO, IEC, and IEEE are developing reports and standards on AI ethics, transparency, and fairness. These international standards are described in Section 11.2.

10.1.1.2.4 Others:

Addressing fairness is also essentially achieving diversity and inclusive development, design and deployment that involves various communities from the beginning of the development process could help prevent further social damage and reduce existing social inequalities. An international multi-stakeholder initiative, Global Partnership on Artificial Intelligence, was established in June 2020 to achieve responsible, human-centered AI development and use, including this inclusiveness.

10.1.2 **Challenges in fairness**

The difficulty of achieving fairness in AI has multiple dimensions. In this section, we organize and discuss them under the following three aspects:

(1) diversity of requirements;
(2) various factors that may result in unfair AI; and
(3) how to handle sensitive attributes and attributes that cannot be directly observed.

[17] *Fairness* here is roughly equivalent to the fairness metrics of the Guideline.

10.1.2.1 Difficulty (1) Diversity of requirements

We described the fundamental definition of fairness in the Guideline in Section 4.4.1. However, there are multiple perspectives to explain, assure, and convince people of being treated equally, and often, just saying being fair is not enough. A fair system would be developed only if fairness is defined in a detailed and mathematical way. In order to make such investigations on fairness, it is crucial to understand the key perspectives of fairness, especially related with various difficulty aspects.

10.1.2.1.1 Treatment and Impact

In this section, we will focus on the following two discriminations often referred to in the subjects related to the equal employment legislation of the United States, and other discussions about fairness.

(A) Disparate treatment discrimination
(B) Disparate impact discrimination

Disparate treatment discrimination refers to unfair treatment in the decision-making process, typically involving intentional differential treatment.

By contrast, *Disparate impact discrimination* pertains to situations where fairness in outcomes is required, such as preventing adverse effects on minority groups in employment results. Defining fairness in terms of outcomes requires specifying target metrics in detail. For example, regarding gender differences in hiring: should fairness be measured by equal numbers of hires, by aligning hiring ratios with workforce demographics, or by equal selection rates adjusted for application rates? Determining which fairness metric is correct in relation to ethical requirements is often a major challenge.

A treatment process without any intentional discrimination is often insufficient to meet the requirements, and in many cases, it becomes necessary to deliberately introduce differentiated treatment into the process in order to eliminate discrimination—a practice that may even involve an intentionally unequal approach known as *affirmative action*.

Regarding the terms *equal opportunity* and *equal treatment* often referred to in Japan, multiple levels exist, which need to be clarified case by case. Taking *equal treatment* as an example, it can refer to *strong equality*, meaning that measures are taken to ensure equality (e.g., algorithms) as a system during the development stage, or it can refer to *weak equality*, meaning that intentional inequities are avoided during the development stage. Especially in statistical machine learning, the latter type of equal treatment alone is often insufficient to ensure fairness due to various factors. This should be understood as distinct from the general approach to avoiding disparate-treatment discrimination assumed in laws and rules, such as in employment decisions made by human decision-makers. In contexts such as personnel evaluations, it may also be necessary for

the individuals concerned to perceive the treatment or decision as fair.

> Group Fairness and Individual Fairness (Informative)
>
> In general, fairness requirements concern attributes that may cause inequity, such as race and gender (attributes requiring consideration). Group fairness avoids discrimination among different groups concerning some attribute value (e.g., disadvantageous treatment of women, which is a value of the attribute gender), while individual fairness seeks for similar individuals to be treated similarly. As described in the following sections, most of the current general-purpose machine learning metrics and measures are based on the premise of group fairness that mainly addresses attributes that require consideration, generally called sensitive attributes, and the Guideline also assumes the group fairness perspective unless otherwise noted.
>
> As noted above, in cases such as personnel evaluation, where the individuals concerned need to be able to see the treatment or decision as fair from their own point of view, general metrics are difficult to define. Therefore, measures to satisfy such individual-fairness requirements should be considered separately for each system. As for individual fairness, research on the degree of similarity using metric learning, for example, has been proposed[123], and further developments are expected.

10.1.2.1.2 Ambiguous social demands for Fairness

When the demand for fairness in the system is caused by the requirements of laws and regulations, even if partially, the concretization level gap between the legal requirements and the technical implementation also becomes an issue. For example, when laws and regulations stipulate that no disadvantageous treatment shall be given, it is not always obvious what acts or omissions in automatic processing by machines and their design are considered to be disadvantageous treatment. Furthermore, whether equal treatment demands on the process of how people build AI or on the AI outcome can make a big difference in actual development work. As norms and best practices such as the Guideline become more widespread, something like a market view of fairness measures is expected to be formed eventually. However, at the moment,

it is strongly necessary to be able to explain the rationale behind the chosen approach and how it was implemented.

In addition, the requirements of laws and regulations are not always self-evident as to the target with respect to which fairness should be ensured. Fairness requirements may sometimes be stated explicitly, such as equal opportunity for men and women, but more often they are presented as an open-ended enumeration, such as “gender, race, etc.” Even in cases where attributes are explicitly specified, careful consideration and a clear explanation of the chosen implementation policy are required if those attributes are not included in the dataset or if their values are subject to non-trivial measurement errors.

10.1.2.2 Difficulty (2) Various factors that may result in unfair AI

To actually realize fair AI, it is necessary to eliminate undesirable biases (i.e., unfairness). However, there are various factors through which such biases can arise, and therefore a wide range of countermeasures is required. In this section, these factors are organized and explained from the following three perspectives[54]:

(1) socially embedded inequities;
(2) biases derived from human thought and cognition; and
(3) statistical and technical biases.

10.1.2.2.1 Inequities embedded in society

When building machine learning components from data, it is important to consider cases where the data, such as training datasets, may contain inappropriate biases. Data obtained from the real world may sometimes inherit the existing biases contained in society. In the hiring judgment case introduced in Section 10.1.1.2.1, the problem was that the past hiring process included discrimination that was not noticed and that tendency was faithfully reproduced by machine learning.

Socially embedded inequities are one reason why a fair system cannot necessarily be created simply by eliminating sensitive attributes from the development process.

Therefore, when developing a machine learning-based system for which fairness is important, fairness requirements should be clarified not only through bottom-up examination of data but also from a top-down analytical perspective, and should be reflected in careful examination at the data preparation stage. Such activities will also clarify social demands for the system that are not always specifically stated, improving its accountability.

10.1.2.2.2 Biases derived from human thought and cognition

Since AI systems are designed and developed by humans, there is a possibility that biased thinking and judgment may be unintentionally introduced[54]. For example, the McNamara fallacy, which places excessive emphasis on quantitative measures at the expense of overlooking the overall picture, can also be regarded as a form of such bias or cognitive bias, and provides important implications for AI quality management.

Such biases are generally considered to be implicit and unconscious, and therefore difficult to control. However, by understanding typical cognitive biases and addressing them through process-oriented measures—such as incorporating stakeholders with diverse attributes into review processes and conducting sufficient examination prior to the formulation of quantitative metrics—it is possible to mitigate them.

10.1.2.2.3 Statistical and technical biases

This type of bias originates from methodological and technical issues in the processes from data collection to model training. Typical examples include problems such as Simpson's paradox described in Section 7.1.3.2, which can be addressed through the application of statistical knowledge.

10.1.2.3 Difficulty (3) Handling of sensitive attributes and important attributes that cannot be observed

As Section 4.4.2 explains the basic ideas, simply removing sensitive attributes from the training dataset will not necessarily ensure fairness.

10.1.2.3.1 Hidden correlations and proxy variables

When building machine learning components from input data with a huge number of attributes and information, other attributes that seem unrelated to discrimination, or input features that are not explicitly identified as attributes, may have hidden correlations with sensitive attributes resulting in statistical reproduction of discrimination. For example, such correlations between attributes such as name and gender, school name and gender, and address and income, may result in unfair output even if the direct contribution of the sensitive attributes to the output is removed.

Furthermore, in such cases, even if one attempts to remove social inequities through data synthesis as described in the previous sections, it may not be possible to synthesize training data

that is appropriate in the real world. For example, if the name of a school has characteristics such as boys' and girls' school, adding data by artificially synthesizing data with only the gender of the input data reversed may not be realistic nor appropriate.

10.1.2.3.2 Variables requiring careful consideration

When attempting to eliminate unfairness in the AI development process, it can be difficult to determine how to treat attributes that have indirect correlation with sensitive attributes, even when such attributes are correctly identified. For example, in a lending scenario, the amount a target customer can repay—an attribute that cannot be measured directly—is an important attribute that we seek to estimate. In this case, if a measurable attribute that may serve as an input variable to the system is clearly identified as a cause of discrimination in the estimation—for example, gender—the appropriate course of action is relatively straightforward. On the other hand, attributes such as *income* are closely correlated with the repayable amount but may also reflect social disparity and discrimination. From the perspective of performance improvement, such attributes may appear useful; however, their use may not be appropriate from a fairness perspective. In such cases, establishing a clear policy for ensuring fairness becomes highly challenging. This is why it is essential to be able to explain the basis for such decisions.

10.1.2.3.3 Important attributes that cannot be observed

For example, in the development of an AI system for assessing experienced hire candidates, the attribute that should ideally be evaluated is the candidate's true job suitability for the company. However, this attribute is generally not directly observable, making it difficult to prepare reliable ground truth labels. If a model is instead trained using proxy indicators, such as how long past hires remained continuously employed without extended leave or what positions they currently hold, the training dataset may become biased against employees who experienced career interruptions associated with childbirth or childcare, a pattern that may disproportionately affect women in some labor markets.

> Case Study: COMPAS (Informative)
>
> The COMPAS (Correctional Offender Management Profiling for Alternative Sanctions) system, used in the United States to estimate recidivism rates as a basis for parole and other criminal justice decisions, was criticized for producing significantly unfavorable outcomes for Black individuals and for potentially reflecting socially embedded biases, such as bias in crime detection, in the training

data. Whether this criticism was justified depended on how the fairness metrics were defined, which also illustrates the importance of defining fairness metrics.

10.1.3 **Framework for process management to ensure fairness**

In this section, we summarize the approach of fairness quality management in machine learning module that the Guideline addresses, based on the issues enumerated in the previous section.

10.1.3.1 Structural model for ensuring fairness

In general, fairness requirements for products and services concern the relationship between users and service providers, that is, quality in use. At this level, they are usually expressed as abstract and qualitative requirements, such as “not to be treated unfairly” or “being treated equally”. By contrast, *fairness metrics* discussed in machine-learning AI and related fields (Section 7.10.1.3) correspond to internal quality indicators in this Guideline. They serve as quantitative indicators of the degree of bias, focusing on specific attributes in training datasets or output results, across various stages from system development to operation.

There exists a significant gap between these two types of expressions. It is crucial to consider which attributes to focus on and which fairness metrics to apply in the evaluation process. This consideration is a key point in ensuring fairness in AI products.

From this perspective, the Guideline assumes the following process for realizing fairness in machine-learning systems:

- First, treat equal treatment as a qualitative requirement at a highly abstract level, such as societal requirements and quality in use.
- Next, translate this qualitative requirement into more concrete fairness requirements through a risk analysis approach, treating possible inequities as risks.
- Then, from the development stage selected as appropriate for the case at hand onward, implement and verify those requirements by using quantitative fairness metrics, such as equality of outcomes.

This approach seeks to ensure the appropriateness of the selection and setting fairness metrics through analysis and design. In this respect, it can be seen as analogous to risk-analysis-based approaches used to realize qualities such as risk avoidance and functional safety.

Figure 15 below illustrates a typical flow of this approach. This diagram is not meant to constrain individual development policies or phases, but to provide a model for shifting from

qualitative considerations to quantitative methods as following.

(a) Fairness demands at the highest level of abstraction, usually demanded from perspectives such as *justice or human rights*, would be thought to derive from matters expressed in keywords such as *equality* and *equal treatment*.

(b) From the viewpoint of social rules such as legal system and implicit ethical behavior, there are two cases, one in which *equal treatment* is required, and another in which *equal outcomes* are required in the form of numerical targets, as shown in Section 10.1.2.1.

In the latter case, it may mean a response assuming that the treatment is not equal in the actual institutional aspect. For example, in the context of equal employment opportunity for men and women, when there is a positive feedback structure of undesirable(negative) phenomenon, such as that the lack of women's advancement in society hindering the development of women with sufficient social experience, which in turn delays women's advancement in society, a positive feedback may be induced to positive phenomenon by forcibly promoting women's advancement through numerical targets. In this case, since the achievement of numerical equality of outcomes becomes the primary objective, all subsequent stages are also attributed to the achievement of numerical equality.

In addition, as mentioned before, *equal treatment* has two levels; a strong requirement to *actively design and implement the system to achieve equal treatment*, and a weak requirement to *avoid intentionally providing unequal treatment*, which is a non-binding, just an effort target. The Guideline mainly deals with the positive activities of the former and does not cover weak requirements at the level of mere effort targets.

(c) and (d)

These phases (quality in use of overall system and external quality of machine learning modules) are still at the stage of desk-based consideration, and do not yet involve detailed examination of the actual training dataset. Nevertheless, if it is possible—on some reasonable basis—to sufficiently anticipate the "result distribution" that should represent the desired fairness, then the target fairness metrics can be defined at this stage.

(e) Next, in the preliminary preparation stage, where we construct (develop) the machine learning system that we want to train and consider some of the internal qualities (A-1 and A-2 as defined in the Guideline), we have the same two options for setting goals. At this stage, analysis may be conducted based on the actual data collected and other information, and goals may be set.

(f) Finally, in the stage of learning and checking internal quality, there are three possible options: analyzing the statistical distribution of results, monitoring statistical and analytical indicators other than the distribution of results, and explaining the equal treatment from the logical structure of the development.

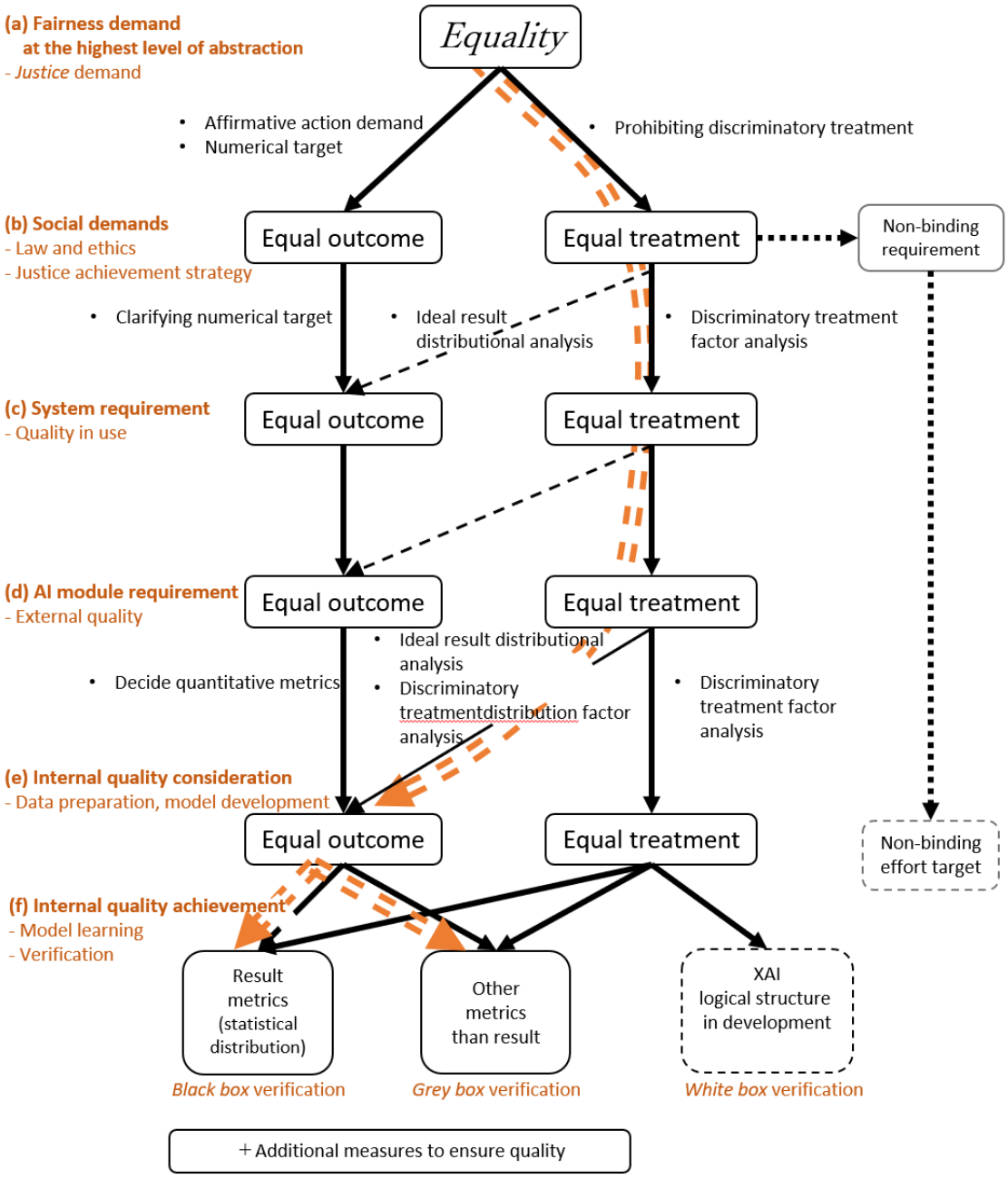


Figure 15: Example process structure for fairness assurance

10.1.3.2 Basic concepts for ensuring fairness

In the remainder of this section 10.1, we assume the process shown by the dotted line in Figure 15 will be an approach that can be generally considered in basic fairness management in machine learning modules. Firstly, at the level of external quality, we discuss the fairness to be

ensured among features and attributes following the definition of fairness given in Section 4.4. At the stage of internal quality consideration, we analyze data distribution, etc., and convert them into numerical requirements for bias in outcome, and others such as in dataset if applicable. Then, based on the results of the previous-step analysis, the requirement for bias in outcome (i.e., inequity regarding distribution of results) will be measured at the system development and testing stages, together with other metrics if defined, thus confirming fairness we are aiming for.

This is based on the following two considerations. (Point 1) the fairness requirements should be reflected in the distributions of training and test datasets, since machine learning implementations are derived from data. (Point 2) to sort out the complex problem analysis discussed in Section 4.4.1, consideration based on actual data at model development stage is necessary, thus keeping the original abstract-level fairness requirements until that stage is important.

Of course, the development processes shown in the figure can be selected according to the situation of the functional requirements of the individual system. For example, in the case of an application where numerical targets are set in advance at stage (b), the implementation will be based on the functional requirements to achieve the numerical targets.

10.1.3.3 Considerations for handling data with sensitive attributes

Among personal information, there are *data with various sensitive attributes*, such as race, gender, place of origin, etc., that require careful handling and one must avoid unfair treatment resulting from these values. In developing machine learning modules that require fairness, the careful handling of these sensitive attributes (data) is required from the early stages of the process shown in Figure 15.

The Guideline *does not take the position that it becomes automatically fair if it does not handle any sensitive data*. Data obtained from the real world has complex structures and correlations, and it is quite possible that AI built from other input data than sensitive data will eventually be found to have correlations with certain sensitive data which is even not included in the input. Furthermore, not using the sensitive attributes in the development, especially in the testing process, can be a major disadvantage because it makes it impossible to check and inspect the quality of fairness. This is because sensitive attributes are often the key points for ensuring fairness. Therefore, in order to build a fair machine learning system, we believe that such sensitive data should at least be obtained during development, and sufficient consideration should be given at the stages of system design and development.

On the other hand, whether sensitive data can be obtained during operation phase for the purpose of quality monitoring and additional training needs to be considered in relation to personal information protection. In addition, there may be cases where it is impossible to handle

sensitive information that requires special consideration, such as race or medical history, even at the system development stage. In these cases, it is necessary to give sufficient consideration to how to eliminate bias and conduct monitoring in the model building and additional learning phases.

10.1.4 Refining fairness requirements

10.1.4.1 Methods for fairness requirements

For analysis described in 7.1.3, a graphical method that can represent these interrelationships in a relatively concise manner, called Causal Bayesian Networks (CBNs) [95] will be described in this section. CBNs are graphical representations that exhibit the causal path between attributes, helping us to consider the patterns of unfairness underlying the training data. It is also useful for examining the types of fairness metrics that should be used.

(Example)

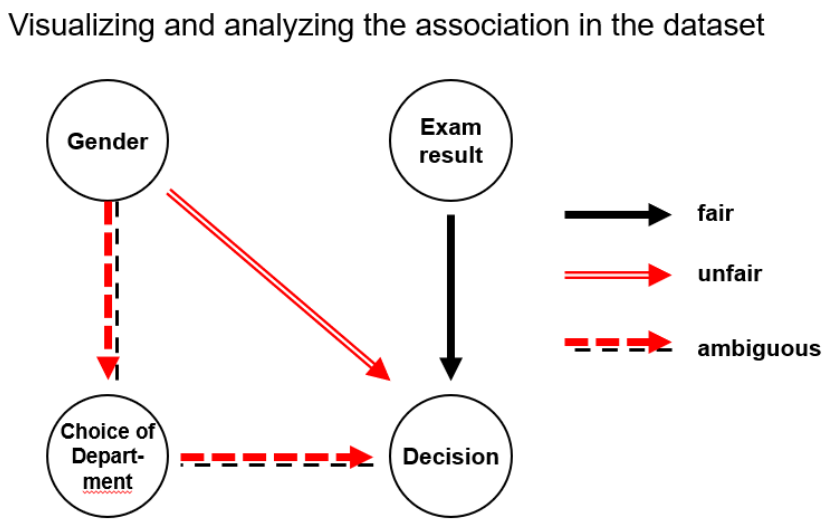


Figure 16: Example visualization of relationships using a CBN (from [95])

Figure 16 shows a college admission scenario, discussed in [95] and other studies, in which applicants are admitted based on qualification exam results, applicants' choice of department, and gender. The CBN visualizes relationships among attributes by representing a direct causal influence of α on β with an

arrow from α to β. Each causal path can then be examined to determine whether the corresponding influence is *fair*, *unfair*, or *could be either fair or unfair*.

In this example,

- the causal path from exam results to admission decisions is clearly fair.
- The direct use of gender in admission decisions is clearly unfair. For example, if two applicants have the same exam results but are treated differently solely because of gender, the relationship between gender and the admission decision is unfair.

The path from gender to choice of department could be regarded as either fair or unfair. If the tendency of women to prefer certain departments is due to implicit environmental pressures on women, then this causal path may be unfair. Similarly, the path from choice of department to admission decision could also be regarded as either fair or unfair. If the admission rates for departments chosen more often by women are intentionally lowered, then the path would be unfair.

Thus, by carefully analyzing the paths that may influence admission decisions, we can see that the path from gender to admission decision via choice of department may impair fairness, depending on the actual situation and the algorithm used. This means that the fairness of the admission decision cannot be guaranteed simply by excluding the direct causal path from gender to the admission decision.

Figure 17 shows an example of Simpson's paradox represented using a CBN. Suppose that we attempt to detect a correlation between the two attributes—exercise days per week and health, we may fail to achieve our goal if we train our model using a dataset collected without taking age into account, where age is a third factor that affects both attributes. For simplicity, assume that age is positively correlated with exercise days per week and negatively correlated with health. In this case, even if exercise frequency and health are actually positively correlated, the analysis may indicate an apparent correlation in the opposite direction. Such a third factor, such as age in this case, is referred to as a confounding factor and

should be handled carefully when preparing the training dataset. Such precautions are necessary in dataset preparation in general, not only from the fairness perspective. Drawing CBNs facilitates the discovery of such confounders, and if a sensitive attribute is a confounder, this indicates in advance that simply removing it may not be appropriate.

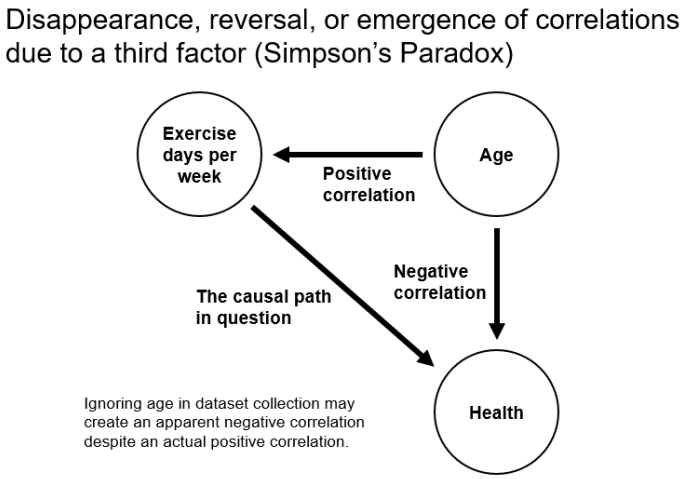


Figure 17 : Example CBN illustrating Simpson's paradox (From [95])

10.1.4.2 Complementary approaches from different perspectives

The refinement of fairness requirements, described above as preliminary preparation, may be difficult for AI engineers alone to carry out sufficiently. The participation of domain experts is therefore highly desirable. In addition, the following approaches, based on perspectives different from those of AI development, can also be useful to some extent.

Correlation analysis (statistical method)

As shown in Figure 15, the Guideline recommends that detailed studies based on datasets should be conducted during the model development process, while requirements analysis should proceed in a top-down manner without going deeply into implementation details.

However, lack of domain knowledge may make it difficult to initiate such top-down analysis. To address this difficulty, the following bottom-up steps can serve as practical alternatives:

(1) Construct a network, such as a CBN, based on correlations observed in the dataset.
(2) Examine the nature of each identified correlation within the network, for example, whether it represents a fair or unfair causal relationship.

That said, it is essential to bear in mind that fairness requirements should ideally be derived from higher-level principles. These requirements cannot always be fully captured by the attributes present in the dataset alone. Therefore, care should be taken to mitigate the potential drawbacks of relying solely on this kind of bottom-up implementation-driven approach.

Social and economic approaches

Fairness and equality in society have long been studied in welfare economics, including through social welfare functions (SWFs), which aggregate the utility functions of individuals and groups. Recently, approaches have been proposed that use SWFs to optimize interventions designed to promote fairness in causal relationships [137]. Such approaches are expected to provide insights into the definition of fairness and help identify possible corrective measures outside the AI system.

10.1.5 Extended topics in fairness implementation

The overall approach to achieving fairness was presented in Chapter 4, and the specific measures from the perspective of each internal quality were explained in Chapter 7. This section provides supplementary explanations and introduces additional topics to complement those discussions.

10.1.5.1 Three types of processing, pre/in/post

As described in Chapter 4, the feasible approaches depend on which part of the machine learning model development process can be carried out—in other words, on the scope of development available to the developer.

If the developer can modify training dataset, then pre-processing is worth trying because it allows for relatively straightforward explanations such as we trained on a dataset with specific metrics and guaranteed fairness and is generally quite effective.

If the developer can implement the learning phase from the beginning, then in-processing approach can be used.

If the developer can only receive a trained model which is basically a black box and can perform some adjustment to it, then only post-processing (or additional learning) would be the choice.

To achieve quality objectives, different types of processing may also be combined or applied iteratively. For example, even if the fairness metrics goal *for the training dataset* is achieved in pre-processing, the fairness metric *for the output* may still not be satisfied, due to how the model is developed or due to trade-offs with other quality attributes. In such cases, in-processing or

post-processing techniques may be incorporated as well. In some cases, it may even be necessary to revisit the training dataset and apply alternative pre-processing methods.

10.1.5.2 Achieving fairness in natural language processing AI

Some of the pre-processing methods explained in the Guideline are also provided in the development platform and tools introduced in Section 10.1.6, and can be used relatively easily. However, those methods assume that the input data are a series of *attributes and values* (structured data) and that *sensitive attributes* are already identified within the data. It is difficult to use those methods directly with machine learning systems that process natural language processing that takes a large amount of text (corpus) as input. As for fairness in natural language processing, defining metrics is also challenging. Mehrabi et al [155] introduces bias mitigation techniques in language modeling and feature learning methods, but further research is still needed before such methods can be widely adopted in practice.

10.1.5.3 Impact of generative AI on the process of achieving fairness

Generative AI, which has been rapidly advanced in recent years, is trained using datasets collected from various information sources and has been carefully fine-tuned. However, it is difficult to evaluate the fairness of its outputs, which may risk amplify real-world bias [150]. Therefore, in addition to the considerations described in Section 7.7.1, further measures may be required—such as clarifying the provenance of data, and in some cases, excluding generative AI-produced content from training datasets.

10.1.6 Development infrastructure and tools supporting fairness assurance

10.1.6.1 Purpose of using development platforms and tools

The tasks described in Section 7.12 are to be carried out within the machine learning lifecycle, alongside the pursuit of quality goals other than fairness. In particular, fairness often requires iterative and exploratory efforts, such as identifying biases embedded in datasets and evaluating fairness metrics for trained models. In addition, qualitative aspects of the development process may also serve as important evidence.

Given this, the use of development platforms and tools can be especially beneficial for the following purposes:

- Efficiently examining the dataset from the perspectives described in Section 10.1.5.

- Conducting visualizations, metric evaluations, and other fairness assessment on trained models.
- Maintaining proper records of development process, the datasets used in the process, and the sequence of detail steps taken.

10.1.6.2 Examples of platforms and tools

10.1.6.2.1 Typical functions and modules

The following key functions are identified based on the purpose outlined in the previous section.

(1) Visualization of datasets for analysis to discover relationships regarding sensitive attributes
(2) Fairness evaluation functions beyond standard fairness metrics (e.g., execution of counterfactual experiments, testing by altering certain attributes).
(3) Various fairness metrics measurement
(4) XAI Library for visualization of which attributes contributed to the decision
(5) Infrastructure for building pipelines to support the lifecycle of continuous monitoring and retraining after operation (so-called DevOps infrastructure + infrastructure for datasets and models)

Google and IBM Fairness360 are described below for reference, but new features from various other vendors will continue to be provided, so please check the latest information and select the one that is most appropriate.

10.1.6.2.2 Example 1: Google tools

Functions (1) through (5) described in the previous section will be covered as follows.

- What-if-tool (1), (2)
  - https://pair-code.github.io/what-if-tool/learn/tutorials/walkthrough/
- Fairness Indicators (3)
- Explainable AI (4)
  - https://cloud.google.com/explainable-ai/
- AI-Platform (5)
  - https://cloud.google.com/ai-platform

The What-If-tool, which provides a variety of functions directly related to fairness, is not just a visualization of the training dataset, but also helps to discover potential bias patterns in the dataset through hypothetical scenarios and analysis functions when sliced by various attributes.

Explainable AI can also contribute to confirming the bias, as it can quantitatively indicate which attributes contributed to the decision during inference. (Note that, as mentioned in

Section 4.4.1 having influence does not necessarily mean unfairness.)

10.1.6.2.3 Example 2: IBM tools

Functions (1) through (5) described in the previous section will be covered as follows.

- AI Fairness 360 … (1), (2), (3)
  - https://github.com/Trusted-AI/AIF360?utm_source=chatgpt.com
- IBM Cloud Pak for Data … (5)
  - https://www.ibm.com/jp-ja/products/cloud-pak-for-data

AI Fairness 360 is open source software developed and released by IBM Research Foundation and donated to the Linux Foundation in 2020. AI Fairness 360 provides a rich set of features to address biases in machine learning models and datasets, and generally covers the methodologies described in the Guideline.

- Pre-processing: Supports reweighing of training data, disparate impact remover, optimized preprocessing of training data, etc.
- In-processing: supports prejudice remover, adversarial debiasing, etc.
- Post-Processing: Equalized odds and calibrated equalized odds are supported.
- Metrics: Predictive parity and others are supported. Equalized odds and demographic parity can be calculated using the provided modules.

IBM Cloud Pak for Data is a multi-cloud software that covers the lifecycle of machine learning models in an integrated manner, and IBM Watson OpenScale is one of the software modules that make it up. The ability to monitor input and output data during model operation and detect time-series changes in fairness metrics is useful for improving biases that occur after operation begins.

10.2 Privacy

This chapter first summarizes social backgrounds in which quality management issues for privacy are considered (Section 10.2.1), then presents some quality characteristics specific to the privacy in machine learning (Section 10.2.2), and finally describes issues to be considered in the quality management (Section 10.2.3).

10.2.1 Privacy protection

This section summarizes the social context in which the notion of privacy in information systems, the right to control personal information, is considered.

10.2.1.1 Privacy risk

10.2.1.1.1 Ethical AI

AI technology, powered with data, enables realization of advanced functions and wide spread use of them. Such application functions may include services that have much impact on the right to privacy, depending on how they are used. For example, remote facial recognition in public places, profiling, and micro-targeting, if used outside the user's control, increase the danger of privacy violation. These observations have led to regulations that restrict the usage of socially unacceptable services and impose legal obligations on service developers and providers so that they protect the right to privacy. The legal issues have been discussed together with the notion of Ethical AI [25], [37].

Ethical AI aims to mitigate threats to human safety and includes concerns about threats to privacy of users in general [97]. The discussions in Europe lead to regulations which enumerate specific services that may violate privacy rights and restrict high-risk AI systems or AI systems being used that pose the unacceptable harms [29].

In developing information systems, including systems to use AI technology, some measures on the protection of information and users' personal data are mandatory [38], [30]. Personal data should not be identifiable from available information in places where they are outside the user's control. Unrestricted sharing and distribution of personal data bring about threats to human safety. It should be ensured throughout the system lifecycle, from development to operation, that personal data are appropriately protected.

10.2.1.1.2 Negative externalities

The views on personal data protection have changed over time. In general, information has positive externality: the greater the amount of information, the higher new values [109]. Until around 1980, restricting data from public use was considered as economical inefficiencies, sometimes resulting in double investment, so that appropriate data sharing mechanisms were pursued. In the mid-1990s, with the development of information technology, it became possible to share large digital data or databases. The benefits of secondary use were emphasized, and digital data distribution was actively promoted.

It is often thought that data protection problems do not arise if personal information that would allow identification of individuals is pre-processed to be removed from public databases that are shared and distributed. However, it has been shown, in a public medical database, that re-identification of personal information is possible from the pre-processed and de-identified medical records [211]. Moreover, if there is public information available externally, the auxiliary information or background knowledge in it can be used in the re-identification attacks. In other

words, the greater the amount of information, the greater the danger of personal data re-identification. This is called negative externality of information [65].

From the viewpoint of service development and provision, the negative externalities directly link to adversarial impacts on business. Indeed, in the 21st century, as Internet services become more active than before, the threat of personal data re-identification has increased, and countermeasures against this have become a major technical challenge [172], [120] M. Hayashitani, J. Mori, and I. Teranishi. Survey of Privacy Threats and Countermeasures in Federated Learning. arXiv:2402.00342 [cs.LG], 2024. https://arxiv.org/abs/2402.00342

[121]. In the future, AI technology can serve as a basis for those advanced services to use personal data that might include sensitive information. It is necessary to understand the dangers posed by the negative externalities, and to establish technical and institutional frameworks to mitigate such business risks.

10.2.1.1.3 Personal data

Personal data represent information that identifies an individual. As information technology has advanced, the context for discussing protection issues has been broadened, and the types of personal data have been varied. They can be generally classified into registered data and activity data.

10.2.1.1.3.1 Registered data

Registered data represent information that directly identifies an individual. Items are covered in Census such as name, date of birth, gender, race, address [249], and sensitive attributes, such as SSN or medical records. Some may also be explicitly subject to regulatory requirements. For example, they include specific sensitive personal information found in JIS Q 15001, Personal Information Protection Management Systems, Japan or sensitive personal information defined by Revised Act on the Protection of Personal Information, Japan.

10.2.1.1.3.2 Activity data

Activity data include not only information resulting from real-world activities, but also information related to Internet service use, online activities. The latter has become understood as personal data as the Internet services grow. Typical examples of activity data include search history when using search engines, like button information in SNS, e-commerce purchase history, on-demand video viewing history, and cookies. Some may also be explicitly subject to legal and regulatory requirements [30]. With the development of information technology, including AI, activity data are considered to become even more diverse in the future.

10.2.1.2 Rights of the data subject

10.2.1.2.1 Data subject

The right to privacy is understood as one of the fundamental human rights and is given a legal basis as a right of natural persons. On the other hand, it is not an absolute right, but rather follows the principle of proportionality. Regulations refer to a balance between positive and negative aspects. Natural persons associated with the personal data discussed in the previous section are called data subjects.

10.2.1.2.2 Control by data subject

The rights of data subjects involve a legal basis. Due to different national basis laws, the rights of data subjects may not be the same. This chapter provides an overview of data subject rights, focusing on the European general data protection regulation (GDPR) [30]. Since its entry into force in 2018, legislations similar to the GDPR have been discussed in many nations, and global markets are being formed in line with the GDPR's approach. This chapter focuses on the GDPR as a representative example of legislation.

Consent: The provision of personal data is subject to consent by the data subject. There are two general approaches: opt-in, which requires prior consent for the provision of personal data, and opt-out, which deems the provision of personal data to be consented to unless there is an explicit refusal to provide it. The GDPR adopts opt-in (Article 7 of the GDPR). And only free consent is valid.

In relation to the consent that has legally basis, the following points should be noted [Article 5.1 of the GDPR].

Purpose limitation: Personal data shall be collected for specified, explicit and legitimate purposes and not further processed in a manner that is incompatible with those purposes. Further processing is considered incompatible.
Data minimization: Personal data shall be adequate, relevant, and limited to what is necessary in relation to the purposes for which they are processed.
Storage limitation: Personal data shall be kept in a form which permits identification of data subjects for no longer than is necessary for the purposes for which the personal data are processed.

Note that, as will be discussed later, these bring about major challenges in the context of data protection related to AI (Section 10.2.2.1.1). The importance of control by the data subject is also discussed as below.

Right to rectification and erasure: Data subjects have the right to obtain from the controller without undue delay the rectification of inaccurate personal data concerning him or her [Article 16 GDPR] and the right to obtain from the controller the erasure of personal data concerning him or her without undue delay [Article 17 GDPR]. The right to erasure is referred to as the right to be forgotten.

10.2.1.3 Protected processed data

Threats to personal data protection arise from re-identification. If personal data were to be released in their unprocessed form, they would clearly not be protected. Whether re-identification is possible is dependent on the method of data protection pre-processing [172], [187]. The GDPR considers multiple methods of data pre-processing and requires that appropriate institutional measures to achieve required level of protection be put in place, depending on the strength of the data pre-processing method (Section 10.2.1.3.2).

10.2.1.3.1 Spectrum of data protection pre-processing

The terminology used for protection pre-processing methods differs in divergent research fields. In this chapter, we avoid preconceptions given by the wording of the terms, and divide them into four levels, from the protection processing level 0 to protection processing level 3, according to the protection strength. Below, the subject matter is equally applicable to both registered and activity data.

Protection processing level 0: Unprocessed or unmodified data. If the data concerned are personal data, technical or institutional measures should be taken to achieve the expected data protection level. For example, mechanisms should be used to prevent the data from being leaked to the outside world.

Protection processing level 1: Data obtained by pre-processing unmodified data. They include pseudonymous data. Those data can be reconstructed by inverse transformations. In other words, data can be re-identified once the function responsible for the pre-processing or its inverse function is obtained. It is mandatory to take technical and institutional measures to achieve the expected data protection level. For example, information related to the pre-processing process should be properly managed to avoid the application of the inverse transformation process, and a mechanism to prevent the leakage of such information to the outside world should be used together.

Protection processing level 2: De-identified data in which unmodified data (protection processing level 0) are processed to make re-identification difficult. While the level 1 allows direct de-identification, the level 2 often requires a trial-and-error method of indirect de-identification while making use of auxiliary information or background knowledge. Differences

in the difficulty of indirect de-identification affect the technical or institutional measures to be taken.

Protection processing level 3: Anonymous data are those pre-processed in such a way that the inverse conversion is proven to be mathematically impossible in principle. Anonymous data are defined as those that are irreversible. Note that at the current level of technology, there is no generic transformation method that guarantees the irreversibility.

10.2.1.3.2 (Supplementary) Views under existing Regulations

The level classification (refer to Section 10.2.1.3.1) of protection processing is based on technical methods. Below is an informal overview of the approaches taken under existing regulations on data protection.

10.2.1.3.2.1 Binary classifications

The binary classification approach was adopted until the mid-1990s, when benefits of secondary use, or publicly sharing, of digital data were primary, in which data are divided into two categories: unmodified data of the protection processing level 0 and the protection processing level 1 or beyond. Unmodified data, if referring to personal information, are subject to regulations, which requires that technical or institutional measures be taken to protect them. On the other hand, pre-processed data, of the level 1 or beyond, are collectively referred to as anonymized data and attempts to re-construct or re-identify these are legally prohibited. However, it is argued that the expected protection cannot be achieved only by imposing penalties on re-construction or re-identification as a violation of the law [187].

It should be noted that these anonymized data are different from anonymous data characterized by the irreversibility (the protection processing level 3).

10.2.1.3.2.2 Ease of re-identification

Deidentification data (protection processing level 2) is closely related to negative externalities, and the threat to data protection varies depending on the volume of auxiliary information or background knowledge available and the technological development of re-construction and re-identification technologies. The basic idea of the GDPR is to regulate unmodified data of protection processing level 0 and pseudonymized data of protection processing level 1.

In addition, Article 11 and the preface of the GDPR [26] refer to de-identified data of the protection processing level 2 equivalent, which are not subject to the regulation if it is demonstrated that they cannot be re-identified by the technology available at the time, at a reasonable cost and time. If it is not subject to the GDPR regulation, there is an incentive to adopt the devised methods of protection by de-identification, because technical or institutional

measures are no longer needed [122].

On the other hand, with technological developments in re-identification methods, it may change whether data by means of particular de-identification methods are subject to the GDPR regulation. Traditionally, census aggregate information, for example, has been considered as statistically processed information equivalent to the protection processing level 2. In other words, statistically processed aggregate information have been regarded as data that are difficult to re-identify. In addition, census aggregate information must be appropriately protected, which is often stated as legal requirements in many nations. For example, in the United States, the law requires that the difficulty of re-identification of census results be guaranteed. However, it was recently pointed out that micro-data or unprocessed data can be re-identified even if conventional protective processing methods are applied [115]. Thus, the application of the method of protection by differential privacy (Section 10.2.1.4.2) is being considered for the 2020 census [250].

10.2.1.4 Technical privacy metrics

The data protection method is accompanied with technical privacy metrics [222]. They discuss the strength of data protection using quantitative metrics. Two representative ideas are introduced below. In this section, the target data with multiple attributes are called records, and a collection of many records is called a database, following the terminology used in the literature in this research field.

10.2.1.4.1 Data similarity

A method that focuses on records. A record is protected if the record is sufficiently similar to other records and cannot be separated from them. When a protected record can be linked to a data subject via particular sensitive attributes, it is sufficient to introduce multiple records such that the quasi-identifiers derived from the sensitive attributes are to have the same value. When the original values are distinct, there is a method to obtain similar records by replacing them with a kind of abstract values through generalization, aggregation, etc., considering the meaning of the attributes. For example, the attribute of birth date is turned to be represented only by the information of year and month, discarding the day. Last, K-anonymity is a systematic approach to obtain K or more similar records [195], [212].

10.2.1.4.2 Outcome indistinguishability

A method that focuses on database query results. Consider two adjacent database D and D' where D' is constructed as a copy of D but a particular record r removed. A record, r in this case,

is protected if the results of queries to both databases are sufficiently indistinguishable. The presence of the record does not affect the results, and thus the record is protected.

The ε-differential privacy (ε-DP) is based on the indistinguishability, and is a theory of provable privacy to guarantee that a given measure ε is an upper bound on the protection strength [104]. The protection is also persistent during a posteriori processing; the same differential privacy strength εis guaranteed when ε-DP protected data are processed afterwards. Furthermore, the composition theorem allows to determine the worst-case protection strength for a combination of multiple queries.

After the proposal of the ε-DP, (ε,δ)-differential privacy was introduced as an approximate differential privacy with relaxed applicability conditions [105]. Subsequently, further research has been conducted on improving the estimated privacy measure required to achieve the expected level of protection strength for multiple queries [159].

Differential privacy protects records as data in a database from data analysts or database users. Local differential privacy (LDP), on the other hand, protects data provided by data subjects from data processors or database creators [225]. A good example of LDP is seen in the process of data transfer from a client to a server via the Internet, in which the client intends to protect his or her own data [108].

10.2.2 Machine learning and personal data protection

This section presents the notion of personal data protection in machine learning systems.

10.2.2.1 Life cycle and protected Information

First gives an overall picture of the lifecycle of machine learning components, from development to operation, as well as how artifacts are delivered or reused, and then introduces personal data protection issues regarding machine learning components.

10.2.2.1.1 Protection interface

The lifecycle of machine learning components goes through the process of collecting raw data, preparing training datasets, conducting training, building systems incorporating trained models, installing the systems in operation contexts, and keeping on working. It should be noted that what is protected is personal data provided by the data subject. The life cycle of machine learning components and the situations in which threats to data protection may arise are explained below.

Case 1: The data subject provides personal data. This is used as raw data. The data are to be provided with appropriate consent by the data subjects.

Case 2: The processing personnel of training data builds training dataset from a group of collected raw data. Since the raw data, referring to personal information, are processed, the processing personnel of training data is a potential threat. Even if the purpose of use is aligned with the scope of users' consent, care should be taken in relation to the detailed content of the consent, as it is often difficult to determine in advance the minimum amount of data definitively needed for the training dataset, which may conflict with the data minimization requirements.
Case 3: The component developer conducts machine learning with the training dataset as input information to derive a trained model. If the training dataset is not properly protected and processed, the machine learning component developers could be a threat because they can potentially refer to the raw data from the training dataset.
Case 4: The developer of the machine learning system builds a machine learning system that incorporates the trained model. If the trained model is not properly protected or processed, the developers of the machine learning system may be a threat because they can potentially refer to the training data indirectly and even to the raw data from the trained model as well.
Case 5: The system operator installs the final product of the machine learning system and supports their operation. If the input data during operations include personal data of a certain data subject, and the system operator can access to the input data information, then the system operator may be a threat. Additionally, if the location information concerning with the system installation is indirectly related to sensitive information, the system operator may again be a threat. For example, the very fact that a remote facial recognition system installed in a certain location recognizes a particular data subject reveals the behavior of that data subject, and thus is a threat.
Case 6: The clients, or primary users, use the machine learning system according to the access right given to them. If training data identification is possible, there may be leakage of personal data and thus those clients can be a threat.

In addition, when the operating status of the machine learning components changes from the assumptions during development, the learning data is revised and retraining is carried out. Retraining is often carried out as a measure against deterioration during operation of the basic predictive inference performance (correctness of the trained model and stability) of the machine learning components. In other words, the retraining is usually meant for the purpose of improving AI performance, and cannot necessarily be said to be done considering enhancement of privacy.

Privacy is generally incompatible with AI performance because of their trade-off relationship. If the main purpose of retraining is to improve AI performance, we cannot exclude the possibility that privacy strength will deteriorate as a side effect. Therefore, it needs to confirm that the revised training data satisfies the validity of the dataset regarding privacy (Section 7.7.2), and also to confirm from multiple perspectives that there is no deterioration of privacy (Section 7.10.2).

10.2.2.1.2 Artifacts and threats

The training dataset (Case 2) and the trained model (Case 3) can also be provided as artifacts for use in the development of new machine learning components. In such reuse cases, threats to data protection arise. First, the reuse development itself can pose threats to the terms of consent approved by the data subject (Case 1), i.e., purpose limitations, data minimization, and storage limitations (Section 9.1.2.2), for example, in the case of the compliance with GDPR.

Second, if the training dataset is not properly protected and pre-processed (Case 2), it is possible to refer to the raw data from the training dataset, which implies that developers who reuse the data are a threat. Additionally, if the trained model is not properly protected (Case 3), it is possible to refer indirectly to the training data or even to the raw data from the trained model, which implies that developers who reuse the data are a threat.

10.2.2.2 Training data identification

The basic problem of personal data protection in machine learning is that membership inference is technically possible to identify a piece of information about the training data.

10.2.2.2.1 Training data re-identification problem

The training process does not achieve removal of specificity sufficiently, and thus trained models are not anonymized. As a result, attempts to re-identify the training data from the pre-trained model are sometimes successful. For classification models, the reason for this is "Memorization" of the label of each individual input training data in the trained model [237], [87].

The memorization problem can be summarized as follows. Compare the likelihood of the predicted label obtained when the input x is given to two trained models M and M', where M is derived from dataset S containing data point <x, y> consisting of data x and label y, and M’ from a training dataset S' with this data point removed (S’ = S\{<x, y>}). In the case of label memorization of training data, the result of the former M(x) is more likely than M'(x). This “memorization” phenomenon appears as a result of the training mechanism leading to poor generalization performance or overfitting and also appears as the trained model is derived from the inadequate data distribution in the training dataset [149].

Training data re-identification problem is classified as membership inference, attribute inference and model inversion, or property inference, and so on [158]. These are methods of re-identification against trained models of the protection processing level 2, and are called “inference” because the results refer to probabilities. For example, a membership question is whether certain data belongs to the training dataset. Since the prediction will always be

accompanied with a certain probability even if answered randomly, it is necessary to decide whether the membership inference is successful or not in terms of a given threshold probability value and to compare it with the case of random answers.

10.2.2.2.2 Membership inference

Membership inference is a basis of the training data re-identification, given a trained model, to find out if specific data were used for training [200], [234], [194]. Note that just revealing such membership relations is a threat to data subjects. For example, imagine that a training dataset is built from a list of defaulters and a trained model is derived from the dataset. If membership inference attempts demonstrate that certain data are used as training data, then it indirectly reveals that the data subjects linked to those training data are defaulters as well.

10.2.2.2.3 Attribute inference and model inversion

Attribute inference is to infer sensitive information about the training data making use of information that is publicly available [173], [207]. Model inversion is to infer training data from output results of the trained model and to reconstruct certain subset of the training dataset [112].

10.2.2.2.4 Property inference

Property inference is an attempt to find certain global properties of a training dataset by means of the information that was not originally intended for the target trained model or the information that is not expected as a result of predictive inference [72], [114]. Global properties are characteristics of the training dataset and are not necessarily threats related to personal data. For example, properties may refer to specific conditions under which data were collected or the fact that the training dataset is biased when focusing on certain attributes. The former is an example of trade secrets, the condition used for the raw data collection in this example, being leaked, and the latter suggests the possibility of problematic issues in fairness because of bias.

10.2.2.3 Quality characteristics overlapped

Privacy is a quality characteristic in relation to the protection of privacy of data subjects or data protection, and relates to the other quality characteristics that the machine learning component should exhibit. Below presents the relationship with fairness and security of both information security and cybersecurity.

10.2.2.3.1 Relationship to fairness

Ethical AI brings up privacy and fairness as the two main aspects to support human safety [28]. Fairness concerns whether the outcomes of AI systems are biased depending on the sensitive attributes of data subjects. Bias, however, is divergent, and thus needs further criteria based on social justice, regarding discrimination, to determine whether bias at hand is threat to fairness. Group fairness addresses bias among groups categorized with different sensitive attribute values, while individual fairness addresses whether outcomes for a particular data subject are biased when compared with the other members in the group that the data subject belongs to.

Fairness has much in common with privacy in that it involves sensitive information such as personal data of data subjects. On the other hand, fairness depends on the system requirements because the notion of bias is based on high-level concepts of social justice, while privacy is discussed from general criteria of whether the sensitive information of the data subject is leaked, which refers to legal regulations.

10.2.2.3.2 Relationship to information security

Privacy is related to information security because privacy violation is considered a specific type of information leakage. Information security has been discussed in terms of three properties (CIA): Confidentiality, Integrity, and Availability [78]. Basically, CIA protects the information managed within the system by means of controlling accesses to the information according to the granted authorization. Specifically, confidentiality is concerned with direct information leakage of information against unauthorized accesses.

Privacy focuses on the issue of information leakage about data subjects, even when the access to the information is legitimately allowed, and is concerned with indirect information leakage of information. Privacy is, indeed, related to confidentiality in that both deals with information leakage, and privacy was once discussed as a part of the information security (CIA+P) [181]. However, in the case of privacy, the target of threats is not the information managed within the system. They are data subjects associated with sensitive information, located outside the system. This protected information is inferred by an indirect method using both system output information and supplementary information available separately (background knowledge).

Because of the inherent technical challenges, privacy has recently come to be considered an area independent of the information security. Privacy is a quality property that ethical AI should satisfy and is one of the main research challenges of machine learning [84].

10.2.2.3.3 Relationship to cybersecurity

The causes of privacy issues can be discussed in relation to cybersecurity when considering

unexpected access to information occurred. Training data identification (Section 10.2.2.2.1) can be divided into two specific methods, either black box or white box. Black box methods make use of results obtained through regular execution such as authorized access, and do not go into the details of the trained model. In a sense, the uses are malicious such that developers and operators had not considered. On the other hand, the white box method uses the internal information that makes up the trained model such as machine learning model, training parameter values, etc.. If the information is obtained through cybersecurity attacks such as model theft, which is the theft of the trained model itself or the learning parameters, it may introduce further threats of indirect information leakage with the white box methods. Privacy and cybersecurity may overlap significantly and require multifaceted measures to counter both threats.

10.2.2.4 Data protection impact assessment

10.2.2.4.1 Social demand

Personal data protection is to reduce the threat that protected data will be re-identified. From the beginning of the development of information systems, including AI, data protection processing techniques are introduced according to the principle of “privacy by design”. Once the system is completed, the impact of the adopted methods on data protection is to be assessed. The GDPR requires the implementation of data protection impact assessment (DPIA) [Article 35 of the GDPR]. In addition, other guidelines also discuss the need for similar privacy risk assessments [248], [48].

Even if the business is in areas not covered by regulatory laws related to data protection (e.g., GDPR), it is important to mitigate business risk by reducing the likelihood of data protection failures during operations.

10.2.2.4.2 Data protection impact assessment tool

There are some research projects to provide technical solutions to DPIA now underway. Below indicates some projects as of writing this document.

ML Privacy Meter [162] is being developed under the support of a national project in Singapore called AI Singapore. It is a checking tool for training data identification for trained models and is intended for use in risk analysis. ML Privacy Meter provides a set of tools based on black box and white box methods for membership inference, as well as a support for estimating ε values needed for requested protection strength when machine learning mechanisms based on differential privacy (e.g., DP-SGD) are used.

ML-Doctor [148] is a systematic assessment tool being developed in a project of the German

National Institute, providing an extensible framework for checking the degree of data protection from various perspectives. It provides membership inference, model inversion, and attribute inference as features to assess the protection strength of trained models in a systematic manner.

10.2.3 Privacy quality levels defined for various artifacts

This section introduces privacy quality levels for different types of artifacts: machine learning systems to be delivered as final products and the other types of artifacts to be delivered for reuse. Since privacy is intended to protect the rights of data subjects, compliance with the underlying regulatory law is mandatory. The required level of protection is achieved through a combination of technical and institutional measures. Technical measures include not only those directly related to data protection, but also information technologies that guarantee technical robustness, such as safety, reliability, and countermeasures to cybersecurity.

Generally, those discussions are divided into the technical measures related to data protection and the other information technology and institutional measures, and the presentation below focuses on the data protection technology. It is because increasing the level of data protection pre-processing reduces the cost of institutional measures and provides providers an incentive to spend their efforts on the data protection technologies. Apparently, if a preliminary analysis shows that no personal data are involved in the target system, there is no need to consider the privacy quality level at all.

10.2.3.1 Machine learning systems

Organize the protection targets following the discussions on machine-learning-based systems (Section 10.2.2.1.1). Measures against the threat to training data identification are dependent on data protection measures during the development (Section 7.1.4.2). The required quality levels that the data protection technology is expected to achieve are divided into the following as below. General technical and organizational measures are employed to address those installation and operation-time threats.

Quality Level 0 (AIPrLc0): No **data** protection measures are taken. Use state-of-the-art training methods that focus on improving the generalization performance.

Quality Level 1 (AIPrLc1): Reduce threats to training data identification by adjusting the data distribution in the training dataset and introducing safeguards, in addition using the latest machine learning algorithms. Moreover, tool-assisted data protection impact assessments will be conducted, if necessary.

Quality Level 2 (AIPrLc2): Privacy-preserving machine learning methods are used giving highest priority to data protection. In addition, the threat of training data identification will be

reduced by adjusting the training data distribution and introducing safeguards. Moreover, tool-assisted data protection impact assessments will be conducted.

10.2.3.2 Artifacts reused in further development

When providing artifacts outside so as to be used in further development, the provider of the artifacts should be responsible for the quality level regarding data protection for the artifacts, depending on the scope of the reuse approved.

10.2.3.2.1 Training datasets

When personal data are included in the training dataset itself, it is a group of unprocessed data of the protection processing level 0, and thus needs some protection pre-processing. In general, the required quality level is determined by those whom the training dataset is delivered to. If the training dataset is reused within the organization that prepares the training dataset, the required quality level to be guaranteed is moderate, because the artifacts are under the organization's control. The level increases if the artifacts are delivered outside of the initial organization, because the third parties are to have responsibility. Furthermore, the quality level should be determined by considering the cost of the institutional countermeasure spent to achieve the strength of the privacy protection as specified by underlying regulations.

Quality Level 0 (AIPrLd0): Protection processing is applied to the training dataset as a measure against the re-identification. The other technical and Institutional measures should be implemented, as a precondition, to prevent the re-identification threats, which refers to the appropriate management of information that are related to the protection pre-processing and thus potentially employed in re-identification methods.
Quality Level 1 (AIPrLd1): Protection processing is applied to the training dataset as a measure against the re-identification. The protection measures should be such that the re-identification requires the external auxiliary information or background knowledge of the protection processing level 2. The privacy-preserving synthetic data may be used to achieve the required level of protection.
Quality Level 2 (AIPrLd2): Privacy-preserving synthetic data methods are used as a countermeasure against the re-identification. In addition, as tool-assisted data protection impact assessments, training data identification for the generative model used in privacy-preserving data synthesis is performed to confirm that threats are kept below the required level.

10.2.3.2.2 Pre-trained models

When the training data refer to personal data, the pre-trained model itself is a kind of de-

identified data of the protection processing level 2, but may be subject to known methods of the training data identification. Therefore, it is necessary to apply protection pre-processing so as to make the re-identification difficult. In general, the required quality level is determined by those whom the pre-trained model is delivered to. If the pre-trained model is reused within the organization that prepare the training dataset, the required quality level to be guaranteed is moderate within the base organization, because the artifacts are under the organization's control. The level increases if the artifacts are delivered outside of the initial organization, because the third-parties are to have responsibility. Furthermore, the quality level should be determined by considering the cost of the institutional countermeasure spent to achieve the strength of the privacy protection as specified by underlying regulations.

Quality Level 0 (AIPrLm0): No data protection measures are taken. Use state-of-the-art machine learning algorithms that focus on improving the generalization performance. Strict institutional measures are taken by the organization.

Quality Level 1 (AIPrLm1): Use state-of-the-art machine learning algorithms that focus on improving the generalization performance, using training datasets after protection pre-processing against the re-identification. Moreover, tool-assisted data protection impact assessments will be conducted, if necessary.

Quality Level 2 (AIPrLm2): Use privacy protection learning methods to counter re-identification threats. In addition, the use of training datasets pre-processed with privacy-preserving synthetic data methods will be considered. Moreover, tool-assisted data protection impact assessments will be conducted.

10.3 AI security

This section summarizes the security of machine-learning-based systems.

First, the overview is explained (Section 10.3.1), followed by the taxonomy of the damages caused by attacks on machine learning-based systems (Section 10.3.2), the taxonomy of threats specific to machine learning, and the explanation of the analysis of assets, stakeholders, attack interfaces, and attackers (Section 10.3.3), as well as threats, vulnerabilities, and controls specific to machine learning (Section 10.3.4).

Based on these, the overall picture of quality management of security for machine-learning-based systems is summarized (Section 10.3.5). AI security controls will be comprehensively and systematically presented for each asset in the system design, development, and operation phases. For further details on controls refer to Chapters 7 and 8.

Related documents are introduced for reference (Section 10.3.6). Furthermore, as reference information, matters to be noted from the security perspective for each chapter of the Guideline are presented (Section 10.3.7).

10.3.1 Overview

10.3.1.1 Importance of AI security

As with conventional information systems, machine-learning-based systems are vulnerable to attacks from outside the system. One of the best-known attacks is an evasion attack, which causes a trained model to malfunction with malicious input data.

Such malfunctions of trained models may degrade the quality-in-use of the system and cause damage to the system, operators, users, and third parties. Thus, security controls to prevent or mitigate attacks are important, especially when the malfunction of trained models results in a high level of human and economic risk, such as in autonomous car and pathological diagnosis systems.

10.3.1.2 Machine learning specific attacks and their controls

Machine learning specific attacks (hereinafter referred to as ML-specific attacks) and their countermeasures have the following characteristics:

- Vulnerabilities in trained models can be mitigated by improving training methods, but are hard to be fixed completely. Therefore, it is essential to take technical controls to mitigate attacks and damage in the overall system design, such as restricting the input to trained models (Sections 7.12.1 and 7.1.5).
- Attacks during data collection, data processing, or system development may cause damage during system operation (Section 10.3.4.1). Therefore, security measures for machine-learning-based systems need to cover the entire process of the data collection and pre-processing, and the system development and operation.
- In the development and operation process of machine-learning-based systems, various stakeholders handle assets such as data and models, and have opportunities to carry out attacks. Therefore, it is important to understand assets and stakeholders throughout the life cycle of a system, and identify the attack interfaces and possible attackers (Sections 10.3.3.2 and 10.3.3.3).
- Attacks against trained models used in systems are often technically difficult to detect. For example, backdoors embedded in models may not be detectable. Furthermore, input data that will cause the model to malfunction cannot be detected in most cases. Additionally, attackers may perform pre-attacks for attacking trained models. Therefore, it is recommended to implement a multi-layered defense strategy to counter attacks that are difficult to detect or involve multiple stages (Section 10.3.3.3.2).

10.3.1.3 Basic security concepts

The Guideline defines assets, threats, vulnerabilities, and security controls for machine-learning-based systems following general information security frameworks, such as ISO/IEC 27000 series [10]. Figure 18 illustrates the basic concept of security for machine-learning-based systems.

Figure 18: Basic concept of security for machine-learning-based systems

Asset is something that has value to an organization and should be protected. Examples of assets related to machine learning include data, models, and systems, which are explained in detail in Section 10.3.3.2.

A *threat* refers to a factor that can cause damage to an asset or the organization owning the asset (i.e., a factor that causes a risk). Typical examples of *threats* include spoofing, falsification/tampering, repudiation, information disclosure, denial of service, and elevation of privilege. Specific threats to machine learning are shown in Section 10.3.3.1.

A *control* refers to measures implemented to protect assets from potential threats (i.e., measures to modify or maintain risks). Section 10.3.4 introduces control measures against threats specific to machine learning.

A *vulnerability* refers to a weakness in an asset or in a security control that can be exploited by a threat. In Section 10.3.5, vulnerabilities to ML-specific threats can be broadly classified into (1) insufficient risk assessment based on credibility checks, (2) insufficient prevention and mitigation measures for attacks, (3) insufficient risk assessment using attack detection techniques, and (4) insufficient prevention and mitigation measures for damage.

10.3.1.4 Scope of AI security quality management covered in this Guideline

This Guideline presents a quality management method for the external quality AI security

(Section 4.6) so that the readers of this document can consider security specific to machine learning from the design phase of machine-learning-based systems. In quality management, the developers should conduct risk assessments (Section 4.6.2.1) for the entire system lifecycle to implement security controls (Section 10.3.5), as in the case of conventional information systems.

Risk assessment and security controls are closely linked to the quality management presented in other chapters of this Guideline. Damage caused by attacks is characterized as a "deterioration in external quality and quality-in-use", and security control measures are defined as "evaluation and improvement of internal quality".

Note that the external quality AI security in the Guideline deals with security against "risks arising through trained models" (Section 4.6.1). Refer to existing standards and frameworks for the security controls common to conventional information systems (Section 10.3.5.3). Furthermore, regarding (1) measures against damage caused by malicious or negligent developers and operators and (2) controls against damage caused by non-human factors such as natural disasters and infrastructure failures, controls for conventional information systems should be implemented, and are not addressed in this Guideline.

Additionally, this guideline section focuses on supervised learning. Security controls for unsupervised, semi-supervised, reinforcement, and distributed learning will be addressed in future editions.

10.3.2 Damage to machine-learning-based systems

We present an overview of the damage caused by threats on machine-learning-based systems.

10.3.2.1 Taxonomy of damage caused by threats

In this Guideline, damage due to threats against machine-learning-based systems (hereinafter referred to as *system*) are classified as follows.

1) Breach of integrity/availability (Section 0)
   a) System malfunction due to unintended behavior of machine learning components
      a1) caused by a malfunction of a trained model (Section 10.3.2.2.1)
      a2) caused by a malfunction of an interpretation functionality of a trained model (Section 10.3.2.2.2)
      a3) due to an unintended functionality of a trained model (Section 10.3.2.2.3)
   b) System malfunction due to other factors (Section 10.3.2.2.4)
   c) Computational resources exhaustion due to machine learning components (Section 10.3.2.2.5)
   d) Computational resources exhaustion due to other factors (Section 10.3.2.2.5)
2) Breach of confidentiality (Section 10.3.2.3)

e) Information leakage on a trained model (Section 10.3.2.3.1)
f) Leakage of sensitive information on training data (Section 10.3.2.3.2)
g) Leakage of other confidential information (Section 10.3.2.3.3)

Table 6 summarizes damage categories and associated threats.

Table 6: Damage categories and associated threats to machine-learning-based systems

10.3.2.2 Breach of integrity and availability

10.3.2.2.1 System malfunction caused by malfunction of a trained model

Attacks that cause malfunction of a trained model can prevent achieving the functional requirements of a machine-learning-based system using that model. For example, it can cause the following impact.

- Reduction of risk avoidance
  - Failure to detect objects during automated driving; missing abnormalities of

drivers when providing driver assistance
  - Bypassing malware detection of information security controls
  - Failure to detect intrusions and anomalous behavior in security systems
  - Failure to perform facial recognition and other biometrics
  - Increase in false positives and false negatives by pathological diagnosis systems
- Decrease in AI performance
  - Inefficient allocation of vehicles in transportation and logistics; increase in traffic congestion and logistics costs
  - Decrease in product recommendation accuracy, and demand prediction in the retail sector
  - Drop in appropriate enrollment, hiring, and staffing
- Decrease in fairness
  - Discriminatory lending by credit screening systems
  - Unfair admissions, hiring, and staffing by personnel rating systems
  - Discriminatory criminal risk assessments by crime prevention systems

Furthermore, a malfunction of a trained model may cause the system to malfunction, resulting in the loss of privacy and drop in other quality-in-use characteristics.

Attacks specific to machine learning can cause a malfunction of a trained model are as follows: data poisoning attacks (Section 10.3.4.1), model poisoning attacks (10.3.4.2), exploitation of poisoned models (Section 10.3.4.3) and evasion attacks (Section 10.3.4.5).

A malfunction of machine learning components may be caused by conventional attacks against the software/hardware implementing the machine learning components, such as buffer overrun and fault injection attacks. Controls against such conventional attacks are analogous to the security controls for conventional information systems, hence not addressed in this Guideline. (The same applies Sections 10.3.2.2.2 and 10.3.2.2.3 described later).

10.3.2.2.2 System malfunction due to a malfunction of interpretation functionality of a trained model

If there is a functionality that interprets the behavior of the trained model, the attacks that cause malfunction this functionality can degrade the system's quality-in-use and worsen the system’s transparency and accountability. For example, there are attacks known to reduce the value of the explanations provided by the interpretation functionality or generate incorrect explanations [102], [204].

Attacks that cause malfunction of interpretation functionality are as follows: data poisoning attacks (Section 10.3.4.1), model poisoning attacks (Section 10.3.4.2), and evasion attacks (Section 10.3.4.5) described later.

10.3.2.2.3 System malfunction due to an unintended functionality of a trained model

The behavior of trained models depends on the training datasets used to train the models. If a model is trained using an unexpected different dataset, it may learn a functionality not expected by the developer. In this case, even if the trained model works appropriately, the system can malfunction as the trained model will not meet the system's requirements.

Attacks that cause a system malfunction by using a model with an unintended functionality are as follows: data poisoning attacks for functionality changes (Section 10.3.4.1) and model poisoning attacks for functionality changes (Section 10.3.4.2).

10.3.2.2.4 System malfunction caused by other factors

System malfunctions may be caused by the malfunction of system components other than machine learning components. Measures against such malfunctions are the security controls for conventional information systems, which are not specific to machine learning and hence not addressed in this Guideline.

10.3.2.2.5 Computational resources exhaustion

A type of attack that violates availability is an attack that intentionally exhausts computational resources. *Sponge attack* (Section 10.3.4.6) inputs malicious data and causes machine learning components to exhaust computational resources. In addition, there are data poisoning attacks (Section 10.3.4.1) and model poisoning attacks (Section 10.3.4.2) that exhaust computational resources. Other attacks, such as conventional attacks on the software and hardware that implement machine learning components, and conventional attacks on the system, intentionally exhaust computational resources and compromise system availability.

10.3.2.3 Breach of confidentiality

10.3.2.3.1 Leakage of information on a trained model

Attacks that leak private information, such as the parameters and functionality of a trained model, may result in the leakage of trade secrets related to the functionality of the trained model. It may also be used for different attacks against the trained model as given below.

- Information about the trained model may be used to generate adversarial examples that malfunction the model or its interpretation functionality.
- Information about the trained model may lead to the leakage of privacy information in the training data used to train the model.

An ML-specific attack that leaks information about trained models is known as a *model extraction attack* (Section 10.3.4.4).

There are also conventional attacks not specific to machine learning: those against vulnerabilities in the development software and development environment (Section 10.3.3.3.2.1), or vulnerabilities in the system, the computing environment, and the operation organization during system operation (Section 10.3.3.3.2.2). Controls against these attacks are the same as those for conventional information systems and are not covered in detail in this Guideline.

10.3.2.3.2 Leakage of sensitive information on training data

When sensitive information is included in training data, an attack to leak training data information may result in the privacy breach, the leakage of trade secrets, and the violation of laws, regulations, and contracts. An attack to compromise training data may cause damage to third parties, for example, if the training dataset includes sensitive information on the third parties, such as medical personal data, customer sales information, or photos of military facilities where photography is prohibited.

Notable examples of ML-specific attacks that leak information on training data are *membership inference attacks* and *model inversion attacks* (Section 10.2.2.2). In these attacks, information on training data used to train a model is compromised by observing the input and output behavior of the model. In this chapter, these attacks are collectively referred to as *attacks for information leakage of training data* (Section 10.3.4.7).

On the other hand, there are conventional attacks that are not specific to machine learning to steal data directly. Controls against this type of attack are the same as conventional information security controls and hence not addressed in this Guideline. As a control to prevent conventional data theft, there is a technology (secret computation) for computing encrypted data without decrypting them, and has been actively studied for its applications to machine learning [160], [183], [73].

10.3.2.3.3 Leakage of other confidential information

In model poisoning attacks (Section 10.3.4.2) for information embedding [206], sensitive information is embedded in model parameters, outputs, etc., and leaked during system operation. This may result in the privacy violation and leakage of trade secrets.

10.3.3 **Attacks on and damage to machine-learning-based systems**

In this section, attacks against machine-learning-based system are described. First, we describe the taxonomy of threats specific to machine learning (Section 10.3.3.1). Next, assets and

stakeholders are listed (Section 10.3.3.2), attack surfaces of machine learning-based systems and possible attackers are presented (Section 10.3.3.3), and information that can be used for ML-specific attacks are described (Section 10.3.3.4).

Also, research regarding machine-learning-based systems is being actively pursued, and understanding the latest trends is important [90] [226] [161]. For example, it is recommended to collect information regarding attack cases related to the development target and the latest cyber security articles (Black Hat, DEF CON, etc.).

10.3.3.1 Taxonomy of threats specific to machine learning

A taxonomy of known machine learning-specific threats to systems developed using supervised learning [135] is shown in Table 7. T1.1 and T1.2 in Table 7 are threats that can be assumed during system development. By contrast, threats T2.1 to T2.5 occur during system operation. In this section, threats T2.1 to T2.5 are collectively referred to as "input of malicious operational data". Section 10.3.4 provides details of each threat.

Table 7: Classification and descriptions of threats specific to machine learning

10.3.3.2 Analyses of assets and stakeholders

In security risk assessment, we first list the assets and stakeholders that appear in the lifecycle of a machine learning system. Typical assets and the stakeholders [135] who may deal with the assets are shown in Table 8.

Table 8: Examples of assets and stakeholders in the lifecycle of machine-learning-based systems

The *learning mechanism* discussed in this section refers to the entire software for developing a model using a learning dataset (and pre-training model), and typically includes hyperparameters, training programs, testing programs, and development software.

Furthermore, a *model provider* refers to a stakeholder that provides a pre-trained model. A *model user* refers to a stakeholder (e.g., system developer) that develops a model using a pre-trained model.

10.3.3.3 Analysis of attack surfaces and attackers

10.3.3.3.1 Assets that can be manipulated and assumptions regarding attackers

Analysis of threats and vulnerabilities of machine-learning-based systems lists the possible attackers and the attack surfaces (set of paths and methods that can be accessed and exploited by attackers) of the system [135].

Table 9: Examples of machine-learning-specific threats, attack surfaces, attack execution phases, attackers, and attack methods

For each type of threat specific to machine learning, examples of assets that make up the attack

surfaces, attack execution phases, possible attackers, and attack methods are presented in Table 9. Among these possible attackers, based on the analysis of assets and stakeholders (Section 10.3.3.2), we assume a stakeholder (Table 8) that may handle assets of attack interfaces. For example, since the training data provider handles the training dataset, it is necessary to assume that the training data provider is an attacker in a data poisoning attack.

10.3.3.3.2 Preparatory attacks for ML-specific attacks and multi-layer defense

In an attack against a machine-learning-based system, an external attacker may combine ML-specific attacks with other attacks to perform a multi-stage attack (Section 10.3.3.1). Specifically, they may perform preparatory attacks to conduct an ML-specific attack. Therefore, system developers and operators may be able to deter, prevent, and mitigate ML-specific attacks by taking control measures against preparatory attacks, and so examining multi-layer defense is recommended.

Therefore, this section describes preparatory attacks to conduct ML-specific attacks. Conventional information security measures are used to counter the pre-attacks, which will be described below.

10.3.3.3.2.1 Preparatory attacks exploiting vulnerabilities in the development software and the development environment

Manipulations of training datasets, learning mechanisms (including training programs), and trained models may cause data poisoning attacks (Section 10.3.4.1) and model poisoning attacks (Section 10.3.4.2), which can be conducted through pre-attacks that exploit vulnerabilities in the development software and environments.

For example, a model poisoning attack may be possible through a preparatory attack that exploits vulnerabilities in software libraries for machine learning, such as TensorFlow and PyTorch, to install backdoors in the development software or in the trained models.

10.3.3.3.2.2 Preparatory attacks exploiting vulnerabilities in the system, the computing environment, and the operation organization during system operation

Manipulation of the trained models in model poisoning attacks (Section 10.3.4.2) and input during system operation in evasion attacks (Section 10.3.4.5) and attacks for information leakage of training data (Section 10.3.4.7) can be conducted through preparatory attacks that exploit vulnerabilities in the system, the computing environment, and the operation organization during system operation.

For example, it may be possible to steal trained models and execute white-box evasion attacks or attacks for information leakage of training data through preparatory attacks exploiting

vulnerabilities, e.g., unauthorized access to the system, reverse engineering of the trained model, or side-channel attacks. In some cases, attackers may be able to steal trained models and execute a white-box evasion attack or attacks for information leakage of training data. If an attacker has physical access to the hardware, a fault injection attack or hardware trojan attack may cause a malfunction of the trained model [235].

10.3.3.4 Information used for ML-specific attacks

In attacks on ML-based systems (Section 10.3.3.1)),) external attackers may collect information in advance to carry out ML-specific attacks. Therefore, the system developers and operators may be able to deter, prevent, or mitigate the ML-specific attack by preventing and mitigating such information collection.

In this section, we will discuss certain information that can be used in the context of ML-specific attacks.

10.3.3.4.1 Access to the models during attacks

Attacks against trained models are classified in terms of the attacker's knowledge as follows:

- White-box attacks, which use parameters of the trained model
- Black-box attacks, which do not use the parameters of the trained model
- Gray-box attacks, which are between the white-box and the black-box attacks

A white-box attack assumes a situation where an attacker has access to the parameters and other information of the trained model. Typical cases are as follows:

Case (1): A publicly available trained model is used in the system

Case (2): The attacker has stolen some parameters of the trained model in advance

In the case (2), to steal parameters and other information of the trained model, attackers conduct preparatory attacks that exploit vulnerabilities in the development software and environment (Section 10.3.3.3.2.1) and in the system, the computing environment, and the operation organization during system operations (Section 10.3.3.3.2.2).

A black-box attack assumes a situation where the attacker does not need to access the parameters of the trained model but can input data into the trained model. Typically, the attacker provides input to the model or system under attack and observes the output to obtain the information that will be used for the attack. Even if attackers cannot fully observe the output of the model or system, they may generate adversarial data in advance and input them into the system.

10.3.3.4.2 Prior knowledge about specifications, models, datasets, etc.

Exploitation of the specification information: Specification information of the system and the trained models can be used for ML-specific attacks. For example, information about the learning algorithm and hyperparameters for training the model can be used in a data poisoning attack (Section 10.3.4.1).
Exploitation of the model: Information on trained models or similar models can be used to generate adversarial examples for evasion attacks (Section 10.3.4.5), and to generate inputs to the models in membership inference attacks (Section 10.3.4.7). Therefore, preventing attackers from obtaining information about the trained model may help in mitigating the attacks. Specifically, the risk of evasion attacks can be greater when the system is built using publicly available trained models without modifying them.
Exploitation of the dataset: The training dataset or similar datasets can be used to build a model that approximates the trained model under attack, and can be used to mount evasion attacks and membership inference attacks. Therefore, preventing attackers from obtaining the training dataset or similar datasets may help in mitigating the attacks. Note that the risk of evasion attacks can be greater when the model is trained using only a single publicly available dataset.

10.3.4 Threats, vulnerabilities, and control measures specific to machine learning

This section describes threats specific to machine learning and introduces control measures against vulnerabilities of assets exploited by threats. For details on vulnerabilities and controls, we recommend referring to [135].

Furthermore, Section 10.3.5 summarizes control measures for each asset and presents the corresponding internal quality.

10.3.4.1 Data poisoning attacks and their control measures

10.3.4.1.1 Overview of threats

A data poisoning attack modifies the source of training data or training datasets to cause (i) unintended behaviors of the trained models, (ii) exhaustion of computational resources due to the trained models, or (iii) leakage of sensitive information from the trained models.

This attack is executed during the dataset collection and pre-processing phase or during the model training phase, and damage occurs during the system operation phase. A data poisoning attack malfunctions the system, exhausts computing resources [96], or decreases risk avoidance, AI performance, and fairness [205], [156]. The attack can also induce the leakage of sensitive

information [154] and may also cause privacy breaches.

Manipulations of datasets in data poisoning attacks are classified into (i) data injection, (ii) data modification, and (iii) label manipulation [226].

The modification targets of data poisoning attacks are not only training data, but also validation data and test data. If validation or test data are manipulated, the developers cannot evaluate a trained model correctly and may miss attacks. Then, the trained model may malfunction during operation.

Typical attackers who manipulate datasets and their situations are:

(1) A third party manipulates the source or environment of the data (the population)
(2) The data provider manipulates the dataset
(3) The person who constructs and processes the dataset manipulates the dataset
(4) A third party manipulates the dataset in the process of providing or using the dataset

There are the 4 types of data poisoning attacks:

(a) Malfunction attacks
    1. Targeted attacks: Attacks that cause a malfunction of a trained model or its interpretation functionality for specific inputs to the model during operations
    2. Backdoor attacks: Attacks that cause a malfunction of a trained model or its interpretation functionality for arbitrary inputs that contains specific information during operations
    3. Non-targeted attacks: Attacks that cause a malfunction of a trained model or its interpretation functionality for unspecified inputs during operations
(b) Functionality change attacks: Attacks that cause a model to learn an unintended functionality
(c) Resource exhaustion attack (sponge poisoning attack): An attack that exhausts computational resources due to a trained model during operations
(d) Information embedding attacks: Attacks that train models to leak sensitive information during operation

(a1) Targeted attacks do not cause a malfunction of the trained model except for certain input data to the model. In addition, (a2) Backdoor attacks [92], [144] do not cause a malfunction for inputs that do not contain trigger information. For example, they cause the model to malfunction for images that contain a specific symbol but not for other images. For this reason, it is more difficult to detect the executions of targeted attacks and backdoor attacks.

(b) In a functionality change attack, a model with functionality not intended by the developer is trained by replacing the training dataset with another. For example, if an attacker adds a significant volume of data containing discriminatory information to the training dataset, then the trained model will output discriminatory information, against the expectations of the developer [196].

In data poisoning attacks other than functionality change attacks, the characteristics of

machine learning algorithms are often exploited to reduce the scale of the dataset manipulation. However, in a functionality change attack, since the goal is to train a model that has an unintended functionality, it tends to require larger-scale manipulations of the dataset, and the characteristics of the machine learning algorithm may not be used to execute the attack.

(d) Information embedding attacks are classified into (d1) attacks that add sensitive information to training data and (d2) attacks that manipulate the dataset to change the model's behavior and induce attacks for information leakage of training data during system operation [154].

10.3.4.1.2 Controls

Controls to prevent or mitigate data poisoning attacks are as follows:

(1) Check the trustworthiness of the data collection and processing
(2) Use data poisoning detection techniques of datasets
(3) Use techniques to improve the robustness of datasets against data poisoning
(4) Train with a robust training method against data poisoning
(5) Remove or mitigate the poisoning of trained models
(6) Implement conventional security measures against vulnerabilities in the development software and environment

To prevent data poisoning attacks, confirming the trustworthiness of the data collection and processing is important. For example, confirmation of the authenticity of the dataset, the credibility of the provider of the dataset, and the trustworthiness of the process of the data collection and processing (Section 7.6.2). However, when input data from external sources are directly used to train the model, e.g., in online learning, it is often impossible to mitigate or reduce malicious data from the dataset.

For data poisoning detection techniques, we can use methods to identify and remove outlier data from training datasets [210]. These techniques are effective for a small number of manipulated data, since they rely on the fact that poison data tend to have characteristics of outlier data. In contrast, functionality change attacks can be more easily detected by checking the contents of the dataset manually or automatically, since they require a larger scale of the dataset manipulation.

As a technique to improve the robustness of a dataset against data poisoning attacks, data augmentation [81] is effective. Using a sufficient number of data can also mitigate the impact of poisoning without decreasing the accuracy of the trained model.

Certain robust training methods such as randomized smoothing [192] and ensemble learning (e.g., Bootstrap Aggregating) [127] can be used for measures against data poisoning.

Removing and mitigating poisoning of trained models is described in Section 10.3.4.2.
To prevent dataset manipulations, conducting conventional security measures against vulnerabilities in the development software and environment is important.

10.3.4.2 Model poisoning attacks and their control measures

10.3.4.2.1 Overview of threats

A model poisoning attack modifies the pre-trained models, learning mechanisms and trained models to cause (i) unintended behaviors of the trained models, (ii) exhaustion of computational resources due to the trained models, or (iii) leakage of sensitive information from the trained models.

This attack is executed during the training and the provision of a pre-trained model or during the system development, and damage occurs during system operation. Similar to data poisoning attacks (Section 10.3.4.1), model poisoning attacks can cause trained models to malfunction, leading to decrease risk avoidance, AI performance, and fairness.

Model poisoning attacks are classified into (a) malfunction attacks (targeted attacks, backdoor attacks, non-targeted attacks), (b) functionality modification attacks, (c) resource exhaustion attacks, and (d) information embedding attacks.

As with data poisoning attacks (Section 10.3.4.1), targeted attacks and backdoor attacks malfunction trained models only for specific inputs that trigger them, while non-targeted attacks degrade the performance of trained models for unspecified inputs. (b) A functionality change attack may produce a trained model with an unintended functionality, e.g., by replacing it with a different model.

In (d) model poisoning attacks for information embedding [206], sensitive information of training data is embedded in model parameters and hyperparameters before the system operation phase, and is leaked during system operation. This may result in the violation of privacy and leakage of trade secrets.

(1) **Manipulation of a pre-trained model**

In model poisoning of a pre-trained model, attackers embed backdoors in the model in advance and provide the pre-trained model to the system developer. When the pre-trained model is used for delta development or transfer learning, the backdoors in the pre-trained model may be effective even after training, and can cause a malfunction of the model or system.

(2) **Manipulation of a trained model**

Model poisoning of a trained model can occur (i) when the model training is outsourced to an external developer, (ii) when Machine Learning as a Service (MLaaS) is used as a development platform for model training, or (iii) when a vulnerability in the development software is exploited to execute the attack.

(3) **Provision of malicious learning mechanisms and intentional modification of learning mechanisms**

In addition to (i) to (iii) of (2) above, there are other situations where a model poisoning attack is carried out against a learning mechanism (e.g., a training program in Section 2.3.1), such as (iv) using a learning mechanism created by a third party to train the model.

In federated learning, where multiple clients cooperate in model training, there is a possibility that a malicious client executes a model poisoning attack, measures against this kind of attack are being studied actively, and the details will be included in a future edition of this Guideline.

10.3.4.2.2 Controls

Controls to prevent or mitigate model poisoning attacks are as follows:

(1) Checking the trustworthiness of the process of model training and provision
(2) Using model poisoning detection techniques
(3) Removing and reducing poisoning in a pre-trained and a trained model
(4) Using learning mechanisms to remove and mitigate poisoning
(5) Conventional security measures against vulnerabilities in the development software and environment
(6) Conventional security measures against vulnerabilities in the systems, computing environment, and operation organization during system operations

To prevent the use of manipulated pre-trained models, confirm the trustworthiness of the process performing training and provision of the model. Specifically, confirm the authenticity of the pre-trained model, the credibility of the provider of the pre-trained model, and the trustworthiness of the training process of pre-trained model" (Section 7.10.3.1).

To identify the poisoning of a pre-trained model, the developers can use model poisoning detection techniques. However, these techniques cannot always detect poisoning, and there is a possibility of attacks that evade detection techniques. For example, an attack method that uses cryptographic techniques to install undetectable backdoors.

To remove or reduce poisoning in a pre-trained and trained model, the developers may process the model. For example, a method to remove backdoors in deep learning models is to prune some neural network nodes after fine-tuning (i.e., updating model parameters through additional learning) [146]. Poisoning may not be removed by processing the pre-trained model, e.g., by only removing the nodes. Furthermore, a model pre-processing may degrade the performance of the model if the additional learning is not performed with a sufficient volume of data.

In addition, the impact of poisoned models can be mitigated by using a learning mechanism that considers the output results of multiple different models, such as ensemble learning.

To prevent the manipulation of training programs and trained models, the developers should implement conventional security measures against vulnerabilities in the development software and environment (Section 10.3.3.3.2.1) and against those in the system, computing environment,

and operation organization (Section 10.3.3.3.2.2).

10.3.4.3 Exploitation of poisoned model and their control measures

10.3.4.3.1 Overview of threats

Exploitation of a poisoned model refers to an attack in which malicious data is input to the model during operation to take advantage of model poisoning and cause (a) unintended behaviors of trained models, (b) exhaustion of computational resources due to the trained models, and (c) leakage of sensitive information from trained models.

The exploitation of a poisoned model is carried out during system operation, and the damage is also caused during system operation. Exploitation of a poisoned model can cause damage similar to data poisoning attacks (Section 10.3.4.1.1) and model poisoning attacks (Section 10.3.4.2.1). However, for poisoning attacks such as targeted attacks and backdoor attacks to cause damage, an operational input that serves as a trigger is required. Therefore, this section addresses operational inputs that exploit poisoned models as a threat.

10.3.4.3.2 Controls

Controls to prevent and reduce exploitation of poisoned models include the following.

(1) Use techniques to detect, process, or restrict malicious input during operations
(2) Use techniques to prevent or reduce model poisoning (Sections 10.3.4.1.2 and 10.3.4.2.2)

A technique [101] known for processing malicious input during operations removes triggers for exploiting backdoors from input during operation. A technique [138] known for detecting malicious input during operations detects triggers.

10.3.4.4 Model extraction attacks and their control measures

10.3.4.4.1 Overview of threats

A *model extraction attack* is an attack that inputs malicious data to a trained model during operation to leak information about the attributes and functionalities of the trained model.

Model extraction attacks are carried out during system operation, and the damage is also caused during system operation. Model extraction attacks may cause the disclosure of trade secrets or other information related to the functionality of the trained model. The information of the leaked trained model may be used for evasion attacks (Section 10.3.4.5) or for attacks for information leakage of training data (Section 10.3.4.7).

In a model extraction attack, the attacker inputs data to the system in operation, and observes the output of the trained model. In other words, an attacker (e.g., a system user) is assumed to have black-box access to the trained model under attack.

Model extraction attacks include (1) attacks that leak information about the attributes of the trained model (architecture [177], hyperparameters [223], parameters [217], decision boundaries, [129] etc.), and (2) attacks to leak information about the trained model's functionalities [202], [178].

10.3.4.4.2 Controls

Controls to prevent or mitigate model extraction attacks are as follows:

(1) Use techniques to detect, process, or restrict input that performs model extraction attacks during operations
(2) Process the output of the model
(3) Use ensemble learning
(4) Use model extraction risk assessment techniques

In addition, the following methods can be used to prevent the use of trained models that have been leaked through model extraction attacks.

(5) Embedding a digital watermark in the model

To detect model extraction attacks, the developers can use a technique that observes the distribution of a series of input data to a trained model and detects inputs for model extraction (PRADA [129], etc.). Furthermore, to inhibit model extraction, a technique [118] has been proposed for processing input during operation.

Measures for model output information are: not outputting confidence values, rounding of confidence values, and adding perturbations to confidence values. While such output modification may help mitigate some model extraction attacks, it often does not mitigate the attacks very well.

Ensemble learning has the potential to reduce model extraction by using multiple models together.

There are tools to evaluate the risk of model extraction, e.g., ML-Doctor [148].

Additionally, digital watermarking technology [218], [66] has the potential to be used for protecting the intellectual property of trained models. Although digital watermarks cannot reduce or prevent model extraction attacks, they may be able to locate the original owner of a trained model and deter the use of stolen trained models.

10.3.4.5 Evasion attacks and their control measures

10.3.4.5.1 Overview of threats

An *evasion attack* is an attack that input malicious data (called *adversarial examples*) to a trained model or a system during operation to cause a malfunction of a trained model.

Evasion attacks are carried out during system operation, and the damage is also caused during system operation. Evasion attacks can cause systems to malfunction, reducing risk avoidance and AI performance. For example, in an evasion attack against an image classifier, an adversarial example is a carefully perturbed image input that appears natural to human eyes but causes the image classifier to misclassify the given image data.

In an evasion attack, a system user is assumed as the attacker. In other words, we assume a situation where a system user inputs adversarial examples to the system during operations. If the system operator uses the system, the attacker is assumed to be the provider of the input data for the system operation.

There are two types of evasion attacks: *white box attacks* and *black box attacks*. These two types have a common aspect of causing malfunction by providing malicious data to the system during operations, whereas they differ in the process and method of generating adversarial examples.

In white box attacks [213], [117], adversarial examples for a particular trained model are prepared in advance using internal information about the model (e.g., parameters of the model). Then the adversarial examples are input into the system during operations.

In contrast, black box attacks [180] generate adversarial examples without using any internal information about the trained model. Typically, to generate adversarial examples, the attacker inputs multiple data to the system during operations and observes the output from the system. Thus, this kind of attack may be prevented or mitigated by restricting the input data to the system or observation of the output.

However, some black box attacks may not require many input and output pairs of the system. Such attacks are realized by exploiting the *transferability* of adversarial examples (i.e., the tendency for adversarial examples against other trained models to also be adversarial examples against the trained model under attack). For example, there are attacks that generate adversarial examples by using (1) an approximate model that mimics the input and output behavior of the attacked trained model or (2) another model trained using another dataset that resembles the training dataset.

Evasion attacks are classified into two types in terms of the *error specificity* of the trained model [77].

(a) *Error-generic*: Attacks that cause some unspecified malfunction of the trained model
(b) *Error-specific*: Attacks that cause a specific malfunction of the trained model

By convention, (a) is often referred to as *non-targeted* and (b) as *targeted*.

There is another classification of evasion attacks regarding the range of adversarial inputs during system operation or attack specificity.

(i) *Indiscriminate*: Attacks that cause malfunctions of trained models for unspecified inputs during system operation

(ii) *Targeted*: Attacks that cause malfunctions of trained models for specific inputs during system operation.

10.3.4.5.2 Controls

Controls to prevent or mitigate evasion attacks are as follows:

(1) Use techniques for improving and evaluating the robustness of trained models against adversarial examples

(2) Restrict inputs to the trained model (e.g., restrictions on access rights and on the number and frequency of accesses)

(3) Use techniques to detect adversarial examples

(4) Use multiple different models and systems together

(5) Control measures to prevent or mitigate model extraction attacks presented in Section 10.3.4.4.2.

Controls against evasion attacks have been studied primarily in terms of the robustness of trained models against adversarial examples [117], [141], [69]. Methods for improving and evaluating model robustness are summarized in Section 9.8.2 of this Guideline and explained in more detail in [246]].]

- Well-known methods to improve robustness are *adversarial training* and *randomized smoothing*.
- Methods to evaluate robustness include evaluation by generating adversarial examples and adversarial verification.
- Tools to evaluate the robustness of models against adversarial examples include, Adversarial Robustness Toolbox [174], RobustBench [99], CleverHans [182], and Foolbox [189].

Note that the application of methods to improve the robustness of trained models may result in models that may leak more information on the training data. For example, adversarial training (Section 9.8.2.2) may increase the risk of membership inference attacks (Section 10.3.4.7) [208].

In general, it is difficult to remove the vulnerability of trained models to adversarial examples. Thus, if there are no restrictions on input to trained models, it is difficult to prevent evasion attacks. Therefore, to mitigate evasion attacks, it is important to restrict the input and output of trained models in the system (Section 7.12.1.1). Moreover, adversarial example detection techniques are available to suppress the input of adversarial examples to the system [233], [152],

[70]. However, these techniques often fail to detect adversarial examples and should be used only as a secondary measure.

Furthermore, evasion attacks may be mitigated by using multiple models trained using different learning algorithms and different hyperparameters (Section 7.10.3.2). However, note that due to the transferability of adversarial examples, attackers may be able to execute evasion attacks against different models effectively.

Evasion attacks that malfunction the interpretation functionality of the model (Section 10.3.2.2.2) tend to be specific to the interpretation method, and may be prevented or mitigated by using multiple interpretation methods together.

10.3.4.6 Sponge attacks and their control measures

10.3.4.6.1 Overview of threats

A *sponge attack* is an attack that input malicious data (called *sponge example*) to a trained model during operation to cause the trained model to exhaust computational resources.

Sponge attacks are carried out during system operation, and the damage is also caused during system operation. Sponge attacks can exhaust computational resources, decreasing risk avoidance and AI performance. For example, sponge attacks on neural networks [201] can increase the energy consumption and processing time of machine learning components, leading to denial-of-service conditions.

Similar to evasion attacks, sponge attacks are also of two types: *white box attacks* and *black box attacks*. Unlike evasion attacks, during interactive black box attacks, it is assumed that attackers can measure the operating time and power consumption of machine learning components.

However, some black box attacks may not require many input and output pairs of the system [201]. Such attacks are realized by exploiting the *transferability* of sponge examples (i.e., the tendency for sponge examples against other trained models to also be sponge examples against the trained model under attack).

10.3.4.6.2 Controls

Practical controls against sponge attacks include monitoring and restricting maximum consumption of energy and other resources [201]. Sponge attacks and their control measures are still under research and a detailed discussion will be included in a future edition.

10.3.4.7 Attacks for information leakage of training data and their control

measures

10.3.4.7.1 Overview of threats

An *attack for information leakage of training data* is an attack that input malicious data to trained models during operations for leaking sensitive information on the training data used for training models during operations.

This attack is carried out during system operation, and the damage is also caused during system operation. This attack can lead to the leakage of sensitive information on training data, possibly resulting in the leakage of privacy or trade secrets.

There are two types of attacks for information leakage of training data: *white box attacks*. In a white box attack, an external attacker obtains a trained model and then observes the behavior of the model. In this case, the attacker does not need to observe the input and output of the system. On the other hand, a black box attack assumes a system user as an attacker, who inputs data into the system during operation and observes the output (and possibly other internal information) of the system.

Attacks for information leakage of training data include the following (Section 10.2.2.2.1 for details).

- Membership inference attacks (Section10.2.2.2.2) [200]
  Attacks that infer whether the training dataset used for model learning contains specific input data (membership information)
- Model inversion attacks (Section 10.2.2.2.3) [112]
  Attacks that (approximately) reconstruct partial data from the training dataset
- Attribute inference attacks (Section 10.2.2.2.3) [207]
  Attacks that infer sensitive information related to the training data
- Property inference attacks (Section 10.2.2.2.4) [72]
  Attacks that infer global properties about the training dataset

Note that data poisoning attacks may also induce attacks for information leakage of training data (Section 10.3.4.1) [154].

10.3.4.7.2 Controls

Controls to prevent and mitigate attacks for information leakage of training data are summarized in Section 10.2.2, Section 7.1.4, and Section 7.7.2 and Section 7.10.2.

10.3.5 Summary and supplementary notes on quality management for AI security

In this section, we summarize the quality management of AI security described in each section

in terms of each asset (Sections 10.3.5.1 and 10.3.5.2). We also describe an overview of security controls that are not specific to machine learning (Section 10.3.5.3).

The controls described in this section were obtained [135] through a risk-based approach to security risk assessment in accordance with the management method of ISO/IEC 27005 [11]. Specifically, for general systems developed using supervised learning, we list assets and stakeholders throughout the lifecycle and comprehensively identify possible attack interfaces and attackers for each type of threat specific to machine learning. Then, we systematize each asset's vulnerabilities and their controls. Risk assessment is explained in Section 7.1.5, asset and stakeholder analysis is explained in Section 10.3.3.2, and attack interface and analysis of attackers is explained in Section 10.3.3.3. Details of threats, vulnerabilities and controls are described in Sections 10.3.4 and [135].

When developers of machine learning systems consider control measures through security risk assessments, they should refer to the controls and literature introduced in this chapter and [135]. However, developers should consider not only the controls introduced in this chapter and the references, but also investigate the security threats and vulnerabilities specific to the system they are designing and developing, and consider and implement additional control measures if necessary.

Hereinafter, this section will mainly focus on controls against attacks from the outside of the system. Regarding (1) controls against damage caused by malicious or negligent developers and operators and (2) controls against damage caused by non-human factors such as natural disasters and infrastructure failures, we should implement controls for conventional information systems, which are not addressed in detail in this Guideline.

Table 10: Examples of security controls for model development in the system design and development phase

10.3.5.1 Summary of controls in the system design and development phase

In this section, we summarize controls that the developers of machine-learning-based systems may need to implement during the system design and development phase. Table 10 and Table 11 show the main control measures for known ML-specific threats and vulnerabilities for a system developed using supervised learning.

In Table 10, we show the control measures for assets handled during model development (sources of training data, training dataset, pre-trained model, learning mechanisms, and trained model). In Table 11, we show the control measures for assets handled during system construction (access control programs, pre-processing programs, machine learning components, post-processing programs, risk monitoring and response programs, etc.). “Input of malicious operation data” described in Table 11 is a generic term for operational threats T2.1 to T2.5 shown in Table 7.

Table 11: Examples of security controls for system construction in the system design and development phase

The classification of threats is based on section 10.3.3.1. Vulnerabilities can be broadly divided into the following 4 types.

(1) Inadequacies in risk assessment based on check of trustworthiness
(2) Inadequacies in prevention and mitigation measures for attacks

(3) Inadequacies in risk assessment using attack detection techniques
(4) Inadequacies in prevention and mitigation measures for damage

The control measures are classified into internal qualities "B-3: Adequacy of data" (Section 7.6), "C-2: Stability of trained models" (Section 7.9), and "C-3pr: Adequacy of trained models in terms of privacy". (Section 7.10.2), “C-3se: Adequacy of trained models in terms of security” (Section 7.10.3), “D-1: Reliability of underlying software systems” (Section 7.11), “D-2se: Adequacy of programs in terms of security” (Section 7.12.1).

As described in Section 10.3.1.4, the developers should design and implement security measures based on a security risk assessment for the specific system, development environment, and operation environment.

We may not need to implement all the controls listed in this section. For example, if the training data do not contain sensitive information, the developers need not implement measures against the attacks for information leakage of training data.

10.3.5.2 Summary of controls in the system operation phase

Controls that should be implemented by operators of machine-learning-based systems are summarized (Table 12). Controls are classified in internal quality "E-0: Continuous monitoring and recording during operation" (Section 8.1).

The developers should assess the security risk in advance, considering the specific system and the operation environment, design controls for system operations, and present them to the operators.

Table 12: Examples of security controls in the system operation phase

10.3.5.3 Security control measures not specific to machine learning

For general security analysis and responses to security risks that are not specific to machine learning, frameworks, such as ISO 27000 series [10], ISO/IEC 15408 (Common Criteria) [3], NIST SP800 series [50], NIST Cyber Security Framework [51], and guidelines issued by IPA (Information-technology Promotion Agency, Japan) are available.

Sector-specific guidelines and references are provided by ISACs (Information Sharing and Analysis Centers). For example, the following ISACs have been established in Japan:

- Japan automotive ISAC (https://j-auto-isac.or.jp)
- ICT ISAC Japan (https://www.ict-isac.jp/)
- Japan Electricity ISAC (https://www.je-isac.jp/)
- Financials ISAC Japan (http://www.f-isac.jp)

For business areas dealing with life and property risks but do not have ISACs, the developers can refer to security standards such as Payment Card Industry Data Security Standard (PCIDSS) [58].

Regarding the control systems in factories and critical infrastructures (IACS, Industrial Automation Control System), the developers can refer to IEC 62443 [18], [19] and the Cyber Physical Security Framework (CPSF) [42] by the Ministry of Economy, Trade, and Industry (METI).

IEC 62443 presents (1) common concepts, reference models, and roles of relevant stakeholders in general, (2) requirements and guidelines for policies and procedures for the management and operation of organizations involved in control systems, (3) security functional requirements and design and technology of security functions required for IACS, and (4) security of the components that make up the system.

METI's CPSF provides a framework for security measures in industrial society in Society 5.0. To properly capture the risk sources, industrial society is viewed as a three-layered structure involving connections between organizations, mutual connections between physical space and cyberspace, and connections in cyberspace and six components, including organization, people, components, data, procedure, and system. Then, the functions at each layer are defined as objects to be protected. CPSF presents examples of security measures from the viewpoint of possible security incidents, risk sources of incidents for each component, and requirements for security measures.

10.3.6 Related documents

Public organizations have published technical documents on machine learning security. Draft NIST IR 8269 by the U.S. NIST [48] defines terms and classifications for the security of machine-

learning-based systems regarding attack, defense, and impact. The report [60] by ENISA gives a glossary of threats to AI systems, classifying assets of the AI ecosystem in six perspectives: data, models, actors/stakeholders, processes, environment and tools, and artifacts. The 2021 report [57] of ENISA classifies threats, vulnerabilities, and controls for machine-learning-based systems. China's National Information Security Standardization Technical Committee's Draft National Standard [62] gives the security requirements, assessment methods, and evaluation metrics for machine learning algorithms.

10.3.6.1 Response to trends in international standards, laws and regulations related to cybersecurity

As of July 2023, security-related laws and regulations enacted in Europe include Cyber Security Act [33] and Cyber Resilience Act [34], subordinates of NIS2 [35]. Cyber Security Act and Cyber Resilience Act [34] refer to ISO/IEC 15408 (Common Criteria CC:2022) [3] for specification and guarantee requirements and ISO/IEC 27000 series [10] (subordinate ISO/IEC 27001 and ISO/IEC 27002 revised in 2022) for management such as proceeding with security risk assessment.

In particular, European legal systems have severe penalties, and some businesses may be affected by certification, so it is preferable to collect information on laws, regulations, and international standards in addition to those listed above.

10.3.6.2 Reference information and case studies of risk assessments

The Adversarial Threat Landscape for Artificial-Intelligence Systems (ATLAS) [54] National Institute of Standards and Technology (United States of America). NIST SP 1270: Towards a Standard for Identifying and Managing Bias in Artificial Intelligence 2022.
https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.1270.pdf

[55] by Microsoft, MITRE, etc., provides reference information for security risk assessment of machine learning-based systems. It describes threats to machine learning-based systems based on the flow of actual attacks and systematizes attack methods in stages and introduces examples of attacks. In addition, an example of risk analysis based on the second edition of this Guideline (released on July 5, 2021) is the initiative [252] of the Special Interest Group on Machine Learning Systems Engineering, which proposes a simple analysis tool for developers.

For the risk assessment for conventional information systems, the developers should refer to ISO 27000 series [10], ISO/IEC 15408 (Common Criteria) [3], and NIST SP 800-30 [50]. For the security measures of control systems in factories and critical infrastructure, the developers

should refer to IEC 62443 [18],` [19] and the Cyber and Physical Security Framework (CPSF) [42] issued by the Ministry of Economy, Trade, and Industry (METI).

10.3.7 **Security considerations (informative)**

Items that should be considered from a security perspective when implementing the provisions in the chapters of this Guideline are listed in Table 13. Developers need to check and add requirements according to the specific system they design and develop. Regarding fairness and privacy, the focus is on cases where fairness and privacy are subject to damage due to cyberattacks.

Table 13: Security considerations

Chapter and section	Guideline description	Security response plan	Remarks
1.2	service developer service provider system developer contracting, sub-contracting, or quasi-entrustment user development entruster development entrustee Self-development entity	Listing and deriving attack interfaces are necessary when conducting a security risk assessment. A list of stakeholders and system configuration should be prepared for this.	
1.3.1	it is desirable to analyze environmental conditions as precisely as possible and understand risks at the early stage of system design especially from the viewpoint of quality management of systems that require compatibility with rare environmental conditions.	Environmental conditions and risks identified in advance are used to consider the definition of security requirements and BIA.	

1.3.2	Lifetime-long requirement for risk assessment Particularly for systems with safety risks, it is essential to ensure a certain level of quality in the early stages of their operation.	When performing security risk assessment, include the results of the functional safety risk assessment and the items to be treated with priority as the targets of potential damage.	Include risks such as privacy and fairness that correspond to the target business for which the system will be used in items.
1.3.3	when a machine-learning-based system is developed by sharing works	When referring to terms and guidelines related to AI and machine learning, include compliance with referenced standards and guidelines for the entire supply chain in the contract.	Machine Learning Quality Management Guidelines (ISO/IEC 27000 series ISO/IEC 15408 Common Criteria), etc. Refer to related laws and regulations such as EU AI act, GDPR, US Executive Orders, China Cyber Security Law, etc., depending on the business to be designed and developed. Specify relevant laws and regulations, international standards, and compliance items that be complied with in contracts in the supply chain.

1.3.3	it has been pointed out at recent academic meetings that there are security risks, such as poisoning of learning results due to the intentional injection of improper data. Simple numerical evaluations cannot detect the effects of such improper data. Such detection requires checks on data sources from broader perspectives.	● Collect information on attacks specializing in AI and machine learning, such as evasion attacks, data poisoning attacks, and model poisoning attacks as illustrated in Chapters 4, 7, and 10 of the Machine Learning Quality Management Guidelines. Analyze attack scenarios, and implement measures for threats and vulnerabilities. ● Collect information on attack and defense methods and update defense measures (control measures) periodically.	● The overall review of control measures should be conducted within 1 year. ● Control measures should be prioritized and implemented starting from the possible items.
2.2.1	ISO/IEC 15408	● ISO/IEC 15408 (Common Criteria) organizes the functional and verification requirements for security. It refers to the ISO/IEC 27000 series for management aspects such as how to proceed with analysis and apply control measures. Select the one that matches the requirement analysis from ISO/IEC 15408 and ISO/IEC 27000 series, SP800-30, etc. ● Assume AI-specific attack scenarios, identify threats and vulnerabilities, and conduct study of measures.	The latest ISO/IEC 15408 is Common Criteria 2022.

2.3.1	machine-learning-based system structure	Identify attack scenarios related to attacks specific to machine learning, list information assets, and use the information to identify threats and vulnerabilities specific to machine learning.	For example, assume an attack scenario for the AI machine learning-specific attack illustrated in Section 10.3 and prioritize the management measures for threats and vulnerabilities extracted from the attack scenario.
2.3.2	Stakeholders of development and their roles	List stakeholders and information property and use it to identify attack scenarios specific to machine learning, which will help identify threats and vulnerabilities specific to machine learning.	
No. 3 item of 2.3.3	Fairness	List damages that infringe on fairness as targets of security damages. ● List the items of fairness that are subject to damage. ● Create an attack tree that targets the listed fairness items	
No. 4 item of 2.3.3	Attack resistance	The following are the viewpoints of security attack resistance. ●Apply designs that avoid attacks on the system. ● Even in the case of tampering due to an attack, suppress the impact of the tampering to a very small level so that the business is not affected. ● The calculation status before the attack remains even if attacked, all the resources are instantly discarded and replaced with the remaining resources without impacting the business.	Consider the implementation scale of each theme.

No. 5 item of 2.3.3	Ethical considerations	Confirm items of ethics and examine threat scenarios in which damage could occur.	
No. 6 item of 2.3.3	Robustness	Assume an attack tree that causes damage to robustness.	
No. 1 item of 2.3.4	System life cycle process	Implement risk assessments (gates) at each stage of the system life cycle.	Regarding the content and scale of implementation, implement items following the order of priority, with periodic implementation (within one year) in mind. (If it does not scaled a feasible scale, there is a high possibility of being reduced to a formality.)
2.3.5	Terms related to use environment	In system components specializing in AI and machine learning, and the environment where learning models are stored, AI machine learning-specific requirements are required in addition to conventional information security.	
2.3.6	Terms related to data used for machine learning	Same as above	
No. 4 item of 2.3.7	A form of operation that involves data collection and additional machine learning training to update the trained machine learning model at appropriate times.	General information security does not assume that the data used by the system will acquire new characteristics (such as fairness or privacy) during operation. The system will include periodic monitoring to ensure that the data stored by the system has acquired new characteristics through additional learning and to consider processes and procedures for the new characteristics if necessary.	

3.1.1	In cases where safety functions are required, either IEC 61508 or one of the standards for the application field and follow that process	Events assumed to cause serious damage due to security risks while examining safety considerations will result in greater damage if attacked, so such events should be handled with higher priority.	Examine safety related to systems using AI and machine learning.
3.1.3	Examining of risk scenarios related to system use	Prioritize high-risk events because the damage will be greater if they become the target of an attack.	Identify risks closely related to AI and machine learning and examine the measures for them.
3.3.1.1	quality-in-use characteristics	If damage is assumed from the perspective of the adopted quality metrics, analyze threats and vulnerabilities and take countermeasures.	When using external quality characteristics, follow the damage assumption of threat and vulnerability analysis and measures regarding external quality characteristics.
3.4	risks of physical or human harm	In this case, because the damage caused by the attack would be significant, it will be treated as a high-risk event, and risk analysis and controls will be implemented as a priority.	Identify risks closely related to AI and machine learning and examine the measures for them.
3.8.1	quality inspections	Include items related to security risks in quality inspection items and establish a process work path that returns to upstream processes for reconsideration if risk issues are discovered during inspection (any inspection item).	Include non-security risk items as necessary.

3.8.1	mixed machine learning lifecycle process	● In information security, security risk assessments are recommended in the definition of requirements phase, external specification completion phase, and design verification phase. ● In AI security, there is a possibility that a trade-off between internal quality will be required between the design and development of the system's internal quality and security control measures, so completion of security risk assessment is preferred at the "external specification completion stage", as referred in information security, by the time the design and development of internal quality is started. (Design and development of AI security control measures and other internal and external qualities should be carried out in parallel.)	● If the AI security risk assessment is a later process than the design and development of the internal qualities that comprise the other external qualities, a rework of the design may be required. ● Risk assessment at the design verification stage is the confirmation that the requirements definitions have been realized down to the last detail, it is recommended that evidence of operation confirmation be attached to the design verification results.
3.8.1	PoC development phase	Security measures are also necessary in the PoC development environment. It is recommended to conduct an AI security risk assessment in the PoC environment. At this stage, it is desirable to take measures against threats and vulnerabilities that have a greater impact on business requirements and the core design.	See sections 4, 7, 8, 9, and 10 for AI security risk assessment.
3.8.2.1	differential development	When reusing existing software components, it is necessary to determine the content of security risk assessments based on the impact on business requirements, design details, and the scale of assignable development resources for new or additional design and development parts and parts to be reused from existing components.	

3.8.3	Development process by multiple stakeholders	Plan to prepare and operate for security analysis and responses based on the supply chain.	Referring to NIST SP800-161, SP800-37, ISO/IEC 27036, etc., is recommended
3.9.1.1	The development entruster and the development entrustee agree on the purpose of the PoC phase, i.e. the PoC exit criteria, and the deliverables to the next phase.	Select policies from consideration and implementation of risk measures, risk transfer, and risk approval by listing security risks after extracting those relevant to the business from attack cases closely related to AI and machine learning when establishing consensus.	● Risk transfer: A situation in which a security requirement is delegated to another party by implementing the system or function that handles the security (e.g., development of a solution on AWS and security measures). ● Risk approval: Accepting the security risk as a business decision and neither taking any measures nor transferring the risk.

3.9.1.2	Clarifying the roles	Confirm that security risk assessments are carried out at the following stages in the progress of the work process, and decisions are made regarding the implementation, transfer, and approval of measures. ● Approval of requirements analysis results ● When the external specifications of the software is approved ● When software operation verification specifications are approved ● When software validation results are approved	In particular, check KPI items that are closely related to AI and machine learning.
3.9.1.3	Quality management during operation	● Treat relationships (e.g., correlation), properties (e.g., fairness and privacy), and hyperparameters among data included in learning data as information assets, perform periodic monitoring to anticipate possible damage and take appropriate measures. ● Periodically review from the perspective of relationship between data and nature.	

3.9.2	Utilizing the foundation model	Analysis and response to generative and foundational models from an AI security perspective will be discussed in the future edition of this document. Precautions for the current situation are as follows. ● The information learned may be made public. ● The information learned, and the contents of the learning model may be extracted. ● Pay attention to the reliability of the service providers used. ● Review the terms and conditions of the service that you will use. ● Refer to guidelines from reliable publishers such as academic organizations, companies, and government agencies, and apply appropriate guidelines to the developed services and products. ● As incidents related to the use of generative models and foundation models have already been reported, collect information and apply the necessary control measures. Examples of attacks. Prompt injection: Modifies model output Prompt leak: Leak of prompts Jail break: Avoids various filters by entering prompts and uses them for individual attacks or information leakage Particular attention should be paid to jailbreaks, as they can be implemented only by knowing the contents of the prompt input.	

		● When using company-provided APIs to develop services, include measures against conventionally known attacks such as XSS and CSRF.	
3.9.3	Utilizing AutoML services	When using external services to carry out service design and development, take control measures against known threats and vulnerabilities, not just those specific AI.	Such as XSS and CSRF.
4.2	Safety and risk avoidance	To confirm and examine threats and vulnerabilities with respect to risk targets. From the perspective of safety, human life, and economic efficiency, items with significant damage spillover are identified, risks are classified according to the magnitude of damage, assumed as attack targets, and used to analyze threats and vulnerabilities. (e.g., fairness, privacy)	
4.3	AI performance	Assuming an attack scenario in which performance is the attack target, identify threats and vulnerabilities and implement measures	
4.4	Fairness	● We will refer to Sections 4.4, 7.10, and 10.1 to envision attack scenarios targeting information that may be subject to attack (e.g., gender, skin color, etc.), identify threats and vulnerabilities, and implement countermeasures. We will collect information about the target business, regularly review the items subject to fairness attacks, and update the monitoring and controls. We will also refer to Sections 4.6.1, 7.1.5, and 10.3 for reference in our analysis.	Carry out analysis referring to Chapter 9.

4.4.3	Pre-processing approach In-processing approach Post-processing approach	Consider whether the means to realize the pre/in/post processing approaches can be included in the security risk assessment and conduct the assessment if necessary.	
4.5	Privacy	● In particular, when the change the nature of the information contained in information assets is assumed (for example, strong correlations generation or lost between multiple information), monitor and implement periodic measures. ● If the new properties acquired above are sensitive, the damage will be greater if attacked, so be careful. ● Confirm whether the new characteristics observed by monitoring violate laws and regulations such as the Japanese personal information protection law, GDPR, China cyber security law, and US CCPA.	● Pay particular attention to GDPR and Chinese Cybersecurity Law (because the regulations and sanctions are very strict) ● Collect information from international standardization organizations such as NIST and ISO/IEC regularly.
4.5.1	Laws related to the protection of personal information, such as GDPR[30] in Europe and CCPA [43] of California in the US are being developed [254].	● Use personal data protection processing regulations as a reference for attack scenarios and damage assumptions in security risk assessments. ● When planning measures, it is advisable to consider a mechanism to monitor violations of personal data protection processing regulations. ● Confirm the trend of damage cases and regulations related to personal data protection processing.	Sanctions for confirmed violations could affect management.

4.7	trade-off relationship	Trade-offs will be made between the internal quality groups required for the security control measures and the internal quality groups required to achieve the service and product's external (non-security) quality. They will be reflected in the requirement definitions, design, and development content.	The content reflected in the requirements definitions and design content differs depending on the target service and product.
4.8.2	Social aspects such as ethical considerations	Examine that the data to be used meets the legal requirements for regulations and contracts regarding the acquisition and use of data for legal rights and public interest policy reasons. Furthermore, examine the need for rights/contractual adjustments.	See [39]and [253] for reference information. Keep as up-to-date information as possible, since it is necessary to follow updates in the legal system.
5.1	Table 2: Estimation criteria for AI safety levels against human risks Table 3: Estimation criteria for AI safety levels against economic risks	For each cell in the table, an attack tree of cases with serious damage from the expected impact is assumed and prioritized.	While safety may not be considered when business solutions are built solely on IT systems, safety considerations are mandatory for safety-related embedded systems, such as brakes to mitigate collision.
5.3	Fairness	Create attack scenarios assuming a case where personal rights and assets are damaged and add it to the threat and vulnerability consideration items.	Attention should be paid to scenarios associated with additional learning
5.3	AIFL	Since the impact of the output of the relevant product and service on the individual rights and the social acceptability of the system is evaluated, anticipate the areas that would be damaged and the damage level if the attacks were successful in parallel.	

5.4	Privacy	Create attack scenarios for cases where privacy damage is anticipated and add them to the list of threats and vulnerabilities to be examined.	
5.4	AIPrL	Anticipate the impact on data subject protection when estimating damage. Pay attention to the content of relevant laws and regulations.	
6	Internal quality characteristics	Organize the status of internal quality characteristics in the system to be developed and anticipate data that will be attacked and attack scenarios that will cause serious damage, identify, organize and take measures against threats and vulnerabilities.	Refer to the reference in the Guideline to check the internal quality status of the development target system.
6	B-1: Coverage of datasets	Assuming an attack scenario in which integrity is compromised among security requirements, analyze threats and vulnerabilities by enumerating the causes that triggered the attack, and consider countermeasures.	
6.1	Sufficiency of problem analysis	Consideration should be given to whether the process results and learning models output by the system to be developed encompass sensitive information. When conducting a BIA (Business Impact Analysis)/PIA (Privacy Impact Analysis), a list of system operation flows should be made to verify the combination of situations (use case review).	Countermeasures are required if the sensitivity of the information changes in time series.

6.2	analysis from the request side, such as risk analysis, failure mode analysis and bottom-up analysis top-down analysis bottom-up analysis	During the development stage, security risk analysis should be conducted in parallel with other risk analyses, such as functional safety. When considering attack scenarios, pay sufficient attention to situations where external quality is degraded.	Conduct both conventional information security and security with a focus on AI/machine learning.
6.3	all combinations of situations identified in the problem domain analysis	Identify threats and vulnerabilities to input/output data for each combination and consider the measures.	
6.5	Uniformity of datasets	Analyze threats and vulnerabilities, anticipate attacks that interfere with the design and functionality related to dataset uniformity, and respond accordingly. Also, see the security response plan in Chapter 7 "Quality Management Characteristics".	
6.6	the absence of inappropriate modifications to the data as authenticity, and the fact that the data are sufficiently new as currentness.	Among the perspectives of security risk assessment, integrity verification is applied to the calculation of the impact of attack scenarios.	Perform a security risk assessment of the environment where the data used for learning is stored.
6.6	adequacy of data selection adequacy of labeling	When identifying the damage caused by an attack, include attacks that violate the adequacy of data selection and labeling.	Identify the causes of damage and apply them to the listing of threats and vulnerabilities.

6.6	the data need to be rearranged to accommodate the new requirements and policies	If the rearrangement of data changes the damage assumptions from an attack, the list of threats and vulnerabilities should also be updated according to the increase or decrease in damage.	Check the scope of system affected by the data rearrangement.
7	quality management characteristics	For each of the internal qualities of the machine learning components, the rationale for setting and designing the quality level determined by the developer is to assume that the most severe damage will occur, identify threats and vulnerabilities based on attack scenarios and implement responses.	From each perspective, identify and take measures to threats and vulnerabilities based on the level (LV) set by the developer and its basis.
7.1.3.1	sensitive attributes	In the attack scenarios used to identify threats and vulnerabilities, include cases where information requiring special consideration is the target.	
7.1.3.2	unfairness	Include attacks that promote unfairness in the attack scenarios used to identify threats and vulnerabilities.	
7.1.4.2	Domain analysis phase	As mentioned above, PIA is implemented to respond to privacy in cyber security.	
7.1.4.2.1	Protected data	In the attack scenarios used to identify threats and vulnerabilities, include cases where data to be protected is the target.	
7.1.4.3	Pre-stage In-stage Post-stage	If there are stage-specific development items, include them in the security risk assessment.	

7.2.2.1	attributes and their viewpoints	Check whether the attribute is sensitive information (eg skin color, race, etc.), and if it is sensitive, consider whether to stop using the attribute or take measures such as anonymization. Scrutinize the relationship between attributes (eg correlation) and the characteristics of attributes (eg fairness and privacy), identify threats and vulnerabilities based on the assumption of damage in the event of an attack, and respond accordingly.	
7.3	Sufficiency of data design	Identify and respond to threats and vulnerabilities assuming attacks that violate designed completeness.	
7.4	For example, check if there is no correlation between data other than attribute values included in combinations of cases using feature extraction or any other technique.	Carefully examine characteristics such as potential correlations between adopted and overlooked features, and if necessary, reflect these in the damage estimates for periodic security risk assessments.	
7.4	Coverage of datasets	Analyze and respond to threats and vulnerabilities by enumerating the causes triggered the attack, assuming an attack scenario compromising the integrity among security requirements.	
7.4.3	Safety	Factors that increase risk cause greater damage when an attacker attacks, so identify threats and vulnerabilities and respond based on the analysis results in accordance with requirements.	The same applies to items not in this section that consider requirements for external quality levels.

7.5	Uniformity of datasets	Identify and address the threats and vulnerabilities involved by anticipating an attack scenario that results in a condition that violates the design approach to uniformity of datasets.	
7.6 7.6.2.1	Adequacy of data Unification and scrutiny of labeling policies	Assuming an attack scenario in which an attack causes data to go against policy, identify and respond to the threats and vulnerabilities involved.	
7.6.2.2	Consistency checking and rechecking of datasets	Assessments to be conducted in response to changes in functional requirements and use environments, and the process for outsourcing work, should be included in the security risk assessment.	It is recommended to include the perspectives of information security and supply chain security.
7.6.2.3	Handling the long tail and determining mismeasurement and outliers	Anticipating an attack scenario in which stored data falls into a state violating the handling policies for long-tail, mismeasurement and outlier, identify and respond to the concerned threats and vulnerabilities.	
7.6.2.4	Address data poisoning	● Refer to the descriptions of data poisoning in Sections 4, 7, and 10.3. ● Respond to relevant threats and vulnerabilities, anticipating attack scenarios in which data are poisoned. ● Consider how to detect poisoning of data. ● Respond to system configurations and usage scenes of development target systems for security requirements other than cybersecurity.	Use guidelines (provided by ISAC and ministries and agencies) according to the business model in which the development system is used to respond to the usage scene. If guidelines are not provided, refer to the guidelines for finance and critical infrastructure.
7.6.2.5	Currentness	Identify and respond to the threats and vulnerabilities involved in a hypothetical attack scenario in which the currentness of training data is compromised.	

7.6.2.6	it is considered that quality control requires the creation of a system and structure to properly manage the process, including the management of changes in policies during development.	Establish security risk assessment rules that is applied to the entire supply chain and include in the contract regular audits and retention of audit trails in accordance with the rules.	
7.6.3	Requirements for quality levels	Reflect on each of the points listed in 7.6.2 of this table.	
7.7.1.1	Training data collection process without introducing bias	Anticipate attack scenarios where the measures aimed at eliminating bias in the system being designed and developed are compromised.	Attacks on learning models are assumed.
7.7.2.1	Accuracy Currentness	Confirm during integrity verification when performing a security risk assessment.	
7.7.2.1	measures should be implemented to prevent the leakage of personal information.	Confirm information leakage measures during the confidentiality verification when performing security risk assessment.	
7.7.2.2	Threat of privacy leakage (training data identification)	Collect information on privacy leaks, anticipate attack scenarios, derive threats and vulnerabilities, and implement control measures.	

7.8.2	in the test phase, the correctness needs to be evaluated using a test dataset that is disjoint from the training dataset, or more generally, by applying cross validation.	● Prepare and monitor means to check for damage (tampering, etc.) to the trained model regarding the behavior of the input in the training dataset. ● Prepare measures anticipating possible causes when trained models are tampered with or otherwise damaged.	Retain evidence (a record of the evaluation index) regarding the behavior for the input included in the training dataset.
7.9.2	Stability can be evaluated and improved	● Prepare and monitor means to check for damage (tampering, etc.) to the trained model regarding the response to the input in the training dataset. ● Prepare measures anticipating possible causes when trained models are tampered with or otherwise damaged.	Retain evidence of responses to inputs not included in the dataset for each of the iterative training phase, quality verification evaluation phase, and quality monitoring operation phase.
7.9.2	Overfitting	Include attacks that force the system to overfit the target when anticipating the attack scenarios.	For example, add or process improper data to training data, etc.
7.10.1	Adequacy of trained models in terms of fairness	Anticipate attack scenarios in which the learning model adjusted by the measures presented in this section is changed to something different than expected.	Refer to Figure 15 in Chapter 10.1.3.1
7.11.2	open-source implementation	Respond to publicly available threat and vulnerability information.	Refer to CWE and CVE.
8.2.2	quality deterioration	Prepare a response process and system infrastructure if the cause of quality deterioration during operation is by attack.	
9.2.2	Modeling of risk factors	Categorizing risk items according to the degree of damage, and assuming those that cause serious damage as the target of damage by attackers, extracting attack scenarios, identify threats and vulnerabilities, and consider countermeasures.	

9.6.1	Data collection policy	Assuming an attack scenario in which an attack causes data to go against policy, identify and respond to the threats and vulnerabilities involved.	
9.6.1	it is particularly important to assess whether the requirements definition has been updated together with the data collection policy, or whether it is necessary to reconfirm the contents identified in the previous stages, such as internal quality A-1 to B-4, etc.	If policies and requirement definitions have been updated, the assumption of security damage may have also changed, and if changes are necessary, they should be reflected in the threat and vulnerability analysis and countermeasures.	
9.7.1.3	the original dataset input is to be discarded.	When discarding data, depending on the device or system that stores data, completely deleting the data simply by using the deletion API is not possible, so examine and apply a method to delete the data completely.	For example, some RDBMSs do not discard the data until a complete deletion is performed. Incidents of information leakage have occurred as a result of incomplete deletion.
9.8.1.3	Fuzz Testing	If the input data are related to threats/vulnerabilities, include the test items to input fuzzing data if necessary.	
9.8.1.5	Automatic generation of test inputs	If necessary, include verification items for model evasion, model extraction, and adversarial examples, which are attacks closely related to AI and machine learning.	

9.8.2.1	regularization	In dropout, the percentage of inactive neurons is a hyperparameter. Such hyperparameters that determine the design of the neural network, optimization, etc., should be listed as information assets.	The considerations indicated in the left-hand cells also apply to other items in Section 9.8.2.
9.8.2.2	Adversarial training	If necessary, anticipate threats and vulnerabilities from attack scenarios that assume attacks using adversarial training and consider measures.	
9.8.2.5	Adversarial attack	If necessary, anticipate threats and vulnerabilities based on attack scenarios assuming adversarial attacks, and examine measures.	
9.9.1	Adequacy of trained models in terms of privacy	The learning methods introduced in this section and the parameters required by the learning methods are managed as information assets.	
9.10.2	open-source libraries	Information on libraries used for AI machine learning and data processing, such as CVE and CWE, are emerging. Therefore, collect sufficient information about the products and services to be developed.	See also Section 9.10.3.
9.10.3	Common Vulnerability Enumeration (CVE)	CVE and CWE are extremely useful as threat and vulnerability information for general information security. Information related to AI and machine learning is beginning to appear, it is necessary to collect sufficient information about the products and services being developed. Collecting information related to AI security other than CVE and CWE in parallel is recommended.	For information on AI security, see Chapters 4, 7, 8, 9, and 10.

9.10.5	Software updates	One of the recent advanced attack methods is related to software updates. Since attacks focusing on machine learning components may be carried out through malware contamination and transaction hijacking, attack scenarios should be envisioned and verified.	
9.13.1	Monitoring	● If there are security requirements for monitoring during operation, a monitoring mechanism should be placed according to the requirements. ● Relationships between features that may cause serious damage if attacked (e.g., hidden correlations), especially in sensitive cases, will also be monitored from a security perspective.	For consideration of monitoring, see Chapters 4, 7, 8, 9, and 10.
9.13.2	Concept drift	Same as monitoring	Same as monitoring
9.13.3	Retraining	Same as monitoring	Same as monitoring
10.1.1.1	Social demands	As for laws and regulations, social principles, guidelines, and international standards, assume damage in the event that a guarantee is requested or have a large impact on business, and incorporate them into attack scenario assumptions. (See Section 10.1.3 for anticipated attack scenarios.)	
10.1.1.1.1	Ethical considerations, fairness	Assuming damage and attack scenarios in the event of a breach of ethics and fairness, identify threats and vulnerabilities leading to the cause and establish countermeasures.	
10.1.1.2.1	discrimination by race, religion, etc.	Consider attack scenarios anticipating damage in order of greatest impact on business in the event of damage, due to race and religion.	

10.1.2.1	Diversity of requirements	Anticipate damage and reflect it in attack scenarios following the diversity of requirements for the products and services being developed.	
10.1.2.3	fairness requirements social demands	Consider attack scenarios in which the content of the fairness requirements or social demands is the final target of the attack.	
10.1.2.4	Hidden correlations	Anticipate attack scenarios where damage to fairness is caused by using hidden correlations.	
10.1.2.5	Variables	Anticipate an attack scenario in which variables are attacked.	
10.1.3.1	Example of a process structure for ensuring fairness quality	Among the process structures introduced in this section, quality abnormalities that differ from expectations are detected during the process of confirming quality. Explore the possibility that the quality abnormalities are due to model modifications.	
10.1.3.2	bias	Anticipate attack scenarios that violate system requirements for bias.	
10.1.3.3	sensitive attributes data with various sensitive attributes	Anticipate attack scenarios that infringe on sensitive attributes requiring special consideration.	
10.2.1.1.2	Negative externalities	Include in the attack scenario that assumes personal data re-identification as the damage of the attack.	
10.2.1.1.3	registered data activity data	Assume registered data and activity data as the damage target in the attack scenario.	

10.2.1.2.2	Consent right to be forgotten Purpose limitation Data minimization Storage limitation	Scrutinize whether consent, right to be forgotten, purpose limitation, data minimization, and storage limitation are subject to attack, and include them in the anticipated attack scenarios in the case that they are the target of attacks.	
10.2.1.3	Protected processed data	Scrutinize the protected processed data to check if it is a target of attack, and envision attack scenarios according to the protection processed level for the product and service.	
10.2.1.4	Technical privacy metrics	If an attack method that violates the metrics is found, it will be included in the scope of assumed damage. (Data similarity or Outcome indistinguishability)	
10.2.2.1.1	Case 1. Case 2. Case 3. Case 4. Case 5.	Anticipate attack scenarios for which the provided personal data, training datasets, trained models, machine learning systems, and operational systems are targeted, include them into the target of security risk assessment.	
10.2.2.2	membership inference is technically possible to identify a piece of information about the training data	The evaluation items anticipated in security risk assessments include "attacks that try to identify information about the training data from a trained model."	
10.2.2.2.2	Membership inference	Include attack scenarios with membership inference in the security risk assessment.	

10.2.2.2.3	infer sensitive information about the training data making use of information that is publicly available	Include threats and vulnerabilities related to inferring sensitive information in your security risk assessment.	
10.2.2.2.3	Attribute inference Model inversion	Include items related to attribute inference and model inversion in security assessments and evaluations.	
10.2.2.2.4	Property inference	Include items related to attacks that infer global properties through property inference attacks in security risk assessment and evaluation items.	
10.2.2.3.3	Black box method white box method	Consider attack scenarios for both black box and white box methods.	
10.2.2.3.3	Relationship to cybersecurity	As an effort toward privacy in cybersecurity, PIA (Privacy Impact Analysis) will be conducted alongside BIA (Business Impact Analysis) in the requirements definition stage of system development. The content of PIA is defined in ISO/IEC 29134 (JIS X 9251). In particular, ANNEX B provides examples of threats to privacy and can be used as a reference when creating a threat list for security risk assessment. In addition, refer to the "Promotion of PIA Efforts" provided by the Personal Information Protection Commission as a practical guideline for conducting PIAs.	

10.2.2.4.1	GDPR	NIS2 obligates competent authorities to notify data protection authorities regarding incidents deemed as personal data breaches based on GDPR, and stipulates penalties. Preparation of a system to investigate and report as required is recommended.	
10.2.2.4.1	protected data will be re-identified	Include the damage of re-identification of protection processed data in the attack scenario assumed in the security risk assessment.	
10.2.3.1	Quality level 0 Quality level 1 Quality level 2	Introduce external penetration testing based on the quality level in parallel with the security risk assessment.	
10.2.3.2	Quality level 0 Quality level 1 Quality level 2	Same as above	Effective both when providing artifacts and when introducing artifacts from outside
10.2.3.1	The achievement of required quality from data protection technology is divided into the following levels.	conduct a PIA that meets the required quality level and reflect the results in your design.	
11.1.1	Social Principles of Human-centric AI	Implement risk countermeasures based on periodic information collection and damage assumptions regarding new attack target perspectives.	Include in the PDCA cycle of security.

11.2.1	Harmonized standard	When considering the security measures required for the products and services being designed and developed, collecting information on the laws and regulations and harmonized standards of the regions relevant to the business is recommended. Collect information on harmonized standards of laws and regulations related to risk items assumed in the products and services being designed and developed. Use this information to predict the risks that may arise from attacks.	In Europe, this includes NIS2 subordinate laws and regulations under (Cybersecurity Act, Cyber Resilience Act, etc.) and harmonized standards of AI Act.
12.1	PoC trial phase	Before construction, thoroughly investigate and review the security settings of the environment and infrastructure in which PoC will be implemented.	
12.1.1	Handling of development processes with several operational stages	● Risk factors (e.g., spurious correlation) should be verified each time a PoC is conducted to verify whether the requirements meet the specifications. (However, the scale of the PoC can be adjusted as needed.) ● Implement a mechanism to monitor risk factors and consider measures to be included in quality inspection and operational performance monitoring.	

12.2.1 Figure 21	Process model in the stage of machine learning building	● Confirm the handling of sensitive information and generation of new sensitive information at the design stage of datasets, models, and tests, analyze threats and vulnerabilities, and incorporate measures. ● Confirm the handling of sensitive information and the generation of new sensitive information based on the inference results output from iterative training and quality confirmation/verification work, analyze threats and vulnerabilities, and incorporate measures.	
12.2.1 Figure 22	Example model of preprocessing for training	Handle sensitive information during each of the preprocessing steps, check for the generation of new sensitive information, analyze threats and vulnerabilities, and incorporate measures.	
12.2.1.1	ML requirements analysis phase	Since the next phase of this process will be the design and development of internal quality. Complete the AI security risk assessment, derive and organize control measures before this process begins.	

12.2.1.1	ML requirements analysis phase	Regarding learning data as information assets, assuming the relationships among data (e.g., correlations) that are sensitive, which are expressed by reorganizing the nature of input/output data and reorganizing them as a requirement for concrete construction as the target of damage, identify threats and vulnerabilities, and consider countermeasures. For quality requirements to be determined based on the nature of the data and the specific dataset itself, threats and vulnerabilities are identified and countermeasures are discussed, assuming damage from attacks.	
12.2.1.2	Training data composition phase	Accounting for datasets as information assets, scrutinizing the relationships (e.g., correlations) between data types stored by the dataset and the nature of the data (e.g., fairness and privacy), identify threats and vulnerabilities and consider how to respond to them, based on the damage that could occur in the event of an attack. ● Investigate new relationships (e.g., fairness or privacy) that have been created between data after preprocessing, identify threats and vulnerabilities based on the assumption of damage in the event of an attack, and consider measures.	

12.2.1.3	Iterative training phase	Extract machine learning models, hyperparameters, and learning models for implementation as information assets, and scrutinize relationships among data stored in datasets (e.g. correlation) and characteristics of data (e.g., fairness and privacy) Then, identify threats and vulnerabilities based on the assumption of damage in the event of an attack, and consider countermeasures. ● Investigate new relationships (e.g., fairness or privacy) that have been created between data after preprocessing, identify threats and vulnerabilities based on the assumption of damage in the event of an attack, and consider measures.	
12.2.1.4	Quality check and assurance phase	Treat test datasets, test results (including test confirmation), and data causing false inferences as information assets, assume damage due to falsification or theft of information that can be used as a reference for attacks, identify threats and vulnerabilities and consider measures.	
12.2.1.4.2	Test step	Confirm the assurance that AI security requirements are being met.	
12.2.2	Integration test phase	Same as above	

12.2.2	System building and integration test phase	Conduct a security risk assessment of the parts of the system other than machine learning components in accordance with the ISO/IEC 27000 series, ISO/IEC 15408 Common Criterial, and guidelines, frameworks, and tools provided by the IPA, and take action based on the results of the assessment and consider measures based on the results.	● Design the assessment methodology with reference to the various ISACs for business sector-specific requirements, or the PCI DSS requirements if no ISAC has been established. ● Pay attention to the selection of guidelines and frameworks when life, safety and property are involved.
12.3	Quality monitoring and operation phase	● Treat relationships (e.g., correlation), properties (e.g., fairness and privacy), and hyperparameters among data included in learning data as information assets, perform periodic monitoring to anticipate possible damage and take appropriate measures. ● Periodically review from the perspective of relationship between data and nature.	

11. Information on related documents (informative)

This chapter is informative.

11.1 Relationships with other documents and rules

11.1.1 Social Principles on Human-centric AI

In Japan, the Cabinet Office has defined an ideal relationship between operators and developers of machine-learning-based systems and the society and public in the form of "Social Principles on Human-Centric AI" [24]. In relation to the Principles, the Guideline is primarily positioned as non-binding guidance that sorts out technical matters developers themselves refer to and practice to improve quality and prevent safety and security from being compromised when operators and developers fully understand those Principles and actually develop machine-learning-based systems and machine learning components therein (Figure 19). Moreover, the Guideline is, together with other guidelines and future international standards, also positioned as an element that constitutes "Social Principles of Human-centric AI" mentioned therein.

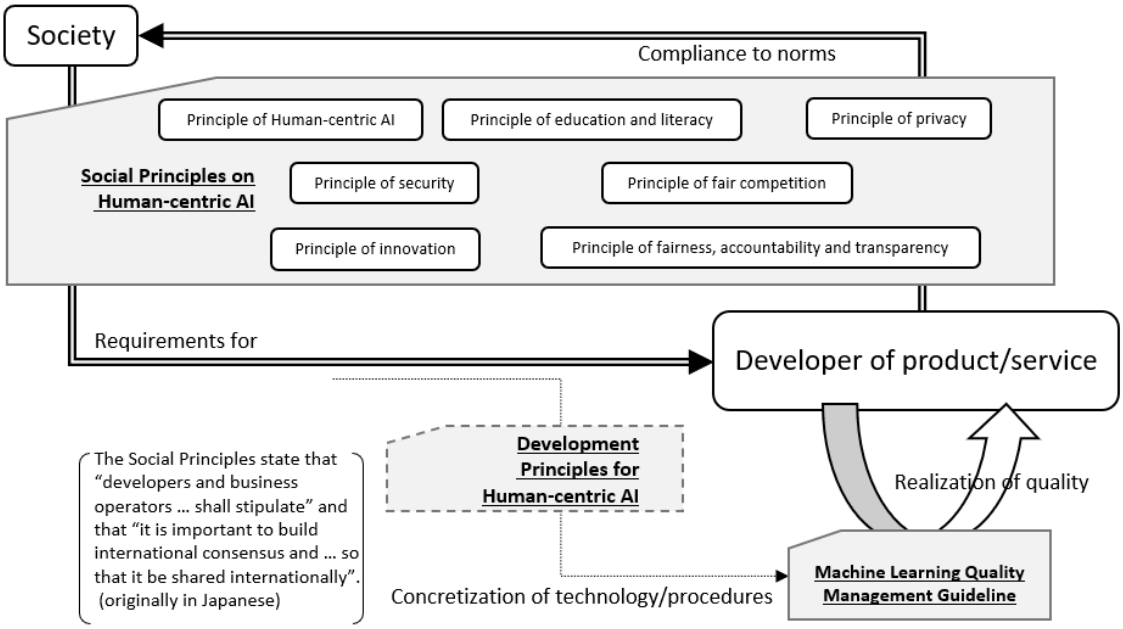


Figure 19: Relationship with the Social Principles on Human-Centric AI

11.1.2 AI-related rules and guidelines of foreign governments and international

organizations concerning AI technology

In addition to the above principles, norms for social nature, safety and ethics concerning the development and use of AI technology have been documented recently in different forms [26], [37]. The Guideline, in relation to these documents, positions the social norms under verbalized documents as in the case of the Social Principles of Human-centric AI mentioned in the previous section and categorizes to present one specific method for realizing some of them in system development.

11.1.3 Contract guidelines for AI of the Ministry of Economy, Trade and Industry

The Contract Guidelines on Utilization of AI and Data [39] published by the Ministry of Economy, Trade and Industry summarizes considerations concerning contracts with business partners when they cooperatively develop systems containing AI (e.g., machine learning) following contracts such as sales orders or quasi-mandates. The Contract Guidelines clarify roles and responsibilities in contracts between business partners engaged in development, while the Guideline summarizes quality of service which providers have to provide to users of system being developed. From the standpoint of the Guideline, quality-in-use characteristics of products and services defined in the Guideline are elaborated with the cooperation of all stakeholders listed in the Contract Guidelines for development, and then they are provided to users. The Guideline does not directly cover how to specifically realize the quality with the cooperation of business partners and how to conclude a contract and share responsibilities. Stakeholders should agree on these matters based on the Contract Guidelines. On the other hand, the Guideline can serve as a base for examining technical matters related to quality when stakeholders share responsibilities and exchange information. In Section 3.9, we briefly analyzed the possibility of sharing roles as one example.

The Contract Guidelines state that a non-waterfall model is appropriate for sales orders in the development process. On the other hand, the Contract Guidelines (p. 46) focus on (3) the development phase and also phases such as (4) the additional learning phase before and after (3). Therefore, the Guideline defines a system lifecycle process by a wide range from system planning to disposal after operation. A model mixed with a waterfall model is used and summarized as Figure 5: Conceptual diagram of mixed machine learning lifecycle process on page 27. Concerning the relation with gradual exploratory development stages recommended in the Contract Guidelines, the developmental phase at the center of Figure 5 in the Guideline corresponds basically to the development phase in the Contract Guidelines.

11.1.4 Relationships with guidelines for quality assurance of machine-learning-

based artificial intelligence (QA4AI)

The Consortium of Quality Assurance for Artificial Intelligence-based Products and Services (QA4AI) in Japan published Guidelines for Quality Assurance of Machine-learning-based Artificial Intelligence August 2020 edition [59] in August 2020. The QA4AI Guidelines propose the following five axes to be considered in the quality assurance of AI products with the aim of giving common guidelines for quality assurance of AI products.

A) Data Integrity
B) Model Robustness
C) System Quality
D) Process Agility
E) Customer Expectation

Moreover, the QA4AI Guidelines present checklists of the 5 axes along with a list of specific quality management technologies.

Regarding the relation between the Guideline with the QA4AI Guidelines, the three quality-assurance axes above (A, B and C) are considered to correspond to the internal qualities listed in Chapter 6 of the Guideline. Therefore, the technologies specifically listed in the QA4AI Guidelines correspond to the quality management method for each internal quality characteristic presented in Chapter 7 and give some insight to engineers responsible for realizing the quality with the complementary help of the two guidelines. Table 14 explains the specific relation between the items listed in the checklists for the 3 axes of the QA4AI Guidelines and the internal qualities listed in Chapter 6 of the Guideline.

Although no clear quality management axis has been established in the Guideline for the quality assurance axes (D and E above), they can be realized through the application process envisioned in the Guideline (Section 3.8) by including a business process with customers.

Consequently, the QA4AI Guidelines published by the Consortium are beneficial as a reference for engineers who create machine-learning-based AI to find the feasibility of technologies to improve and elaborate internal qualities of machine learning components. On the other hand, the Guideline intends to comprehensively analyze matters necessary for businesses that plan and develop overall systems containing machine-learning-based AI to ensure quality-in-use characteristics throughout the lifecycle process and list them as much as possible. Therefore, these two guidelines are considered to complement each other.

Table 14: Analysis of the relationship with the QA4AI Guidelines (August 2020 edition)

Internal quality characteristics in the Guideline	Checklist in the QA4AI Guidelines
A-1: Sufficiency of problem domain analysis	2.2.1 Data Integrity (b) Adequacy of training data (b.i)

	2.2.2 Model Robustness (19) Has verification been conducted with a sufficiently diverse set of data considering mathematical, semantic, and social-cultural diversity?
A-2: Sufficiency of data design	2.2.1 Data Integrity (a) Sufficiency of the amount of training data (a.i) (a.ii) (a.iii) (b) Adequacy of training data (b.i)
B-1: Coverage of datasets	2.2.1 Data Integrity (d) Impartiality of training data (f) Consideration of characteristics of training data
B-2: Uniformity of datasets	2.2.1 Data Integrity (d) Impartiality of training data
B-3: Adequacy of data	2.2.1 Data Integrity (b) Adequacy of training data (b.ii) (b.iii) (c) Suitability of training data for requirements (c.ii) (e) Complexity of training data (g) Adequacy of value ranges in training data
C-1: Correctness of trained models	2.2.1 Data Integrity (i) Adequacy of test data 2.2.2 Model Robustness (a) Sufficiency of model accuracy (b) Sufficiency of model generalization performance (c) Sufficiency of model evaluation (d) Adequacy of learning process (e) Adequacy of model structure 2.2.3 System Quality (a) Suitability of value provision by the system (a.ii)
C-2: Stability of trained models	2.2.2 Model Robustness (b) Sufficiency of model generalization performance (d) Adequacy of learning process (e) Adequacy of model structure (f) Adequacy of model testing (g) Model robustness
D-1: Reliability of underlying software systems	2.2.1 Data Integrity (k) Adequacy of data processing programs 2.2.2 Model Robustness (k) Adequacy of model implementation in code
E: Maintainability of quality characteristics during operation	2.2.1 Data Integrity (j) Consideration of effects of online learning

	2.2.2 Model Robustness (i) Sufficiency of testing after model updates (j) Consideration of model obsolescence 2.2.3 System Quality (i) Consideration of system quality deterioration
Items corresponding to external qualities	2.2.3 System Quality (a) Suitability of value provision by the system (c) Adequacy of units of system evaluation (D) Mitigation of impacts due to accidents (E) Prevention of accident occurrence (F) Mitigation of AI influence

11.2 Relationships with international initiatives for AI quality

Currently, diverse international initiatives for the quality of AI have been taken, including those mentioned below. The Guideline adopts adaptable parts from these initiatives. Moreover, we will actively present outstanding knowledge acquired from the Guideline toward international standardization.

11.2.1 Quality and safety

ISO/IEC JTC 1/SC 42/WG 3 has started the examination of quality and safety. The working group is analyzing gaps with existing standards for quality characteristics and quality assurance technologies of AI (ISO/IEC 25000 (SQuaRE) [7][8], ISO/IEC/IEEE 29119-4 (testing technology) [11] and functional safety (IEC 61508 [13], ISO 26262 [9])). In addition, SC42 has started discussions on data quality, etc.

In Europe, the European Commission published the European AI Policy Guidelines [31] in 2018, including ensuring an appropriate ethical and legal framework, and established the AI-HLEG (AI Expert Group) [33]. In 2019, the AI High-Level Expert Group (AI-HLEG) formulated the European AI Ethics Guidelines [37] based on four ethical principles, including respect for human dignity and fairness. In 2020, the European Commission published the European AI White Paper [32], which requires six requirements for AI systems in high-risk industrial domains and high-risk applications, including training data and accuracy. In April 2021, following a public consultation on the European AI White Paper, the European Commission published the European AI Bill [29], which requires eight requirements for data and accuracy, and the European Parliament adopted the amendment in June 2023. Furthermore, the European Commission requested European standardization bodies to create Harmonized Standards

following the European AI Bill. The system of the European Commission recognizes compliance with the Harmonized Standard as evidence of adherence to the relevant EU regulations. The technical requirements for compliance with the European AI legislation are anticipated to be specifically stipulated in the system. It is possible that the European AI Bill, the world's first legally binding AI hard law proposal, will ultimately establish the objectives or goals to be protected, where the Guideline can provide the technical means to achieve the goals in the future.

In its data and data governance requirements, the bill requires the use of high-quality training, validation, and test data for development and the consideration of the characteristics, properties, and elements of the geographic, behavioral, and functional use environments. The bill suggests the importance of B-1: Coverage of datasets and B-2: Uniformity of datasets of the Guideline corresponding to the requirements. The bill also calls for accuracy, robustness, cybersecurity requirements to ensure consistent performance throughout the lifecycle, declared accuracy levels and standards, and cybersecurity against adversarial data. The bill suggests the importance of C-1: Correctness of trained models and C-2: Stability of trained models of the Guideline corresponding to the requirements. Furthermore, in September 2022, the European Commission proposed the Artificial Intelligence Liability Directive [28], premised on European AI legislation, establishing fundamental rules on legal responsibility for the provision and use of artificial intelligence systems. The aim is to ensure that individuals who suffer damages from artificial intelligence systems receive protection levels equivalent to those who suffer damages from other technologies.

In Europe, in addition to the EU, Fraunhofer IAIS (Fraunhofer Institute for Intelligent Analysis and Information Systems IAIS) in Germany issued the AI Assessment Catalog [63] in German in July 2021 and English in February 2023. The Catalog is a structured guideline for systematically identifying AI-specific risks, formulating assessment criteria, assessing and improving trustworthiness, and documenting the results.

In the United States, NIST (National Institute of Standards and Technology) solicited public comments on the AI Risk Management Framework in July 2021, conducted multiple workshops to gather feedback from around the world, and released AI RMF 1.0 in January 2023. The AI RMF is intended for voluntary use. While the AI RMF illustrates a high-level framework, including governance levels, a companion AI RMF Playbook provides examples of specific measures.

This Guideline, the AI Assessment Catalog, and AI RMF 1.0 have the following similarities concerning the basic approach to quality control.

- The quality requirements for AI should be identified based on the business requirements and social demands of AI systems and solutions.
- Various quality requirements for AI may not always be compatible; hence, it is necessary to strike a balance.
- The quality of AI needs to be assessed and improved by selecting evaluation indicators for each quality requirement type. Documenting and disclosing the assessment and

improvement process is a means of explaining the quality to others.

11.2.2 Transparency

In Europe, the European AI Ethics Guidelines and the European AI Bill have outlined requirements for transparency, which are expected to have an international impact, especially in EU member states. In April 2019, AI-HLEG presented a checklist for ensuring transparency, which will be verified through actual demonstrations with companies, and in 2020, published a self-assessment list of trusted AI [38], including seven requirements for transparency and accountability. The European AI Bill requires transparency and provision of information to users, which means transparency of operation by design and development of transparent operation so that the user can understand and control the processing process.

On the other hand, IEEE is currently examining IEEE P7001 (transparency of autonomous systems) [21], and it may have a certain level of influence over future standardization in terms of the definitions of terms and concepts. The six transparency levels from 0 to 5 have been defined for the five types of stakeholders such as users, accident investigation committee, etc. A higher number does not always mean that it is more stringent. The certification of its compatibility is demonstrated through the pilot validation project called ECPAIS.

ISO includes terms and concepts related to transparency in the document TR24028 [6] at WG3 (trustworthiness) of ISO/IEC JTC 1/SC 42.

11.2.3 Fairness (bias)

EU AI high-level expert group (AI-HLEG) compiles high-level principles for problems of ELSI of AI, including bias. The European AI Bill calls for data and data governance requirements to ensure that datasets are relevant, representative, error-free, have appropriate statistical properties, and monitor, detect, and correct bias.

The above IEEE P7003 (algorithmic bias) [23] specifies a method to identify negative bias, including legally prohibited discrimination based on race or gender and non-legal discrimination in algorithm development and keep bias within the acceptable range in the lifecycle from system planning to operation. A demonstration experiment of this method is underway in the pilot project (ECPAIS (Ethics Certification Program for Autonomous and Intelligent Systems)) to validate its conformity.

ISO/IEC JTC 1/SC 42/WG 3 (trustworthiness) is also preparing documents such as TR 24028 (Overview of trustworthiness in Artificial Intelligence) [6] and TR 24027 (Bias in AI systems and AI-aided decision making) [5].

11.2.4 Other perspectives on machine learning quality management

Privacy [22], nudging, and other issues are being considered in the IEEE P7000 series, and governance and other issues are being considered in ISO/IEC JTC1/SC42. In addition to the requirements already mentioned, the European AI bill requires a risk management system, technical documentation, record-keeping, and human oversight.

In 2023, the development of generative AI garnered attention, prompting governments worldwide to respond to this trend. In Europe, as mentioned above (Section 11.2.1), the European Parliament adopted an amendment to the pending European AI legislation in June 2023, introducing new obligations related to generative AI. In July 2023, the White House obtained voluntary commitments from major IT companies to manage the risks posed by AI in the United States. These commitments will serve as the key guideline in addressing the widespread concerns surrounding AI technology in the future. In China, in April 2023, CAC (Cyberspace Administration of China) released the Measures for the Management of Generative Artificial Intelligence Services (Draft for Soliciting Opinions), stipulating the points to be observed and penalties to be imposed regarding the provision of services of generative AI in the country.

12. Reference models for development processes (informative)

This chapter describes the development process reference models for machine-learning-based systems, which were examined in the discussions of the Guideline.

The reference models mentioned in this Chapter do not force developers to adopt a certain development method for machine-learning-based systems. Rather, they aim to add referable names to internal components and development processes of general machine-learning-based systems and compare recommendations in the Guideline with actual development processes. This way, the respective configurations and development processes can be compared with this model.

As shown in Figure 5 (page 27), the Guideline roughly divides the development process of machine learning components (and the principal parts of machine-learning-based systems that contain them) into three phases.

12.1 PoC phase

In the earliest stage of system development, the **PoC trial phase** as a development lifecycle process is sorted out as the preparatory stage to identify the possibility of performance attained by a system, quality deterioration risks and its possible cause before defining the functions of the whole process. Quality management activities in the PoC phase are organized as important preparatory works of quality management activities in the main development phase for the quality management of the Guideline. From the viewpoint of data science, analytical activities in this phase are closely associated with subsequent risk analysis and requirements analysis, and their results are often used in the subsequent main development phase. Efforts are essential in the PoC trial phase to attain sufficiency of problem domain analysis (Section 7.1). The Guideline stipulates that the adequacy of the activities will ultimately be checked again in each step of the main development phase. This allows quality management activities to be conducted on a trial-and-error basis in the PoC phase without restrictions.

12.1.1 Handling the PoC phase, including trial operation

In general, activities called a proof of concept may go beyond simple data collection, analysis, trial training, or model building. They may also include trials conducted in an actual operational environment under usage conditions different from those of the final products, such as operation under human supervision. In some cases, operation may proceed through multiple stages with different quality requirements. For development involving such multiple stages of actual operation, this Guideline treats the system construction stage for each operational stage as

corresponding to the overall development lifecycle, including the main development phase, as defined in this Guideline. In addition, all results obtained in earlier operational stages are treated as knowledge gained in the PoC phase for later development stages (Figure 20).

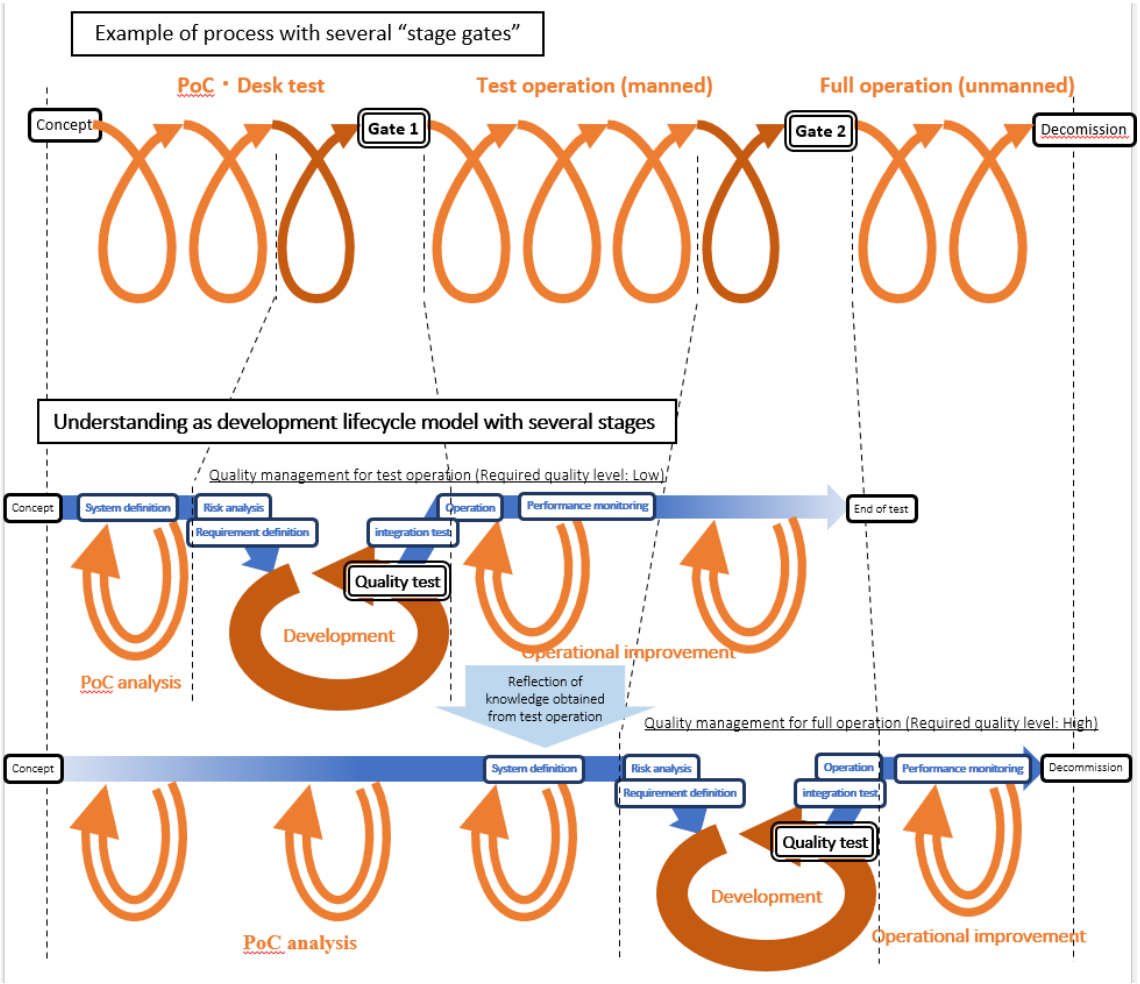


Figure 20: Handling of a development process with multiple operational stages (example)

12.2 Main development phase

The **main development phase** is a process from when system’s functional requirements are confirmed by means of the efforts in the PoC trial phase to immediately prior to the in-operation phase. This phase includes system definition equivalent to very early stages of development of conventional systems and integrated tests prior to final operation and shipment.

In the main development phase, a lifecycle that mixes a part of the conventional waterfall early-stage development and iterative development processes specific to machine learning components is defined as a model. To be more specific, system requirements for each component,

including functional definitions and risk analysis of the overall system, analysis and decomposition of effects of components and machine learning components, are decided. A series of processes that set the goals for the last integrated tests before shipment are sorted according to the conventional waterfall model. Moreover, software and hardware components other than machine learning components are assumed to adopt the conventional waterfall development model or agile development process[18] capable of ensuring the same level of quality as needed in the late stages of development. On the other hand, the development process of a specific machine learning model equivalent to the late stages of development of machine learning components defines the iterative development process model again in the following section.

12.2.1 Machine learning model building phase

After the risks to be addressed by machine learning components and the related quality requirements have been identified in the main development phase, a machine learning model is built, and its external quality is evaluated using using indicators of internal quality. This process is repeated until the model satisfies the required criteria. This stage is referred to as the **machine learning model building phase**.

Figure 21 shows a model that breaks this phase down into more specific processes. The individual processes in this model are intended to help developers map their own development processes to the descriptions in this Guideline by comparing those processes with this model. They are not intended to prescribe a single development process.

[18] Agile development process here means test-driven development or other non-waterfall style of development process, which includes quality assurance activities comparable to those of waterfall processes so that expected levels of quality are reasonably explainable. However, it does not include all forms of non-waterfall development.

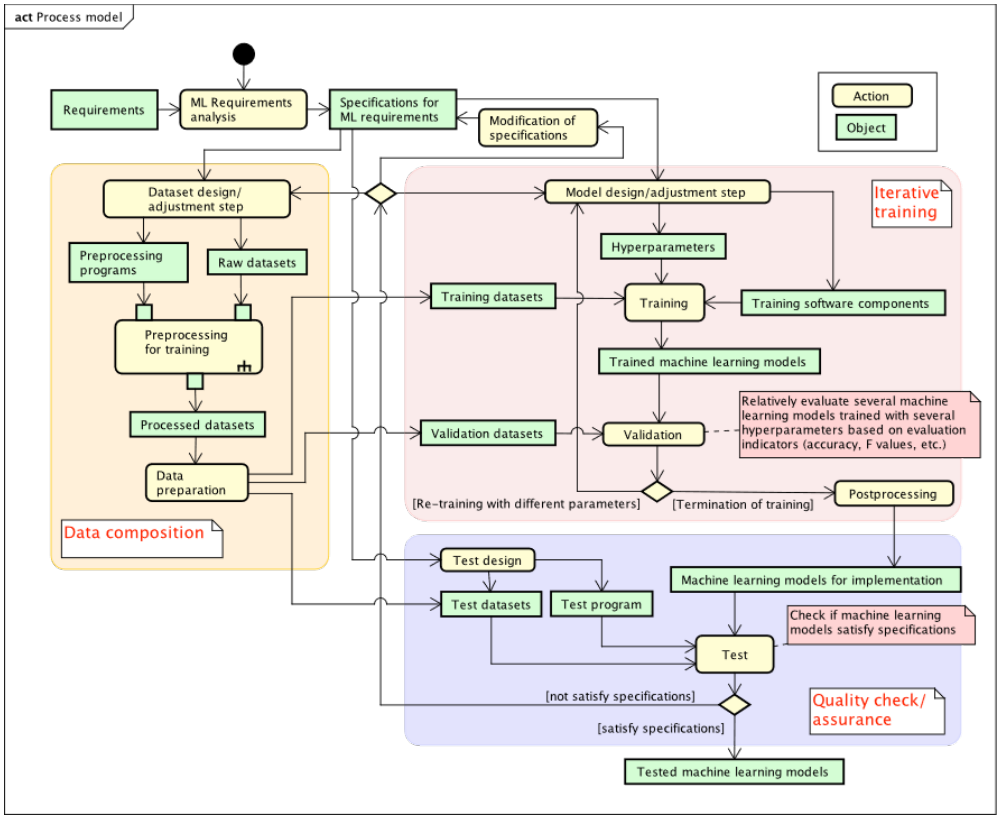


Figure 21: Process model for the machine learning model building phase (example)

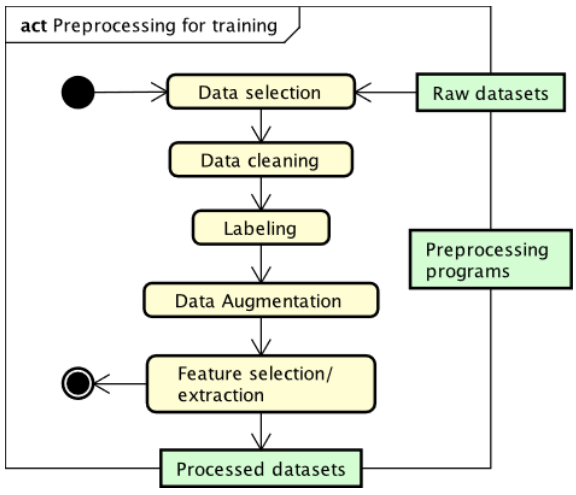


Figure 22: Example of preprocessing for training

12.2.1.1 ML requirements analysis phase

In the machine learning requirements analysis phase, conceptual quality requirements for machine learning components are concretized by sorting out data input characteristics to or output from machine learning components and reorganized as specific requirements for

building data in the **ML model building phase**. In developing machine learning-based systems, performance and quality requirements are often specified based on data characteristics or specific datasets. By explicitly including this analysis phase, it becomes possible to sort out a specific method of managing data quality and its relationship with other components from the quality management aspect. This phase is closely related to internal qualities mentioned in Sections 7.1 and 7.2.4.

12.2.1.2 Training data composition phase

In the training data composition phase, training, validation, or test datasets are generated according to the ML requirement specifications. This phase is explained below in line with training data composition shown in the upper left side of Figure 21.

12.2.1.2.1 Dataset design and adjustment step

In the dataset design and adjustment step, sufficient data are collected for each case according to the ML requirement specifications, and design is carried out to generate datasets suitable for training. The output of this step is unprocessed datasets and preprocessing programs. In cases where it is found that trained machine learning models do not satisfy the specifications after training, the process may return to this step to adjust the design of datasets.

12.2.1.2.2 Preprocessing step for training

In the preprocessing step for training, raw datasets generated in the dataset design and adjustment step in Section 12.2.1.2.1 are processed into datasets suitable for training. As Figure 22 shows, there are cases where each process is performed in random order, in parallel, or repeatedly.

12.2.1.2.2.1 Data selection

Datasets used for training, validation and tests are selected considering coverage and uniformity from a large number of raw datasets. Since coverage and uniformity have a trade-off relationship, the balance between them should follow the ML requirement specifications.

12.2.1.2.2.2 Data cleaning

Noise removal, data forming, complement of missing values and removal of outliers are carried out so that original characteristics of the datasets can be trained.

12.2.1.2.2.3 Data Augmentation

In order to ensure the amount of training data or test data and to prevent overfitting, data variants are generated and added to datasets. For example, controls such as rotation, scale-down, reverse, change in luminosity and background replacement of image data are carried out to generate variants.

12.2.1.2.2.4 Label addition

Datasets are labeled as the correct answer (for teaching) according to the ML requirement specifications.

12.2.1.2.2.5 Feature extraction and selection

Feature extraction is performed to add new features useful for model training, and feature selection is performed to reduce unnecessary features. For example, feature extraction uses techniques such as calculating frequency components by Fourier transform and finding the principal components of a plurality of features. If useful features are obtained, more unnecessary features can be reduced by feature selection in many cases.

12.2.1.2.3 Data preparation step

In the data preparation step, processed datasets after generation are divided into training, validation, and test datasets. Training datasets and validation datasets may be replaced during iterative training.

12.2.1.3 Iterative training phase

In the iterative training phase, machine learning models are designed according to ML requirement specifications and training and validations are repeated using datasets created in the training data composition phase. This phase is explained below concerning the flow of iterative training shown at the top right of Figure 21.

12.2.1.3.1 Model design and adjustment step

In the model design and adjustment step, hyperparameters necessary for training, including the structure of learning models, learning algorithms and various parameters, are designed following the ML requirement specifications and training software components are created. Hyperparameters may be adjusted in response to the results of validations and tests.

12.2.1.3.2 Training step

Machine learning models are trained using training datasets based on designed hyperparameters.

12.2.1.3.3 Validation step

Validation datasets are input into trained machine learning models to evaluate models based on evaluation indicators such as accuracy, precision, recall, F values, etc., of learning models. Generally, several machine learning models are trained with several hyperparameters and evaluated relatively to judge their adequacy.

12.2.1.3.4 Postprocessing step

In the postprocessing step, models are transformed to adapt trained machine learning models to the in-operation environment of inference servers, edge devices, etc. and learning models for implementation are generated.

The following are some specific cases.

- Transformation of models appropriate for the in-operation environment by computational performance
  - Changes in calculation precision
  - Compression and speeding up of models (such as distillation, quantization, etc.)
- Optimization of models appropriate for in-operation environment (efficient inference)
  - Compiling the models (such as parallelization, vectorization, etc.)

The figure of a process model envisioned in the Guideline expects this postprocessing to be placed immediately after the iterative training phase and the quality check and assurance phase to be carried out under conditions close to the time of in-operation. However, postprocessing is often carried out as part of the building process of the whole system after the quality check and assurance phase of machine learning components in actual development. In this case, the quality checked in the quality check and assurance phase may change or deteriorate at the time of in-operation, and it may become necessary to re-validate numerical equivalence in the test stage of the whole system.

12.2.1.4 Quality check and assurance phase

In the quality check and assurance phase, tests are designed following the ML requirement specifications to conduct evaluation tests of learning models for implementation. This phase is explained below with reference to the flow of quality checks and assurance shown at the bottom

right of Figure 21.

12.2.1.4.1 Test design step

In the test design step, programs required for tests are created, and test data are added following the ML requirement specifications. Test data to be added may include cases where data are generated by means of pre-processing data augmentation and cases where trained models generate data which tend to make erroneous inferences.

12.2.1.4.2 Test step

In this step, the accuracy of learning models for implementation is evaluated using test datasets based on regular indicators such as accuracy, precision, recall, F values, etc., and its stability is evaluated using other data, such as data made by adding noises to ones in the datasets. In cases where any evaluation result does not satisfy the ML requirement specifications, the process returns to the previous phase to adjust learning models and training datasets and modify the ML requirement specifications.

12.2.2 **System building and integration test phase**

This stage is not part of the machine learning building, but it is described here due to the order of the procedures.

In an ideal development situation, all quality requirements for machine learning components are pre-arranged through the PoC trial phase, and all achievements can be confirmed up to the machine learning building phase. Hopefully, no problems will occur at the subsequent system building and integration testing stage. In actual system development, complex real environment requirements analysis may not be perfect, unexpected interactions may occur with other components, and the effects of defects in other components may spread. Quality defects in machine learning components due to these various factors often occur during the so-called integration testing stage. Moreover, prior data are inevitably insufficient as test data, especially in a large-scale and complicated system. In many cases, inspection in the actual environment/system after integration is indispensable. In addition, inspections for rare cases that cannot be prepared in advance during the data composition phase could be reproduced and performed during the integration test phase

If a test fails in any of these cases, ML requirements analyses are partially modified, and the whole building phase of the machine learning model is repeated. To avoid this enormous amount of repeated work, it is preferable to accurately record the content of quality management activities conducted in each phase of each stage of machine learning building so that the effect

of modifications can be grasped correctly and partial modifications are made without fail.

12.3 Quality monitoring and operation phase

After the system is deployed in actual operation, the activity to maintain the performance by continuously monitoring the quality at the operation stage and making necessary corrections is positioned as the **quality monitoring and operation phase**. Although this phase is placed outside the development phase in a more restricted sense in the waterfall V-shape development model, it already exists in the concept of system lifecycle, such as the RAMS standards.

We consider that, in many cases, quality needs to be managed during operation from the viewpoint of:

- When qualitative requirements analysis based on prior data does not capture the actual environment
- When the environment during operation has changed from the environment when prior data was prepared

In various machine learning applications, there are various patterns to update trained machine learning models in actual operation, because they cannot be classified uniformly. Therefore, the Guideline classifies such patterns into two categories.

(1) A pattern of re-training machine learning in the development environment and deploying it in the operation environment after conducting tests.
(2) A pattern of carrying out additional learning automatically in the operation environment so that the system is self-adapted.

Section 8.2 summarizes the principles for these patterns individually.

13. Bibliography

13.1 International standards

[1] ISO 13849-1:2015: Safety of machinery — Safety-related parts of control systems — Part 1: General principles for design.

[2] ISO/IEC/IEEE 15288:2015: Systems and software engineering — System life cycle processes.

[3] ISO/IEC 15408-1:2022: Information security, cybersecurity and privacy protection — Evaluation criteria for IT security — Part 1: Introduction and general model.

[4] ISO/IEC IS 22989:2022 Information technology — Artificial intelligence — Artificial intelligence concepts and terminology.

[5] ISO/IEC TR 24027:2021 Information technology — Artificial Intelligence (AI) — Bias in AI systems and AI aided decision making.

[6] ISO/IEC TR 24028:2020: Information technology — Artificial intelligence — Overview of trustworthiness in artificial intelligence.

[7] ISO/IEC 25010:2011: Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models.

[8] ISO/IEC 25012:2008: Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Data quality model.

[9] ISO 26262-1:2018: Road vehicles — Functional safety — Part 1: Vocabulary.

[10] ISO/IEC 27000:2018: Information technology — Security techniques — Information security management systems — Overview and vocabulary.

[11] ISO/IEC 27005:2022 - Information security, cybersecurity and privacy protection — Guidance on managing information security risks

[12] ISO/IEC/IEEE 29119-4:2015: Software and systems engineering — Software testing — Part 4: Test techniques.

[13] ISO/IEC/IEEE 29148:2018 - Systems and software engineering — Life cycle processes — Requirements engineering.

[14]] IEC 61508-1:2010: Functional safety of electrical/electronic/programmable electronic safety-related systems - Part 1: General requirements.

[15] IEC 61508-3:2010: Functional safety of electrical/electronic/programmable electronic safety-related systems - Part 3: Software requirements.

[16] IEC 61508-4:2010: Functional safety of electrical/electronic/programmable electronic

safety-related systems - Part 4: Definitions and abbreviations.

[17] IEC 62278:2002: Railway applications - Specification and demonstration of reliability, availability, maintainability and safety (RAMS).

[18] IEC TS 62443-1-1:2009: Industrial communication networks - Network and system security - Part 1-1: Terminology, concepts and models.

[19] IEC 62443-2-1:2010: Industrial communication networks - Network and system security - Part 2-1: Establishing an industrial automation and control system security program.

[20] IEC TS 62998-1:2019: Safety of machinery - Safety-related sensors used for the protection of persons.

[21] IEEE P7001 - IEEE Standard for Transparency of Autonomous Systems.

[22] IEEE P7002 - IEEE Standard for Data Privacy Process.

[23] IEEE P7003 - Algorithmic Bias Considerations. An active project.

13.2 Documents and regulations issued by countries and international organizations

[24] Council for Science, Technology and Innovation, Cabinet Office of Japan. *Social Principles of Human-Centric AI*. March 2019.
https://www8.cao.go.jp/cstp/aigensoku.pdf (In Japanese),
https://www.cas.go.jp/jp/seisaku/jinkouchinou/pdf/humancentricai.pdf (tentative translation).

[25] United Nations Educational, Scientific and Cultural Organization (UNESCO), *Draft Text of the Recommendation on the Ethics of Artificial Intelligence*, 41 C/73 Annex, November 2021.

[26] Organization for Economic Co-operation and Development (OECD). *OECD Principles on Artificial Intelligence*. May 2019.
https://www.oecd.org/going-digital/ai/principles/

[27] Organization for Economic Co-operation and Development (OECD). Recommendation of the Council concerning Guidelines Governing the Protection of Privacy and Transborder Flows of Personal Data, OECD/LEGAL/0188, 2013.

[28] European Commission. Proposal for a Directive on adapting non contractual civil liability rules to artificial intelligence (Artificial Intelligence Liability Directive, AILD). September 2022.
https://commission.europa.eu/business-economy-euro/doing-business-eu/contract-rules/digital-contracts/liability-rules-artificial-intelligence_en

[29] European Commission. Proposal for a Regulation of the European parliament and of the council laying down harmonized rules on artificial intelligence (Artificial Intelligence Act) and amending certain union legislative acts. April 2021.

https://digital-strategy.ec.europa.eu/en/library/proposal-regulation-laying-down-harmonised-rules-artificial-intelligence-artificial-intelligence

[30] European Commission. Regulation (EU) 2016/679 of The European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation). http://data.europa.eu/eli/reg/2016/679/oj

[31] European Commission. *Communication Artificial Intelligence for Europe*. 2018. https://digital-strategy.ec.europa.eu/en/library/communication-artificial-intelligence-europe.

[32] European Commission. *The White Paper on Artificial Intelligence – A European approach to excellence and trust*. February 2020. https://ec.europa.eu/info/publications/white-paper-artificial-intelligence-european-approach-excellence-and-trust_en

[33] European Commission. Regulation (EU) 2019/881 of the European Parliament and of the Council of 17 April 2019 on ENISA (the European Union Agency for Cybersecurity) and on information and communications technology cybersecurity certification and repealing Regulation (EU) No 526/2013 (Cybersecurity Act) (Text with EEA relevance). 2019. http://data.europa.eu/eli/reg/2019/881/oj http://data.europa.eu/eli/reg/2019/881/oj

[34] European Commission. Proposal for a REGULATION OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL on horizontal cybersecurity requirements for products with digital elements and amending Regulation (EU) 2019/1020. 2022. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex:52022PC0454

[35] European Commission. Directive (EU) 2022/2555 of the European Parliament and of the Council of 14 December 2022 on measures for a high common level of cybersecurity across the Union, amending Regulation (EU) No 910/2014 and Directive (EU) 2018/1972, and repealing Directive (EU) 2016/1148 (NIS 2 Directive) (Text with EEA relevance). 2022. http://data.europa.eu/eli/dir/2022/2555/oj http://data.europa.eu/eli/dir/2022/2555/oj

[36] European Commission. High-Level Expert Group on Artificial Intelligence. 2018. https://digital-strategy.ec.europa.eu/en/policies/expert-group-ai

[37] The High-Level Expert Group on Artificial Intelligence, European Commission. *Ethics guidelines for trustworthy AI*. April 2019. https://ec.europa.eu/digital-single-market/en/news/ethics-guidelines-trustworthy-ai

[38] The High-Level Expert Group on Artificial Intelligence, European Commission. *The Assessment List for Trustworthy Artificial Intelligence (ALTAI)*. July 2020. https://ec.europa.eu/digital-single-market/en/news/assessment-list-trustworthy-artificial-intelligence-altai-self-assessment

[39] Ministry of Economy, Trade and Industry, Japan. *Contract Guidelines on Utilization of*

AI and Data. June 2018. In Japanese.
https://www.meti.go.jp/press/2018/06/20180615001/20180615001-3.pdf

[40] Ministry of Economy, Trade and Industry, Japan. Study Group for Architecture of AI Social Implementation. *AI Governance of our country -version 1.0*. January 2021. In Japanese. https://www.meti.go.jp/press/2020/01/20210115003/20210115003-1.pdf

[41] The IEEE Global Initiative on Ethics of Autonomous and Intelligent Systems. *Ethically Aligned Design: A Vision for Prioritizing Human Well-being with Autonomous and Intelligent Systems, First Edition*. IEEE, 2019.
https://standards.ieee.org/content/ieee-standards/en/industry-connections/ec/autonomous-systems.html

[42] Ministry of Economy, Trade and Industry, Japan. *Cyber Physical Security Measures Framework Ver1.0*. April 2019. In Japanese.
https://www.meti.go.jp/press/2019/04/20190418002/20190418002-2.pdf

[43] California Consumer Privacy Act (CCPA). Cal. Civ. Code. Div. 3. Part 4. Title 1.81.5
https://leginfo.legislature.ca.gov/faces/codes_displayText.xhtml?division=3.&part=4.&lawCode=CIV&title=1.81.5

[44] Ministry of Economy, Trade and Industry, Japan. Governance Guidelines for Implementation fo AI Principles ver. 1.1 January 2022.
https://www.meti.go.jp/shingikai/mono_info_service/ai_shakai_jisso/pdf/20220128_2.pdf

[45] Cabinet Office of Japan. AI Strategic Council. 2023.
https://www8.cao.go.jp/cstp/ai/ai_senryaku/ai_senryaku.html (In Japanese)

[46] Chiba Prefecture. Usage Guidelines of Generative AI such as ChatGPT ver. 1.0. June 2023.
https://www.pref.chiba.lg.jp/dejisui/press/2023/documents/guideline20230619.pdf (In Japanese)

[47] Ministry of Education, Culture, Sports, Science and Technology, Japan. *On the announcement of the "Tentative Guidelines on the use of Generative AI in the Elementary and Secondary Educational Stages"*. July 2023.
https://www.mext.go.jp/content/20230704-mxt_shuukyo02-000003278_003.pdf (In Japanese)

13.3 Official standards and forum standards

[48] National Institute of Standards and Technology (United States of America). Federal Information Processing Standards (FIPS) Publication 199. Standards for Security Categorization of Federal Information and Information Systems. February 2004.
https://csrc.nist.gov/pubs/fips/199/final

[49] National Institute of Standards and Technology (United States of America). Draft NIST IR 8269: *A Taxonomy and Terminology of Adversarial Machine Learning*. October

2019.
https://nvlpubs.nist.gov/nistpubs/ir/2019/NIST.IR.8269-draft.pdf

[50] National Institute of Standards and Technology (United States of America). NIST SP 800-30: *Guide for Conducting Risk Assessments*. September 2012.
https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-30r1.pdf

[51] National Institute of Standards and Technology (United States of America). *Cyber Security Framework Version 1.1*. April 2018.
https://doi.org/10.6028/NIST.CSWP.04162018

[52] National Institute of Standards and Technology (United States of America). Special Publication (SP) 800-34, Revision 1, Contingency Planning Guide for Federal Information Systems, April 2021.

[53] National Institute of Standards and Technology (United States of America). AI Risk Management Framework.
https://www.nist.gov/itl/ai-risk-management-framework

[54] National Institute of Standards and Technology (United States of America). NIST SP 1270: Towards a Standard for Identifying and Managing Bias in Artificial Intelligence 2022.
https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.1270.pdf

[55] The MITRE Corporation. ATLAS - Adversarial Threat Landscape for Artificial-Intelligence Systems.
https://atlas.mitre.org/

[56] National Institute of Standards and Technology (United States of America). *Official Common Platform Enumeration (CPE) Dictionary*.
https://nvd.nist.gov/products/cpe

[57] The MITRE Corporation. *Common Vulnerability Enumerations*.
https://cve.mitre.org/

[58] Payment Card Industry Security Standards Council. Payment Card Industry Data Security Standard (PCI DSS).
https://www.pcisecuritystandards.org/.

[59] Consortium of Quality Assurance for Artificial-Intelligence-based products and services (QA4AI Consortium). Guidelines for Quality Assurance of Machine-learning-based Artificial Intrelligence - 2020.08 version. August 2020. http://qa4ai.jp/download/.

[60] European Union Agency for Cybersecurity (ENISA). Artificial Intelligence Cybersecurity Challenges. December 2020.
https://www.enisa.europa.eu/publications/artificial-intelligence-cybersecurity-challenges

[61] European Union Agency for Cybersecurity (ENISA). Securing Machine Learning Algorithms. December 2021.
https://www.enisa.europa.eu/publications/securing-machine-learning-algorithms

[62] National Information Security Standardization Technical Committee of the People's Republic of China (TC260). *Information Security Technology — Security Specification and Assessment Methods for Machine Learning Algorithms* (draft). August 2021. https://www.tc260.org.cn/front/bzzqyjDetail.html?id=20210804165243&norm_id=20201104200030&recode_id=43653

[63] Fraunhofer Institute of Intelligent Analysis and Information Systems (IAIS). *AI Assessment Catalog*. February 2023 https://www.iais.fraunhofer.de/en/research/artificial-intelligence/ai-assessment-catalog.html

13.4 Research articles

[64] Martín Abadi, Andy Chu, Ian J. Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep Learning with Differential Privacy. In *Proc. of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS'16)*, pp. 308–318, 2016.

[65] Alessandro Acquisti, Curtis Taylor, and Lian Wagman, The Economics of Privacy, Journal of Economic Literature 54(2), pp.442-492, 2016.

[66] Y. Adi, C. Baum, M. Cisse, B. Pinkas, J. Keshet. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In USENIX Security'18, pp. 1615-1631, 2018.

[67] Charu C. Aggarwal, On k-Anonymity and the Curse of Dimensionality, Proc. *31st VLDB*, pp.901-909, 2005.

[68] Charu C. Aggarwal, *Outlier Analysis (2ed.)*, Springer 2017.

[69] Naveed Akhtar, and Ajmal Mian. Threat of Adversarial Attacks on Deep Learning in Computer Vision: A Survey. *IEEE Access* 6, pp. 14410–14430, 2018. https://arxiv.org/pdf/1801.00553.pdf

[70] Ahmed Aldahdooh, Wassim Hamidouche, Sid Ahmed Fezza, and Olivier Deforges. Adversarial example detection for DNN models: a review and experimental comparison. Artif Intell Rev (2022). https://arxiv.org/abs/2105.00203

[71] P. Ammann, and J. Offutt. *Introduction to Software Testing*, Cambridge University Press, 2008.

[72] Giuseppe Ateniese, Luigi V. Mancini, Angelo Spognardi, Antonio Villani, Domenico Vitali, and Giovanni Felici. Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers. *International Journal of Security and Networks*, 10(3):137–150, 2015.

[73] Nuttapong Attrapadung, Koki Hamada, Dai Ikarashi, Ryo Kikuchi, Takahiro Matsuda, Ibuki Mishina, Hiraku Morita, and Jacob C. N. Schuldt. Adam in Private: Secure and Fast Training of Deep Neural Networks with Adaptive Moment Estimation.

Proceedings on Privacy Enhancing Technologies, 2022 (4), pp.747–768, 2022.

[74] Eugene Bangdasaryan and Vitaly Shmatikov, Differential Privacy Has Disparate Impact on Model Accuracy, arXiv:1905.12101v2, 2019.

[75] Guy Barash, Eitan Farchi, Ilan Jayaraman, Orna Raz, Rachel Tzoref-Brill, and Marcel Zalmanovici. Bridging the gap between ML solutions and their business requirements using feature interactions. In *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE 2019)*, pp. 1048–1058, August 2019. https://dl.acm.org/citation.cfm?id=3340442

[76] E. T. Barr, M. Harman, P. McMinn, M. Shahbaz, and S. Yoo. The Oracle Problem in Software Testing: A Survey. In *IEEE Transactions on Software Engineering*, 41(5):507–525, 2015.

[77] Battista Biggio, and Fabio Roli. Wild patterns: Ten years after the rise of adversarial machine learning. Pattern Recognition, Vol. 84, pp. 317-331, 2018.

[78] Matt Bishop, *Computer Security – Art and Science*, Addison Wesley 2003.

[79] Kallista Bonawitz, Peter Kairouz, Brendan McMahan, and Daniel Ramage, Federated Learning and Privacy, Comm. ACM, 65(4), pp.90-97. 2022.

[80] Akhilan Boopathy, Tsui-Wei Weng, Pin-Yu Chen, Sijia Liu, and Luca Daniel. CNN-Cert: An Efficient Framework for Certifying Robustness of Convolutional Neural Networks. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI 2019)*, pp.3240–3247, 2019.

[81] Eitan Borgnia, Valeriia Cherepanova, Liam Fowl, Amin Ghiasi, Jonas Geiping, Micah Goldblum, Tom Goldstein, and Arjun Gupta. Strong Data Augmentation Sanitizes Poisoning and Backdoor Attacks Without an Accuracy Tradeoff. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'21)*, pp.3855-3859, 2021. https://arxiv.org/abs/2011.09527

[82] H. Brendan McMahan, Galen Andrew, Ulfar Erlingsson, Steve Chien, Ilya Mironov, Nocolas Papernot, and Peter Kairouz, A General Approach to Adding Differential Privacy to Iterative Training Procedures, arXiv:1812.06210v2, 2019.

[83] Hannah Brown, Katherine Lee, Fatemehsadat Mireshghallash, Reza Shokri, and Florian Tramer, What Does it Means for a Language Model to Preserve Privacy?, arXiv:2202.05520v2, 2022.

[84] Miles Brundage, et al., Toward Trustworthy AI Development: Mechanisms for Supporting Verifiable Claims, arXiv:2004.07213v2, 2020.

[85] T. Byun, V. Sharma, A. Vijayakumar, S. Rayadurgam, and D. Cofer, Input Prioritization for Testing Neural Networks, Proc. AITest, pp. 63-70, and arXiv:1901.03768

[86] Flavio Calmon, Dennis Wei, Bhanukiran Vinzamuri, Karthikeyan Natesan

Ramamurthy, and Kush R. Varshney. Optimized Pre-Processing for Discrimination Prevention. In *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, December 2017. https://papers.nips.cc/paper/6988-optimized-pre-processing-for-discrimination-prevention.pdf.

[87] Nicolas Carlini, Chang Liu, Ulfar Erlingsson, Jernej Kos, and Dawn Song, The Secret Sharer: Evaluating and Testing Unintended Memorization in Neural Networks, arXiv:1802.08232v3, 2019.

[88] Nicolas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingson, Alina Oprea, and Coloin Raffel, Extracting Training Data from Large Language Models, arXiv:2012.07805v2, 2021.

[89] Nicholas Carlini, and David Wagner. Towards Evaluating the Robustness of Neural Networks. In *Proceedings of the 38th IEEE Symposium on Security and Privacy (SP)*, pp. 39–57, 2017.

[90] Chakraborty, Anirban, Manaar Alam, Vishal Dey, Anupam Chattopadhyay, and Debdeep Mukhopadhyay. A Survey on Adversarial Attacks and Defences. *CAAI Transactions on Intelligence Technology*, 6 (1), pp.25–45. https://doi.org/10.1049/cit2.12028

[91] Hongyan Chang and Reza Shokri, On the Privacy Risks of Algorithmic Fairness, arXiv:2011.03731, 2021.

[92] Bryant Chen, Wilka Carvalho, Nathalie Baracaldo, Heiko Ludwig, Benjamin Edwards, Taesung Lee, Ian Molloy and Biplav Srivastava. Detecting Backdoor Attacks on Deep Neural Networks by Activation Clustering. In *Proceedings of the Workshop on Artificial Intelligence Safety 2019*, 2019. http://ceur-ws.org/Vol-2301/paper_18.pdf

[93] T. Y. Chen, S. C. Chung, and S. M. You. Metamorphic Testing – A New Approach for Generating Next Test Cases, Technical Report HKUST-CS98-01, Department of Computer Science, The Hong Kong University of Science and Technology, 1998.

[94] T. Y. Chen, F.-C. Kuo, H. Liu, P.-L. Poon, D. Towey, Y. H. Tse, and Z. Q. Zhou. Metamorphic Testing: A Review of Challenges and Opportunities. *ACM Computing Surveys*, 51(1):1–27, 2018.

[95] S. Chiappa, and W. S. Isaac. A Causal Bayesian Networks: Viewpoint on Fairness. In *Privacy and Identity Management: Fairness, Accountability, and Transparency in the Age of Big Data*, IFIP Advances in Information and Communication Technology, vol. 547, 2019.

[96] A.E. Cina, A. Demontis, B. Biggio, F. Roli, M. Pelillo. Energy latency attacks via sponge poisoning. CoRR abs/2203.08147, 2022. https://doi.org/10.48550/arXiv.2203.08147

[97] Mark Coeckelbergh, *AI Ethics*, The MIT Press Essential Knowledge Series.

[98] Jeremy M Cohen, Elan Rosenfeld, and J. Zico Kolter. Certified Adversarial Robustness

via Randomized Smoothing. *The 36th International Conference on Machine Learning (ICML 2019)*, 2019.

[99] Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, Edoardo Debenedetti, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein. RobustBench: A Standardized Adversarial Robustness Benchmark. 2021. http://arxiv.org/abs/2010.09670, https://robustbench.github.io

[100] George Deckert, NASA Hazard Analysis Process. Johnson Space Center, National Aeronautics and Space Administration, 2010. https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20100040678.pdf.

[101] B.G. Doan, E. Abbasnejad, D.C. Ranasinghe. Februus: Input purification defense against trojan attacks on deep neural network systems. In ACSAC'20, pp. 897-912, 2020.

[102] Ann-Kathrin Dombrowski, Maximilian Alber, Christopher J. Anders, Marcel Ackermann, Klaus-Robert Müller, and Pan Kessel. Explanations can be manipulated and geometry is to blame. In *Proceedings of the Annual Conference on Neural Information Processing Systems 2019 (NeurIPS'19)*, pp. 13567–13578, 2019. https://arxiv.org/pdf/1906.07983.pdf

[103] Yizhen Dong, Peixin Zhang, Jingyi Wang, Shuang Liu, Jun Sun, Jianye Hao, Xinyu Wang, Li Wang, Jin Song Dong, and Dai Ting. There is Limited Correlation between Coverage and Robustness for Deep Neural Networks. arXiv:1911.05904 [cs.LG]. https://arxiv.org/abs/1911.05904

[104] Cynthia Dwork, Differential Privacy, In *Proc. ICALP'06*, pp.1-12, 2006.

[105] Cynthia Dwork and Aaron Roth, The Algorithmic Foundations of Differential Privacy, *Foundations and Trends in Theoretical Computer Science*, 9(3-4), pp.211-407, 2014.

[106] S. Elbaum and D. S. Rosenblum. Known Unknowns – Testing in the Presence of Uncertainty, In *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering (FSE 2014)*, pp. 833–836, 2014.

[107] Arisa Ema (ed.). *Special Issue: "AI Principles to Practice: An Introduction to International Activities"*. [Journal]Artificial Intelligence Vol. 36 No. 2. The Japanese Society for Artificial Intelligence. March 2021. (In Japanese)

[108] Ulfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova, RAPPOR: Randomized Aggregatable Privacy-Preserving Ordinal Response, arXiv:1407.6981v2, 2014.

[109] Itay P. Fainmesser, Andrea Galeotti, and Ruslan Momot, Digital Privacy, HEC Paris Research Paper, 2021.

[110] E. R. Faria, J. Gama, and A. C. Carvalho. Novelty detection algorithm for data streams multi class problems, In *Procedings of the 28th annual ACM symposium on applied computing*, pp. 795–800, 2013.

[111] Y. Feng, Q. Shi, X. Gao, J. Wan, C. Fang, and Z. Chen, DeepGini: Prioritizing Massive Tests to Enhance the Robustness of Deep Neural Networks, Proc. 29th ISSTA, pp. 177-188, and arXiv:1903.00661v2, 2020.

[112] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS '15),* pp. 1322–1333, 2015. https://dl.acm.org/doi/10.1145/2810103.2813677

[113] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, A survey on concept drift adaptation. In *ACM computing surveys*, 46(4):1–37, April 2014.

[114] Karan Ganju, Qi Wang, Wei Yang, Carl A. Gunter, and Nikita Borisov, Property Inference Attacks on Fully Connected Neural Networks using Permutation Invariant Representations, In Proc. 25th ACM CCS, pp. 619-633, 2018.

[115] Simson Garfinkel, John M. Abowd, and Christian Martindale, Understanding Database Reconstruction Attacks on Public Data, ACM Queue, September-October, pp.1-26, 2018.

[116] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems 27 (NIPS 2014)*, pp. 2672–2680, 2014.

[117] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. In *Proc. of the 3rd International Conference on Learning Representations (ICLR'15)*, 2015. https://arxiv.org/pdf/1412.6572.pdf

[118] J. Grana. Perturbing inputs to prevent model stealing. In CNS'20, pp.1-9, 2020.

[119] F. Harel-Canada, L. Wang, M. A. Gulzar, Q. Gu, and M. Kim. Is Neuron Coverage a Meaningful Measure for Testing Deep Neural Networks? In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE 2020)*, pp. 851–862, 2020.

[120] M. Hayashitani, J. Mori, and I. Teranishi. Survey of Privacy Threats and Countermeasures in Federated Learning. arXiv:2402.00342 [cs.LG], 2024. https://arxiv.org/abs/2402.00342

[121] Ori Heffetz and Katrina Ligett, Privacy and Data-Based Research, Journal of Economic Perspectives 28(2), 75-98, 2014.

[122] Mike Hintze, Viewing the GDPR through a De-Identification Lens: A Tool for Clarification, Compliance, and Consistency, International Data Protection and Law, 8(1), pp.86-101, 2018.

[123] Christina Ilvento. Metric Learning for Individual Fairness. arXiv:1906.00250 [cs.LG]. https://arxiv.org/abs/1906.00250

[124] L. Inozemtseva, and R. Holmes. Coverage is not Strongly Correlated with Test Suite

Effectiveness, In *Proceedings of the 36th International Conference on Software Engineering (ICSE 2014)*, pp. 435–455, 2014.

[125] Matthew Jagielski, Alina Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru and Bo Li. Manipulating Machine Learning: Poisoning Attacks and Countermeasures for Regression Learning. In *Proceedings of the 2018 IEEE Symposium on Security and Privacy (S&P'18)*, pp. 19–35, 2018.
https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8418594.

[126] Bargav Jayaraman and David Evans, Evaluating Differentially Private Machine Learning in Practice, In Proc. 28th USENIX Security Symposium, and also arXiv:1902.08874v4, 2019.

[127] Jinyuan Jia, Xiaoyu Cao, and Neil Zhenqiang Gong. Intrinsic Certified Robustness of Bagging against Data Poisoning Attacks. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI'21)*, 35(9), pp.7961-7969, 2021.
http://arxiv.org/abs/2008.04495

[128] J. Jia, A. Salem, M. Backes, Y. Zhang, and N. Z. Gong. MemGuard: Defending against Black-Box Membership Inference Attacks via Adversarial Examples. In *Proc. of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS'19)*, pp. 259–274, 2019.

[129] Mika Juuti, Sebastian Szyller, Samuel Marchal, and N. Asokan. PRADA: Protecting Against DNN Model Stealing Attacks. In *Proc. of the IEEE European Symposium on Security and Privacy (Euro S&P'19)*, pp.512–527, 2019.
https://arxiv.org/pdf/1805.02628.pdf

[130] Peter Kairouz, M. Brendan McMahan et al., Advances and Open Problems in Federated Learning, arXiv: 1912.04977v3, 2021.

[131] F. Kamiran, A. Karim, and X. Zhang. Decision theory for discrimination-aware classification. In *Proceedings of the IEEE International Conference on Data Mining (ICDM 2012)*, 2012.

[132] Faisal Kamiran, and Toon Calders. Data preprocessing techniques for classification without discrimination. In *Knowledge and Information Systems*, 33(1):1–33, 2012.

[133] T. Kamishima, S. Akaho, H. Asoh, and J. Sakuma. Fairness-aware classifier with prejudice remover regularizer. In *Machine Learning and Knowledge Discovery in Databases*, Lecture Notes in Computer Science, 7524:35–50, 2012.
https://link.springer.com/content/pdf/10.1007%2F978-3-642-33486-3_3.pdf

[134] Guy Katz, Clark Barrett, David Dill, Kyle Julian, and Mykel Kochenderfer. Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks. In International Conference on Computer-Aided Verification (CAV), 2017.

[135] Yusuke Kawamoto, Kazumasa Miyake, Koichi Konishi, and Yutaka Oiwa, Threats, Vulnerabilities, and Controls of Machine-learning-based Systems: A Survey and Taxonomy, arXiv:2301.07474v2 [cs.CR], 2023 https://arxiv.org/pdf/2301.07474.pdf

[136] J. Kim, R. Feldt, and S. Yoo. Guiding Deep Learning System Testing Using Surprise Adequacy, In *Proceedings of the 41st International Conference on Software Engineering (ICSE '19)*, pp. 1039–1049, 2019.

[137] Keita Kinjo. Fair causal effect using Social Welfare Function. The 35th Annual Conference of the Japanese Society for Artificial Intelligence, 2021

[138] P. Kiourti, W. Li, A. Roy, K. Sikka, S. Jha. MISA: online defense of trojaned models using misattributions. In ACSAC'21, pp. 570-585, 2021.

[139] Rob Kohavi. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. *The 14th International Joint Conference on Artificial Intelligence*, 2(12):1137–1143, 1995.

[140] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet Classification with Deep Convolutional Neural Networks. In *Procedings of the Advances in Neural Information Processing Systems 25 (NIPS 2012*), pp.1097–1105, 2012.

[141] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial Examples in the Physical World. In *Proc. of the 5th International Conference on Learning Representations (ICLR) Workshop*, arXiv:1607.02533 [cs.CV], July 2016, https://arxiv.org/pdf/1607.02533.pdf

[142] Mathias Lecuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. Certified Robustness to Adversarial Examples with Differential Privacy. *The IEEE Symposium on Security and Privacy* (SP), 2019.

[143] S. Lee, J. Kim, J. Jun, J. Ha, and B. Zhang. Overcoming catastrophic forgetting by incremental moment matching. In *Proc. of the Advances in Neural Information Processing Systems 30 (NIPS 2017)*, pp. 4652–4662, 2017.

[144] Yiming Li, Baoyuan Wu, Yong Jiang, Zhifeng Li, and Shu-Tao Xia. Backdoor Learning: A Survey. arXiv:2007.08745 [cs.CR], July 2020, https://arxiv.org/pdf/2007.08745

[145] P. Lindstrom, B. M. Namee, and S. J. Delany. Drift detection using uncertainty distribution divergence. In *Proceedings of the 2011 IEEE 11th International Conference on Data Mining Workshops*, pp. 604–608, 2011.

[146] Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *Proceedings of the International Symposium on Research in Attacks, Intrusions, and Defenses (RAID'18)*, pp. 273–294, 2018. https://arxiv.org/abs/1805.12185

[147] X. Liu, M. Cheng, H. Zhang, and C. Hsieh. Towards robust neural networks via random self-ensemble. In *Proceedings of the European Conference on Computer Vision (ECCV 2018)*, 2018.

[148] Yugeng Liu, Rui Wen, Xinlei He, Ahmed Salem, Zhikun Zhang, Michael Backes, Emiliano De Cristofaro, Mario Fritz, and Yang Zhang, ML-Doctor: Holistic Risk Assessment of Inference Attacks Against Machine Learning Models, USENIX Security

Symposium 2022.

[149] Yunhui Long, Vincent Bindschaedler, Lei Wang, Diyue Bu, Xiaofeng Wang, Haixu Tang, Carl A. Gunter, and Kai Chen, Understanding Membership Inferences on Well-Generalized Learning Models, arXiv:1802.04489, 2018.

[150] Alexandra Sasha Luccioni, Christopher Akiki, Margaret Mitchell, and Yacine Jernite, Stable Bias: Analyzing Societal Representations in Diffusion Models, arXiv:2303.11408 [cs.CY]

[151] Lei Ma, Felix Juefei-Xu, Fuyuan Zhang, Jiyuan Sun, Minhui Xue, Bo Li, Chunyang Chen, Ting Su, Li Li, Yang Liu, Jianjun Zhao, and Yadong Wang. DeepGauge: Multi-Granularity Testing Criteria for Deep Learning Systems, In *Proceedings of the 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE 2018)*, pp. 120–131, 2018.

[152] Shiqing Ma, Yingqi Liu, Guanhong Tao, Wen-Chuan Lee, and Xiangyu Zhang. NIC: Detecting Adversarial Samples with Neural Network Invariant Checking. In *Network and Distributed Systems Security Symposium (NDSS)*, 2019.

[153] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. In *the Sixth International Conference on Learning Representations (ICLR 2018)*, 2018.

[154] Saeed Mahloujifar, Esha Ghosh, Melissa Chase. Property inference from poisoning. In Proceedings of the 2022 IEEE Symposium on Security and Privacy (SP), pp. 1569-1569, 2022. https://arxiv.org/abs/2101.11073

[155] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A Survey on Bias and Fairness in Machine Learning. arXiv:1908.09635 [cs.LG], 2019.

[156] Ninareh Mehrabi, Muhammad Naveed, Fred Morstatter, and Aram Galstyan. Exacerbating Algorithmic Bias through Fairness Attacks. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI'21)*, pp. 8890-8938, 2021.

[157] B.P. Miller, L. Fredricksen, and B. So. An Empirical Study of the Reliability of UNIX Utilities, *Communications of the ACM*, 33(12):32–44, 1990.

[158] Fatemehsadat Mireshghallah, Mohammadkazem Taram, Praneeth Vepakomma, Abhishek Singh, Ramesh Raskar, and Hadi Esmaeilzadeh, Privacy in Deep Learning: A Survey, arXiv:2004.12254v5, 2020.

[159] Ilya Mironov, Renyi Differential Privacy, arXiv:1702.07476v3, 2017.

[160] Payman Mohassel and Yupeng Zhang. SecureML: A System for Scalable Privacy-Preserving Machine Learning. In Proceedings of the 2017 IEEE Symposium on Security and Privacy (S&P'17), pp.19-38, 2017.

[161] Ikuya Morikawa. *A Guide to Research Frontier of Machine Learning Security*. IEICE

ESS FUNDAMENTALS REVIEW, 15(1): 37-46. July 2021. (In Japanese)

[162] Sasi Kumar Murakonda and Reza Shokri, ML Privacy Meter: Aiding Regulatory Compliance by Quantifying the Privacy Risks of Machine Learning, arXiv:2007.09339, 2020.

[163] Moin Nadeem, Anna Bethke, and Siva Reddy, StereoSet: Measuring Stereotypical Bias in Pretrained Language Models. arXiv:2004.09456, 2020.

[164] Shin Nakajima. *Software Testing under Dataset Diversity*, Computer Software, 35(2):26-32, 2018. In Japanese.

[165] Shin Nakajima. Distortion and Faults in Machine Learning Software, In *Proc. 9th International Workshop on SOFL + MSVL for Reliability and Security*, pp. 29–41, 2020, and also arXiv:1911.11596 [cs.LG], 2019.

[166] Shin Nakajima. *Quality Issues in Machine Learning from Software Engineering Viewpoints*. Maruzen Publisher. ISBN 978-4-612-30573-7, November 2020. In Japanese.

[167] Shin Nakajima. *Distortion Metrics for Trained Machine Learning Models*. IEICE Technical Report. SS2019-44. March 2020. In Japanese.

[168] Shin Nakajima. Statistical Partial Oracle for Machine Learning Software Testing, In *Proceedings of the 10th International Workshop on SOFL + MSVL for Reliability and Security*, 2021.

[169] Shin Nakajima and Tsong Yueh Chen. Generating Biased Dataset for Metamorphic Testing of Machine Learning Programs, In Proceedings of the 31st IFIP International Conference on Testing Software and Systems (IFIP–ICTSS 2019), pp. 56–64, 2019.

[170] Shin Nakajima and Takako Nakatani, AI Extension of SQuaRE Data Quaity Model, In Proc. 1st IEEE Workshop on FPDRE, 2022.

[171] Takako Nakatani, Shin Nakajima. *Software Engineering*. Foundation for the Promotion of the Open University of Japan. ISBN 978-4-595-14119-5. March 2019. In Japanese.

[172] Arvind Narayanan and Vitaly Shmatikov, Robust De-anonymization of Large Sparce Datasets, In *Proc. SSP*, pp.111-125, 2008.

[173] Milad Nasr, Reza Shokri, and Amir Houmansadr, Comprehensive Privacy Analysis of Deep Learning: Passive and Active White-box Inference Attacks against Centralized and Federated Learning, arXiv:1812.00910v2, 2020.

[174] Maria-Irina Nicolae, Mathieu Sinn, Minh Ngoc Tran, Beat Buesser, Ambrish Rawat, Martin Wistuba, Valentina Zantedeschi, et al. Adversarial Robustness Toolbox v1.0.0. 2019.
http://arxiv.org/abs/1807.01069, https://adversarial-robustness-toolbox.readthedocs.io

[175] Steven Nowlan, and Geoffrey Hinton. Simplifying neural networks by soft weight-sharing. *Neural Computation*, 4(4), 1992.

[176] L. Oneto, N. Navarin, A. Sperduti, and D. Anguita. Recent Trends in Learning From Data. *Tutorials from the INNS Big Data and Deep Learning Conference (INNSBDDL2019)*, January 2020.

[177] Seong Joon Oh, Bernt Schiele, Mario Fritz. Towards Reverse-Engineering Black-Box Neural Networks. In Proceedings of the 6th International Conference on Learning Representations (ICLR'18), 2018.

[178] Tribhuvanesh Orekondy, Bernt Schiele, Mario Fritz. Knockoff Nets: Stealing Functionality of Black-Box Models. In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR'19), pp.4954-4963, 2019.

[179] Tinghui Ouyang, Vicent Sanz Marco, Yoshinao Isobe, Hideki Asoh, Yutaka Oiwa, and Yoshiki Seo. Corner Case Data Description and Detection. arXiv:2101.02494v2 [cs.LG]. https://arxiv.org/pdf/2101.02494.pdf

[180] Nicolas Papernot, Patrick D. McDaniel, Ian J. Goodfellow, Somesh Jha, Z. Berkay Celik, Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security (AsiaCCS 2017)*, pp.506-519, 2017.

[181] Nicolas Papernot, Patrick McDaniel, Arunesh Sinha, and Michael P. Wellman, SoK: Security and Privacy in Machine Learning, In *Proc. IEEE Euro S&P*, pp.399-414, 2018.

[182] Nicolas Papernot, Fartash Faghri, Nicholas Carlini, Ian Goodfellow et al. Technical Report on the CleverHans v2.1.0 Adversarial Examples Library. http://arxiv.org/abs/1610.00768 https://github.com/cleverhans-lab/cleverhans

[183] Arpita Patra and Ajith Suresh. BLAZE: Blazing Fast Privacy-Preserving Machine Learning. In Proceedings of the 27th Annual Network and Distributed System Security Symposium (NDSS'20), 2020

[184] Judea Pearl. Understanding Simpson's Paradox. *The American Statistician*, 68(1):8–13, February 2014. Also UCLA Cognitive Systems Laboratory Technical Report R-414, University of California, Los Angeles, December 2013.

[185] Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. DeepXplore: Automated Whitebox Testing of Deep Learning Systems, *The 26th Symposium on Operating Systems Principles (SOSP 2017)*, pp. 1–18, 2017.

[186] G. Pleiss, M. Raghavan, F. Wu, J. Kleinberg, and K. Q. Weinberger. On fairness and calibration. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS '17)*, December 2017.

[187] Jules Polonetsky, Omer Tene, and Kelsey Finch, Shades of Gray: Seeing the Full Spectrum of Practical Data De-identification, Santa Clara Law Review, 56(3), pp.593-629, 2016.

[188] Maria C. Psaoletti and Alessandro Simonetta, Data Quality and GDPR, UINOFO, 2019.

[189] Jonas Rauber, Wieland Brendel, and Matthias Bethge. Foolbox: A Python toolbox to benchmark the robustness of machine learning models. https://arxiv.org/abs/1707.04131 https://github.com/bethgelab/foolbox

[190] D. M. dos Reis, P. Flach, S. Matwin, and G. Batista. Fast unsupervised online drift detection using incremental Kolmogorov Smirnov test. In *Proc. of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1545–1554, 2016.

[191] Lucas Rosenblatt, Xiaoyan Liu, Samira Pouyanfar, Eduardo de Leon, Anuj Desai, and Joshua Allen, Differentially Private Synthetic Data: Applied Evaluations and Enhancements, arXiv:2011.05537, 2020.

[192] Elan Rosenfeld, Ezra Winston, Pradeep Ravikumar, and J. Zico Kolter. Certified Robustness to Label-Flipping Attacks via Randomized Smoothing. In *Proceedings of the 37th International Conference on Machine Learning (ICML'20)*, pp.8230–8241, 2020. https://arxiv.org/abs/2002.03018

[193] G. Rothermel, R.H. Untch, and M.J. Harrold, Prioritizing Test Cases for Regression Testing, IEEE TSE 27(10), pp. 929-948, 2001.

[194] Ahmed Salem, Yang Zhang, Mathias Humbert, Pascal Berrang, Mario Fritz, and Michael Backes, ML-Leaks: Model and Data Independent Membership Inference Attacks and Defenses on Machine Learning Models, arXiv:1806.01246, 2018.

[195] Pierangela Samarati and Latanya Sweeney, Protecting Privacy when Disclosing Information: k-Anonymity and Its Enforcement through Generalization and Suppression, 1998.

[196] Oscar Schwartz. 2019. In 2016 Microsoft's Racist Chatbot Revealed the Dangers of Online Conversation. IEEE Spectrum 11 (2019).

[197] T. S. Sethi, and M. Kantardzic. On the reliable detection of concept drift from streaming unlabeled data. *Expert Systems with Applications*, 82:77–99, 2017.

[198] B. Settles. Active Learning Literature Survey. Technical Report #1648, Computer Science Department, University of Wisconsin-Madison, 2009.

[199] S. Shalev-Shwartz. Online learning and online convex optimization, *Foundations and Trends® in Machine Learning*, 4(2):107–194, 2012.

[200] Reza Shokri, Marco Stronati, Congzheng Song and Vitaly Shmatikov. Membership Inference Attacks Against Machine Learning Models. In *Proceedings of the 2017 IEEE Symposium on Security and Privacy (S&P'17)*, pp. 3–18, 2017. https://arxiv.org/pdf/1610.05820.pdf

[201] I. Shumailov, Y. Zhao, D. Bates, N. Papernot, R.D. Mullins, R. Anderson. Sponge examples: Energy-latency attacks on neural networks. In: EuroS&P'21, pp. 212-231,

2021.

[202] Jacson Rodrigues Correia da Silva, Rodrigo Ferreira Berriel, Claudine Badue, Alberto Ferreira de Souza, Thiago Oliveira-Santos. Copycat CNN: Stealing Knowledge by Persuading Confession with Random Non-Labeled Data. In Proceedings of the International Joint Conference on Neural Networks (IJCNN'18), pages 1–8, 2018.

[203] Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Yarin Gal, Nicolas Papernot, Ross Anderson, The Curse of Recursion: Training on Generated Data Makes Models Forget, arXiv:2305.17493, 2023. https://arxiv.org/abs/2305.17493

[204] Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. Fooling LIME and SHAP: Adversarial Attacks on Post hoc Explanation Methods. In *Proc. of the AAAI/ACM Conference on AI, Ethics, and Society 2020 (AIES'20)*, pp.180–186, 2020. https://arxiv.org/pdf/1911.02508.pdf

[205] David Solans, Battista Biggio, Carlos Castillo. Poisoning Attacks on Algorithmic Fairness. In Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD' 20), Part I, p.162-177, 2020.

[206] Congzheng Song, Thomas Ristenpart, Vitaly Shmatikov. Machine Learning Models that Remember Too Much. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, (CCS'17)*, pp. 581-601, 2017.

[207] Congzheng Song and Vitaly Shmatikov, Overlearning Reveals Sensitive Attributes, Proc. ICLR 2020, and also arXiv:1905.11742v3, 2020.

[208] Liwei Song, Reza Shokri, Prateek Mittal. Membership Inference Attacks against Adversarially Robust Deep Learning Models. In *Proceedings of Deep Learning and Security Workshop (DLS)*, May 2019.

[209] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting, *Journal of Machine Learning Research* 15(56):1929–1958, 2014.

[210] Jacob Steinhardt, Pang Wei W Koh, and Percy S Liang. Certified Defenses for Data Poisoning Attacks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17)*, pp.3520–3532, 2017. https://arxiv.org/abs/1706.03691

[211] Latanya Sweeney. Weaving Technology and Policy Together to Maintain Confidentiality, The Journal of Law, Medicine and Ethics 25(2-3), pp.98-110, 1997.

[212] Latabya Sweeney. k-Anonymity: A Model for Protecting Privacy, International Journal on Uncertainty, Fuzziness and Knowledge-based Systems, 10(5), pp.557-570, 2002.

[213] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, Rob Fergus. Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199 (2013). https://arxiv.org/abs/1312.6199

[214] H. Tian, M. Yu, and W. Wang. Continuum: A platform for cost aware, low latency continual learning, In *Proceedings of the ACM Symposium on Cloud Computing*. pp. 26–40, 2018.

[215] Y. Tian, K. Pei, S. Jana, and B. Ray. DeepTest: Automated Testing of Deep-Neural-Network-driven Autonomous Cars. In *Proceedings of the 40th International Conference on Software Engineering (ICSE 2018)*, pp. 303–314, 2018.

[216] Vincent Tjeng, Kai Xiao, and Russ Tedrake. Evaluating robustness of neural networks with mixed integer programming. In *International Conference on Learning Representations*, 2019.

[217] Florian Tramèr, Fan Zhang, Ari Juels, Michael K. Reiter, and Thomas Ristenpart. Stealing Machine Learning Models via Prediction APIs. In *Proceedings of the 25th USENIX Security Symposium (USENIX Security'16)*.
https://arxiv.org/pdf/1609.02943.pdf

[218] Y. Uchida, Y. Nagai, S. Sakazawa, S. Satoh. Embedding watermarks into deep neural networks. In ICMR'17, pp. 269-277, 2017.
https://doi.org/10.1145/3078971.3078974

[219] Matej Uličný, Jens Lundström, and Stefan Byttner, Robustness of Deep Convolutional Neural Networks for Image Recognition, Communications in Computer and Information Science, CCIS 597, pp.16-30, 2016.
https://link.springer.com/chapter/10.1007/978-3-319-30447-2_2

[220] Guillermo Valle-Pérez and Ard A. Louis. Generalization bounds for deep learning. arXiv:2012.04115v2, 2020. https://arxiv.org/abs/2012.04115

[221] A. Vostrikov and S. Chernyshev. Training sample generation software. In *Intelligent Decision Technologies 2019*, Smart Innovation, Systems and Technologies (SIST), 143:145–151, 2019.

[222] Isabel Wagner and David Eckhoff, Technical Privacy Metrics: A Systematic Survey, ACM Computing Survey 51(3), Article 57, 2018.

[223] Binghui Wang, Neil Zhenqiang Gong. Stealing hyperparameters in machine learning. In Proceedings of IEEE Symposium on Security and Privacy (SP), pages 36–52, 2018.

[224] J. Wang, J. Chen, Y. Sun, X. Ma, D. Wang, J. Sun, and P. Cheng. RoBOT: Robustness-Oriented Testing for Deep Learning Systems, In *Proceedings of the 43rd International Conference on Software Engineering (ICSE 2021)*, 2021.

[225] Tianhao Wang, Jeremiah Blocki, Ninghui Li, and Somesh Jha, Locally Differentially Private Protocols for Frequency Estimation, *Proc. 26th USENIX Security Symposium*, pp.729-745, 2017.

[226] Zhibo Wang, Jingjing Ma, Xue Wang, Jiahui Hu, Zhan Qin, and Kui Ren. Threats to Training: A Survey of Poisoning Attacks and Defenses on Machine Learning Systems. *ACM Computing Surveys*, 2022. https://doi.org/10.1145/3538707

[227] Tsui-Wei Weng, Pin-Yu Chen, Lam Nguyen, Mark Squillante, Akhilan Boopathy, Ivan Oseledets, and Luca Daniel. PROVEN: Verifying Robustness of Neural Networks with a Probabilistic Approach. In *Proceedings of the 36th International Conference on Machine Learning (ICML 2019)*, PMLR 97:6727–6736, 2019.

[228] Tsui-Wei Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Duane Boning, Inderjit S. Dhillon, and Luca Daniel. Towards Fast Computation of Certified Robustness for ReLU Networks. In *Proceedings of the 35th International Conference on Machine Learning*, PMLR 80:5276-5285, 2018.

[229] E. J. Weyuker. On Testing Non-testable Programs, Computer Journal, 25(4):465–470, 1982.

[230] Eric Wong and J. Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. *The 35th International Conference on Machine Learning (ICML 2018)*, PMLR vol. 80, pp. 5283–5292, 2018.

[231] Min Wu, Matthew Wicker, Wenjie Ruan, Xiaowei Huang, and Marta Kwiatkowska. A Game-Based Approximate Verification of Deep Neural Networks with Provable Guarantees. In *Theoretical Computer Science,* 2019.
https://doi.org/10.1016/j.tcs.2019.05.046

[232] Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou, Differentially Private Generative Adversarial Network, arXiv: 1802.06739, 2018.

[233] Weilin Xu, David Evans, and Yanjun Qi. Feature Squeezing: Detecting Adversarial Examples in Deep Neural Networks. In *Network and Distributed Systems Security Symposium* (NDSS), 2018.

[234] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha, Privacy Risk in Machine Learning: Analyzing the Connection to Overfitting, *IEEE Computer Security Foundations Symposium* (CSF), and also arXiv:1709.01604v5, 2018.

[235] Kota Yoshida and Takeshi Fujino. *Hardware Security for Edge AI Devices*. IEICE ESS Fundamentals Review, Vol. 15, No. 2, pp. 88-100. 2021. In Japanese.

[236] B. H. Zhang, B. Lemoine, and M. Mitchell. Mitigating Unwanted Biases with Adversarial Learning. In *AIES '18: Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, December 2018.
https://dl.acm.org/doi/pdf/10.1145/3278721.3278779.

[237] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals, Understanding Deep Learning Requires Rethinking Generalization, arXiv:1611.03530v2, 2017.

[238] Jie M. Zhang, Mark Harman, Lei Ma, and Yang Liu. Machine Learning Testing: Survey, Landscapes and Horizons. arXiv:1906.10742 [cs.LG]. https://arxiv.org/abs/1906.10742

[239] P. Zhang, Q. Dai, and P. Pelliccione. CAGFuzz: Coverage-Guided Adversarial Generative Fuzzing Testing of Deep Learning Systems. arXiv:1911.07931, 2019.

[240] P. Zhang, J. Wang, J. Sun, G. Dong, and X. Wang. White-box Fairness Testing through Adversarial Sampling. In *Proceedings of the 42nd International Conference on Software Engineering (ICSE 2020)*, pp. 1–12, 2020.

13.5 Miscellaneous

[241] Information-technology Promotion Agency, Japan. *STAMP/STPA for beginners*. June 2019. In Japanese. https://www.ipa.go.jp/ikc/reports/20190329.html

[242] Security Center, Information-technology Promotion Agency, Japan. *Guidance for quality assurance for connected worlds*. June 2018. In Japanese. https://www.ipa.go.jp/sec/publish/tn18-001.html

[243] Security Center, Information-technology Promotion Agency, Japan. *Overview of Common Platform Enumeration (CPE)*. October 2008. In Japanese. https://www.ipa.go.jp/security/vuln/CPE.html

[244] Secuity Center, Information-technology Promotion Agency, Japan. *Overview of Common Vulnerability Enumeration (CVE)*. January 2009. In Japanese. https://www.ipa.go.jp/security/vuln/CVE.html

[245] National Institute of Advanced Industrial Science and Technology, Report on quality evaluation/improvement technology (2nd Edition), Technical Report, Digital Architecture Research Center / Cyber Physical Security Research Center / Artificial Intelligence Research Center, Digiarc-TR-2023-02 / CPSEC-TR-2023002, 2023.

[246] National Institute of Advanced Industrial Science and Technology, Report on quality evaluation/improvement technology (3rd Edition), Technical Report, Digital Architecture Research Center / Cyber Physical Security Research Center / Artificial Intelligence Research Center. In Japanese. Digiarc-TR-2023-04 / CPSEC-TR-2023004, 2023. (The 3rd edition has not been published in English yet. See the 2nd edition written in English, instead)

[247] IBM Corporation. AI Fairness 360 - Resources. http://aif360.mybluemix.net/resources

[248] Information Commissioner's Office (United Kingdom), Guidance on the AI Auditing Framework – draft guidance for consultation, 2020.

[249] United States Census Bureau, Your Guide to the 2020 Census, 2020.

[250] United States Census Bureau, Disclosure Avoidance for the 2020 Census: An Introduction, 2021.

[251] National Institute of Advanced Industrial Science and Technology. *Machine Learning Quality Management Reference Guide*. Technical Report, Digital Architecture Research Center, Cyber Physical Security Research Center, Artificial Intelligence Research Center. In Japanese. DigiARC-TR-2022-03 / CPSEC-TR-2022004, 2022

[252] Machine Learning Engineering Research Group. *Machine Learning System Security Guidelines Version 1.0*. 2022. In Japanese.

[253] Taichi Kakinuma. *The impact of the Revision of the Copyright Law on AI Development*. 2nd Symposium on Automatic Translation. March 2019. In Japanese. 2019 年 3 月 https://h-bank.nict.go.jp/event/event_190306.html

[254] Nobuyoshi Ashibe. *The Right of Privacy*. Chapter 7 Section 1 Item 3. *Kenpo* (The Constitutional Law). pp.101-107 Iwanami Shoten, Publishers. 1993. In Japanese.

[255] Masao Horibe. *Historical Development of Privacy Protection Legislation*. Horitsu Bunka pp.18-21 November 2002. In Japanese.

14. Revision history

14.1 English version of the fourth edition (June 2026)

- Changes introduced in the Japanese version of the fourth edition have been incorporated.

14.2 Japanese version of the fourth edition (December 2023)

- An internal quality A-0: Sufficiency of preliminary problem analysis has been introduced.
- An internal quality E-0: Continuous monitoring and recording during operations has been introduced.
- Internal qualities concerning privacy, fairness, and AI security have been introduced. Requirements, controls, and how to apply technologies for these quality characteristics have been described in the same chapter as the other internal qualities.
- The description of AI security in Section 10.3 has been expanded to provide a more organized overview of assets, stakeholders, threats, vulnerabilities, and controls.

14.3 English version of the third edition (January 2023)

- Changes introduced in the Japanese version of the third edition have been incorporated.
- Section 7.6.1.7 has been reinstated together with three of the four bibliographic items, which had been added to the English version of the second edition but lost in the Japanese version of the third edition.

14.4 Japanese version of the third edition (August 2022)

- The description of privacy as an external quality, including its analysis and handling methods, was considered and added.
- Security was positioned as an external quality, and its relationship with existing characteristics was organized, and the descriptions in the second edition were expanded.
- Reorganized the descriptions in the chapter on fairness.
- Updated other descriptions.
- Table numbers for Table 65 in Section 9.3 and all tables after that were corrected in Rev. 3.1.1.0077 issued on September 2022.

- An editorial mistake in Section 7.6.1.6 was corrected.
- Added Section 7.6.1.7. (Rev. 3.2.1.0078, January 2023)

14.5 English version of the second edition (February 2022)

- Section 7.6.1 has been reorganized and extended with a new section, Section 7.6.1.7
- Four new items have been added to the bibliography and cited in Section 7.6.1

14.6 Japanese version of the second edition (July 2021)

- The definition of fairness in Section 1.5.3 was updated based on more detailed discussions.
- Internal qualities in Section 1.7 and Chapter 6 have been categorized into groups A to D and given labels from A-1 up to E-1.
- The internal quality B-3: Adequacy of data has been detached from the quality C-1: Correctness of trained models and expanded.
- Some of the internal qualities have been given new names.
- Chapter 8 on fairness and Chapter 10 on Security have been added.

Editors and authors

Intelligent Platforms Research Division (IPRI)/
Artificial Intelligence Research Center (AIRC)/
Cyber Physical Security Research Division (CPSEC),
National Institute of Advanced Industrial Science and Technology (AIST)

The Committee for Machine Learning Quality Management (FY2023)

Shin Nakajima	National Institute of Advanced Industrial Science and Technology/ National Institute of Informatics (chair)
Yoshiki Seo	National Institute of Advanced Industrial Science and Technology (vice-chair)
Isao Echizen	National Institute of Informatics
Takashi Egawa	National Institute of Advanced Industrial Science and Technology
Shintaro Fukushima	Toyota Motor Corporation
Chinami Hamatani	Ad-Sol Nissin Corporation
Kenichi Kobayashi	Fujitsu Laboratories Limited
Hideaki Komiyama	Konica Minolta, Inc.
Koichi Konishi	National Institute of Advanced Industrial Science and Technology
Hiroshi Kuwajima	Denso Corporation
Yusei Nakashima	TechMatrix Corporation
Takaaki Namba	Panasonic Corporation
Hideto Ogawa	Hitachi, Ltd.
Yutaka Oiwa	National Institute of Advanced Industrial Science and Technology
Tamao Okamoto	Panasonic Corporation
Naoto Sato	Hitachi, Ltd.
Tomomichi Suzuki	Tokyo University of Science
Shingo Takada	Keio University
Satoshi Tsuchiya	Fujitsu Limited
Naoya Wakamatsu	NEC Corporation
Atsushi Yamada	IBM Japan, Ltd.

List of authors of the Machine Learning Quality Management Guideline

Japanese edition:

- Overall structure, writing and editing:
  Yutaka Oiwa and Koichi Konishi (AIST)
- Fairness: Sections 3.9, 4.4, 7.1.3, 7.7.1, 7.10.1, 10.1
  Yutaka Oiwa, Chinami Hamatani (Ad-Sol Nissin)
- Privacy: Sections 4.5, 7.1.4, 7.7.2, 7.10.2, 9.7.1, 9.9.1, 10.2
  Shin Nakajima (NII)
- AI Security: Sections 4.6, 7.1.5, 7.6.2.4, 7.10.3, 7.12.1, 8.1.2.1.2, 9.9.2, 9.11.1, 10.3.1, 10.3.6
  Yutaka Oiwa, Yusuke Kawamoto (AIST),
  Kazumasa Miyake (Sumitomo Electric), Koichi Konishi
- Section 9.8.1 Shin Nakajima
- Section 9.8.2 Yoshinao Isobe (AIST)
- Section 9.13 Kenichi Kobayashi and Yoshihiro Okawa (Fujitsu)
- Section 10.3.7 Kazumasa Miyake
- Section 11.2 Takashi Egawa (AIST), Hiromu Kitamura (CDLE AI Legal, NEC), Hiroshi Kuwajima (Denso)
- Supervision: Members of
  - Committee for Machine Learning Quality Managemet
  - Guideline Taskforce

English edition: Members of the Guideline Taskforce

- Editors: Yutaka Oiwa and Koichi Konishi

Members of the Guideline Taskforce (FY2023)

Toshinori Araki	NEC Corporation
Takashi Egawa	National Institute of Advanced Industrial Science and Technology
Shintaro Fukushima	Toyota Motor Corporation
Chinami Hamatani	Ad-Sol Nissin Corporation
Masahiro Hayashitani	NEC Corporation
Yoshinao Isobe	National Institute of Advanced Industrial Science and Technology
Yusuke Kawamoto	National Institute of Advanced Industrial Science and Technology
Hiromu Kitamura	CDLE AI Legal (NEC Corporation)
Kenichi Kobayashi	Fujitsu Limited
Hideaki Komiyama	Konica Minolta, Inc.
Hiroshi Kuwajima	Denso Corporation
Kazumasa Miyake	Sumitomo Electric Industries Ltd.
Takeshi Miyake	Cyber Creative Institute Company Limited
Shin Nakajima	National Institute of Advanced Industrial Science and Technology
Yusei Nakashima	TechMatrix Corporation
Takaaki Namba	Panasonic Holdings Corporation
Kyoko Ohashi	Fujitsu Limited
Yutaka Oiwa	National Institute of Advanced Industrial Science and Technology
Tamao Okamoto	Panasonic Holdings Corporation
Yoshiki Seo	National Institute of Advanced Industrial Science and Technology
Takashi Tabe	NEC Corporation
Satoshi Tsuchiya	Fujitsu Limited
Naoya Wakamatsu	NEC Corporation