Rev. 1.0.2.0037.1（2021/03/09）

機械学習品質マネジメントガイドライン

第 1 版

2020 年 6 月 30 日

（訂補 2 版: 2021 年 3 月 9 日）

国立研究開発法人産業技術総合研究所

サイバーフィジカルセキュリティ研究センター

テクニカルレポート CPSEC-TR-2020001

人工知能研究センター

テクニカルレポート

前書き（Foreword and Disclaimer）

本ガイドラインは、国立研究開発法人新エネルギー・産業技術総合開発機構（NEDO）からの受託事業の一部として、国立研究開発法人産業技術総合研究所（産総研・AIST）と大学共同利用機関法人情報・システム研究機構国立情報学研究所（NII）が企業・大学等の有識者委員とともに構成した「機械学習品質マネジメント検討委員会」においてとりまとめたものである。

本ガイドラインの内容に寄与した委員の意見は技術者としての個人の知見に基づくものであり、各々が所属する会社等の意見を代表するものではない。

本ガイドラインは、機械学習人工知能を利用したシステム・サービスの開発を主導する企業等が、そのビジネス等への影響を踏まえて主体的にその採用の有無を選択し、共同開発者等とともに実践するものであって、法令・公的指針等との関係においては非拘束的なものである。本ガイドライン中で規範的（normative）とされる規定は、あくまで本ガイドラインを任意に採用した場合に限り、規範的な意味を持つものである。

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

目次

1　ガイドライン全体概要

本章の内容は参考（informative）である。本章に含まれ、本ガイドラインの規範の一部を構成する（normative）内容については、後の章で再掲する。

本概要の全体構成は以下の通りである。第 2 章以降の本編に対応する内容がある場合には、その対応する章節も示す。

- 1.1 節では、本ガイドラインを作成した背景と目的を示す。
- 1.2 節では、本ガイドラインが主に想定する「使われ方」を提示する。（本編 2 章）
- 1.3 節では、背景として掲げた「AI の品質管理が困難である理由」を分析する。
- 1.4 節では、本ガイドラインがベースとする、品質管理のプロセスについての全体的な考え方を述べる。
- 1.5 節では、本ガイドラインが「目標」として設定する 3 つの「外部品質」の観点（実装手段にあまり依存せず、使用を通じてしか評価できない品質観点）を提示する。（本編 3 章）
- 1.6 節では前節を補足して、一般に「AI の品質」として議論される要素のうち、前節で採用しなかった要素について、その理由や本ガイドラインでの考え方を説明する。
- 1.7 節では、本ガイドラインが「管理手段」として設定する 8 つの「内部品質」の観点（実装手段に依存し、計測や作り込みでの管理が可能な品質観点）を提示する。（本編 6 章・7 章）
- 1.8 節では、本ガイドラインを作成するに当たって想定する開発プロセスの全体のイメージを提示する（本編 4 章・5 章）
- 1.9 節では、外部の規範文書などとの関係を明らかにする。（本編 8 章）
- 1.10 節では、本ガイドラインの残りの部分の構成を整理する。

1.1　目的と背景

人工知能（AI）、とりわけ機械学習技術は、製造業、自動運転、ロボット、ヘルスケア、金融、小売などの幅広い応用分野で有効性が確認され、社会実装が本格化する兆しを見せている。その一方、AI を利用した製品・サービスの品質を測定し説明する技術の不足に起因

し、万が一の事故の際に原因が特定できず、また投資に見合う AI システムの優位性を説明できず、結果として、社会的な受容性を得るための制度設計の遅れや、AI 開発ビジネス拡大への大きな障害となっている。

本ガイドラインは、機械学習を利用したシステム、特にその中に含まれる機械学習で実装されたソフトウェアコンポーネント（機械学習要素）の品質に関する基準と達成目標を定めることにより、企業が自ら構築した AI を利用するシステムの品質を測定し向上させ、また AI の誤判断による事故や経済損失などを減少させる一助となる事を目的とする。さらに、このようなアプローチを取ることで、達成した品質の認識を開発のステークホルダ間で共有したり、社会に対し具体的に示し説明することができるようになったりすることで、受発注などの条件の明確化や問題点の特定、高品質による付加価値の提示などが実現し、「良いシステムを作る事業者がより高く評価される」健全なビジネス環境の実現にも寄与することを目的とする。

1.2　本ガイドラインの使われ方

本ガイドラインが想定する一次的な読者は、機械学習を利用して作られる製品やサービスの提供者（ここでは「サービス開発者」とする [1]）と、実際に製品・サービスをソフトウェアとして実装するシステム開発者である。このサービス提供者とシステム開発者は、自社で製品・サービスを開発しエンドユーザに提供する単一の主体（本ガイドラインでは、「自己開発者」と呼ぶ。）である場合もあれば、契約に基づく分担（請負や準委任）の関係により異なる主体である場合も有り得る。さらに、システム開発者の中でも例えばシステム全体の設計開発と機械学習コンポーネント部分の実装などについて、さらに個別の分担関係がある場合もありえる。製品・サービスが利用される状況に応じて必要とされる品質に関して、これらの主体が明確な目標を共有し、システム開発プロセスの全体を通じてその品質を実現するために参照されることが、本ガイドラインが主に想定する利用形態である。

さらに二次的な利用形態として、その製品やサービスの利用者に対し、サービス提供者が本ガイドラインに従って設定し達成した品質を示すことで、安心して利用できる判断根拠を示すことが想定される。その延長として、われわれの社会全体が製品・サービスに一定の

[1] ソフトウェア事業者などが、他者が提供・自己使用するサービスを想定したシステムを予め企画・開発しておき、パッケージとして、あるいはカスタマイズを行って販売するような事業形態については、この企画・開発を行う主体をサービス提供者に準ずるものとして扱う。

品質を求める目的で、規格策定団体や第三者検証機関などが品質を評価し認定する基準を定める際の技術的な出発点となることも将来的には期待される。

このガイドラインは、できるだけ広範な応用に適用できるような汎用的な記述内容になっており、利用者が具体的な応用に即して、記述内容を取捨選択・具体化して用いることを想定している。実際の利用場面においては、各利用者の状況に応じて、応用分野ごとにこのガイドラインの内容を具体化した個別ガイドラインを作ることを想定している。本ガイドライン検討プロジェクトでも、いくつかの事例について、参照事例を作成・公表することを計画している。

本ガイドラインの具体的な利用形態としては、例えば次のようなものが挙げられる。

- AI システム開発に関する体制構築（契約など）の場面
  - 機械学習を利用するシステムまたはその一部となる機械学習の製品要素の開発を依頼する主体（本ガイドラインでは「開発依頼者」と呼ぶ[2]。）が、ガイドラインを開発対象の応用に合わせて具体化し、受託する主体（「開発協力者」とよぶ。）に対して契約要件となる仕様として指定する。
  - 開発協力者となる予定の者が、開発依頼者に対して品質を保証するための工程管理の標準として参照し、工数や受注価格を算出する根拠あるいは参考とする。
- 設計・開発の場面
  - 機械学習利用システムまたは機械学習要素を設計・開発する者（開発協力者または自己開発者）が、その開発の工程設計・システム設計及び品質管理の基準とする。
- 社会的な位置づけ
  - 自己開発者または開発依頼者が、システムを社会に提供・自己業務で活用する際に、社会規範としての品質要求に対する自己適合宣言の拠り所とする。
  - 将来的に機械学習利用システムの品質に関する社会合意としての基準とする。
  - 機械学習利用システムの品質に関する第三者評価制度を設計・運用する際の基準とする。

[2] 経済産業省がまとめた「AI・データの利用に関する契約ガイドライン」は、主に B2B（ビジネス当事者間）の開発契約の視点からまとめられており、本ガイドラインの「開発依頼者」を『ユーザ』と、「開発協力者」を『ベンダ』または『SIer』と整理している。本ガイドラインでは最終的な利用時品質の享受者を B2C（ビジネス～消費者間）の製品・サービス利用者に置く立場から、「ユーザ」の語は「Customer」にあたる製品・サービス利用者を指すものとしてのみ用いる。

なお、本ガイドラインでは、機械学習要素の作り方として主に「教師あり学習」(supervised learning) による実装を想定している。基本的な考え方はその他の実装方法、例えば教師なし学習（unsupervised learning)、半教師あり学習（semi-supervised learning)、強化学習（reinforcement learning) などにも通用するものもあるが、これらの具体的な扱い方については、今後の改訂時に記述を追加する予定である。

1.3　機械学習の品質管理に関する課題

機械学習による人工知能の実装は、単純に言ってしまえばソフトウェアの一種であるといえる。しかし、従来のソフトウェアに対する品質管理の考え方だけでは、機械学習を利用したシステムの品質を向上させ維持していくには技術的に不足する点がいくつかある。本節では、そのような「従来ソフトウェアとの差異」について、いくつかの観点から課題を整理する。

1.3.1　環境分析の重要性

機械学習を利用する製品やサービスは、往々にして人間がその複雑さを把握しきれないような環境条件で用いられることが多い。全ての環境条件の複雑さを人が分析・特定し判断ルールとして逐一プログラムコードを実装しなければならない通常のプログラムと比較して、環境条件の子細を人が分析作業により特定しなくてもデータから自動的に判断ルールを構築できる機械学習技術は、高い複雑さを持つ環境条件で動作するシステムの実装を効率的に行う手段として大いに期待されている。

一方で、理想的な品質管理の観点、特に稀な環境条件への適合性が問われるようなシステムの品質管理の観点からは、システム設計の初期段階でできるだけ環境条件を緻密に分析し、リスクなどを把握しておくことが望ましい。機械学習技術をこのようなシステムに利用すると、従来はプログラムコードの実装時に行われていた環境条件の分析が省かれることになることから、初期段階での環境分析がより高い重要性を持つことになる。

このような実装の効率化と環境適合性の確保の２つの特質を両立させる上で、利用状況・環境条件の分析を設計段階でどのような詳細度で行うべきかは、従来のソフトウェア開発との相違点も含めて、システム設計の初期において検討すべき重要な事柄となる。

1.3.2　継続的なリスクアセスメント

一般に実社会で動作し、いわゆるサイバーフィジカルシステム（cyber physical systems, 以下「CPS」という。）の一部を構成するシステムにおいては、通常のソフトウェアシスエムに存在するリスクに加え、外部の環境変化に起因するリスクが常にある。未知の状況をある程度論理的に分析し想定・対策できる可能性のある通常のソフトウェア実装と異なり、実在するデータを元に構築する機械学習利用システムは、（その汎化性能に期待することはあっても）周囲状況の変化に起因するデータ傾向の大きな変化に追従できない可能性がある。

また一般に、機械学習を利用したシステムにおいては、過去のデータに基づく初期の訓練段階だけでなく、運用開始時の実データを用いた追加的学習を経て初めて実用的な品質を達成するような設計がされる場合も想定できる。

このような観点から、機械学習を利用するシステムにおいては、いわゆる「バグ」「初期不良」への対応として必要な修正だけでなく、当初から運用開始後の状況変化への対応を想定し、開発・運用を一体化した継続的ライフサイクルプロセスを導入することが必要となる場合が多い。開発計画の初期段階であらかじめ運用段階を含むライフサイクルを想定し、運用計画や受発注の契約内容などにも運用段階で必要な作業を折り込んでおく必要がある。

ただし、品質管理を継続的に行う場合においても、初期構築された運用開始時点での成果物について一定の品質の担保は必要である。特に、安全性などへの脅威となるリスクを伴うシステムにおいては、初期段階で一定の品質を確保することは不可欠である。本ガイドラインではこのような観点から、特に実装段階から運用を開始するまでに至る段階での品質検査に重点を置いている。さらに、運用の初期と本格運用時でシステムの運用形態（例えば人間による監視・介入の有無などの回避可能性の状況）を変化させることで、リスクの影響を変化させ、必要となる品質レベルを変更することも考えられる。このような場合には、運用形態を変化させる時点と品質管理の目標を変更する時点を同期させる必要がある。

また、運用時点でシステムの実装を更新する場合、その更新が品質を向上させる目的であっても、時によっては品質がかえって劣化する（ソフトウェア工学におけるリグレッション）リスクもある。そのことから、何らかの形での運用時の品質監視・対策も必要になると考えられる。そのような対策は開発・運用の形態によって異なることから、いくつかの運用モデルを 4.3 節に整理した。

1.3.3　データに依存した品質確保

機械学習などのデータ主導による開発において、構築に用いる「データの品質」に関する要求はしばしば言及される。本ガイドラインでは、データの品質そのものは品質管理における最終的な目的ではなく、品質劣化の原因や品質確保の手段であるとの考え方に立つ。もちろん、実際に機械学習の品質を、特に本ガイドラインのようなライフサイクルプロセス管理を通じて確保するためには、データに一定の品質を確保することがほぼ必須の手段となる。また、機械学習利用システムを分業で開発する際には、学習用のデータそのものが売買や開発役割分担の対象となることがあり、このような場合には「データの品質」を単体の性質として議論する余地がある。更には、最近では学会などにおいて、不正データの意図的混入による学習結果の汚染のセキュリティリスクも指摘されている。このような不正データの影響は単純な数値評価では検出できず、データの出所についての何らかの広い視点からの担保が必要である。

本ガイドラインはできるだけ品質を数値評価で担保することを是としつつも、必要な場面ではデータに対する定性的な品質管理を通じて品質を実現することが必要となると考える。

1.4　品質管理の基本的な考え方

参考）本ガイドラインでは、システム全体で最終的な利用者に提供すべき品質として「利用時品質」を捉える。他方、実際にシステムが提供する品質としては、Systems Engineering の考え方から、システムの構成要素を階層的に整理し、その構成要素毎に「外部品質」「内部品質」を考える[3]。

各構成要素の「外部品質」は、その構成要素がシステムの部品として要求される、客観的な視点の品質のことを言い、例えば、セキュリティ、信頼性、一貫性などが挙げられる。一般論としてこの外部品質は、定性的や定量的なものを含んでいて、必ずしも、計測可能な単一の指標によって示せるものではない。

[3] 本ガイドラインで用いる「外部品質」「内部品質」の語は、ISO 9126 およびその後継である ISO 25000 シリーズにおいて用いられる External/Internal Quality の語とは、厳密には一致しない。特に外部品質に関して本ガイドラインは、要求される品質の「レベル」を設定するが、品質指標として必ずしも直接的に測定できるものではないと考える点に、注意が必要である。

一方で、各要素の「内部品質」は、構成要素を作成する際に具体的に測定したり、設計などの開発行為を通じて評価したりする、その要素が固有で持つ特性としての品質のことを言う。

このような整理をもとに、各要素の外部品質はその要素の内部品質の向上を通じて実現され、1 つ外側の要素の内部品質を達成するために要求されるという、図 1 のような階層的な品質モデルを想定する。システム全体の「外部品質」は、最終的な製品・サービス利用者から見た「利用時品質」のために、製品・サービスの提供者が実現し提供するものとして考えられる。

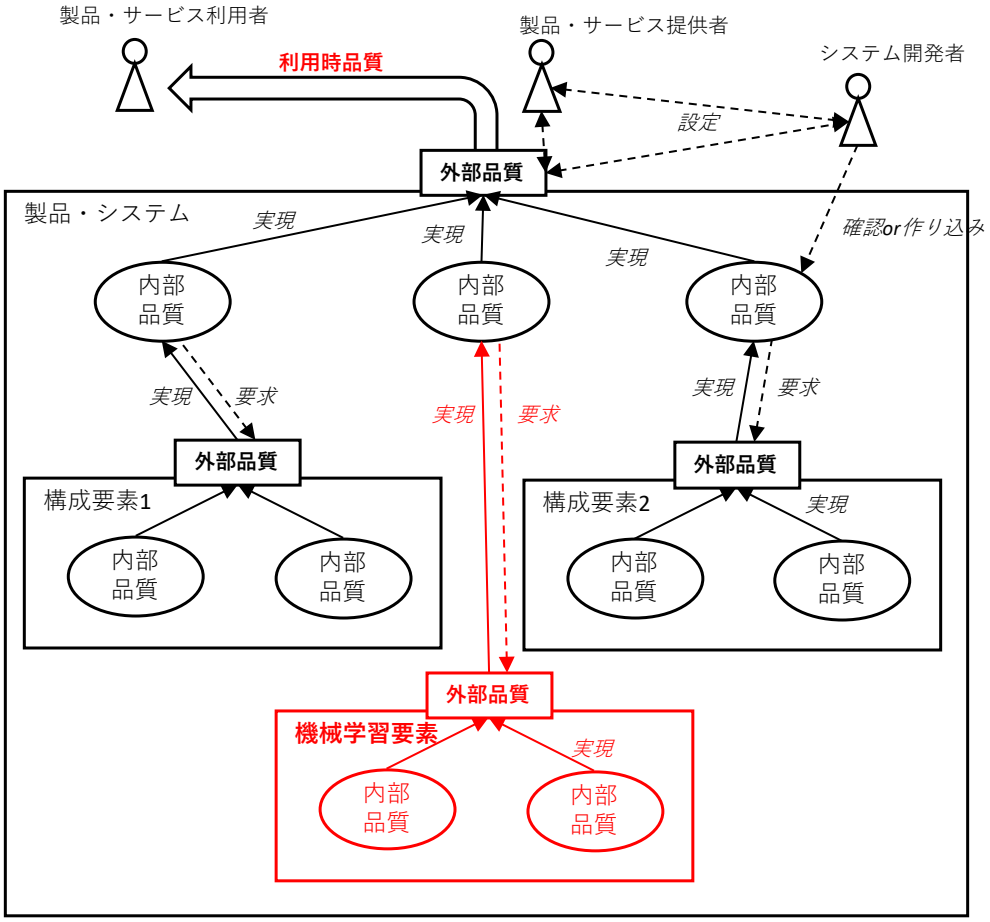

図 1　階層的な品質モデル

本ガイドラインでは、機械学習システムにおける「品質」そのものを、

- システムがその全体として利用時に満たすことが期待される「利用時品質」

- システムのうち機械学習で構築された構成要素が満たすことが期待される「外部品質」
- 機械学習による構成要素が固有に持つ「内部品質」

の 3 つに分けて理解し、機械学習要素の「内部品質」の向上を通じてその「外部品質」を必要となるレベルで達成し、最終的なシステムの「利用時品質」を実現するものと整理する（図 2）。

品質管理の目標とする機械学習要素の外部品質としては 1.5 節に掲げる 3 つの観点を設定した。そして、その達成手段としての内部品質としては機械学習要素特有の観点に注視し、現時点では 1.7 節に掲げる 8 つの観点をそれぞれ抽出した。外部品質の 3 つの観点に対して品質目標のレベル付けを行い、そのレベルに応じて 8 つの内部品質観点それぞれに達成目標を設定し、様々な技術や開発プロセス管理を通じてその目標を達成するという流れが、本ガイドラインにおける品質管理の基本的な考え方となる。

なお、最終的に実現するシステム全体の利用時品質については、その応用毎に着目すべき内容が異なることから、本ガイドラインでは具体的に規定しない（下記補足も参照）。

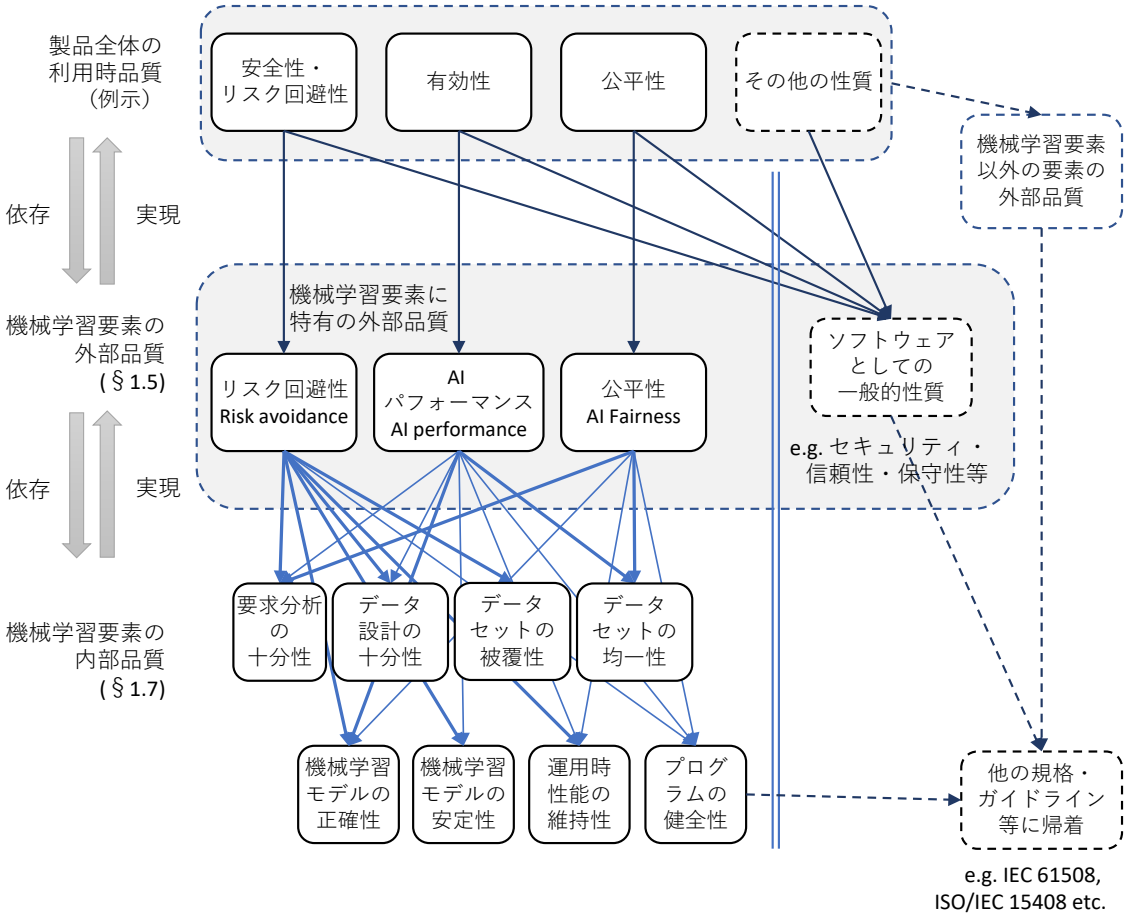

図 2: 製品品質実現の全体構造

例 1）

自動運転を行う自動車に搭載される前方映像からの物体認識モジュールを想定する。自動車全体の利用時品質の 1 つとしては「あらゆる運転可能な環境条件下で障害物に衝突しない安全性」が考えられ、それを実現する為に物体認識モジュールには、「想定される天候や時間帯などの全てにおいて、障害物を正しく認識する」性質が外部品質として求められる。それを実現する内部品質として、例えば学習状況の網羅性などがあり、これを機械学習の訓練用データの構成プロセスなどにより実現する。

例 2）

株の自動取引を行うサービスに内包される、株価予想 AI を想定する。サービス全体の利用時品質の 1 つとしては「利益の最大化」が考えられ、株価予想を行うモジュ

ールには、「株価予測の誤差の最小化や、想定される取引結果の総和を最大化する」などの性質が外部品質として求められる。それを実現する内部品質としては、例えば機械学習の推論精度などがあり、これを機械学習の訓練パラメータの最適化などを通じて実現する。

（補足）IEC 61508[4]や IEC 62278[5]などの、産業システム全体の安全性・信頼性の評価を行う厳格なプロセスを用いて開発する場面においては、従来からのシステム全体の設計プロセスにおいて、その一部の構成要素としての機械学習要素に対する外部品質要求は当然に特定される。

一方で、特に機械学習技術の需要が大きい IT 系サービスなどの応用においては、産業システムほど想定されるリスクが大きくない場合が多く、厳格なリスク管理プロセスをシステム開発全体にわたって適用することが必要でないこともあるだろう。

また、いわゆる人工知能技術全般の使い方として、「賢い判断」を機械学習要素に多少なりとも期待している場合、機械学習要素の外部品質のうち、特に本節で掲げた 3 つの観点は、システム全体の外部品質や利用時品質に、ある程度直接的な対応が考えられるような場合が多いことも想定される。上に掲げた 2 つの例は、そのような事例の具体的な例示になっている。

そのような観察から、図 2 ではシステム全体の利用時品質と機械学習要素の外部品質に対応があるような表記とし、また利用時品質として「安全性・リスク回避性」「有効性」「公平性」の 3 つを例示した。

1.5　実現目標とする外部品質特性

本ガイドラインでは、以下の 3 つの異なる性質（リスク回避性、AI パフォーマンス、公平性）を機械学習要素の外部品質特性の軸として抽出し、それぞれに品質レベルを設定した。

[4] IEC 61508-1:2010: Functional safety of electrical/electronic/programmable electronic safety-related systems - Parts 1.

[5] IEC 62278:2002: Railway applications - Specification and demonstration of reliability, availability, maintainability and safety (RAMS).

1.5.1　リスク回避性

『リスク回避性』（危害・危険回避性／安全性）は、機械学習要素が望ましくない判断動作を行うことを抑制し、システムを用いたサービス提供者・システムにより提供されるサービスの利用者または第三者などに人的被害や経済損失・機会損失などの悪影響を及ぼすリスクを低減する品質特性である。昨今、一層重要視されている「セーフティ＆セキュリティ」や「安全・安心」に深く関係する。

具体的な事例として、

- 自動運転のための物体認識における、障害物の見落としや種別の誤認識
- 食品工場ラインの混入物検知における異物の見落とし
- 有価証券の自動取引における、不正な入力（見せ玉など）による許容範囲を超えた発注

などが挙げられる。

リスク回避性については、多数の人命や企業体の存続に影響するレベルから、軽微な利益機会の逸失に留まるレベルまで、7 レベルを設定した（3.1 節）。リスク回避性の特性は、従来のシステムの安全性規格などで扱われてきた性質と密接に関係することから、品質レベルの設定に当たって、これらの規格との併用や親和性を考慮し、主に強い要求については、既存規格（IEC 61508 の SIL など）と対応する 4 レベル（レベル 4〜1）を設定した。一方で、IT サービスやスマートデバイスなどの機械学習の応用では、従来の工業製品でリスクとして捕捉しないような軽微な損害しか想定しないような需要も多く、これらの品質要求は既存安全性規格では「対象外」の単一レベルに対応してしまうことから、実用性の観点からさらに 3 レベルを追加することとした。

一般に機械学習システムにおいて、「どんなときにでも必ず安全な動作をする」といったような厳密な性質の保証は本質的に馴染まず、機械学習要素単体だけではシステム全体に必要の利用時品質を達成できないことも考えられる。実際のシステム構築においては、その実現を機械学習要素に依存せずに、周辺のソフトウェア実装による「安全確保弁」などに依ることも多いと想定される。本ガイドラインでは、従来の安全性の規格同様、システム全体の利用時品質の要求と機械学習要素の外部品質の要求を別個として認識し、システム全体のリスクアセスメントと、システム構成から機械学習要素の外部品質要求レベルを設定する（5.1.1.5 節）こととする。

1.5.2　ＡＩパフォーマンス（有用性）

2 つ目の特性軸として、機械学習提供機能の有用性が重視される分野への応用に着目し、「**AI パフォーマンス**」の特性軸として整理する。リスク回避より AI パフォーマンスが重要視される典型的な応用としては、個別の誤判断による悪影響よりも平均的な成績の高さが要求されるシナリオがあり、一例として小売店の仕入れにおける需要予測や、投資判断の予測などが挙げられる。

もちろん応用によっては、「AI パフォーマンス」と「リスク回避性」の双方が要求されることもありえる。例えば自動運転においては、事故を起こさないリスク回避性が第一優先ではあるが、同時に乗り心地の向上や目的地への到着時刻の最適化も第 2・第 3 の達成目標として考えられる。また、需要予測においても、極端な過剰仕入れなどは運用環境の想定を超えた間接的な悪影響をもたらし、利潤の平均値の大小だけでは評価できない大きな経済損失を引き起こす場合なども考えられる。このような場合は、「AI パフォーマンス」と「リスク回避性」二つの特性軸について、最適な妥協（バランス）ポイントを、要件としても明確にすることが必要となる。

AI パフォーマンスについては、達成すべき具体的な目標値そのものはいわゆる Key Performance Indicator（KPI）として応用毎に異なることから、その目標達成がどの程度強く求められるかという観点から、3 つのレベルを設定した。

1.5.3　公平性

AI にはしばしば、ある種の社会規範性や倫理性が要求されることがある。人文科学的な立場ではなく工学的な観点からは、これらの社会規範性の多くは既存の様々な利用時品質特性と、その特性の設定過程に帰着されることが多い。そのような中で、特に近年の統計的な技術による AI に特有の、従来あまり考慮されていない品質観点として**「公平性」**を抽出することができる。

もちろん従来のソフトウェアでも「公平な判断」をしなければならない場面は多いが、その実現は主に設計段階での検討で完了しており、実装段階ではその設計通りに正しく動作する「信頼性」などの別の品質特性に帰着されていた。しかし機械学習においては、実装過程や学習結果そのものに確率・統計的な振る舞いが含まれるため、事前の検討だけでは公平性の実現を担保しきれず、ソフトウェア要素としての機械学習要素が、直接「公平性」を品質として実現する必要が生じる。

このような視点から、本ガイドラインは 3 つ目の外部品質特性として「公平性」を掲げる。本ガイドラインにおいて「公平性」とは、機械学習利用システムの出力に対して、ユーザ視点から見て望ましくない偏りがないことなど、出力全体の分布に関して一定の統計的性質が要求されることと定義する。公平性の事例としては、例えば健康保険の見積もりや人事採用のスコアリングにおける人種差や性別の取扱いなどに偏りが生じていないことを明示的・暗示的に要求されるケースが想定される。

一般に機械学習利用システムに要求される社会的要求として、「FAST（fairness, accountability, sustainability, transparency）」の 4 要素が指摘されることがあるが、本ガイドラインではそのうちでも特に統計的な性質として直接的に分析可能な公平性にまず着目する。

公平性については、まだその目標の設定方法や実現方法に議論の余地が大きい。現時点では暫定的に、社会的な要求の強さなどから 3 レベルの分類を行った。

なお、どのような属性に対しても一定のリスク回避性や AI パフォーマンスが求められることについては、ここでは公平性とみなさず、リスク回避性や AI パフォーマンスとして整理する。例えば、安全性のための防護装置において性別や体格の違いの影響に関する品質の要求は、「どのような性別・体格であっても安全であるべきとするリスク回避性の要求」と整理し、「性別で安全性の差が出ない公平性の要求」とは整理しない。これは、公平性の技術的実現方法が他の性能指標とのトレードオフになることが十分考えられ、「一方の安全性を下げて公平性を向上する」といった解決が望ましくないためである。

また、本外部品質特性では、「どのような公平性が計測可能で品質管理の対象にできるか」と、「それらの公平性を、機械学習要素にどう実現するか」の 2 点を本ガイドラインの検討対象とし、「特定のシステムにおいて、どのような公平性が社会的正義を実現するか」はガイドラインのスコープ外であらかじめ検討すべき事柄とする。これらの社会規範性や倫理性については、1.6.3 節でさらに議論する。

1.6　その他の「AI 品質」の観点についての取扱い

本ガイドラインでは、上記の通り「リスク回避性」「AI パフォーマンス」「公平性」の 3 つの観点に着目して品質管理を行うこととしたが、一般に「AI の品質」としては他にも様々な観点が議論されている。本節では、これまで整理した 3 つ以外のいくつかの観点について、本ガイドラインとの関係の考え方を整理する。

1.6.1　セキュリティ・プライバシー

機械学習利用システムを含む AI のセキュリティには、「外部環境への作用としてのセキュリティ」と、「情報処理システムとしてのセキュリティ」の 2 つの観点がある。

前者には例えば、敵対的データ（adversarial example）に相当する入力が故意に行われることに依る誤動作の発生などがある。これについては、次節で「耐攻撃性」として詳しく検討する。

後者について、いわゆるセキュリティの 3 要素「秘匿性」「一貫性」「可用性」のうち後者 2 つについては、純粋にソフトウェアとしてのセキュリティとして議論ができる。訓練済み機械学習モデルを用いた推論の計算は、計算時間が入力値によってほとんど変化しない単純な数値計算であり、従来型のソフトウェアとして信頼性が確保されていれば十分と考えられる。これらの要素については、必要に応じて従来規格などにより、例えばISO/IEC 15408の評価保証レベル（EAL）を設定するなどして管理することとし、本ガイドラインでは検討の対象としない。

また「秘匿性」については、一定の条件下で訓練済み機械学習モデルの内容から訓練用データの一部を推測可能であることが学術的な研究成果として指摘されており、個人情報・プライバシーの保護の観点から将来の課題になる可能性がある。ただし、現段階においては、本ガイドラインでは学術研究の進展状況の注視に留め、必要に応じて個人情報保護法の匿名加工データなどの取扱いを参考として、従来のプライバシー保護の枠内で考えることとする。

1.6.2　耐攻撃性

現在の機械学習による人工知能実装が、入力データの意図的改変に脆弱であることは、学会などで近年繰り返し指摘されている。さらに、画像カメラなどへの入力となるシステム外部の状況そのものの改変を通じて、そのような脆弱性への攻撃を行いうる可能性も指摘されている。一方で、その実運用上のリスクなどについてはまだ評価が定まっていない。また、そのような攻撃に対して対策を講じることが、通常運用の機械学習要素の性能に対しては負の影響を与える可能性も指摘されている。

本ガイドラインでは現時点においては、単独の品質要件として耐攻撃性を評価するよりは、前節に掲げた品質特性軸のそれぞれについて、動作時の周辺環境に攻撃者が意図的な改変を加える可能性やその影響度合いをきちんと評価することが現実的であると考え、その

上で、実運用の意図的改変が除外できるかどうかに応じて、品質マネジメントプロセスの中で意図的改変への対策の必要性を検討することとする。対策が必要な場合には、基本的には 1.7.6 節に掲げる「機械学習モデルの安定性」を通じてその効果を評価することにした。

1.6.3　倫理性などの社会的側面

機械学習利用システムに求められる品質として、機械学習要素が行う判断処理結果の倫理性や社会的正当性が含まれることがある。例えば、個人情報に強く関連する分野（人事採用の事前スコアリング、犯罪予測など）においては、性別や人種などについて、法律的・社会的にその差異が判断に影響しないことを保証することが求められることがある。また、人的・経済的被害を与えるリスクが高い分野では、複数のリスクを伴う判断の選択肢の正当性が問われる場面（何らかの人的被害は不可避な状況下における自動運転システムの判断など）が有り得る。

本ガイドラインでは、このような場面で機械学習要素が「どのような判断を行えば」社会的正当性を持つかについては、システムの開発の最初に要求定義の一部として事前に人間によって整理されるべきものとして、その直接の検討の対象としない。その上で、機械学習利用システムが、人間が整理した「正しい」出力に向かって可能な限り高い確からしさで処理結果として導出することを、利用時品質の要求と捉える。

そのような中で、特に従来のソフトウェア工学では直接的な外部品質としてあまり扱われてこなかった「公平性」について、外部品質軸の 1 つとして 1.5.3 節に掲げた。なお、公平性・透明性などの倫理的側面については参考として、国際的な取り組みを 8.2 節にまとめた。

1.6.4　外部環境の複雑性への対応限界

人工知能の一般的な問題として、対応できる環境の複雑性の限界に関する議論が着目されることがある。機械学習利用システムの利用形態には、路上や公共空間などの開放環境でいわゆるサイバーフィジカルシステムの一部として組み込まれる形態も多くあり、全ての環境条件において期待通りの判断を行う事は、極限的な状況も含めると不可能である。この問題は本質的には、機械学習に依らず、開放環境で動作する装置やソフトウェア全般に共通する問題であり、従来の信頼性工学関係の規格においても、例えば IEC 62998[6]などがこの

[6] IEC TS 62998-1:2019: Safety of machinery - Safety-related sensors used for the protection of persons.

問題に多かれ少なかれ対応しようとしていると考えられる。

本ガイドラインでは、全体的な考え方は従前の規格などを踏襲しつつ、複雑な外部環境での品質の実現のために必要なリスク分析やシステム設計の妥当性などについて、従来のソフトウェアとの特性の差異や、それに起因する固有の分析・設計の留意点などに限って、品質管理手法の検討対象とする。

1.7　品質管理の対象とする内部品質特性

本ガイドラインでは、「リスク回避性」「AI パフォーマンス」の 2 つの外部品質の達成に対応して、機械学習要素の内部品質の管理する具体的な特性として、以下の 8 特性を品質管理の特性軸として抽出した。この抽出に至る分析については、9.1 節にその概要を述べる。

「公平性」については、今後更なる分析を行い、必要な品質管理特性軸を今後追加する。9.3 節も参照されたい。

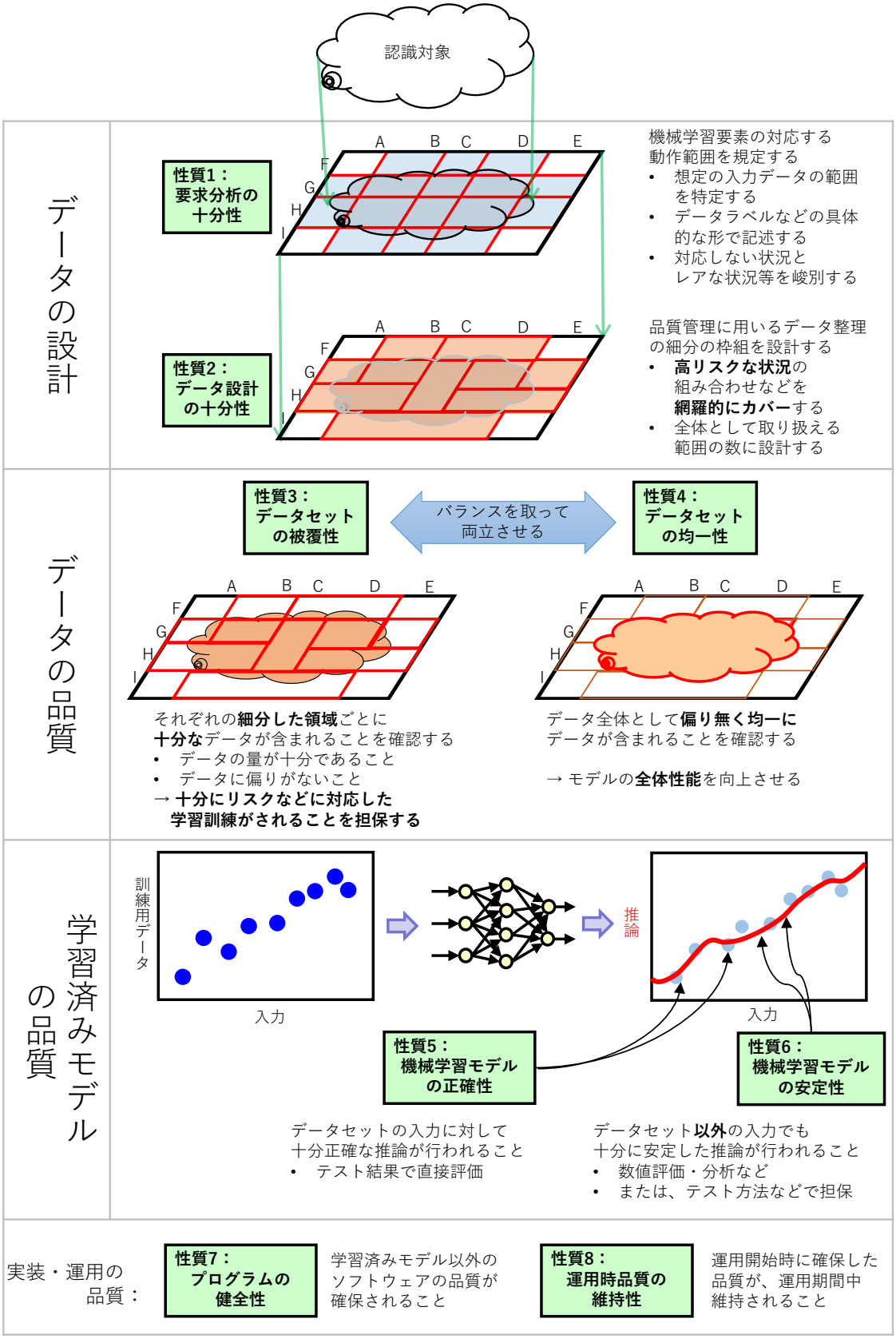

図 3: 着目する内部品質特性

1.7.1　要求分析の十分性

まず初めに、機械学習利用システムの実世界での利用状況に対応して機械学習要素に入力されると想定される運用時の実データの性質について分析が行われ、その分析結果が想定される全ての利用状況を被覆していることを、「要求分析の十分性」とする。

通常、基本的な要求分析は上流設計の一部として完了していると考えられるが、この段階においては、後の段階で訓練用データの整理や必要な検査用データの有無の確認などに用いられるよう、具体的な個々のデータと紐付けられるレベルまで、その分析した要件を具体的な言葉で書き下し、データを元に品質を分析する「軸」を定めることを目標とする。

このような具体的なデータ分析の「軸」の定め方にはいくつかの方法が有り得るが、本ガイドラインの基本的な考え方としては、従来からあるソフトウェアプロダクトライン工学における feature tree の考え方や、それを簡易化した、下の例に挙げるようないくつかの独立した条件の箇条書きとして分類整理し、特定の利用状況をそれらの組み合わせとして把握する手法を想定する。また、この段階でリスク分析・故障モード分析などの手法による要求側からのトップダウン分析と、PoC（Proof of Concept）段階での予備的なデータ分析によるボトムアップな分析を両方行い、外部品質が変化する（特に劣化する）可能性のある状況などを十分に整理する。

（例 1）

屋外で走行する自動運転車両において、交通信号機の現示を画像から認識する機械学習要素においては、遭遇する状況として例えば以下のような書き出しができる。

（この例は実応用上十分に網羅的ではない）

表示の意味： 緑色、黄色、赤色

時間帯： 昼間、夜間

天候条件： 晴れ、曇り、小雨、大雨、雪、霧 など

その他

このように分析すると、例えば「**雨**の**夜間**に、**赤色**の信号が点灯している」状況が、1 つの「状況の組み合わせ」となる。

（例 2）

小売店の販売傾向の予測を行う機械学習要素においては、その利用状況について例えば以下のような書き出しができる。（この例も、網羅的ではない）

曜日： 月曜日・週中日・金曜日・土曜日・休日
天候条件： 晴れ、曇り、小雨、大雨、雪
時間帯： 朝、午前、昼、午後、夕、夜、深夜
季節： 春、夏、秋、冬
近隣のイベント： あり、なし

この例で、例えば「**冬**の**晴天**下の**休日**の**朝**に近隣イベントが**ある**」が１つの組み合わせとなる。

この要求分析の十分性は、従来型のソフトウェアにおけるリスク要因の分析や、ブラックボックステストを行う際にそのリスク要因をきちんと検査対象にするためのテスト要件の分析に対応し、品質を把握し確認する単位を規定する重要な特性となる。一方で、機械学習の実装においては、ある程度以上に些細な特徴は機械学習の訓練段階での学習処理に委ね、システムの実装者が個別の条件判定を細かく指示しないことが最大のメリットでもあり、また場合によっては人による実装よりも高い性能を期待することがありえる。このような観点から、「どの程度の詳細さで要件分析を行うか」の設定は、品質を考慮した機械学習要素の実装戦略を考える上で極めて重要なポイントとなる。

また、このような分析を行うと、実環境で実際には遭遇しない・極めて稀で動作を保証することが現実的でない（対応を要求されない）状況（例えば、夏期の雪）や、逆に収集する訓練用データに出現しないが対応しなければならない稀な状況（例えば、東京地方での春の雪）などが識別できることがある。具体的に、このような２つの状況を峻別することは、特にリスク回避性を要求されるシステムにおいては極めて重要で、システム全体の安全性・堅牢性の設計と直結する。実環境から収集したデータだけでは発見が困難な対応すべき稀な状況（レアケース）の状況を特定し、同時にまた対象システムの設計を進める過程で、実際には発生し得ない状況を定義し、後の段階での検討対象から排除することも、この「要求分析の十分性」を考える段階での重要な作業となる。

具体的な設定の考え方については、6.1 節でさらに詳しく述べる。

1.7.2　データ設計の十分性

要求分析の十分性を前提として、システムが対応すべき様々な状況に対して十分な訓練用データやテスト用データを収集し整理するためのデータ設計の十分な検討を、「データ設計の十分性」として要求する。

極めて単純なシステムでは、前項の要求分析において特定された「状況の組み合わせ」の全てに対して、各々対応するデータが訓練用データセットやテスト用データセットなどに含まれていることをいえばよい。しかし、システムの想定する利用状況が複雑な場合には、取りうる組みあわせの数が膨大になるため、全ての組み合わせについてデータセットで網羅することは現実的でない（例えば、上記の例 2 の簡易な事例だけでも、組み合わせの総数は 700 になる。実際の事例では、1 万のオーダになることも想定される）。その場合には、複数の状況を包括する粗い粒度レベルで広く網羅性を確認しつつ、危害や性能低下の可能性が高い細かな状況の組み合わせにも十分に対応し漏れがないことが必要になる。ソフトウェア工学の分野ではテスト設計などに「網羅性基準」といった考え方があり、応用毎に適切な手段を選択することで、現実的かつ実用上十分な網羅性を達成することを目標とする。

1.7.3　データセットの被覆性

前項で設計した「対応すべき状況の組み合わせ」の各々に対して、状況の抜け漏れがなく、十分な量の学習データが与えられていることを、「データセットの被覆性」とする。

通常のソフトウェア開発においては、ソフトウェアの動作が依存する全ての実世界の特徴の子細については、要求分析から実装までの少なくともいずれかの段階で把握され、最終的にプログラム内の条件分岐や計算式などとして反映されることになるが、機械学習要素の構築においては、ある程度より子細な状況は、訓練用データセットの「特徴量」や「正解ラベル」などとして明示的に把握されず、訓練用データセット内に暗黙的に含まれるのみであり、機械学習の訓練段階を通じて最終的な動作に反映されることになる。要求分析やデータ設計において特定された状況やケースについて、データの不足による学習不足や、偏ったデータによる特定の状況への学習漏れが起こらないことを保証するのが、本特性軸を設定する目的である。

（例 1）

前記の交通信号機の画像認識のケースでは、各都道府県の設置形態や信号機の塗色、設置高や距離、前後の道路の形態などが偏り無くデータとして含まれ、例えば特定の市の狭い範囲のデータのみで訓練していないことがこの特性に相当する。

（例 2）

街中の小動物から「猫」を認識する画像認識 AI を構築する際に、猫の品種や体長などを個別の特徴として認識しないこととした際に、その製品が利用が想定される環境において十分に多様な「猫」や「犬など他の小動物」の画像がデータとして用意されること、例えば三毛猫など特定の品種だけが学習対象になっていないことが、この特性に当たる。

1.7.4　データセットの均一性

上記の「被覆性」と対となる概念として、想定する入力データ全体に対するデータの均一性がある。データセット内の各状況や各ケースが、入力されるデータ全体におけるそれらの発生頻度に応じて抽出されているとき、均一であるとする（図 4)。この均一性と、先述の被覆性の間のバランスが評価の対象となる。機械学習の技術は一般には、入力環境に対して均一に抽出したサンプルを訓練用データセットとして用いれば予測精度は高くなるとされる。しかし、実際の応用やそこで求められる品質特性によっては、前項の状況に対する被覆性が重要視される場合もある。被覆性と本項の全体に対する均一性のどちらを優先するか、またどのように両立させるかは考慮する必要がある。

一般論として、リスク回避性が強く求められるケースでは、正しい判断で回避すべき高リスクな状況に対して十分な訓練用データがあることが求められるであろうが、特にそのような状況が稀に発生する場合、その稀なケースに対する十分なデータ量を確保しつつ、他の全ての状況について均一性を保ちながら訓練しようとすると、必要なデータ量が膨大になる可能性がある。このような場合、特に稀で高リスクな状況を重点的に訓練することは十分に考えられる。

一方で、全体的な性能（AI パフォーマンス）が求められる場合には、稀なケースを実際の発現確率以上に重点的に訓練することで、他のケースにおける推論結果の確からしさが劣化し、全体としての平均性能を悪化させる可能性もある。このような場合には、前節の状況ごとの被覆性基準は適切でないといえる。

また、公平性が強く求められる場合には、「どのような公平性」が求められているかに依存して、データを選別あるいは追加するといった人為的な処理を行ってケース間で人工的に同等な学習をさせるべきか、抽出された訓練用データの分布に即して無作為に学習をさせるべきかが変わってくる可能性がある。

このように、前 1.7.3 節と本節の 2 つの観点は、両立することも相反することもあり、妥当なレベルで両立させるための適切な訓練用データの調整が求められる可能性がある。ま

た、訓練の段階と品質検査の段階でも、求められる性質が違うことも有り得る。

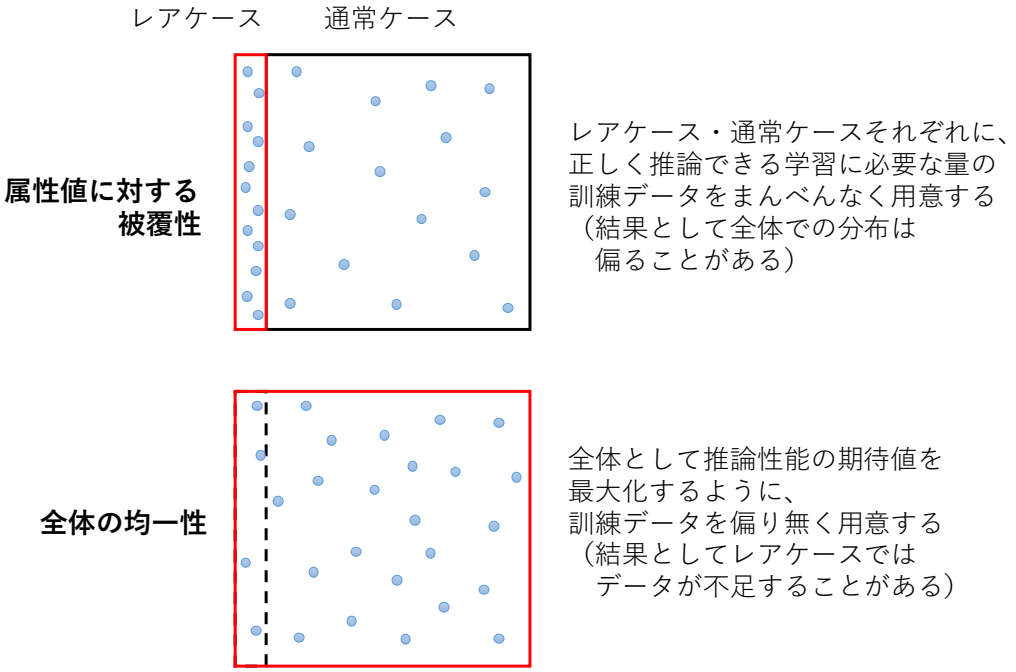

図 4: 被覆性と均一性の関係

（例 1）

例えば完全自動運転車における信号画像認識を東京で年中運用する場合には、年に 1～2 日程度と想定される、（東京としては）視界の極めて悪い雪の場合の学習画像が十分に含まれている必要がある可能性がある。これに対して、他の天候の画像を 360 倍程度用意することが過大な場合、あるいはそのような学習では雪の場合が例外として十分に学習されないと想定される場合には、雪の画像の含まれる割合を実際の雪天の発生確率より多くする場合が有り得る。

この場合、「データセットの被覆性」を「データセットの均一性」より優先する必要が考えられる。

（例 2）

一方で、同じ東京でも小売店の売り上げ予測においては、このような年 1～2 日程度の雪の事例を強く訓練することで、他の天候における予測性能を悪化させ、平均の利潤を最大化できない可能性もある。この場合、「データセットの均一性」のほうを重視する可能性が考えられる。もちろん、実際にどのようなデータを訓練用データセ

ットとして用意するかは、最終的に双方の内部品質特性のバランスを考えながら実装工程で検討することになる。

1.7.5　機械学習モデルの正確性

学習データセット（訓練用データ、テスト用データ、バリデーション用データからなる）に含まれる具体的な入力データに対して、機械学習要素が期待通りの反応を示すことを、「機械学習モデルの正確性」とする。

訓練段階での収束性などの数値的な振る舞いの他に、訓練などに用いるデータに誤りがないこと（誤ったデータが訓練に問題ないレベルで十分少ないこと）なども、この特性に含む。

通常、学習に用いられる訓練用データセットは、運用時に環境から与えられる入力データのすべてを捉えきれず偏りを含むこともある。そのため、機械学習モデルが訓練用データに対してのみ性能が高く、それ以外のデータに対しては性能が低いという場合（過学習）がありえる。学習データセットのみを用いた評価では、環境からの入力データ一般に対して機械学習要素が意図した動作を行うかどうかを確認できるわけではない。そこで、本項で掲げる正確性と、次項において取り上げるモデルの安定性はその両立をはかることが重要となる。

1.7.6　機械学習モデルの安定性

学習データセットに含まれない入力データに対して、機械学習要素が学習データセット内のそれに近いデータに対する反応と十分に類似した反応を示すことを、「機械学習モデルの安定性」と称する。低い汎化能力や敵対的データによる予測不可能な振る舞いを排除することにより、機械学習要素の振る舞いの予測可能性を高める。

1.7.7　プログラムの健全性

機械学習の訓練段階に用いる訓練用プログラムや、実行時に使われる予測・推論プログラムが、与えられたデータや訓練済み機械学習モデルなどに対してソフトウェアプログラムとして正しく動作することを、「プログラムの健全性」とする。アルゴリズムとしての正しさの他、メモリリソース制約や時間制約の充足、ソフトウェアセキュリティなど一般的なソフトウェアとしての品質要求がここに包含される。

1.7.8　運用時品質の維持性

運用開始時点で充足されていた内部品質が、運用期間中を通じて維持されることを「運用時品質の維持性」と称する。システム外部の環境変化に十分に追従できることと、その追従のための訓練済み機械学習モデルなどの変更が品質の不用意な劣化を引き起こさないことの 2 点を含む。

具体的な実現方法は運用の形態、特に追加学習・再訓練の行われ方に強く依存する。これについては、6.8.1 節で述べる。

1.8　開発プロセスについての考え方

1.8.1　反復訓練による開発と品質管理ライフサイクルの関係

機械学習を利用したシステムにおいては、いわゆる従来の V 字型プロセスの開発のみでなく、Proof of Concept（PoC）と呼ばれる予備的実験段階の導入やアジャイル開発など、より柔軟で適応的な開発プロセスの適用が広く行われている。また、運用段階で得られる実データを元にして、運用中により高い品質の実現や環境変化への追従を行う継続的開発も、機械学習では効果的と考えられる。

このような背景から本ガイドラインでは、実際にソフトウェアを構築する上流〜下流工程だけではなく、システムの仕様策定の段階から、実際の運用段階までを含む一貫した工程を、開発・運用一体型のシステムライフサイクルプロセスとして位置づける。特に、システム設計前の要件分析段階における品質目標の把握と、運用中の動作監視や追加データの取得・追加学習などの工程を、システム開発の重要な一工程として位置づけ、品質管理の取り組み対象の一部として扱う。

その上で、ライフサイクル全体の構成については、品質検査（テスト）結果により進捗を管理するアジャイル型（反復型）の開発段階と、工程管理に注力するトップダウンの V 字開発の全体工程を複合した、ハイブリッドなプロセスとして全体をモデル化する（図 5）。

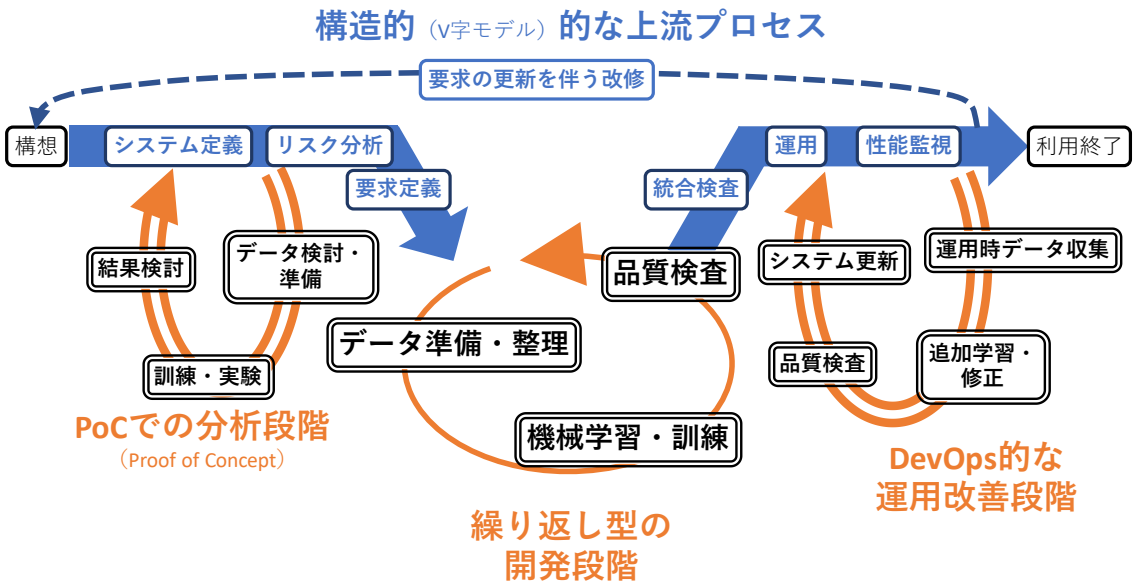

図 5: 混合型機械学習ライフサイクルプロセスの概念図

具体的には、システム全体の品質の必要要件の分析作業については、整理としてはトップダウン型プロセスに類似した流れ式の工程として捉え、実際のシステム開発やデータ整理などに入る前に、独立した工程としてきっちりと分析を行う事を想定する。この段階での手戻りや反復作業については、最終的な結果として開発工程途中から改めてやり直したのとの同等の一貫性を確保することを前提とする。また、運用時のデータ追加取得やモニタリングなどの工程についても、DevOps の考え方に基づき、繰り返し型の品質管理プロセスの一部として位置づけ、必要な場合には RAMS 規格などでのトップダウン型の運用フェーズの考え方とも対応付けできるようにする。

さらに、AI 開発でしばしば行われるいわゆる PoC 開発の段階については、品質管理上の重要な段階として、要求定義に至るまでの事前分析段階のプロセスの一部として整理する。これは、PoC で得られた知見やデータを活用する際に、改めて品質に関する要件を洗い出して再整理することにして、本格的な開発段階における品質マネジメントと、PoC 段階での自由な探索的開発を両立させる考え方である。もちろん、実際の開発工程においては、PoC 段階から本格的な開発段階での品質マネジメントの考え方を部分的に導入することで、再整理の手間を減らすようなやりかたも考えてよい。

なお、本節で掲げたライフサイクルプロセスはあくまで「モデル」であり、各開発者が実際に行う開発プロセスをここで 1 つに集約するものではない。このモデルは、各々の事情に合わせて工夫される開発プロセスの個々の工程を、本ガイドラインと「対応づけて」理解し、本ガイドラインの各章で書かれる品質管理技術などが、各々の実践する開発工程におい

てどの段階に対応するかを探す手がかりとなることを意図している。

（参考）従来、AI 開発プロセスは、非ウォーターフォール型の反復型プロセスとして整理されることが多い。反復訓練を行ううちにおいては、アルゴリズム実装のコード変更以外に、収集データの追加、選別の変更、パラメータ調整などさまざまな作業を行い、時にはその変更内容をもとに実装方針や仕様を修正し、新しい仕様に従って訓練を行い、精度を向上させることで目標達成を目指すことも広く行われる。一方で品質を要求されるプロダクトにおいて、現実には、開発プロセスモデルとしては循環型を採用していても、運用前の合否判定や、受発注の検収作業などの形で「品質の関門」を設けることが多い。

図 6 に示すモデルは、このような非ウォーターフォール型の循環型プロセスとして理解される AI 開発プロセスをもとに、本ガイドラインの基礎となる図 5 の混合型のプロセスとの対応関係を例示するものである。

具体的には、実装方針（実装仕様と呼ぶことも有り得る）が固まるまでの複数回の開発試行を、品質を検討する PoC 段階の複数回の循環と対応付ける。その上で、最終的な仕様に基づいて訓練・品質検査を行いリリースに至る、運用段階の直前の 1 回ないし数回の試行を、本ガイドラインの混合プロセスにおける「本開発段階」と位置づける。

時には、本開発段階として開発を行った結果、さらに仕様の修正が必要となることも十分有り得る。ウォーターフォール型のプロセスとして捉えれば、実装段階から要求分析段階への手戻りに相当するが、反復的なプロセスとして捉える場合には、「結果的にまだ PoC 段階であった」というだけであり、深刻な手戻りと捉える必要は無い。PoC 開発段階は実装の先読み（Implementation Forecast）の要素があり、この段階で得られた知見は暗黙に本開発段階に反映されるため、ウォーターフォール型として捉えた場合であっても、PoC 開発段階での実装の労力が無駄になることはない。

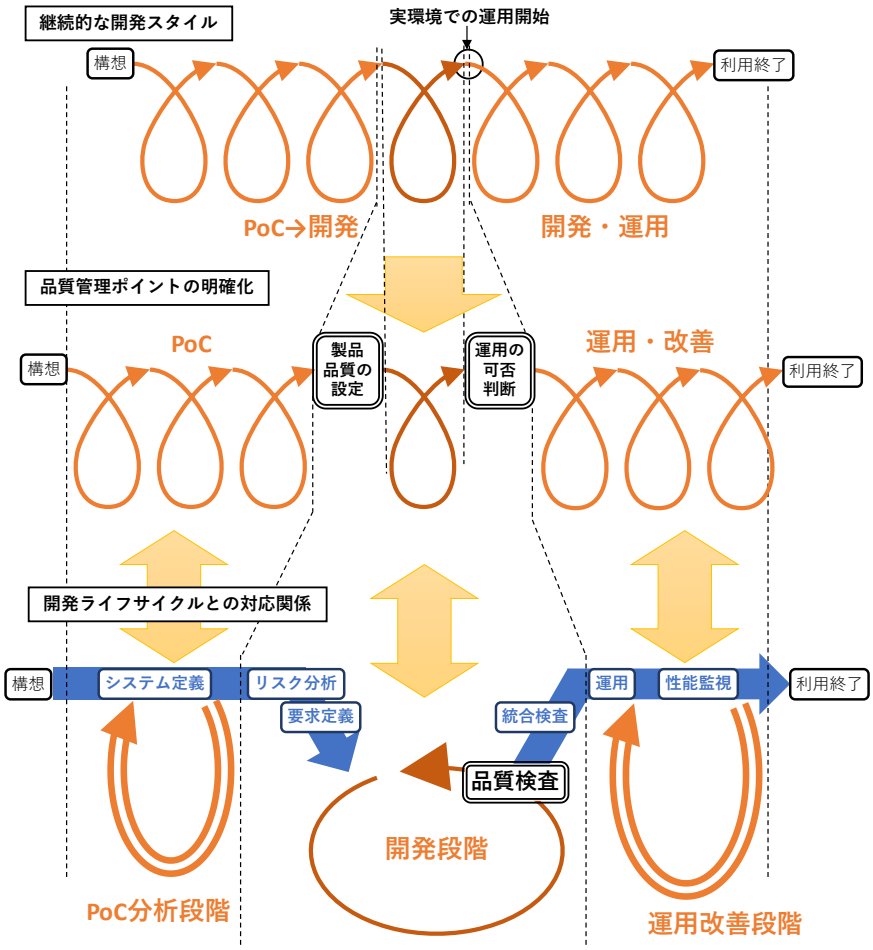

図 6: 循環型のプロセスモデリングと本ガイドラインのモデルの対応（例示）

（参考 2）応用事例として、運用状況の変化や多段の「関門」を有する運用の考え方を 4.1.1 節（49 ページ）に掲載している。

1.8.2　分業による開発と開発プロセスとの関係

実際の AI 開発においては、サービス提供者が自ら企画・開発・運用を一手に行う場合もあれば、訓練段階のみを外注する、システム設計から訓練モデルの構築・システムへの組み込みまでを委託するなど、様々な形で受発注関係などでの業務分担が考えられる。本ガイドラインで定義する品質管理プロセスのうえでは、製品企画を行う開発依頼者を中心とし開発者などを全て含んだ一体の品質管理活動の中で、プロセスの個々の段階の管理作業を作業者間の合意の元に分担する形として整理する。結果として、応用システムの目的に基づく利用時品質や外部品質の設定については、その分担の形態により、開発依頼者側の担当にな

ったり、開発協力者側の担当になったりすることが有り得る[7]。いずれの場合も、最終的な利用者にとって十分な品質を共同で確保することと、そのためのコスト負担を明確に契約などに反映し、運用時まで続く品質管理プロセスの活動が破綻しないように合意しておくことが重要と考えられる。

委託開発や差分開発など、複数の主体が開発に関与する場合の分担モデルの在り方や、その際の留意点については 5.2 節および 5.3 節で述べる。

1.9　他の文書・規範類との関係について

1.9.1　「人間中心の AI 社会原則」

日本においては、内閣府が「人間中心の AI 社会原則」[8] として、主に機械学習利用システムの運用者や開発者と、社会及び一般市民との間のあるべき関係についてまとめている。

同原則との関係では、本ガイドラインは一義的には、このような原則を運用者や開発者が十分に理解した上で、実際に機械学習利用システムやその内部の機械学習要素を開発する際に、その品質を作り込み、開発者の意図しない形で「安全・安心」などを損なわないようにするために、開発者が自ら参照して実践する技術的な事項を整理する、非拘束的なガイダンス類として位置づけられる（図 7）。また、本ガイドラインは、他のガイドライン類や将来の国際規格類などと合わせて、同原則中で言及されている「人間中心の AI 開発原則」を構成する要素とも位置づけられる。

---

[7] 特に、AI 開発にしばしば見受けられる発注要件の形態として、「開発依頼者より与えられたデータ上で一定の性能指標（Accuracy など）を達成すること」を検収要件とした場合には、本ガイドラインの 1.7.1 から 1.7.4 に掲げた「そのデータが製品の目的に見合う学習に十分適する」こと、いわゆる「データ品質」の担保は、開発依頼者が予め完了させておき責任を持つことになることに留意する必要がある。開発依頼者側でデータ品質の検証・担保ができない場合には、その作業の一部を「設計支援作業」などとして明示的に委託することが必要と考えられるほか、実際に訓練を行った際にデータ品質の不足によりシステム構築が成功しない（開発依頼者側まで手戻りが生じる）などの可能性も考慮しておく必要がある。

[8] 内閣府 統合イノベーション戦略推進会議. 人間中心の AI 社会原則. 2019 年 3 月 29 日. https://www8.cao.go.jp/cstp/aigensoku.pdf .

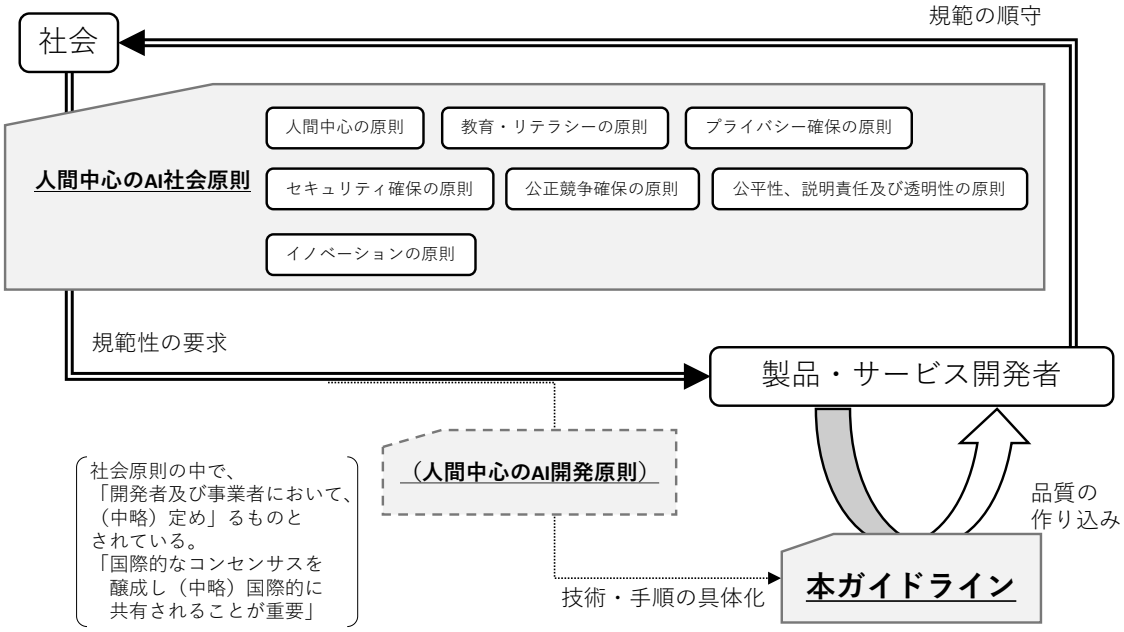

図 7: 「人間中心の AI 社会原則」との関係

1.9.2　人工知能技術に関する海外・国際機関の規範・ガイドライン類

人工知能技術の開発や利用については、上記の原則の他にも近年、その社会性や安全・倫理などに関する規範が、様々な形で文書化されている [9,10]。本ガイドラインは、これらの文書類との関係においては、前節の「人間中心の AI 社会原則」と同様、これらの社会規範を言語化した文書類の下位に位置づけ、これらの規範の一部をシステム開発において実現するための具体的方法論の 1 つを提示するものとして整理する。

1.10　本ガイドラインの構成

本ガイドラインの残りの部分は、以下の構成となっている。

---

[9] Organisation for Economic Co-operation and Development (OECD). *OECD Principles on Artificial Intelligence* (人工知能技術に関する OECD 原則), May 2019. https://www.oecd.org/going-digital/ai/principles/.

[10] The High-Level Expert Group on AI, European Union. *Ethics guidelines for trustworthy AI* （信頼できる AI の倫理ガイドライン). 8 April 2019. https://ec.europa.eu/digital-single-market/en/news/ethics-guidelines-trustworthy-ai .

- 2 章では、本ガイドラインのスコープや既存規格類との関係について改めて整理する。
- 3 章では、1.5 節で述べた利用時品質について、レベル分けの決定を含む詳細について整理する。
- 4 章では、1.8 節で概要を説明した開発プロセスについて、その参照モデルを示す。
- 5 章では、本ガイドラインの具体的な運用・適用方法について述べる。
- 6 章では、1.7 節で述べた内部品質特性をより詳細に整理する。
- 7 章では、6 章の内部品質特性毎に、その留意点や具体的な実現方法の可能性を整理する。
- 8 章では、他のガイドラインなどとの関係性などの参考情報を示す。
- 9 章では、本ガイドラインの検討委員会が 1.7 節（および 6 章）で掲げた内部品質を洗い出すまでの分析過程および考え方を、参考情報として示す。

（参考）

本ガイドラインが想定する（先頭から順番に読む他の）読み方は、以下のようなものである。

- 5 章でプロセスの手順の概略を理解する。
- 3 章を読み、開発プロセスで参照する品質レベルを決定する。
- 6 章の各節（と 10.1 節の表）を参照し、各レベルで必要なチェック項目を洗い出す。
- 必要に応じて 7 章の各節を参照し、上記のチェック項目の具体的な考え方や適用可能な技術の参考とする。
- 各節で参照されている開発段階などの定義は 4 章にまとめられている。

2　基本的事項

2.1　ガイドラインのスコープ

2.1.1　対象とする製品・システム

本ガイドラインが品質マネジメントの対象とする製品・サービスは、産業製品・消費者機器・情報サービスなど情報処理を行うシステム全般のうちで、内包される一部のソフトウェア要素の構築に機械学習技術を用いたもの（以下、本ガイドラインにおいて「機械学習利用システム」という。）とする。

なお、本ガイドラインでは、機械学習要素の作り方として主に「教師あり学習」(supervised learning）による実装を想定している。基本的な考え方はその他の実装方法、例えば教師なし学習（unsupervised learning)、半教師あり学習（semi-supervised learning)、強化学習（reinforcement learning) などにも通用するものもあるが、これらの具体的な扱い方については、今後の改訂時に関連する内容を追加する。

2.1.2　品質マネジメントの対象

本ガイドラインが目標を設定し管理・保証しようとする品質特性は、機械学習利用システムを構成する内部要素である、機械学習技術を用いて構築されたソフトウェア要素（以下、「機械学習要素」という。）が、これを用いるシステムに対して及ぼす影響に対応した外部品質とする。

一方で、本ガイドラインにおける品質マネジメントの直接の対象は、機械学習要素がもつ特性としての内部品質とする。

システム全体の利用時品質は、そのシステムを構成する要素部品の外部品質が総合的に実現するものとして、また各要素の外部品質はその内部の部分要素の品質と、その要素自身の内部的に持つ品質である内部品質特性に依存するものと考える。機械学習要素に内部品質特性として求められる要件を整理し、その要件の達成にいたる工程を管理することで品質管理を行う（図 8)。ただし、システムの構成要素のうち、機械学習要素以外のソフトウェア・ハードウェアなどについては、最低限本ガイドラインの目的に必要な範囲で関

係性などを述べるに留める。

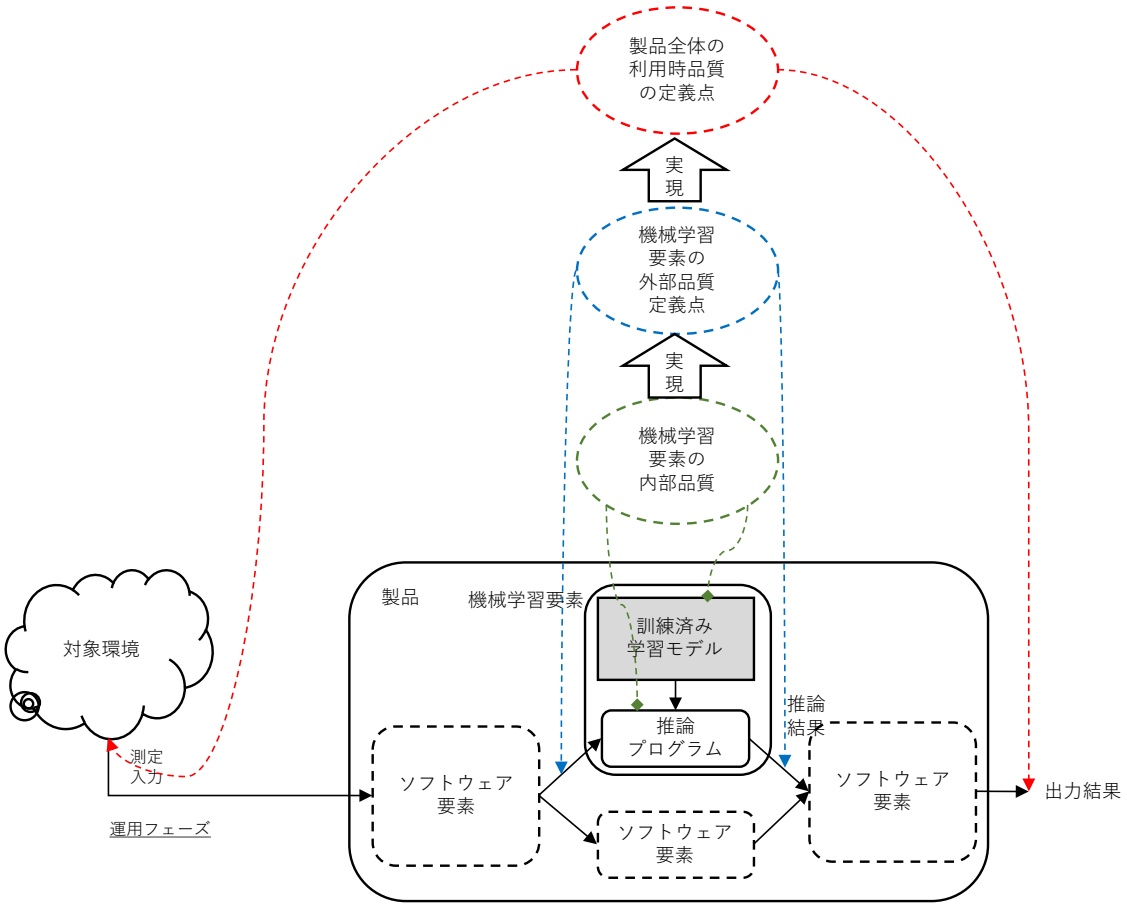

図 8: AI 利用システム全体の品質要求と達成手段の概念

2.1.3　品質マネジメントの範囲

本ガイドラインは「品質マネジメント」を、機械学習利用システムの企画段階から運用までのライフサイクル全体に渡る品質活動プロセスを指す用語として、品質目標の設定、計画、確認、品質保証、管理を包含する意味で用いる。これは、一般のソフトウェアの品質保証や管理よりは広範であるが、例えば ISO 9001 における品質マネジメントに含まれる組織計画や責任・資源などの管理などは含まない。

2.2　システムの品質に関する他の規格などとの関係

本節では、主に情報システムの品質に関する既存の国際規格との関係を整理する。社会性などの上位概念に相当する指針などとの関係については 1.9 節も参考にされたい。

本ガイドラインは、機械学習の様々な応用に対してその品質管理の方法を提示するものであるが、特に安全性を要求されるシステムについては、従来の機能安全性規格の一部（IEC 61508 における第 3・第 4 分冊など）を補完・拡張する位置づけとする。

- 機能安全が強く要求されるシステムについては、機能安全規格 IEC 61508-1 または相当する規格を優先して適用する。
- その上で、そのシステム中の機械学習要素について、要求される安全性レベルを担保するために、機械学習固有の安全性確保の課題や手法について整理し、従来のソフトウェアと比較して IEC 61508-3 などの手法を補完・部分的に置換する方法論を提案する。

2.2.1　セキュリティ規格 ISO/IEC 15408

ISO/IEC 15408 などの情報システムのセキュリティ規格と本ガイドラインは独立した関係にあり、必要な場合は同時に適用されるべきである。本ガイドラインが対象とするリスク回避性の実現の為に一貫性や可用性が求められるシステムでは情報セキュリティは必須要件であるが、基本的な対策は機械学習利用システムであっても同規格で対応できると考えられる。

2.2.2　ソフトウェア品質モデル ISO/IEC 25000 シリーズ

ソフトウェアの品質モデルを定義する ISO/IEC 25000 シリーズ、特に ISO/IEC 25010 は部分的に機械学習利用システムに適応しうる。

ただし、特にソフトウェア要素に関する製品品質については、従来ソフトウェアの観点から分析されている点もあり、同規格が整理した品質特性は機械学習要素でも達成されていると考えられるが、本ガイドラインで着目する分析・要素分解とは異なる部分も存在するため、明確な対応関係は無いと考えられる。現在国際的に同規格の機械学習・人工知能向け拡張の提案なども存在するが、今後の標準化なども踏まえて検討を進める。

2.3　用語の定義

2.3.1　機械学習システムの構成に関する用語

1. 機械学習利用システム
   machine learning based systems / systems using machine learning technology
   機械学習技術を応用して実装されたソフトウェアコンポーネント（機械学習要素、2.3.1.2）を、コンポーネントとして内包するシステム。
   （参考）一般的な「機械学習システム」(machine learning system）は、その文脈により機械学習利用システムか、機械学習要素（2.3.1.2）に対応すると考えられる。

2. 機械学習要素
   machine learning component
   機械学習技術を応用して実装されたソフトウェアコンポーネント。訓練済み機械学習モデル（2.3.1.5）の機能をソフトウェアとして実現する。通常、ソフトウェア実装としての予測・推論プログラム（2.3.1.6）と、固定的入力として組み込まれる訓練済み機械学習モデル（2.3.1.5）から構成される。

3. 機械学習アルゴリズム
   machine learning algorithms
   機械学習の予測・推論に用いる計算方法と、その計算方法を訓練により獲得するための計算方法を与えるアルゴリズム。それぞれの計算方法は、予測・推論プログラム(2.3.1.6)と訓練用プログラム（2.3.1.7）に対応する。
   　ニューラルネットワーク(neural networks)、サポートベクターマシン(support vector machines)、決定木（decision trees）など様々なものがあり、機械学習要素に獲得させる知識の種類や目的により適切なものを選択する。

4. ハイパーパラメータ
   hyperparameter
   機械学習の訓練を実行するうえで、設定として訓練用プログラム（2.3.1.7）に入力する設定値。訓練の進捗に応じて、調整が必要な可能性がある。ハイパーパラメータが無い

場合や、あえてハイパーパラメータ調整を省く場合、またハイパーパラメータの調整を自動で行う技法を用いる場合もある。

（参考）機械学習の対象となりうる数学的構造の多様性について、どの範囲を「機械学習アルゴリズム」として訓練の開始前に固定し、どの範囲を「ハイパーパラメータ」として訓練中に設定・調整するかは、ソフトウェアの実装方法や開発者の訓練方針の選択による任意性がある。場合によっては、計算式としての構造そのものもデータとして扱い、ハイパーパラメータの一部として自動的な最適化調整の対象とすることもある。

5. 訓練済み機械学習モデル

trained model / trained machine learning model / knowledge base / trained parameter

訓練の出力として求められる、機械学習の機能的動作を規定する情報。

（参考 1）本ガイドラインにおいて「機械学習モデル」は、訓練済み機械学習モデルか、それを得るための途中段階のデータを指す。

（参考 2）機械学習技術の文脈で単に「パラメータ」と呼ばれるものは、この訓練済み機械学習モデルを、またはその中に含まれる数値データに着目して指すことが多い。

（参考 3）ニューラルネットワークやベイジアンネットワークなどの数学的構造を「グラフモデル（graphical model）」「統計的モデル（statistical model）」と称することがあり、それに対応して「機械学習モデルの選択」「モデル設計」「未訓練のモデル」などの用語を、機械学習アルゴリズムの選択及びそのパラメータ設定を指して用いることがある。

（参考 4）trained model の語は諸文献では、「モデル訓練の出力」として、あるいは具体的なソフトウェアとして実装された訓練済み機械学習モデルとして trained parameter と対比して用いているが、本ガイドラインでは（訓練自体の出力と、事前実装された予測・推論プログラムを明確に区別する観点から）前者の意味として用い、後者は「機械学習要素」として整理する。

（参考 5）機械学習あるいは人工知能に関する文脈が明らかな場合には、単に trained model と称しても混乱が無いものと考えられる。

6. 予測・推論プログラム

prediction/inference software component

運用中に用いられる機械学習要素のうち、機械学習モデルのソフトウェア実装として固定されている部分。訓練済み機械学習モデル (2.3.1.5) を静的 (準静的) な入力とし、

さらに実環境などから取得されたデータを動的な入力として受け取る。

7. 訓練用プログラム

training software component

訓練用データセットを用いて、訓練済み機械学習モデル（2.3.1.5）を生成するためのプログラム。予測・推論プログラム（2.3.1.6）とは同じ機械学習アルゴリズムの異なる側面からの実装として暗黙に対応していて、通常は組として用いられる。

運用中に用いられる機械学習要素には含まれないことが多いが、運用中にオンライン学習を行うような系・強化学習などでは、一部として含まれることもある。

（参考）本節で整理する各用語の相互関係を次図に示す。

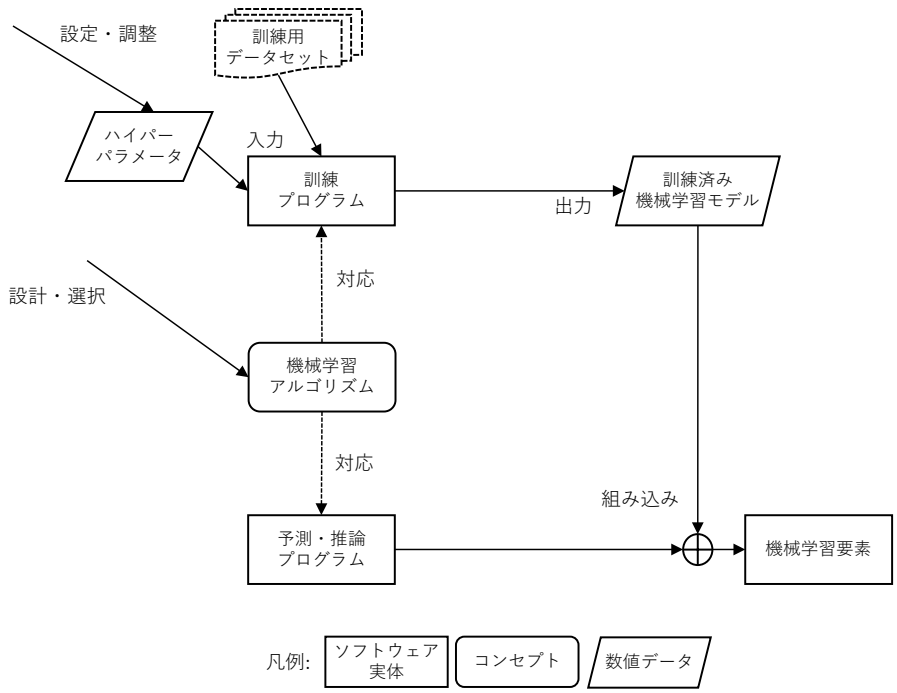

図 9: 機械学習システムの構成に関する用語の相互関係

2.3.2　開発の当事者・ロールに関する用語

1. サービス提供者

service provider

機械学習利用システムを自らの目的で、または顧客向けの販売・サービス提供に用いる主体。
また、ソフトウェア事業者などが、他者が提供・自己使用するサービスを想定したシステムを予め企画・開発しておき、パッケージとして、あるいはカスタマイズを行って販売するような事業形態については、この企画・開発を行う主体をサービス提供者に準じて扱う。

2. 自己開発者
   self-development entity
   機械学習要素の設計・実装を自ら行う製品開発主体。

3. 開発依頼者
   development entruster
   機械学習要素の実装を他者に依頼する製品開発主体。契約の形態（役務・委託など）を問わない。

4. 開発協力者
   development entrustee
   機械学習要素の実装を開発依頼者から依頼されて行う者。

5. 開発当事者・ステークホルダー
   (development) stakeholders
   機械学習利用システムの開発・運用に関わる全ての当事者。製品運用主体・自己開発者・開発依頼者・開発協力者を含む。

2.3.3　品質に関する用語

1. リスク回避性（危害回避性・安全性）
   harm avoidance (risk avoidance / safety)
   機械学習要素の望ましくない判断動作によって、その製品の運用者・利用者または第三者などに人的被害や経済損失・機会損失などの悪影響を及ぼすことを回避する品質特

性である。「リスク回避性」を高めることが、安全分野における「リスク低減」の概念と対応する。

[本ガイドライン 1.5.1 および 3.1 で定義]

2. AI パフォーマンス

AI performance

機械学習要素が、機械学習利用システム及びその利用者が期待する出力を、長期的に平均してより高い精度・確率で出力するという性質。個々の出力の是非よりも、総合的な性能を元に評価する。

[本ガイドライン 1.5.2 および 3.2 で定義]

3. 公平性

fairness

機械学習要素の出力またはその分布が、入力の元となる人などのもつ属性のうちのいくつかについて、その差異に影響されない（影響が十分低く抑えられる）こと。

（本ガイドライン 1.5.3 および 3.3 を参照）

4. 耐攻撃性

attack resistance

機械学習要素（または機械学習利用システム）が、「攻撃者」によって意図的に構築された入力データまたは外部環境に対して、運用者の期待に反する（「攻撃者」の意図に沿う）反応を示さないことが期待できる性質。

5. 倫理性

ethicalness

機械学習利用システムの動作が、人間を中心とする社会のなかで適切なものであること。

6. 堅牢性

robustness

システムが性能のレベルをどのような状況下でも維持できる性質。

2.3.4　開発プロセスに関する用語

1. システムライフサイクルプロセス
   system lifecycle process
   システムの企画立案から運用終了廃棄までの一連の流れを俯瞰した工程管理モデル。

2. アジャイル型開発プロセス
   agile development process
   価値駆動型の俊敏性のある開発アプローチの総称。
   （参考）2001 年のアジャイルマニフェストに端を発したもので、スクラムや XP など様々な手法が活用可能。

3. PoC
   proof of concept
   製品としての実現ではなく、実現したいアイデアや想定する課題解決方法についての実現可能性の検証を目的として行う、予備的な開発活動。

4. システムズエンジニアリング
   systems engineering
   多様な利害関係者のニーズに対応するバランスの取れたシステムソリューションを展開するための複数の分野にまたがるアプローチ。マネジメントプロセスと技術的プロセスの両者を適用し、これらのバランスをとってプロジェクトの成功に影響するリスクを低減する。

5. RAMS
   Specification and demonstration of Reliability, Availability, Maintainability and Safety (RAMS)
   本ガイドラインでは主に、鉄道分野の信頼性管理プロセスについて定めた IEC 62278 (EN 50126) 規格に定められた総合的なシステムライフサイクルプロセスの概念を指すものとして用いる。
   （参考）「RAMS 規格」としては、一般に IEC 62278 自体のことを指すこともある。

2.3.5　利用環境に関する用語

1. 環境
   environment
   本ガイドラインでは 3 つの文脈で用いる: 1）外部環境 / 2）計算機環境 / 3）運用環境

2. 外部環境
   external environment
   機械学習利用システムに対する入力の源となるシステム外部の実環境（physical environment）または情報空間（cyberspace）であって、その環境からシステムへの干渉またはシステムから環境への干渉があるもの。

3. 開放環境
   open environment
   外部環境のうち特に利用者以外の人・自然などのモノの状況がそのシステムの動作に大きく影響を及ぼすようなもの。

4. 環境条件
   environmental condition
   外部環境の持つ状況の違いを識別するような特徴。機械学習品質マネジメントの観点からは、特に危害の状況やリスクなどが変化するような特徴に着目する。

5. 計算機環境
   computing environment
   機械学習要素（推論・予測プログラム）や訓練用プログラムなどを実行するソフトウェア実行環境。状況により、その基盤となるハードウェア環境や、オペレーティングシステム・ミドルウェアなどを含み得る。

6. 運用環境
   operational environment
   計算機環境と、それを用いる（人間の）運用体制などを含む。

7. 実運用環境
   in-operation environment
   機械学習利用システムが実際に実用に供される運用環境。

8. 推論実行環境
   runtime (computing) environment
   実運用環境中で、機械学習要素を含む機械学習利用システムを実際に運用する計算機・クラウド・IoT デバイスなどの計算機環境。

9. 開発環境
   development environment
   機械学習要素を構築・修正する計算機環境で、その環境における修正が実運用に直接影響を及ぼさないもの。また、その計算機環境を含む運用環境。

2.3.6　機械学習構築に用いるデータなどに関係する用語

1. 訓練用データセット
   training dataset
   反復訓練フェーズにおいて、機械学習モデルの訓練に用いるデータの集合。

2. 訓練用データ
   training data (instance)
   訓練用データセットに含まれるデータ。
   （参考）一般的には、「データ」は単数のインスタンスの意味でも、複数のインスタンス（インスタンスの集合）の意味でもどちらでも用いられる。本ガイドラインでは、この多義性を避けるため、「データ」は前者の意味にのみ用い、後者の意味には「データセット」を用いる。すなわち、「データセット」は、集合としての分布や総量などを議論対象とできるような、1 つの塊として扱う場合に用いる。「データ」は集合として捉える前の個々の、あるいは散在的なデータ点を表す場合に用いる。これらの間の関係は、「データセットに加える」「データセットに含まれる」など、集合と要素の関係の用語で記述する。

3. バリデーション用データセット
   validation dataset
   反復訓練フェーズにおいて、機械学習モデルの収束性などを評価するために用いるデータの集合。

4. テスト用データセット
   test dataset
   品質確認・検証フェーズにおいて、構築された機械学習モデルが所用の品質・性能を満たすことを確認する手段（の 1 つ）としてのテストに入力として用いるデータの集合。

5. 敵対的データ
   adversarial example(s)
   機械学習要素に入力すると想定・直感と異なる推論結果が出力されるよう、意図的に構成されたデータ。

6. 過学習
   overfitting
   訓練済み機械学習モデルが訓練データに過剰に適合し、訓練データ以外の入力データに対し望ましい出力を返せなくなること。

7. 属性
   attribute
   ML 要件分析において、環境条件の特徴を分析・分類する項目立ての各要素。

8. 属性値
   value
   ML 要件分析において分類した各属性に含まれる具体的な環境の特徴の種別。

9. 正解ラベル
   labels
   教師あり学習において、分類問題に用いられるデータに付随し、属する正しいクラスを

示す識別子。

2.3.7　その他の用語

1. リグレッション
   (software) regression
   （参考）ソフトウェア工学分野で、ソフトウェアを改良した際、以前は期待通りに動作していた入力に対して、期待通りの動作をしなくなること。
   （注）機械学習・統計分析の分野では回帰分析の意味で用いられる。混乱を招くため、基本的には本ガイドラインでは使用しない。

2. SIL
   safety integrity level (SIL)
   IEC 61508 で定められる、製品の機能安全性の達成に関するレベル分け。
   [定義元: IEC 61508-1]

3. KPI
   key performance indicator
   機械学習要素の出力が機械学習利用システムを通じて達成する機能要件の目標達成度合いを、数値化して定量的に示す指標。

4. 継続的学習
   continuous learning
   運用期間中にデータの収集と追加的な機械学習の訓練を行い、随時・適時に訓練済み機械学習モデルの更新を行う運用の形態。
   （参考）下記の「オンライン学習」だけでなく、「オフラインでの追加学習」などを含む概念である。

5. オンライン学習
   online learning
   実運用環境において、開発環境での検証プロセスを経ることなく継続的学習とその結

果の反映が行われるシステム実装の形態。

（参考）オンライン学習を行わず、開発環境で追加学習を行い、検証プロセスを行ってから、ファームウェアアップデートなどの形態でモデルパラメータの更新を行う形態を、本ガイドライン中では「オフラインでの追加学習」と称している。

3　機械学習利用システムの外部品質特性レベルの設定

本ガイドラインでは 1.5 節で述べたとおり、機械学習利用システムに内包される機械学習要素に対し 3 つの外部品質の特性軸を定義し、それぞれ下記の基準により品質レベルを設定する。開発における具体的な設定手順については、5.1.2 節で規定する。

3.1　リスク回避性

機械学習利用システムのリスク回避性のレベルについては、人に対する傷害などの人的リスクと、その他の経済的リスクに細分し、それぞれ表 1 及び表 2 により 7 レベルの「AI 安全性レベル」（AISL 4, AISL 3, AISL 2, AISL 1, AISL 0.2, AISL 0.1, AISL 0）に分類する[11]。

但し、表中の※に該当する項については、先に機能安全性規格 IEC 61508-1（または機能安全性に関する各応用分野毎の個別国際規格）を適用し、当該装置が IEC 61508-1 の SIL4～1（または同等と考えられる個別規格の安全性レベル）に相当する場合、そのレベルを同等の数値の AISL に読み替える。また、※に該当しない場合についても、IEC 61508-1 などを適用する場合には、同規格の SIL を AISL に読み替えて良いものとする。

なお当面の間、AISL 4 および 3 に相当する極めて高いレベルの信頼性が機械学習要素に（直接的に）要求される場合については、現時点ではシステム全体の設計を見直すことなどにより対処するものとし、本ガイドラインでは対策基準を設定せず将来の検討課題とする。

表 1: 人的リスクに対する AI 安全性レベルの推定基準

想定される影響 ＼　回避可能性	回避操作が不可能	人の監視により 回避操作が可能	人による確認・ 手動操作が 常に必要
複数人の同時死亡	※	※	※
単一の人の死傷	※	※	※

[11] AISL の各レベルの表記については、IEC 61508（機能安全）規格の安全性インテグリティレベル SIL 4～SIL 1 と概ね対応させつつ、従来「SIL なし」とされているレベルをさらに 3 段階に分割するために、大小関係を保つために小数を用いて 0.2、0.1、0 というレベル表記を採用した。

障碍の残る傷害	※	AISL 2	AISL 1
重傷	※	AISL 1	AISL 0.2
軽傷	AISL 1	AISL 0.2	AISL 0.1
軽傷(想定される被害者により容易に回避できる場合)	AISL 0.2	AISL 0.2	AISL 0.1
損害の想定無し	AISL 0	AISL 0	AISL 0

表 2: 経済リスクに対する AI 安全性レベルの推定基準

影響 ＼ 回避可能性	回避操作が不可能	人の監視により回避操作が可能	人による確認・手動操作が常に必要
企業体としての存続等に著しい影響	(AISL 4)	(AISL 3)	AISL 2
業務の運営を揺るがす重大な損害	(AISL 3)	AISL 2	AISL 1
無視できない・具体的な損害	AISL 2	AISL 1	AISL 0.2
軽微な利益の逸失にとどまる	AISL 1	AISL 0.2	AISL 0.1
損害の想定無し	AISL 0.1	AISL 0	AISL 0

3.2 AI パフォーマンス

AI パフォーマンスについては、以下の標準とし、サービス提供者が、または開発ステークホルダ間の協議により決定する。

- AIPL 2 (mandatory requirements):
  - 当該製品・サービスが一定の性能指標（正答率・適合率・再現率など）を満たすことが、製品・サービスの運用上の必須または強い前提である場合。
  - 受発注などの契約において、前記の一定の性能指標の充足が受入要件として

明確に記載される場合。

- AIPL 1 (best-effort requirements):
  - 一定の性能指標が製品・サービスの目的として特定されているが、AIPL 2 に該当しない場合。

    特に、リリースまでの日程スケジュールが重視される場合、または品質をモニタリングしながら試験運用を行い漸次性能向上を行う運用が許される場合。
- AIPL 0
  - 性能指標が開発時点で特定されず、性能指標そのものを発見することが開発の目的となる場合など。
  - 所謂 PoC の段階で終了する開発を行う場合。

3.3 公平性

公平性については、以下を基準とする。

- AIFL 2 (mandatory requirements)
  - 法令・規則・社会的なガイドラインなどにより、一定の公平な取扱いを行う事があらかじめ要求・強く想定されている場合。
  - 当該製品・サービスがパーソナルデータを扱い、その出力が、個人の権利などに直接影響を及ぼす場合。

- AIFL 1 (best-effort requirements)
  - 当該製品・サービスが偏りを持たないことについて、明確な要件が特定できる場合。
  - 当該製品・サービスが内包する機械学習要素（ないし AI）の出力が公平であることを説明できないことが、システムの社会受容性などに影響する場合・運用の障害になる場合。

- AIFL 0
  - 当該製品・サービスに対する公平性の要求が存在しない場合。

- 当該製品・サービスが潜在的に不公平・不均一であったとしても、性能とのトレードオフなどの観点から積極的に肯定されうる応用である場合。
  （例えば、機械的損傷の検知のような応用において、特定の損傷の検知率が高くなった場合に、他の損傷に合わせて検知率を下げる必要は無い。）

4　機械学習利用システムの開発プロセス参照モデル

本章では、本ガイドラインが議論の前提として参照する機械学習利用システムの開発プロセス参照モデルについて述べる。

本章の参照モデルは、機械学習利用システムについて特定の開発方法を開発者に強制するものではなく、一般的な機械学習利用システムの内部構成要素や開発工程に参照可能な名称を付加し、それぞれの固有の構成や開発プロセスを本モデルと対比することにより、本ガイドラインが規定する推奨事項などを実際の開発工程と対応づける為のものである。

本ガイドラインは図 5（25 ページ）に示したとおり、機械学習要素（およびそれを含む機械学習利用システムの主要な部分）の開発工程を大きく 3 段階に分析する。

4.1　PoC 試行フェーズ

開発ライフサイクルの工程としての「**PoC 試行フェーズ**」は、システム開発の最初期段階において、システム全体の機能定義などに先行してシステムの達成できる性能の可能性や、品質劣化リスクやその原因となる状況などを特定するための準備段階として整理する。本ガイドラインの品質マネジメント上、PoC 試行フェーズにおける品質マネジメント活動は、本格開発フェーズにおける品質マネジメント活動のための重要な準備作業と整理する。この段階におけるデータサイエンスの観点からの分析活動は、後のリスク分析や要求分析と密接に関連し、その分析結果が後の本格開発フェーズにおいて利用されることも多い。特に、「要求分析の十分性」（6.1 節）を達成するためには、PoC 試行フェーズの取り組みが欠かせない。本ガイドラインでは、これらの活動の妥当性を最終的に本格開発フェーズの各段階で改めて確認することと整理する。これにより、PoC フェーズでは品質マネジメント活動そのものについても試行錯誤を制約無く自由に行うことができることになる。

4.1.1　試験運用を含む PoC フェーズなどの取扱い

また、一般には、単なるデータ収集や分析・試行的な訓練・モデル構築に留まらず、最終製品とは異なる利用状況（例えば有人の監視下）ではあるものの、実運用の環境で試行するようなものもひとくくりに「Proof of Concept」と呼ばれることがある。また、運用の段階

が複数にわかれ、品質要求が異なる運用状況へ連続的に移行していく場合もある。このような複数段階の実運用を伴う開発については本ガイドラインでは、それぞれの段階のシステムの構築段階に対して各々、本ガイドラインにおける「本番開発フェーズ」を含む全体の開発ライフサイクルに準ずるものとし、かつ早期の運用段階の成果すべてを後期の開発における PoC 段階の知見として取り扱う（図 10）。

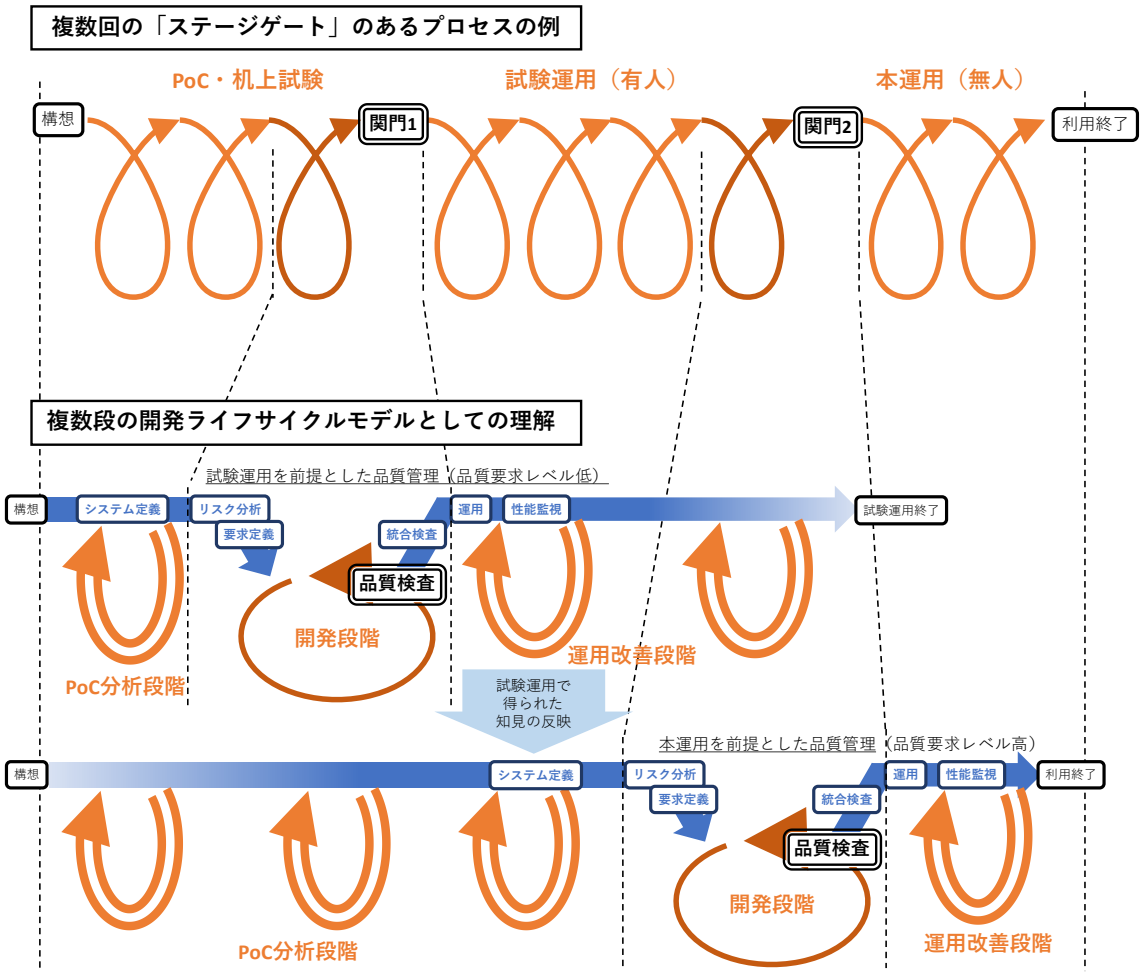

図 10: 複数の運用段階を伴う開発プロセスの取扱い（例示）

4.2　本格開発フェーズ

PoC 試行フェーズにおける取り組みによりシステムの機能要求が確定して以降、実運用に至る直前までのライフサイクル上の工程を、「**本格開発フェーズ**」とする。このフェーズには、従来のシステム開発の超上流工程に相当するシステム定義から、最終的な運用・出荷

前の統合検査までの工程を含む。

本格開発フェーズにおいては、従来のウォーターフォール型の上流開発工程の一部と、機械学習要素に特有な反復型の開発工程を混合したライフサイクルをモデルとして定義する。より具体的には、従来「上流工程」とされてきた、システム全体の機能定義やリスク分析と構成要素の影響分析・分解、機械学習要素を含む各構成要素のシステム要求を決定し、出荷前の最終の統合検査におけるゴールを設定する一連のプロセスは、従来のウォーターフォール型モデルに準じて整理する。また、機械学習要素以外のソフトウェア・ハードウェアの構成要素については、下流工程についても従来のウォーターフォール型開発モデルや、同等の品質を確保できるアジャイル開発プロセス[12]を適宜適用するものと想定する。一方で、機械学習要素についての「下流工程」に当たる具体的な機械学習モデル開発のプロセスは、次節で改めて反復型の開発プロセスモデルを定義する。

4.2.1　機械学習モデル構築フェーズ

本格開発フェーズ中で、機械学習要素が対応すべきリスクや品質要件が特定された後、機械学習モデルを構築しその外部品質を（内部品質の指標を用いて）検査し、合格するまで反復する段階を、**「機械学習モデル構築フェーズ」**とする。

このフェーズをより具体的な工程に分解したモデルを図 11 に示す。このモデルの各工程も、各開発者それぞれ固有の開発プロセスを本モデルと対比することにより、本ガイドラインの記述と対応づける為のものであり、開発プロセスを一義的に制約する目的ではない。

[12] ここでいう「アジャイル開発プロセス」は、テスト主導型開発など、ウォーターフォール型の品質確保プロセスを代替するような品質確保活動を含み、その期待される品質面の効果をきちんと説明可能であるようなプロセスを想定しており、非ウォーターフォール型の開発の全ての形態を容認する趣旨ではない。

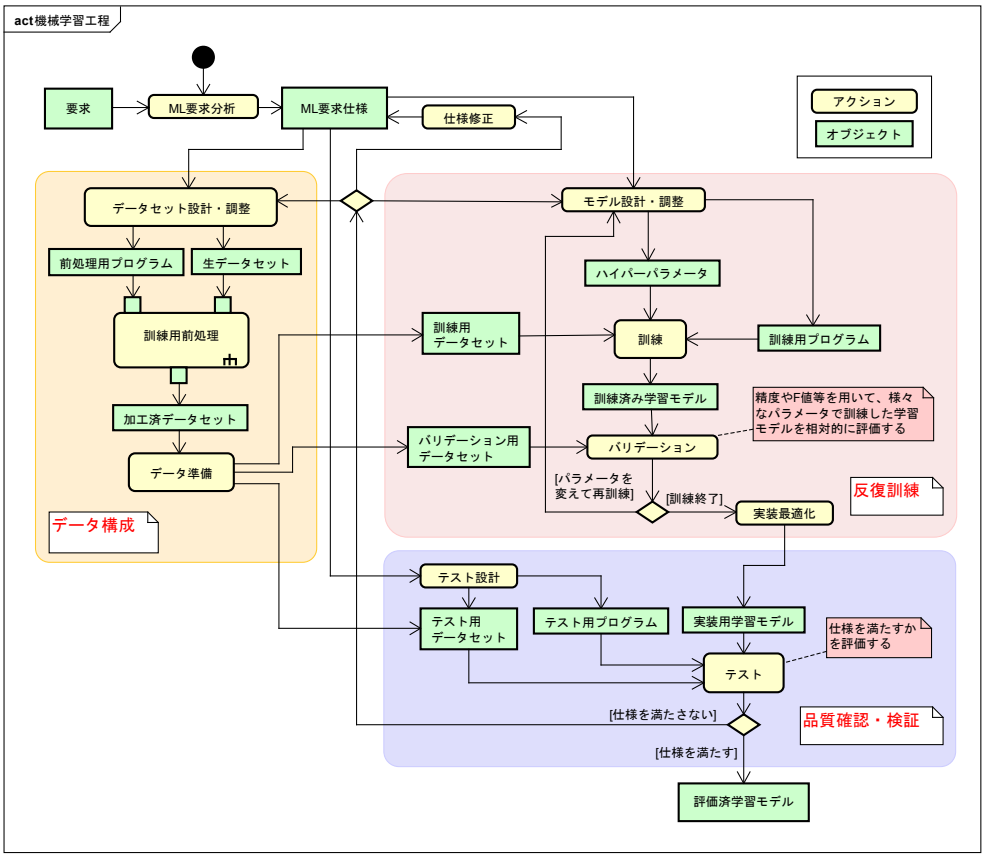

図 11: 機械学習構築の段階におけるプロセスモデル（例示）

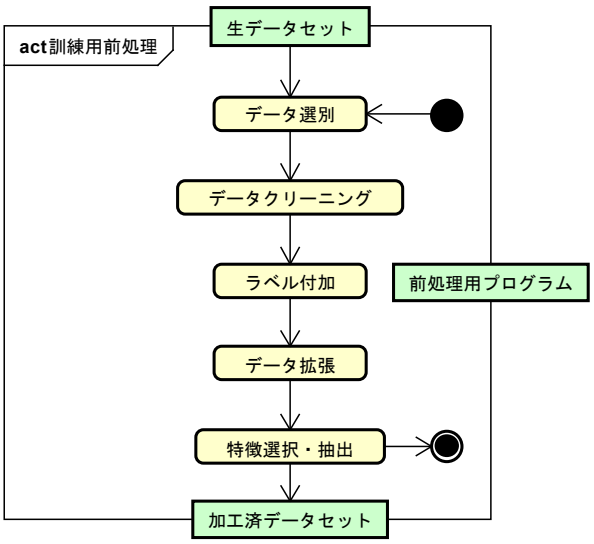

図 12: 訓練用前処理の一例

4.2.1.1　ML 要求分析フェーズ

機械学習要素に対する概念的な品質要求を、特に機械学習要素への入出力データデータの性質の整理として具体化し、機械学習モデル構築フェーズにおける具体的な構築への要求として再整理するフェーズを、「**ML 要求分析フェーズ**」とする。実際の機械学習利用システムの開発においては、しばしばデータの性質や具体的なデータセットそのものに基づいて性能要求（品質要求）が定められることも多いが、このような分析フェーズを明示的に置くことで、データ品質の具体的な管理方法や、他の構成要素との品質管理上の関係などを整理することができる。このフェーズは主に、6.1 節・6.2 節の内部品質に強く関係する。

4.2.1.2　訓練用データ構成フェーズ

訓練用データ構成フェーズでは、ML 要求仕様にしたがって、訓練用、バリデーション用、テスト用のデータセットを生成する。以下、図 11 左上の「訓練用データ構成」の流れにそって説明する。

4.2.1.2.1　データセット設計・調整ステップ

データセット設計・調整ステップでは、ML 要求仕様にしたがって、ケースごとに十分なデータを収集し、訓練に適したデータセットを生成するための設計を行う。このステップのアウトプットは、加工していないデータセットと前処理用プログラムとなる。なお、訓練後に訓練済み機械学習モデルが仕様を満たさない場合は、このステップに戻り、データセットの設計を調整することもある。

4.2.1.2.2　訓練用前処理ステップ

訓練用前処理ステップでは、4.2.1.2.1 節のデータセット設計・調整ステップで生成した生データセットを訓練に適したデータセットに加工する。図 12 はその訓練用前処理の一例である。実際には、図 12 の訓練用前処理の挿入図に掲げた各処理を順不同で行う、並列に行う、繰返し行う場合などもあるが、ここでは一例として、図に掲げた順序にそって説明する。

4.2.1.2.2.1　データ選別

大量の生データセットから、データセットの被覆性、均一性を考慮しながら、訓練と評価（バリデーションとテスト）に用いるデータセットを選別する。被覆性と均一性はトレードオフ関係にあるため、そのバランスは ML 要求仕様にしたがう。

4.2.1.2.2.2　　データクリーニング

選別したデータセットに対して、本来の特徴を訓練できるように、ノイズ除去、データ整形、欠損値補完、外れ値除去などを行う。

4.2.1.2.2.3　　ラベル付加

クリーニングしたデータセットに対して、ML 要求仕様にしたがい、正解（教師用）のラベル付けを行う。

4.2.1.2.2.4　　データ拡張

訓練用のデータ数の確保や過学習の防止など、または、テスト用のデータ数の確保のため、データの変種を生成してデータセットに追加する。例えば、画像データでは、変種を生成するために、回転、縮小、反転、明度変更、背景の置換えなどの操作を行う。

4.2.1.2.2.5　　特徴選択・抽出

モデルの訓練に有用な新たな特徴量を加えるために特徴抽出を行い、不要な特徴量を減らすために特徴選択を行う。例えば、特徴抽出ではフーリエ変換による周波数成分を計算したり、複数の特徴量の主成分を求めたりなどの技法を用いる。有用な特徴量が得られれば、多くの場合その数以上に不要な特徴量を特徴選択により削減できる。

4.2.1.2.3　データ準備ステップ

データ準備ステップでは、生成した加工済データセットを訓練用、バリデーション用、テスト用に分割する。反復訓練中に、訓練用とバリデーション用のデータセットを入れ替えることもある。

4.2.1.3　反復訓練フェーズ

反復訓練フェーズでは、ML 要求仕様にしたがって機械学習モデルを設計し、訓練用データ構成フェーズで作成したデータセットを用いて訓練とバリデーションを繰り返し行う。以下、図 11 右上の「反復訓練」の流れにそって説明する。

4.2.1.3.1　モデル設計・調整ステップ

モデル設計・調整ステップでは、ML 要求仕様にしたがって訓練に必要なハイパーパラメ

ータ（学習モデルの構成、学習アルゴリズム、各種パラメータなどを含む）を設計するとともに、訓練用プログラムを作成する。バリデーションやテストの結果を受けて、ハイパーパラメータを調整することもある。

4.2.1.3.2　訓練ステップ

設計したハイパーパラメータをもとに、訓練用のデータセットを用いて、機械学習モデルを訓練する。

4.2.1.3.3　バリデーションステップ

訓練した機械学習モデルにバリデーション用のデータセットを入力し、学習モデルの評価指標（正解率、適合率、再現率、F 値など）によってモデルを評価する。一般には、複数のハイパーパラメータで複数の機械学習モデルを訓練し、それらを相対的に評価して、その妥当性を判断する。

4.2.1.3.4　後処理ステップ

後処理ステップでは、訓練済み機械学習モデルを実運用環境（推論用サーバやエッジデバイスなど）に合わせるためのモデル変換などを行い、実装用学習モデルを生成する。

具体的に行われる事例としては、

- 実運用環境（計算性能）に適したモデルの変換
  - 計算精度の変更
  - モデルの圧縮・高速化（蒸留、量子化など）
- 実運用環境に適したモデルの最適化（効率的な推論）
  - モデルのコンパイル（並列化、ベクトル化など）

などがある。

本ガイドラインの想定するプロセスモデル図においては、この後処理は反復訓練フェーズの直後に置き、品質確認・検証フェーズは実運用時に近い条件で行われることを想定しているが、実際の開発においては、機械学習要素単体での品質確認・検証フェーズのあとで、システム全体の構築の工程の一部として行われることもしばしばある。この場合、品質確認・検証フェーズで確認した品質が、実運用時に変化する（悪化する）可能性があることに注意し、場合によっては数値的同等性やシステム全体での検査段階などでの再検証が必要となることを想定しておく必要がある。

4.2.1.4　品質確認・検証フェーズ

品質確認・検証フェーズでは、ML 要求仕様にしたがってテストを設計し、実装用学習モデルの評価（テスト）を行う。以下、図 11 右下の「品質確認・検証」の流れにそって説明する。

4.2.1.4.1　テスト設計ステップ

テスト設計ステップでは、ML 要求仕様にしたがって、テストに必要なプログラムの作成やテスト用データの追加を行う。追加するテスト用データとしては、前処理のデータ拡張で行うような操作でデータを生成する場合や、訓練済みの学習モデルが誤推論しやすいデータを生成する場合もある。

4.2.1.4.2　テストステップ

実装用学習モデルに対し、テスト用データセットを用いた通常の指標（正解率、適合率、再現率、F 値など）による正確性の評価の他、データセットの各データの近傍データ（ノイズを付加したデータなど）を用いた安定性の評価なども行う。評価結果として ML 要求仕様を満たさない場合は前のフェーズに戻り、学習モデルの調整、訓練用データセットの調整、ML 要求仕様の修正などを行う。

4.2.2　システム構築・統合検査フェーズ

本段階は機械学習構築の一部ではないが、手順としての順序の関係上ここに記述する。

理想的な開発状況であれば、機械学習要素に対する品質の要求は PoC 試行フェーズを通じて全て事前に整理され、その達成が機械学習構築フェーズまでで全て確認でき、その後のシステム全体の構築や統合テストの段階では問題が起こらないことが望まれるが、実際のシステム開発においては、複雑な実環境の要求分析が完璧でなかったり、他の構成要素と想定外の相互作用を起こしたり、他の構成要素の不具合の影響が波及するなど様々な要因で、いわゆる統合テストの段階で、機械学習要素の品質上の不具合が発見されることはしばしば起こる。また、特に大規模で複雑なシステムでは、事前のデータだけではどうしてもテスト用データとして不足し、統合後の実環境・実システムでの検査が不可欠である場合も多いほか、データ構成フェーズではどうしても事前に用意しきれないレアケースに対する検査

を、統合テストの段階で再現して行うことなども考えられる。

このような場合は、検査が失敗したときには ML 要求分析に対する部分修正などが起こり、機械学習モデル構築フェーズ全体がもう 1 周行われることになる。このような場合に膨大な作業の再発生を避けるためにも、機械学習構築の各段階においては、それぞれのフェーズでの品質管理活動内容をきちんと記録し、修正の影響などをきちんと把握できるようにしておくことで、部分修正を確実に行えることが望ましい。

4.3　品質監視・運用フェーズ

システムが実運用に展開された後に、品質を運用段階で継続的に監視し必要な修正を加えることで性能を維持し続ける為の活動を、「**品質監視・運用フェーズ**」として明確に位置づける。このようなフェーズは、ウォーターフォール型の V 字開発モデルでは狭義の開発工程の外側に置かれるが、例えば RAMS 規格などのシステムライフサイクルの考え方では既に存在するフェーズである。

特に機械学習利用システムにおいては、

- 事前データに基づく定性的な要求分析が実環境を捉え切れていないケース
- 事前データを用意した際の環境から、運用時の環境が変化しているケース

の双方の観点から、品質の運用時管理が必要な場合が多いと考えられる。

機械学習の多様な応用においては、実際に運用時に訓練済み機械学習モデルを更新する手段として様々なパターンが既に存在しており、単一の類型化では扱いきれないことから、本ガイドラインでは主なパターンとして、

① 開発環境で再訓練し、検査後に運用環境に展開するパターン

② 運用環境で自動的に追加学習し、自己適応するパターン

の 2 類型に分類整理する。その上で、6.8 節でそれぞれのパターンの対応方針を個別に整理することとする。

5　本ガイドラインの適用方法

5.1　基本的な適用プロセス

本節では、想定する本ガイドライン適用プロセスを概観する。

本節では機械学習要素の構築に焦点を当てているが、説明上必要な範囲において、従来既に行われているであろう、システム全体の分析などの過程も含んでいる。そのため、開発担当者において、国際規格対応のために具体的な開発プロセスが既に構築されていて、その妥当性を説明できる場合には、その既存プロセスを基に本節で掲げる段階の必要なもののみを追加するなど、改変を行っても良い。

5.1.1　機械学習要素のシステム内での担当機能の特定

機械学習要素がシステム内で果たすべき安全性などの機能が、例えば IEC 61508 などの従来型システム開発プロセスにおいて十分に特定できている場合、本項はスキップして良い。

機械学習システム全体の利用時品質が未特定の場合、以下のプロセスを通じてシステム全体の利用時品質と機械学習要素の外部品質を特定する。

5.1.1.1　安全性機能の事前検討・適用規格の確認

- 機械学習利用システムが一定以上（おおむね SIL1 以上・無視できない人身障害が想定される程度）の安全性機能を必要とするかをあらかじめ検討する。
- 安全性機能が必要な場合は、IEC 61508 または各応用分野の規格から適用すべきものを選択し、そのプロセスに従う。

5.1.1.2　システムの機能要件（目的・目標）の特定

- システムの利用目的・想定される利用環境の範囲、および達成すべき KPI を概要レベルで特定する。

5.1.1.3　システム利用のリスクシナリオ検討

- システムの機能要件が達成されない、またはシステムが利用目的や社会の要求に不適合となる状況の可能性を検討し、その場合に起こる被害その他のデメリットを列挙する。いわゆるリスクアセスメントに対応する。

5.1.1.4　システム全体の利用時品質特性の要求の検討

- 何らかのシステムの品質メトリクスを用いて、システムの利用時に要求される品質を検討する。
  - 他に適切な選択肢が無い場合の手段の一例としては、3 章で掲げる 3 つの外部品質特性軸を利用時品質に流用し、前節で検討したデメリットのいずれに当たるかを判断し、それぞれの深刻度を 3 章各節の基準で判断し品質レベルに対応づける。
  - 3 つの軸ごとに、最大のレベルを特定し、システムの達成すべき利用時品質レベルとする。
- 但し、5.1.1.1 で従前の安全性規格を採用することとした場合には、物的損害に対するリスク回避性レベルは従来規格の該当する品質指標（機能安全性など）による。

5.1.1.5　システムの構成要素の設計・機能要件及び利用時品質特性の達成に寄与する要素の特定

- システムの構成要素の組み合わせを設計し、機械要素や従前のソフトウェアなどで達成される機能と、機械学習要素により実現される機能を分別する。
- また、利用時品質特性の達成がシステムのどの構成要素に依存するかを分析する。

5.1.2　機械学習要素の外部品質達成要求レベルの特定

- 機械学習要素が具体的に利用時品質に寄与すべき外部品質の達成レベルを、以下の手順により特定する。
- リスク回避性のうち物的・人的損害リスクについて、従来の機能安全性規格を適用する際は、従来の規格による分析を優先し、同規格による機械学習要素がソフトウ

ェアとして達成すべき機能安全性レベルを、機械学習要素のリスク回避性の達成要求レベルに読み替える。

- その他のケースについては、以下の通りとする。
  - 基本的には、システム全体に要求される利用時品質の特性レベルを、機械学習要素に要求される外部品質の達成要求レベルとする。
  - 機械学習要素と並列・直列に処理されるソフトウェア要素（図 8）により、機械学習要素の望ましくない出力に対して監視・補正（出力の上書き修正）が行われる場合で、かつ同ソフトウェア要素が従前の手法で十分な品質を確保できると評価できる場合、機械学習要素への品質特性レベルへの要求は、システム全体のレベルから 1 段階軽減させたものとする。
  - システム全体に要求される品質特性レベルの達成に、機械学習要素が全く寄与・干渉しないと認められる場合、機械学習要素には当該品質特性の達成レベルを設定しない（レベル 0 とする）。

5.1.3　機械学習要素の内部品質の要求レベルの特定

- 6 章の各節に掲げる内部品質特性の各項目について、それぞれの特性の要求レベルを、同節内に掲げる対応関係に従い、前項の利用時品質の特性達成要求レベルから導出する。

5.1.4　機械学習要素の内部品質の実現

- 前項の内部品質特性の要求レベルを実現する方法を、8.2 節の各小項目に従って検討する。
- 各項に掲げられた技術・プロセスや同等と考えられる技術により、各特性の実現を担保する。

5.2　（参考）AI 開発の依頼

本節の内容は参考（informative） である。

機械学習要素開発に必要な、前節までで述べたフェーズやプロセスを遂行する作業は、単

一の組織では遂行しきらず、作業依頼（受発注）が発生する場合も多い。本節では、開発依頼者と開発協力者が、AI 特有の側面への対応を含め協力して作業推進するうえで留意すべき点を述べる。なお、知財面での権利帰属問題や損害賠償の分担など「契約」に関しては 8.1.1 節 経産省 AI 契約ガイドラインも参考にされたい。

5.2.1　　探索的アプローチへの対応

機械学習要素開発においては、開発依頼者から開発協力者への依頼開始時に明確な KPI や受け入れ条件の定義が困難な場合も多く、探索的段階型開発アプローチが必要となる。このアプローチは、いわゆるアジャイル開発を、厳密なコスト・品質が要求される製品開発プロセスに適用しようとした際に発生しがちな「品質管理が難しい」「必要費用が分からない」といった問題と類似する為、企業向けアジャイル適用のプラクティスが有益と考えられる。

たとえば、 4 節で述べた PoC フェーズは、 DA(Disciplined Agile)[13] でいう「Inception」フェーズと見做し、Inception で推奨されている成果物（テスト戦略や、基本アーキテクチャ決定など）を参考に、AI の PoC フェーズの目的（➡出口条件）や、次フェーズへ引き渡されるべき成果物を開発依頼者、開発協力者の双方で合意形成し、ステージゲートにて双方にて確認する、といった施策が取り得る。このように PoC ステージゲートの目標設定は大変重要な意味を持つ。　また、特に AI 開発を念頭に置いた場合、下図に示すとおり PoC フェーズの活動を、「要求明確化」に関するものと、「実現可能性確認」（フィージビリティスタディ）に大別し、その二つの視点から、

- どんな成果物が次フェーズに移行するうえで条件とすべきか
- 開発協力者から開発依頼者へ成果物として引き渡されるか

を検討する事で、より抜け漏れ防止、またその後のフェーズの適切な推進へ繋がる。

---

[13] 企業がアジャイル開発を実際に取り入れるためのベストプラクティスをまとめたもの。2019 年 PMI(Project Management Institute）に獲得されより幅広く活用が開始
https://disciplinedagiledelivery.com/

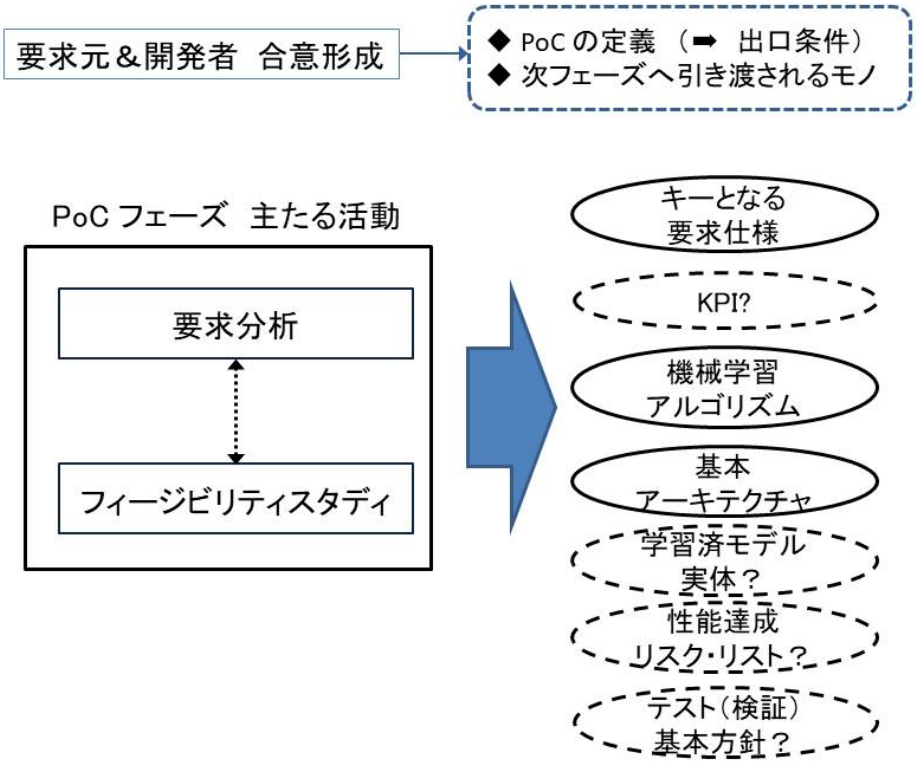

図 13:　PoC フェーズの活動と成果物イメージ

5.2.2　　各工程における作業内容の明確化

5.1 節で述べたプロセスに基づく作業ごとに、開発依頼者、開発協力者双方が担うべき役割を明確化する事が、双方のリスク対策になるとともに、為すべき作業の抜け漏れ防止、結果として機械学習要素の品質確保に繋がる。

たとえば、ゼロから作成するオーダーメイドなＡＩ開発を想定した場合の一例を以下に示す。

表 3　役割定義表のサンプル

各ステップ	開発依頼者	開発協力者
1.安全性適用規格確認	実施	確認
2.システムの機能要件	○○からの要件をもとに、自然文章にて提供 利用時品質特定の定性的な表現を含む	レビューし、目的・目標の明確化を図る。

3.リスクシナリオ	類似製品○○のリスクリストの用意 制約条件の提示	○○手法にて分析し、提示。必要に応じ更新する。
4.利用時品質特性特定	優先度、制約条件を提示する(達成必須な品質特性など)	三つの品質特性について目指すべき品質レベルを提示する。
5.システム構成要素設計	レビュー	遂行 レビューの便宜上○○を提供する
6.機械学習要素品質達成要求レベル特定	レビュー	レベル提案
7.機械学習要素の内部品質要求レベルの特定	レビュー	内部品質管理・保証の方法(テスト手法など)を検討し、目標とするレベルを提示。
8.規格学習要素の内部品質の実現	データセットの提供 テストレポートで確認	７で検討した方法の遂行

プロジェクトにより、行う内容の詳細は異なり、また繰り返し実施される場合一度定めた内容が不変とは限らない。

（例１）
現実解として、「この訓練用データにて学習させること」趣旨での作業依頼となる場合もある。この場合は６節で述べるデータ関連内部品質分析を実施した結果、追加の訓練用データが必要になる可能性は少なくない。　このリスクを認識したうえで、表３の７の内容を考え、また必要な段取り（開発依頼者、開発協力者双方が何をするか）を予め合意することが望まれる。

（例２）
当初、開発依頼者認識に機械学習要素が達成すべき品質への「理解不足」や「曖昧さ」が存在する場合、表３に記載したように、開発依頼者が利用時品質の側面から「優先度、制約」を適切に提示し、開発側が相当するレベルを定義することが出来ない。　こ

の場合は、最初の PoC フェーズは、利用時品質については、開発側が(システム設計の概要を踏まえたうえで、ボトムアップに)初期アイデアを出すことが有り得る。

その後の本格開発フェーズにおいては、開発依頼者から利用時品質への提示を想定した役割定義表に変更をしていく。

PoC フェーズの場合において、表３の各工程ごとの作業内容は変わってくるが、工程が丸ごと不必要になるわけではない。いっぽう本格開発フェーズにおいては、前述したようにステージゲートの目標を適切に設定しクリアしているのであれば、通常ステップ８の中でのループ（繰り返し）で済むことになる。

しかしながら、プロジェクトによってはステージゲートの目標を落とさざるを得ない、すなわち、PoC フェーズと本格開発フェーズの区切りが実態としてあまり無い場合もあり得るが、この場合は品質管理の見通しが立たないリスクをとるのは開発依頼者になり、受け入れるかどうか、また作業分割をどうするかを含め通常開発依頼者の判断に委ねられる。

5.2.3　作業分担の詳細分けにあたって留意すべき点

前節で述べた開発ライフサイクル中の作業具体化＆取り決めをする際に、同時に留意すべき事項を、IPA/SEC「つながる世界の品質確保に向けた手引き」[14]を参考に AI 視点にて抽出・検討した結果を以下に述べる。

1）説明責任が果たせる体制・仕組み

AI 開発においては品質に関するエビデンスがプロセス視点にならざる得ない部分も多いため、その定義や承認者の明確化が必要となるうえ、さらにはプロセスに内在するリスクも存在する。たとえば、

- 訓練用データの準備について、生データの用意は発注者と定義するだけでは不足であり、前処理（クリーニングなど）は誰がどう行うのか、その目標を達成したことのエビデンスは誰がどう残し確認するのか？

などが例としてはあげられる。いずれの場合も、時間・費用制約がある中で、「双方が納得できる妥協・論理」を、事前に検討し、定めることが重要となる。

[14] https://www.ipa.go.jp/sec/publish/tn18-001.html

2)テスト

モデルの訓練実施までは受注側に完全に任された場合でも、「品質確認・検証フェーズ」においては発注側として内容に踏み込んだ理解が通常求められる。そのためには、エビデンスの定義はもちろん、テストそのものの方法・スコープ、あるいは実効性・効率性の面で何を断念するのか、など可能な限り受発注時に納得感を持てるべく、取り決めが望まれる。この際、受発注両者ともに、「最終的な品質向上」を目的に協力をする事が重要であり、たとえば、提示されたテスト用データセットを訓練に用いるなど「検証クリアのみを重視した行為」は、無論避けねばならない。

3)運用時の品質管理

モデル更新の即時性如何によらず、　4.3「品質監視・運用フェーズ」で述べたとおり、機械学習はその性質上、リリース後も継続的な品質管理は欠かせない。　したがって、システムの機能要件に応じた最適な運用時品質管理方法の設計・実装を作業スコープに含める必要がある。

具体例としては、性能評価用データや、開発時には把握しきれない環境データなど、「取得が必要なデータ」の特定と、取得・保存の仕組みの実現に関する取り決めがあげられる。

5.3　差分開発などにおける留意点

機械学習利用システムに限らず、ソフトウェア要素を利用したシステムやサービスにおいては、既存のソフトウェア部品の再利用が頻繁に行われる。機械学習においては特に、一定のデータを訓練済みのモデルを基に、追加学習を施してカスタマイズするような応用が行われることがある。このような再利用に関する品質の保証には取扱いの難しさがある。

まず基本原則として、従前からの機能安全性の確保においても、本ガイドラインの対象とする機械学習の品質確保においても、ソフトウェア部品を再利用したシステムの品質については、あくまでその新しいシステムが利用される状況（文脈・コンテキスト）において品質を保証する方針が、利用状況の分析から機能要件定義・実装から検査にいたるまで、一貫して整合している必要がある。この一貫性を実現する方法としては、例えば以下のような方法が考えられ、システムの要求する品質レベルや部品の提供された状況などに応じて選択すべきである。

1. 再利用元の機械学習要素を構成したデータセット（旧データセットと呼ぶ）の精査を含め、新しいシステムの文脈において新たに品質管理プロセスを立ち上げ、元の要素とは独立して品質管理を行う。
2. 再利用元の機械学習要素について本ガイドラインと同等な品質管理活動が行われ、新しいシステムの要求以上の品質管理が行われていることと、特に「要求分析の十分性」について、想定する利用状況が旧要素の想定状況の部分集合（同じ場合を含む）であることが明確に確認できる場合には、元の品質管理に関する記録類を参照し、必要に応じて追加学習による品質の低下がないことなどを確認する。

   元の要素についての品質管理が十分に行われ記録されていることが前提となるため、当該部品が当初より品質を意識していない場合には適用が難しい。また、利用状況の分析は暗黙に一定の利用文脈を前提としてしまうことが多いため、包含性の判断には困難がある場合もありえる。

   また、機械学習要素を汎用部品として提供する開発者は、あらかじめこのような再利用を想定して品質管理活動を行っておき、特に前提とした品質レベルを明確にしておくことで、再利用性を向上させることができる。
3. 比較的低い品質要求レベルの応用においては、既存データセットを詳細不明なブラックボックスとした前提で、テストフェーズなどに重点を置いた品質管理活動を検討する。具体的には例えば、
   - 6.1「要求分析の十分性」および 6.2「データ設計の十分性」については、新システムで新たに分析を行い目標を設定する。
   - 6.3「データセットの被覆性」6.4「データセットの均一性」については、新たな要求分析に基づいたテスト用データセットを新たに用意し、テストフェーズにおいてこれらの性質の一定の達成を確認する。
   - 6.5「機械学習モデルの正確性」6.6「機械学習モデルの安定性」については、追加学習を行う場合にはその際の指標を用い、新たな要求分析に基づいて追加された訓練用データセットでバリデーションフェーズ・テストフェーズで評価を行う。追加学習を行わない場合には、新たな要求分析に基づいて、テストフェーズまたはシステム全体の結合テスト以降のフェーズで検証を行う。

   但し、少なくとも現在の 6 章・7 章の記述の想定では、訓練に用いたデータセットの分析を行わずに訓練結果に対するサンプリング的な検査のみを用いて十分な品質管理活動を行うことは困難であると考えている。できるだけ旧データセットなど

に関する詳細な情報を入手し、前項 2. のようなホワイトボックス的なアプローチと併用することが、現時点では現実的と考えられる。

6　品質保証のための要求事項

本章では、「リスク回避性」「AI パフォーマンス」の 2 つの利用時品質の達成に対応して、機械学習要素の内部品質の管理を具体的に行うための特性として、以下の 8 つの内部品質を品質管理の特性軸として設定する。この設定に至る分析については、9.1 節にその概要を述べる。「公平性」については、今後更なる分析を行い、必要な品質管理特性軸を追加する。

6.1　要求分析の十分性

6.1.1　基本的な考え方

1.7.1 節で述べたとおり、機械学習利用システムの実世界での利用状況について、十分に要求分析が行われ、その分析結果が想定される全ての利用状況を被覆していることを、「要求分析の十分性」とする。

要求分析は主にリスク回避性（安全性など）を要求される従来のソフトウェアの構築の超上流段階において重要とされている。本ガイドラインが想定する機械学習実装における要求分析の目的は、

① 主にリスク回避性を要求される応用において、実世界の利用状況の中でリスク対策が必要な状況を十分に特定すること、

② 主に公平性が要求される応用において、不公平であってはならない比較対照集合としての属性を十分に特定すること、

③ AI パフォーマンスを含む全ての性能要求に対して、十分な訓練用データセットやテスト用データセットが実世界を十分に網羅的かつ妥当に抽出したものであることを確認するために必要な、実世界の分析を与えること

の 3 つが主なものと考えられる。

この「要求分析の十分性」は、端的には機械学習利用システムに限らずソフトウェアによる制御を伴う機器やサービスでは必ず要求される性質である。ただし、機械学習利用システムにおいては特に、この段階でシステムの利用状況に関する見落としがあると、データの収集から実際の訓練過程・テスト工程に至るまでほとんどの段階で誤りに気付く機会がなく、

実環境での最終的なシステムテスト段階、または実際の運用段階において初めて発生する誤動作の原因になる可能性が高い。

一方で、全ての利用状況の詳細をその微細な差異に至るまで全て事細かに分析することを原則論として徹底すると、分析結果をそのまま通常のプログラムとして実装することができ、そもそも機械学習技術を用いるメリットが皆無となる。これは裏を返すと、このような網羅的かつ徹底的な分析が事前にできないような応用のシステムであるからこそ、何らかの形で機械学習により知識獲得を行わせたいという需要があることになる。

このような２つの観点から、機械学習利用システムにおける「要求分析の詳細度」の適切な設定は、品質確保と実現性・実装の効率性の両面から極めて重要であると考えられる。これは、「何を人が分析するか」「何を機械学習に獲得させるか」の判断に対応していると考えられる。この 2 つのバランスを適切に維持することが、機械学習品質管理における要求分析の重要なゴールとなる。

6.1.2　具体的な取扱い

本項の内部特性軸のゴールは、

- 機械学習要素が対応すべき要求内容の対象を明らかにすることと、
- 機械学習要素が対応すべき範囲の限定を明示すること

の２点となる。

6.1.2.1　属性（特徴の軸）及び属性値（具体的な特徴）の列挙抽出

まず初めに本ガイドラインでは、具体的な特徴として抽出しうる実世界の入力を、以下の観点で整理する。これ以降、その整理されたそれぞれの観点を「属性」と、その観点に属する具体的な特徴の種類を「属性値」と、それぞれ称することとする。
ここで、「属性」の候補として考えられる特徴としては、以下のようなものが挙げられる。

1. 機械学習要素に機能要件として要求される出力の違いを特徴付ける属性。
   教師あり機械学習においては、「教師ラベル」が異なるもの。
   例えば、数字の文字認識では「0 から 9」までの 10 通りの区別、あるいは異状検知などにおいては、異状の有無などが該当する。
2. 出力が同一であっても、機械学習要素に機能仕様レベルで明示的に対応が要求される

属性。

たとえば、文字認識における「l と ℓ」や「1 と /」などが、「どちらも正しく認識すること」と仕様で定められている場合が相当する。

あるいは、自動運転において回避すべき物体として「歩行者」「自転車」「交差交通の自動車」などが指定されている場合に、それぞれの物体が存在する状況などもこれに当たる。

あるいは、公平性の要求される人事採用スコアリングにおいて、例えば「性別」「年齢層」なども本項に該当する。

3. 機械学習要素に期待される性能について、実運用時の性能劣化リスクが大きく異なる可能性の高い属性。

   たとえば、屋外の物体認識において、「天候」「昼夜の別」「逆光・順光」「移動速度」などが考えられる。

4. 機械学習の特性上、期待される出力が同一であっても、学習結果のモデルが共通性を見いだしづらい・別のものとして内部的に認識する可能性のある要素。

   例えば、屋外の物体認識において、昼間と夜間では物体の異なる特徴を捉える可能性や、交通信号機の縦横の区別などが挙げられる。

   あるいは、手書き数字における「1 と /」のような形態の相違はその筆記者の出身国などにより大きく異なるいくつかの特徴に分類され、異なるタイプを全て認識させるためには、たとえ機能仕様で明示されていない場合であっても、異なる母集団として少なくとも訓練用データセットとテスト用データセットに意図的に含めることが必要となる。

5. 学習用にデータを収集する際に、「均等に集めてくる」方法をプロセスとして定めることが難しい特性。

6. 上記のような差異はないが、人が容易に把握できる対象の差異。例えば、動物の種類の違いや、皮膚の色など。

7. 人が言語で特徴を認識・説明できない対象で、かつ一方で十分に偏りのないサンプル抽出が手続き的に可能であるもの。例えば、数字の 8 のありとあらゆる書き方を事前に分析することは困難であるが、人も普段意識して書き分けていない場合には、十分に大量のサンプルを無作為的に用意すれば、偏りのない標本抽出になっていると期待できる。

8. 差異を機械的に網羅補完することが極めて容易なもの。例えば、画像の並行移動や色の変化、一定の範囲での回転拡大などは、要求仕様が一旦定まれば画像処理により機械的

にサンプルを生成することが可能であり、意図的に訓練用データやテスト用データを合成することができる。

システム全体について分析された要求などからこれらの属性の候補を列挙した上で、それぞれのリスクの高さや、機械学習を利用する応用の特性（PoC 試行の結果を参考にする）に応じて、それぞれの特性を属性として抽出して開発者による管理の対象とするか、抽出せずに「機械学習に任せる」かを検討する。一般論としては、上記の列挙で始めの方、1〜3 に掲げたような特性ほど、属性として認識する必然性が高いと考えられる。また 7〜8 項のような特徴は、最終的には属性とせず、ほぼ確実に機械学習に「任せる」ことが妥当であると考えられる。ただし、機械学習に「任せた」属性については、後に性質 3「データセットの被覆性」においてその属性に対応するデータ種別の多様性を検討することになるので、このことも踏まえた分析が必要である。

6.1.2.2　除外する属性組み合わせの検討

また、本項では同時に、「あり得ない属性値の組み合わせ」についても検討を行う。これは、例えば「（東京地方での）夏の雪」のような、機能要件として排除すべきレアなケースを、例えばごく稀に起こる「冬の積雪」のような、機能として対応すべきレアケースと峻別することになる。このような整理は極めて重要で、この段階で区別をしておかない限り、あるデータ欠損が妥当なものか望ましくないものかを判断するすべがなく、この後の品質管理において「品質管理手法自体の品質」を説明することができなくなる。

具体的には、属性とそれに対応する属性値を列挙した後に、システムへの要求などに基づきあり得ない属性値の組み合わせを書き出すことになる。

6.1.3　品質レベル毎の要求事項

要求分析の十分性については、各外部特性のレベルに応じて、以下の 3 レベルの取扱いを要求事項とする。

外部特性レベルと本内部特性の要求レベルの対応は以下の通りである。

「リスク回避性レベルについて」

AISL 0.1 → Lv 1 以上

AISL 0.2 → Lv 2 以上

AISL 1 → Lv 3

AISL 2～4 → Lv 3 に追加すべき要求について、今後検討する。

「AI パフォーマンスについて」

AIPL 1 → Lv 1 以上

AIPL 2 → Lv 2 以上

「公平性について」

AIFL 1 → Lv 1 以上

AIFL 2 → Lv 2 以上

上記の内部特性の要求レベルごとの要求事項は以下の通りである。

- Lv1
  - 主要な品質低下リスクが発生する原因について検討を行い記録する。
  - その検討結果に基づき、データの設計を行い必要な属性などに反映する。
- Lv2
  - システム全体での利用時品質低下リスクとその影響について、工学的に一定の網羅性をもつ分析を行い、文書として記録する。
  - それぞれのリスクについて対策の要否を分析し、機械学習要素への入力においてそのリスクに対応する特徴となる属性について分析を行う。
  - また、応用に即した機械学習要素の入力をもたらす環境の特徴について、機械学習の容易さなどの分析を行い記録する。
  - これらの分析結果に基づいて属性と属性値のセットの検討を行い、その決定の経緯を記録する。
- Lv3
  - Lv2 に加えて、以下の活動を行う。
  - システムの利用環境の特徴量として捉えるべき要素について、過去の自己・他者の検討結果などの文献調査を行い、必要な集合の抽出に至る検討経緯を記録する。
  - システム全体の利用時品質低下リスクについても、そのシステムの応用分野に即した過去の検討結果などを調査し、取捨選択の経緯も含めて検討経緯を記録

する。

- また、システム全体の利用時品質低下リスクについては、Fault Tree Analysis などの工学的分析を用いた抽出も行い、その結果を記録する。

6.2　データ設計の十分性

6.2.1　基本的な考え方

前項の要求分析の十分性を前提として、システムが対応すべき様々な状況に対して十分な訓練用データやテスト用データを確保するためのデータ設計の十分な検討を、「データ設計の十分性」として要求する。より具体的には、訓練用データセット準備以降、テスト工程までの段階で着目する属性値の組み合わせの数や内容についてこの段階で検討を行う。

理想的な状況として、例えば、前項で十分と考えられる属性の組み合わせに対して、全ての属性値の組み合わせ（属性の直積）に対応する十分なデータがあれば、実世界の全ての状況を網羅していると言うことができる。しかし、現実のシステム開発においては、属性の数が 10 以上になることは十分に考えられ、属性値の組み合わせの数が 1 万～100 万程度になることもよくある。このような場合に適切な「網羅性」を考え「設計する」ことが、本項での品質管理の要点となる。

実際に品質管理を行う上では、2 つの観点に着目することが重要となる。1 つは、誤動作・誤判定などを引き起こす可能性のある属性の組み合わせは、確実に訓練や検査段階で対応しておく必要がある。また同時に、実装する機械学習利用システムが運用時に遭遇しうる全状況に対しても、訓練の品質と成果物の品質の双方の観点からできるだけ網羅する必要がある。

このような問題は、従来のソフトウェア開発において「テスト設計」の課題として取り組まれてきていたものであるが、テストだけでなく実装工程もデータセットを元に行う機械学習においては、同じ課題が実装工程において必要になっていることがユニークな点である。

従来のソフトウェア工学ではこのようなテスト設計の組み合わせ爆発にはいくつかの現実的な解決策が既に示されており、本ガイドラインでは、この既存の知見を機械学習応用に適用することで、現実的かつ十分な訓練や品質検査を行うことを目指している。

6.2.2　具体的な取扱い

まず、認識すべき属性が極めて少なく、全ての属性の属性数の積（前項のあり得ないケースを差し引いたもの）が高々10～20 程度であれば、これらの組み合わせをそのまま「ケース」とよび、後の段階でテスト用データセットや訓練用データセットに全てのケースが含まれることを確認することにする。

一方で、これらの属性の積をそのまま用いたのでは組み合わせ数が多すぎる場合や、特にレアケースにおいて十分なデータを得ることができない場合には、属性値のうちのいくつかを取りだした組み合わせを一定の基準で抽出して「ケース」として設定し、これらの組み合わせを網羅することとする。例えば、1.7.1 節の例 1 に掲げた交通信号機の例においては、例えば「昼間の緑」「昼間の黄」「夜間の赤」のほか、「昼間の雨の状況」「夜間の雨の状況」「雨の中での赤信号」などの組み合わせを抽出し、それぞれをケースと呼び、必ずテスト用データセットにこれらの組み合わせに該当するデータが含まれるようにすることで、完全な属性全ての網羅が不可能な場合でも、例えば「夜間の雨」のような高リスクな事例が全くデータセットに含まれていないことや、「赤信号」を全く学習させていないといった事例を排除して、ある程度の「網羅性」を得ることを目指す。なお、このような「間引いた網羅性」を考える場合には、「昼間の雨の赤信号」のデータは、「昼間の雨」「昼間の赤信号」「雨の赤信号」など、複数のケースに同時に含まれることになる。

さらに、高い品質を要求される応用では、これらの抽出作業に数学的な「網羅性基準」を導入することで、完全性をより具体的に保証することにする。ソフトウェア工学のブラックボックステストにおいては、ランダムサンプリングのサンプル率の他にも直交表やペアワイズテストなどの手法とそれに基づく「網羅性基準」が知られており、応用毎に適切な手段を選択することになる。

6.2.3　品質レベル毎の要求事項

要求分析の十分性については、各外部特性のレベルに応じて、以下の 3 レベルの取扱いを要求事項とする。

外部特性レベルと本内部特性の要求レベルの対応は以下の通りである。

「リスク回避性レベルについて」

AISL 0.1 → Lv 1 以上

AISL 0.2 → Lv 2 以上

AISL 1 → Lv 3

AISL 2〜4 → Lv 3 に追加すべき要求について、今後検討する。

「AI パフォーマンスについて」

AIPL 1 → Lv 1 以上

AIPL 2 → Lv 2 以上

「公平性について」

AIFL 1 → Lv 1 以上

AIFL 2 → Lv 2 以上

上記の内部特性の要求レベルごとの要求事項は以下の通りである。

- Lv1
  - 主要なリスク要因に対応する属性について、それぞれに対応したケースを設定すること。
  - さらに、複合的なリスク要因については、その組み合わせに対応したケースを設定すること。
  - また、特に重要と考えられる環境要因の差異に対する属性を抽出し、大きなリスクの要因との組み合わせに対応するケースを用意すること。
- Lv2
  - Lv1 の要求を全て満たすこと。
  - 特に重要と考えられるリスク要因については、原則として pair-wise coverage の基準を満たすこと。具体的には、「その原因の組み合わせの属性値」と、「その属性値の属する属性以外の全ての属性について、属性に含まれる属性値を 1 つずつ個別に選択したもの」の組み合わせのケースを含むこと。
- Lv3
  - 工学的な検討に基づき、属性の網羅基準を設定し、その網羅基準を満たす属性値の組み合わせの集合をケースとして設定すること。
  - 網羅基準の厳密さ（pair-wise coverage, triple-wise coverage など）は、システムの利用状況やリスクの重大さなどを加味して設定されること。必要な場合には、個別のリスクに応じてリスク毎に基準を個別に設定することも考えられる。

6.3　データセットの被覆性

6.3.1　基本的な考え方

前項で基準を定めて網羅したそれぞれのケースに対して、それぞれのケースに対応する入力の可能性に対して抜け漏れなく、十分な量のデータが与えられていることを、「データセットの被覆性」と称する。

通常のソフトウェアの開発においては、ソフトウェアの動作が依存する全ての実世界の特徴の子細については、いわゆる要求分析フェーズから実装フェーズまでの少なくともいずれかの段階で把握され、最終的にプログラム内の条件分岐や計算式などとして反映されることになるが、機械学習要素の構築においては、6.1 節で述べたとおり、ある程度以上の子細な状況の差異は ML 要求分析の属性や訓練時の「正解ラベル」などとして把握されずに、学習に用いるデータセットの分布として、機械学習の訓練フェーズを通じて最終的な動作に反映されることになる。このような「属性値」として特定されていない子細の特徴について、不足したデータによる不適当な学習の振る舞いが起こらないことを保証するのが、本特性軸を設定する目的である。

6.3.2　具体的な取扱い

本項の目的は主として「量」と「入力の状況に対して網羅的であること」の 2 点があるが、属性としてその内部の差異が特定されていない特徴がある以上、後者の網羅性は一般的には、データを収集・処理するプロセスにその多くを依存せざるを得ないと考えられる。一方で、6.2 節で網羅性基準を導入している場合には、それぞれのケース毎に、「ケースに含まれない」属性が偏り無く分布していることを検査することは十分に考えられる。

また、量に関しては基本的には前節でのケースの取り方の粒度を適切に取ることが重要となるが、例えば発生頻度の低いレアケースでは特定のケースのデータがどうしても十分に得られない場合も考えられる。その場合には、例えば特定のデータの欠落したケースに対しては「データセットの被覆性」を部分的に諦め、より緩い網羅性基準でカバーをした上で、システム全体のテストなどにおいてその対応状況を評価するなどの対応が必要になるなど、開発プロセス全体での対応となる場合も考えられる。

6.3.3　品質レベル毎の要求事項

外部特性レベルと本内部特性の要求レベルの対応は以下の通りである。

「リスク回避性レベルについて」

AISL 0.1 → Lv 1 以上

AISL 0.2 → Lv 2 以上

AISL 1 → Lv 3

AISL 2～4 → Lv 3 に追加すべき要求について、今後検討する。

「AI パフォーマンスについて」

AIPL 1 → Lv 1 以上

AIPL 2 → Lv 2 以上

「公平性について」

AIFL 1 → Lv 2 以上

AIFL 2 → Lv 3 以上

（公平性を損なう要因で重要なデータの偏りに対応するためには、
この品質特性が重要となることを考慮している。）

上記の内部特性の要求レベルごとの要求事項は以下の通りである。

- Lv1
  - テスト用データセットの取得源や方法を検討し、応用の状況に対して偏りがないことを期待できるようにすること。
  - 各ケース毎に、元データから偏りのないサンプル抽出などを行い、偏りがないことを期待できるようにすること。
  - これらの偏りを入れないために行った活動について、記録を行うこと。
  - 分析した各ケースについて訓練用データおよびテスト用データが十分に存在することを、訓練フェーズやバリデーションフェーズなどで確認すること。
  - ケースに対して訓練用データが十分に取得できない場合には、網羅基準を見直して緩めた上で、当初の基準に照らして個別にシステム結合テストなどで確認すべきことを記録しておくこと。
- Lv2

- Lv1 に加えて、以下の取り組みなどを行うこと。
- 各属性値または各ケース毎に、およその出現確率の想定を把握すること。
- 取得できたデータがその分布から外れていないことを確認すること。
- 各ケース毎に、中に含まれるデータの被覆性について、取得方法以外の何らかの積極的な確認を行うこと。
- 例えば、各ケース毎に、そのケースに含まれない属性がある場合、その属性に関する分布を抽出して、著しい偏りがないことを確認すること。

・ Lv3

- Lv2 に加え、各ケース毎に、中に含まれるデータの被覆性について、一定の指標を得ること。
- 例えば、特徴量抽出などの技法を用いて、ケース組み合わせに含まれる属性値以外のデータ間相関がないことなどを確認すること。
- あるいは、各ケース毎の、ケースに含まれない属性の分布について、あらかじめ想定される分布を検討し、相違について分析を行い記録すること。

6.4　データセットの均一性

6.4.1　基本的な考え方

一方で、上記の「ケース毎の被覆性」の評価だけでは、データセット全体が入力データの表現する環境全体に対する良いサンプリングになっているとは必ずしも言うことができない。ケース毎の発生確率が大きく異なる場合に、ケース毎にサンプルを準備しただけでは、全体としては大きな偏りを持ったデータセットが生成され、特に AI パフォーマンスの観点での性能を大きく損なう可能性がある。一方で、とりわけ発生頻度の少ないレアケースに対する性能が要求される場合には、入力全体にわたって偏りない均等なデータを現実的な量で用意することと、レアケースに対して十分な量のデータを用意することは一般に両立しない。例えば、百万分の一の頻度で発生する事象に 100 件の訓練用データが必要な場合、不偏な全データの件数が一億件となることは一般に許容できないであろう。このような観点から、本項の均一性は、前項の被覆性と場合によっては相反して適切な妥協が必要となることが考えられる。

一般論として、リスク回避性が強く求められるケースでは、正しい判断で回避すべきリス

クのある属性値の組み合わせに対して十分な訓練用データがあることが求められるであろうが、特にそのようなリスクが稀に発生する場合、その稀なケースにおける十分な「データ量」で他の全てのケースを訓練しようとすると、必要なデータ量が膨大になる可能性がある。このような場合、特に稀なリスクケースを「重点的」に訓練することは十分に考えられる。

一方で、全体的な性能（AI パフォーマンス）が求められる場合には、稀なケースを実際の発現確率以上に重点的に訓練することで、却って他のケースにおける推論精度が劣化し、全体としての平均性能を悪化させる可能性もある。このような場合には、前節の詳細なケースに対する被覆性を深く追求することは、必ずしも適切で無い可能性がある。

また、公平性が強く求められる場合には、「どのような公平性」が求められているかに依存して、ケース間で人工的に同等な学習をさせるべきか、抽出された訓練用データセットの分布に即して無作為に学習をさせるべきかが変わってくる可能性がある。

6.4.2　具体的な取扱い

基本的な考え方そのものは、前 6.3.2 節の被覆性において、「全体」というケースに着目することと変わらない。データセット全体を取得するプロセスに偏りが生じないよう配慮しつつ、個々の属性値の発生頻度などを適宜監視することとなる。

むしろ基本的な考え方で述べたとおり、本項では前節の網羅性とどのように両立させるかの検討やデータ設計が重要になると考えられる。

6.4.3　品質レベル毎の要求事項

外部特性レベルと本内部特性の要求レベルの対応は以下の通りである。

「リスク回避性レベルについて」

AISL 0.1 → Lv S1 以上

AISL 0.2・1 → Lv S2 以上

AISL 2～4 → Lv S2 に追加すべき要求について、今後検討する。

「AI パフォーマンスについて」

AIPL 1 → Lv E1 以上

AIPL 2 → Lv E2 以上

「公平性について」

AIFL 1・2 → Lv E2 以上

上記の内部特性の要求レベルごとの要求事項は以下の通りである。

- Lv E1
  - 前節「データセットの被覆性」Lv 1 に同じ。
- Lv E2
  - 前節「データセットの被覆性」Lv 2 に同じ。但し、想定する出現確率については想定事象の全集合に対して比較する。
- Lv S1
  - 前節 L1 で検討したケース毎のデータ量に関して、リスクに対応するケースにおいて十分なデータ量が存在することを明示的に確認すること。
  - 訓練用データの全体集合の量、レアケースの出現確率を比較して、レアケースのデータが訓練に不足する場合には、レアケースの学習を重点化することを検討すること。但し、特に Lv E2 が要求される場合には、重点化に伴い他のケースの学習が弱化することの、システム全体の品質への影響について必ず検討を行うこと。
- Lv S2
  - Lv S1 に加え、リスク事象毎・ケース毎の出現確率の想定に基づき、各ケースのデータ量を事前に見積もり設計すること。

6.5　機械学習モデルの正確性

6.5.1　基本的な考え方

学習データセットに含まれる入力に対して、機械学習要素が期待に反しない反応を示すことを、「モデルの正確性」と称する。

訓練工程の妥当性や収束性などの数値的な振る舞いの他に、入力データに誤りがないこと（あるいはそのような「外れ値」が訓練に問題ないレベルで十分少ないこと）なども、この特性には含まれる。

6.5.2　具体的な取扱い

基本的には主にバリデーションのフェーズで訓練用データセットに対する収束性をみて判断し、テスト用データセットでその達成を確認することになるが、入力値に確率的な広がりやノイズが含まれるような現実的な場合には、特に出力値が連続的である場合に「100%の再現性達成」がゴールとならない（むしろ過学習を起こしている状況に対応する）場合が有り得る。

そのため、データの量を変化させたり、いわゆる交差検定の手法を用いるなど、Accuracy などの指標の相対的な振る舞いを見て、学習の達成度を評価する必要が一般にある。

具体的な手法は応用に応じてデータサイエンティストが選択し、きちんとその選択の背景を説明できるようにする必要がある。いくつかの具体的に適用可能かもしれない技術については、7.5 節に例示する。

6.5.3　品質レベル毎の要求事項

外部特性レベルと本内部特性の要求レベルの対応は以下の通りである。

「リスク回避性レベルについて」

AISL 0.1 → Lv 1 以上

AISL 0.2 → Lv 2 以上

AISL 1 → Lv 3

AISL 2〜4 → Lv 3 に追加すべき要求について、今後検討する。

「AI パフォーマンスについて」

AIPL 1 → Lv 1 以上

AIPL 2 → Lv 2 以上

「公平性について」

AIFL 1 → Lv 1 以上

AIFL 2 → Lv 2 以上

上記の内部特性の要求レベルごとの要求事項は以下の通りである。

- Lv1

- テスト用データとして必要なデータ量を PoC 仮定や過去の経験から導き出し、「データの被覆性」を満たす抽出プロセスを通じて用意すること。
- テスト用データのラベルなどの誤り及び外れ値の除去方法について検討し実施・記録すること。
- 訓練用データセットについても上記に準じた取扱いとする。ただし、データの分布の取り方については違う方法を採用して良い。
- テスト段階において一定量の誤判断を許容する場合（false negative/false positive で扱いを変える場合を含む）については、その判定基準を合理的に事前に決定し、記録しておくこと。
- 公平性が要求される場合には、予め公平性の比較手段を定めておくこと。対照テストの結果による場合には、その合格基準を予め定めておくこと。

・ Lv2

- Lv1 に加えて以下の対応を取る。
- テスト用・訓練用データのラベルの正当性について、何らかの追加的な確認手段を検討すること。
- 正解率（Accuracy）などのバリデーション段階での合否判定についても、その合理的な判定基準を事前に決定し記録しておくこと。
- 実データでのテストと、可能な範囲でのデータ変形などでの機械的な増量テストを同時に行うこと。
- 可能であれば、入力の影響度分析・ニューロンの発火状況その他の内部的な情報の分析の適用を検討し、可能な範囲で明らかな誤りを手動で排除すること。

・ Lv3

- Lv2 に加え、学習成熟状況の内部確認手段などを事前に検討すること。
- 結合テスト以降のシステム全体での検証計画と機械学習要素のテスト計画の対応を明示すること。
- 特にリスクが大きいケースを中心に、システムレベルでのテスト時の機械学習要素の要件との対応をテスト計画に反映し、その被覆状況を監視・確認すること。

6.6　機械学習モデルの安定性

6.6.1　基本的な考え方

機械学習モデルの安定性とは、データセットに含まれない入力に対して、機械学習要素が近傍の訓練用データセット内の入力に対する反応と十分に類似した反応を示すことである。低い安定性は、未知の入力に対する予測性能の低下（AI パフォーマンスの低下）や、潜在する重大な誤判断によるリスクの増加（リスク回避性の低下）をもたらす。そのため、特に安全性が要求される場合の安定性の評価は重要である。例えば、安定性に関しては次の 2 つの問題がよく知られている。

- 訓練用データセット以外の入力に対して大きく外れた推論をする問題：
  このような問題は、例えば、訓練用データセットに対する過度な適合（過学習）によって生じることがある。
- 意味を変えない程度の入力の微小変化に対して出力が大きく変動する問題：
  これは機械学習特有の問題として知られている。微小変化は自然界のノイズ（例えば、カメラレンズの汚れ）による場合（擾乱）の他、敵対的データと呼ばれる意図的な攻撃の場合（摂動）もある。攻撃には、サイバー空間のデータ改ざんや物理世界の対象物への細工（例えば、道路標識への微小なシールの貼付け）などがある。

6.6.2　具体的な取扱い

安定性は機械学習ライフサイクルの次に示す 3 つのフェーズとの関係が強く、安定性の目標達成に向けて、主にこれらのフェーズで安定性の評価と向上を行う。そのために適用可能な技術については 7.5.2 節で例示する。

- 反復訓練フェーズ：訓練用データセットとバリデーション用データセットの分離、入力の微小変化の出力への影響評価、訓練過程の監視などによって、訓練用のデータセットへの過剰な適合を回避する。
- 品質確認・評価フェーズ：テスト用データセットによる正解率や適合率などの評価の他、入力の微小変化に対する出力の変動の評価を行う。
- 品質監視・運用フェーズ：運用時に誤推論しやすい入力を検知して排除する。

6.6.3　品質レベル毎の要求事項

機械学習モデルの安定性を確認できるように、その安定性を向上するために適用した技術とその評価結果を記録することが求められる。特に、各レベルにおいて記録すべき事項を以下に示す。ここで、近傍データとは、元データに微小変化を加えて生成されるデータのことである。適用可能な具体的な技術については 7.5.2 節で説明する。

外部特性レベルと本内部特性の要求レベルの対応は以下の通りである。

「リスク回避性レベルについて」

AISL 0.1 → Lv 1 以上

AISL 0.2 → Lv 2 以上

AISL 1 → Lv 3

AISL 2～4 → Lv 3 に追加すべき要求について、今後検討する。

「AI パフォーマンスについて」

AIPL 1 → Lv 1 以上

AIPL 2 → Lv 2 以上

「公平性について」

AIFL 1 → Lv 1 以上

AIFL 2 → Lv 2 以上

上記の内部特性の要求レベルごとの要求事項は以下の通りである。

- Lv1：安定性向上のために適用した技術を記録すること
  - Lv1 では、過学習を防止するために広く利用されている交差検証や正則化などの技術の適用が推奨される。
- Lv2：近傍データによる安定性の評価結果を記録すること
  - Lv2 では、データセットの各データの近傍に対する安定性を評価することが求められる。例えば、近傍の敵対的データによる攻撃を防御する技術の適用が推奨される。敵対的データを生成して安定性を評価する技術、敵対的データの攻撃を受けにくくする訓練技術、敵対的データの動的検知技術などがある。これらの技術を適用することは容易ではないが、現在、そのための実用的な開発・

評価するための環境整備が進められている。

- Lv3：近傍データに対する安定性を保証すること
  - レベル 3 では、近傍データに対して一定の安定性をもつことを保証することが求められる。例えば、近傍には敵対的データが存在しないことを保証する技術などがある。これらの技術はまだ研究段階であるが、将来的にはレベル 3 での適用が期待される技術である。

6.7　プログラムの健全性

6.7.1　基本的な考え方

訓練用プログラムや予測・推論プログラムなど、訓練済み機械学習モデルを除く純粋なソフトウェア部分（C 言語などで記述された従来のプログラム部分）が、ソフトウェアプログラムとして仕様通り正しく動作することを、「プログラムの健全性」と称する。アルゴリズムとしての正しさの他、メモリリソース制約や時間制約の充足、ソフトウェアセキュリティなど一般的なソフトウェアとしての品質要求がここに包含される。

組み込み機器などにおいて、MPU などの電子的ハードウェアが開発に含まれる場合には、それらの健全性の確保も本項に準ずる。

機械学習の応用においては、学習訓練及び実運用の双方において、いわゆるオープンソースの実装を用いることが非常に多い。世に知られるオープンソース実装には極めて多数の利用者がいてその品質が間接的に評価されているものもある一方で、品質がまだ十分に検証されていないものや、性能向上のためのバージョンアップが頻繁で新たなバグ混入のリスクが高いものなどもあり、最終的には開発者の責任において十分な品質が確保されることが求められる。

また、一般にソフトウェアの正しさの検査は実際の動作環境と同一性を確認できる環境で行うことが大前提であり、動作環境が異なる場面ではその動作環境の差異の影響をきちんと評価する必要がある。しかし、機械学習の応用においては特に、訓練などにもちいる環境（例えばクラウドや GPU 付き計算機環境）と実際の動作を行う環境（例えば組み込み計算機）が異なり、数値的な計算の振る舞い（浮動小数点演算の精度など）ですら変化する場面も多く見受けられるほか、さらにモデルの圧縮など数値的処理そのものを変化させることも稀にある。実際の機械学習実装において、これらの環境変化などが予測精度などに影響

を与えることがあることも既に知られており、慎重な取扱いが必要となる。

6.7.2　具体的な取扱い

純粋なソフトウェアとしての品質確保そのものは、従来の計算機応用などでの品質管理手法を適宜適用する。

オープンソース実装の利用に関しては、その応用の信頼性の重要性や事故リスクの重大性などに鑑み、次に掲げるように適切な品質確保手段を講じることとする。

また、実行時環境が訓練工程の環境と異なる場合（モデル圧縮などを行う場合も含む）には、少なくともテストフェーズにおいては実行時環境と同じ計算（精度やアルゴリズム・ソースコードなど）を再現するソフトウェアを用いて検査することを、本来あるべき姿として本ガイドラインでは推奨する。そのような扱いが困難な場合には、実機でのシステム全体の検査工程において、品質の劣化が許容範囲内であることを追試することが必要と考えられる。

6.7.3　品質レベル毎の要求事項

「リスク回避性レベルについて」

AISL 0.1 → Lv 1 以上

AISL 0.2 → Lv 2 以上

AISL 1 → Lv 3

AISL 2～4 → Lv 3 に追加すべき要求について、今後検討する。

「AI パフォーマンスについて」

AIPL 1 → Lv 1 以上

AIPL 2 → Lv 2 以上

「公平性について」

AIFL 1 → Lv 1 以上

AIFL 2 → Lv 2 以上

- Lv1
  - 利用するソフトウェアについては、信頼できる実績を持つソフトウェアなどを選定し、その選定経緯を記録すること。

- 選定したソフトウェアについて、その欠陥の発見などを運用期間中モニタリングし、必要に応じて修正などの措置をとること。
- 学習からテストフェーズに至るまでの環境と、実用段階で用いる環境の相違について、その影響などをあらかじめ検討しておくこと。

・ Lv2

- 利用するソフトウェアについて、検査・実験などによりその信頼性を自己評価すること。
- 可能な場合には、SIL 1 相当のソフトウェア信頼性を得られたソフトウェアを用いること。
- システムの運用期間中のソフトウェアの健全性の維持に関する保守体制を必ず構築すること。
- バリデーションおよびテストフェーズにおいては、原則として実用段階で用いられる計算環境（浮動小数点精度・モデル規模など）を模倣した環境でバリデーション・テストを行うこと。または、テスト済み学習モデルと実用環境での学習モデルの動作の一致性について、何らかの検証を行うこと。

・ Lv3

- SIL 1（またはシステムの要求する SIL レベル）のソフトウェア品質の確認を必ず行うこと。
- 実用環境の計算環境での学習モデルの振る舞いに基づくテスト（または形式検証など）を必ず行うこと。
- また、そのモデルと実用環境での動作の一致の確認を、結合テスト以降の段階で必ず行うこと。

6.8　運用時品質の維持性

6.8.1　基本的な考え方

運用開始時点で充足されていた内部品質が、運用期間中を通じて維持されることを「品質の維持性」と称する。システム外部の動作環境の変化に十分に追従できることと、その追従のための訓練済み機械学習モデルなどの変更が品質の不用意な劣化を引き起こさないことの 2 点を包含する。

具体的な実現方法は運用の形態、特に追加学習・再訓練の行われ方に強く依存する。本ガイドラインでは、追加学習・再訓練の形態として、次の 2 つのパターンを想定する。

(a) 追加学習後に、サービス提供中の実システム（運用環境）の外（開発環境など）における品質検査を経て、明示的なソフトウェアの更新などを通じて初めて運用環境に追加学習結果が反映される場合。現在想定されている自動運転などのソリューションは概ねこちらのパターンに属する。

(b) 運用中にリアルタイムに訓練済み学習モデルの更新がおこなれ、運用環境で人手を介した検査を介することなく更新処理が完結する場合。ストリームデータ処理などで用いられるオンライン学習や、その他にも開発・運用一体の案件での言語認識や会話応用などでもこのパターンが見られる。

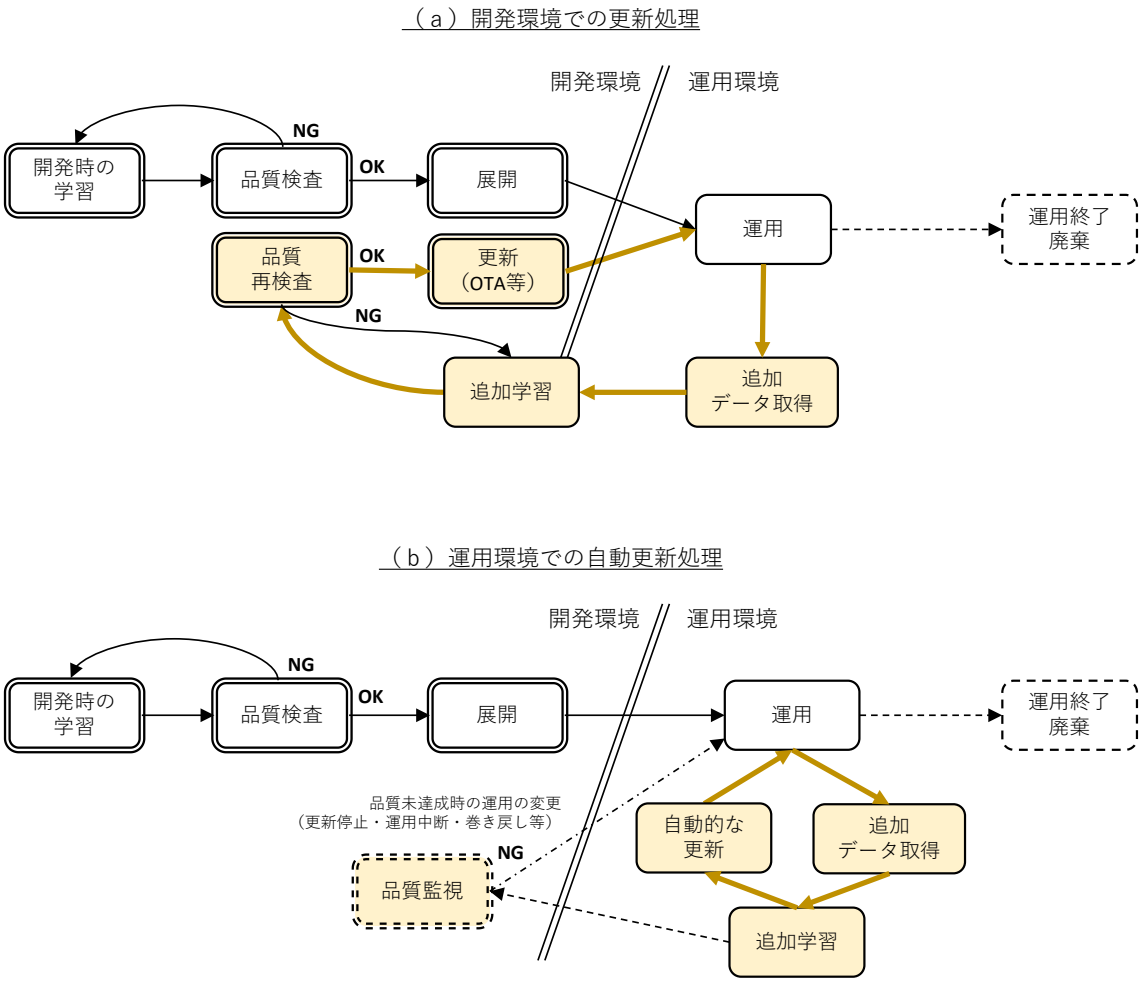

図 14: 機械学習要素の運用中の更新形態

(a) の開発環境などでの更新パターンでは、上図に掲げたように品質検査が必ず更新前に

行われ、基本的には初回開発時のテストフェーズと同様の内容で検査を行えば、一定の品質は確保できると考えられる。一方で（b）の運用環境での更新パターンでは、更新そのものは全自動で行われるため、品質が劣化したモデルがそのまま運用に反映されるリスクが高い。このような場合には、品質のモニタリングの仕組みや、劣化した際の対処までを運用体制にあらかじめ組み込んでおくことが重要になる。

また、運用時品質の維持性においては、全体としての性能の劣化の他にも、特定の入力について、更新前に正答を導いていた機械学習要素が更新後に誤判断を行う[15]問題についても配慮が必要な場合がある。本来は全体として 1.5 節の外部品質が向上ないし維持できていれば必要十分であると考えたいところであるが、実際の産業の現場においては、一旦正しく実装され動作したものが、後日正しく動作しないことについて大きな抵抗感がある。一方で、機械学習システムの特性上追加学習は従来の入力に対して異なる出力値を導くことが必然的であることから、あらかじめ運用方針として取扱いを検討しておく必要があろう。

また、本ガイドラインの直接の対象外ではあるが、追加学習に用いた実環境のデータについての契約やプライバシーなどの法的位置づけも、運用の検討においては配慮しておく必要がある。品質管理の観点からは、追加学習も含め学習訓練に用いたデータは全て保存され、必要に応じて品質検証やモニタリング・上述した過去に正解した事例での性能劣化の確認などのために用いることができることが望ましいが、実際の応用においては、特にプライバシーなどの観点からデータの取扱いに制約が掛かることも想定される。機械学習利用システムの設計時において品質管理の前提としたデータが、運用時に入手できないこととなった場合には、システムの品質担保のロジック全体が崩壊することになりかねない。そのため、入手可能・保存可能なデータの特定は、運用体制の重要な検討要素となる。

6.8.2　具体的な取扱い

システムの更新時に、開発者などによる品質検査が行われるか否かにより、前記した 2 パターンに分けて取り扱う。

（a）更新前に品質検査プロセスを経る更新処理の場合

[15] ソフトウェア工学分野ではしばしば「レグレッション」とも呼ばれる。とりわけ本節で述べている内容はいわゆる「レグレッション・テスト」に相当する。しかし、機械学習の分野では同じ意味の英単語を全く違う文脈で用いるため、カタカナの「レグレッション」は全く違う意味をもつ。そのため、本文書では基本的にはどちらの意味でも用いないこととする。

- 更新頻度の見積もり または 更新の必要性の判断の基準を事前に検討する。
- 運用時に必要となる更新プロセスについて、設計段階で分析と概要の想定を行い、運用開始時までに具体的な手順を設計する。少なくとも、下記のような点に留意が必要と考えられる。
  - 運用環境から取得可能なデータと、更新を行う開発環境などへの収集・展開方法。
  - 追加訓練用データの前処理・フィルタ・ラベル付けの方法。
  - 追加訓練・モデル更新時に利用するデータ範囲。特に、時間経過で環境の変化が想定される場合、どのように古いデータを除去するか。
- 更新時の品質検査の方法、特に更新の可否の判断基準（または意志決定の方法）について、運用開始時までに検討する。
- 個別の正解事例について品質が悪化する場合の運用上の取扱いについて、応用によってはあらかじめ決定しておく必要が有る可能性がある。

(b) 開発環境での品質検査プロセスを経ない更新処理の場合

- 追加学習による顕著な品質劣化が起こる可能性と、発生した場合のシステムへの影響波及の範囲について、あらかじめ想定をする。
- その影響が許容できない可能性がある場合には、追加学習による学習モデルの品質劣化が発生させる、システム全体へのリスクを許容範囲に収める技術的または運用上の仕組みを検討し、運用に組み込む。例えば、以下のような方法が考えられる。
  - 技術的に出力値域を限定する。事前に想定される正常範囲を逸脱しないよう、周囲のシステム構成要素（ソフトウェアなど）で強制する。
  - 運用環境外部からの性能監視により、学習の巻き戻しや、運用の停止・中断などを行う。
- 運用環境・運用システム内部において、更新前の品質監視ができる可能性があると、品質管理に有用と考えられる（図 15)。

（ｂ－２）運用環境での品質監視の可能性

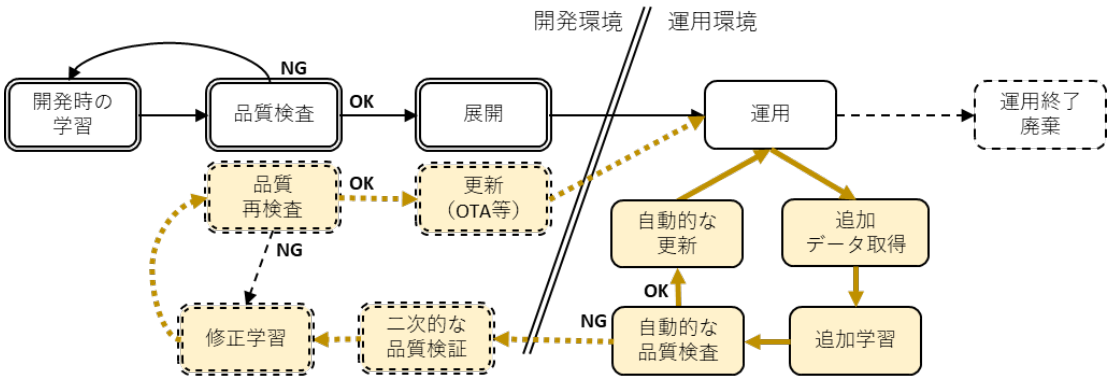

図 15: 運用環境での自動的な品質監視の可能性

6.8.3　品質レベル毎の要求事項

外部特性レベルと本内部特性の要求レベルの対応は以下の通りである。

「リスク回避性レベルについて」

AISL 0.1 → Lv 1 以上

AISL 0.2 → Lv 2 以上

AISL 1 → Lv 3

AISL 2～4 → Lv 3 に追加すべき要求について、今後検討する。

「AI パフォーマンスについて」

AIPL 1 → Lv 1 以上

AIPL 2 → Lv 2 以上

「公平性について」

AIFL 1 → Lv 2 以上

AIFL 2 → Lv 3 以上

上記の内部特性の要求レベルごとの要求事項は以下の通りである。

- Lv1
  - ➢ 外部環境変化によりシステムの品質が著しく失われたときの対応について、あらかじめ検討しておくこと。
  - ➢ オンライン学習を行う場合には、予想外の品質の低下がもたらす影響について

あらかじめ検討しておき、必要な場合には動作範囲の限定などのシステム的な対応を取ること。

- オフラインで追加学習を行う場合には、前 7 項に準じた品質管理を行うこと。

・ Lv2

- システムの利用状況が許す範囲において、システムの品質について、動作結果との対照などから品質劣化・誤判断のモニタリングを行うこと。モニタリングにおいては、プライバシーなど製品品質以外の要因を十分に検討すること。
- オンライン学習を行う場合には、追加学習結果を何らかの方法で定常的にモニタリングすること。モニタリングの結果で性能要求からの逸脱が判明した場合には、直ちに対処を行うことができること。
- オフラインでの追加学習を行う場合には、システム開発段階で用いたテスト用データセットでの「性能劣化の回帰テスト」を行い、更新前に品質が失われていないことを確認すること。必要な場合には、システム開発段階と同等の手法でテスト用データセットの更新を行うこと。

・ Lv3

- プライバシーなどと両立するシステム品質の監視手段を、運用体制を含めて必ず構築すること。
- オンライン学習を行う場合には、追加学習結果をシステムに反映する前に、システム内部で一定の品質確認を行う仕組みを実装し、想定外の品質劣化が無視できない場合には更新を中止する仕組みとすること。また、オフラインでの更新・修正手段を必ず確保すること。
- オフラインでの追加学習においては、運用での収集データと、システム初期構築時のテスト用データセット、および同じ手法で定期的に更新するテスト用データセットを用いて品質を管理すること。

7　品質管理のための具体的技術適用の考え方

7.1　要求分析の十分性

本項で必要とされる要求分析（ML 要求分析と称する）は、システムやサービスの利用される実世界に対応して、機械学習要素に入力されるデータの多様性を様々な観点から分析して言語化し、リスクやシステムの要求の差異を機械学習の訓練・テストで用いるデータの属性として具体化する工程である。これは基本的には、従来のシステムズエンジニアリングにおいて行ってきたリスク分析やハザード分析・要求分析の手法と同じ領域の考え方であり、これらの既存領域の取り組みや文献が参考になるところも多い。

従来型のソフトウェア工学の観点ではこれらの工程は、超上流工程の一部として位置づけられながら、定石としての確立した方法論は与えられていない。一方で機械学習の分野の観点では、PoC などを通じてデータサイエンティストが職人芸的なノウハウとして行ってきたことを、品質を説明する為の工程として形式化・知識化することに相当する。

7.1.1　全体的な取り組みの方向性について

システムズエンジニアリングの観点からの分析工程全体の概観としては、「ソフトウェア工学」[放送大学 2019] など一般的な教科書をまず参考にされたい。また、特にリスク回避性について、より詳細化された分析工程の例としては、例えば Causal Loop Diagram や、STAMP/STAP [IPA 2016-2019] などを事例として掲げられる。これらの分析を通じて得られた結果は、従来では Fault Tree Analysis, Fault Mode and Effects Analysis, Loop Diagram, Feature Tree などの形で具体化されることが多い。本ガイドラインでは特に機械学習要素の構築に用いるデータの性質として具体化することから、これらの分析のいくつかを元に、最終的には Feature Tree を構築することを想定して全体を構成している。

これらの分析を具体化すると、「入力事象（特にリスク要因）の推定」「分析の出力としてのデータの構造の推定」の 2 点が特に大きな問題となり得る。本節ではさらに、この 2 つの観点から機械学習利用システム及び機械学習利用要素に特有の考え方について付記する。

[放送大学 2019] 中谷 多哉子，中島 震.「ソフトウェア工学」。放送大学大学院教材，放送

大学教育振興会, 2019 年 3 月.
[IPA 2016-2019] 独立行政法人情報処理推進機構,「はじめての STAMP/STPA」. 2016 年～2019 年 6 月. https://www.ipa.go.jp/ikc/reports/20190329.html

7.1.2　入力側のリスク要因の推定

入力側の実空間などに存在するリスク要因の洗い出しは、基本的には正解のないブレインストーミングであり、その成果の充実度をデータで説明することはなかなか難しく、プロセス的な担保が基本となる。

この概説的な説明として、米国航空宇宙局（NASA）の「NASA Hazard Analysis Process」[Deckert 2010] はコンパクトながら、特に初期段階のハザード分析においてブレインストーミングの起点として着目すべき事柄として、

① 標準ハザードリスト
② 過去の経験・従来システムからの文書類
③ エンジニアリングの訓練と経験

を掲げている。さらに、同文献では①標準ハザードリストの起点として、26 項目の「NASA 一般ハザードリスト」を提示している。これらの中には他の応用では問題にならないようなハザード原因も含まれてはいるが、このリストは特に屋外環境で動作するシステムの機械的動作に伴うハザードの原因としては十分に汎用性が高く、他のシステムの検討においても分析の起点となり得る。

また、②はこれまでの機械学習システムの構築においてデータサイエンティストが行ってきたことと同種のものと考えられる。過去の類似のシステムや、同一システムの前バージョンの分析データは貴重なものであり、積極的に活用すべきものである。加えて、PoC 段階ではこのような出力を意図して構築実験を行うことで、本開発システムの構築に必要な対象の性質理解を進めることが推奨される。さらに③は、いわゆる「ドメイン知識」の活用に相当すると考えられ、機械学習の専門家だけでなくシステム発注側の業務内容に関する知識を盛り込むことに相当すると考えられる。同スライドでは、これらの観点を元に分析を行い、「明確な基準でハザードの分類を判断すべき」とまとめられている。

本ガイドラインでも、基本的には上記①②③の 3 つの観点全てを俎上に上げた検討を行い、さらに PoC 段階において得られた知見を踏まえ、その検討過程と結果をきちんと記録することを推奨する。具体的には、

・ ①として、既存の機能安全設計の基礎知識の活用や、応用ごとの既存のハザードリス

ト、あるいは上記の NASA ハザードリストを元にした「読み替え」のブレインストーミングなど

- ②として、先行システムや過去の機械学習を利用した類似システムの分析事例の活用、
- ③として、企画者（受託開発における開発依頼者）とのブレインストーミングによるドメイン知識の導入、さらに
- PoC 段階での予備的なデータおよび機械学習訓練の試行において得られた例外的ケースなどの知見

の全てを統合し、1 つのリスク要因リストとして整理する。

[Deckert 2010]　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

7.1.3　出力としてのデータの構造の推定

一方で、この工程の出力は機械学習に用いるデータの属性付けであり、次工程である実際の機械学習・訓練プロセスの特徴が、要求として必然的に影響してくる。次工程の訓練プロセスの観点から見た本工程への要求は、「機械学習工程を通じて『別のもの』として扱う属性」を特定することであり、これには

④　出力値やリスクの違いから、「別のもの」と認識しないと困るもの

⑤　入力値や機械学習モデルの特性上、出力値が同じであっても「別のもの」として扱われてしまう可能性のあるもの

の 2 つの側面がある。

このうち④は、システム仕様と前記のハザード分析から主に導出される。出力値の違いは訓練用データのラベルの違いでもあり、またリスクの違う事象はハザードの原因を特定すれば基本的には特定される。

一方で⑤には例えば、画像認識の背景の違いや文字の書体の違いなど、応用ドメインごとに特有の事情があり、また実装上の前処理や学習に用いるネットワークの種類などにも影響される可能性がある。このような分析には、どうしても最終的な実装形態をある程度想定した分析が必要となる。これはソフトウェア工学的には Implementation Forecast と呼ばれる事柄であり、不適切に行うとリスク分析を歪まされる危険性もあるものの、機械学習要素

の品質管理においては避けがたいものと考えられる。具体的には、本実装段階を意識した PoC 段階での机上検討と、データを用いた分析が対応すると考えられる。

実際に Forecast を行うべき観点の抽出そのものは、「構築された AI が似た状況で判断を異にするリスク」の観点として考えると、前節で掲げた 3 つの分析上の観点が再び有効であると考えられる。このような観点から、前節の②③及び本節の⑤を合わせて、各応用分野ごとに、ある程度のシステムの分析結果を集積し、分野毎のより詳細化された標準ハザードリストとして整備することが、将来的には望ましいと考える。本ガイドラインの執筆者としても、この観点からの周辺文書の整備を将来の方向性として積極的に考えたい。

7.2　データ設計の十分性

7.2.1　基本的考え方

前節の分析が十分になされていることを前提にして、本内部品質の目的は 2 つあり、機械学習利用システムが「全く学んだ・試されたことのない未知の状況」に遭遇しないようにするために、必要なデータを訓練時・テスト時にきちんと用意しておくことと、前節で膨大に準備した属性の組み合わせについて、機械学習の構築を現実的にするための「間引き・統合」をシステマチックに行うことにある。

仮に、極めてシステムの解こうとする問題がシンプルで、属性の組み合わせが高々数十程度な場合（例：数字認識で 10 文字×2 書体程度を認識すれば良く、紙質の差異なども考慮しなくて良く 20 通りしか属性組み合わせがない場合）であれば、それぞれの属性毎に認識率を一定以上に与えるデータの量を考え、それぞれを満たすデータの総量があれば、網羅的に十分な訓練を説明することが出来る。しかし実際には従来型のシステムであっても、属性の組み合わせは単純なかけ算では数万通り以上になることも多く、全ての組み合わせのデータを検査用に用意することは現実的でないことが多い。まして機械学習要素においては、属性の組み合わせが細分化された結果として訓練用データや検査用データがそれぞれ 0 件から数件となるようでは、大数の法則が使えず訓練の収束性などの判断が不可能になってしまう。

このような問題への対策として、従来のソフトウェア工学では「組み合わせテスト」と呼ばれる技術があり、これは「網羅性基準」と呼ばれる考え方を用いて、属性の分類の組み合わせの詳細度と、テスト工程における検査項目の充実度の基準を、関連させつつも別々に設

定する考え方である。この考え方は従来のソフトウェア構築ではテスト工程での検査用データの評価によく用いられるが、機械学習の訓練工程もまたデータ主体のプロセスであることから、特に有効と考えられる。

実際に組合せテスト技術（t-way）を元にした機械学習用のデータセット評価の事例が[Barash 2019]にも紹介されている。

[Barash 2019] 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, ESEC/FSE 2019, pp.1048-1058. 2019.
https://dl.acm.org/citation.cfm?id=3340442

7.3　データセットの被覆性

データセットの被覆性は、機械学習の品質管理でも極めて難しい問題で、ここに誤りがあると、判断が安定しなくなる・特定の想定内の状況で判断を誤るなどの結果を招き、公平性の失陥の原因にもなりうる。

既に「要件分析の十分性」において、「言語化できる」状況の差異についてはカバーをしているので、ここではその他の状況の微少な差異をデータ構築時に欠落させないことが求められる。

ライフサイクルにおいては、データの事前準備・データの評価が主に対応するが、テスト段階においても補助的な確認を行うことが考えられる。

7.3.1　データ取得段階における配慮

本項目の基本的な実現手段は、データ準備段階でのデータ取得計画に頼らざるを得ない。

実運用時に想定される状況を偏り無く取得するためには、データを収集する範囲や期間・規模などの計画が不可欠である。具体的な方法は応用ごとにブレインストーミング的に考えざるを得ないが、「要求分析の十分性」の分析の段階で過去の同種の応用の知見を踏まえつつ深くブレインストーミングを行うことで、ある程度の多様性の度合いを事前に検討することができる可能性が高い。

7.3.2　データ整理段階における追加的検査

「要求分析の十分性」の段階で、洗い出した上で属性として採用しなかった属性候補がある場合や、「データセットの十分性」の検討段階で統合され隠れた属性が有る場合には、各ケースの内部で、これらの追加的な属性の分布を確認することは重要である。特に後者の段階で統合された属性に関しては、明確な言語化定義がされている上、場合によってはデータラベルが付与され機械的に確認できることも有り得る。

もちろんブレインストーミングの段階から完全に見落とされていた属性はここでも発見できないため完全な手法ではないが、検討に値すると考えられる。

またデータの種類によっては、前処理段階でデータ固有の特徴量のようなものが得られ、その分布を確認することも有り得る。

7.3.3　テスト段階での追加的検査

そもそもデータの分布自体に問題がある事例で、テスト段階で再確認できることは限られているが、例えば属性として採用した特徴量と、見落としている隠れ特徴量の間に明らかな隠れ相関がある場合には、機械学習モデルの入力値と出力値の間の相関・影響度の分析を行うことで、本来有るべき特徴と異なる特徴を捉えて判断していることが分かる場合もある。安全性などが特に要求される分野においては、このような分析も検討に値する。

7.4　データセットの均一性

データセットの均一性は、前節のデータセットの被覆性を、入力データ全域を単一の「ケース」として検討することに相当する。7.3 節の各節に掲げた対策が、この場合にも適用できると考えられる。

7.5　機械学習モデルの正確性・安定性

機械学習モデルの正確性・安定性の検査は、従来のソフトウェア開発においては「単体テスト」などのソフトウェア・テスティングに対応する作業と考えられる。本節では、

- ソフトウェア・テスティングの技術［1］を、機械学習モデルを含む「推論エンジン」全体に適用する際に、機械学習ソフトウェアが持つ特徴によって特に考慮すべき問題と、
- 安定性を直接的に検証する技術の可能性

について述べる。

7.5.1　機械学習要素におけるソフトウェア・テスティング

問題点の第 1 に，推論結果の品質を議論する際に、その正確性の根源として検討すべき対象が，訓練用データセットを入力し訓練済み学習モデルを求める訓練学習プログラムと，得られた訓練済み学習モデルが機能振舞いを規定する予測・推論プログラムの 2 つに分かれていることが挙げられる．第 2 に，計算結果が正しいか否かを判断する基準が明らかでなく，オラクル問題[2]への対応を考える必要がある．訓練学習プログラムは，その結果の正解値を予め知ることが困難である．また，予測・推論プログラムが導く答えは確定的ではなく確からしさを伴う相対的な値であり，絶対的な正解を定義することが難しい[3]．機械学習プログラムはこの観点からはテスト不可能プログラム[4]に分類される。

7.5.1.1　テスティングの２つの観点

最初に，テスト入力生成の問題に関連する特徴を説明する．プログラムの品質を確認する際，正常系テスティング（Positive Testing）と例外系テスティング（Negative Testing）を適宜行う．

正常系テスティングは，検査対象が想定した機能振舞いを示すことの確認である．説明の都合上，プログラムが，事前・事後条件を伴うとしよう．この時，「事前条件を満たす時，プログラム本体実行後の状態で事後条件を満たす」という関係が成り立つ．事前条件を満たすテスト入力データに対して，上記の関係が成り立つ時，プログラムはテストに合格したとする．

例外系テスティングは，想定外の状態でプログラムが破滅的な不具合を生じないことを確認する．一般に，プログラムは事前条件を満たさない入力に対しても，それなりの振舞いを示すことが期待される．暴走して異常終了するようなことがあってはならない．実行結果が異常終了を導くか否かを調べる際には，実行結果の正しさの基準を与えるオラクルが存在しない．このような場合，暗黙オラクル（Implicit Oracles）と呼ぶ．

従来から，システム・ソフトウェアやセキュリティ・システムなどのオープンなソフトウェアの例外系テスティングでは，条件付きランダム生成したデータをテスト入力とするファズ・テスティング（Fuzz Testing）の方法[5]が知られている．この方法は，例外系テスティングでも有効である．テスト対象ごとに適切なファズ（Fuzz）を生成する方法を考案する必要がある．

7.5.1.2　メタモルフィック・テスティング

メタモルフィック・テスティング（MT）[6][7]は，数値計算あるいはコンパイラを含むトランスレータなど，入力に対する計算結果の正解値を予め知ることが難しいテスト不可能プログラムの検査方法として考案され，その後，暗黙のオラクルを用いるシステム・ソフトウェアやセキュリティ・システムの検査に適用された．現在では，機械学習プログラムに対する標準的なテスティング方法論になっている[8]．

7.5.1.3　テスト入力の自動生成

MT では，テスト対象ごとに適切なテスト入力データを生成する方法が重要となる．また，正常系テスティングあるいは例外系テスティングの目的に応じて適切な方法を考案する．特に，対象が訓練学習プログラムの時，その入力はデータの集まり（データセット）であることから，データセット多様性（Dataset Diversity）[9][10]の考え方が，正常系ならびに例外系テスティングの入力生成方法論に指針を与える．

予測・推論プログラムのテスティングでは，訓練学習時に想定していなかったような多様なデータを入力し予測結果を確認する．つまり，機械学習の分野でテスト時補完（Test-time Augmentation）と呼ばれている手法である．実際，機械学習の技術を応用してデータ生成を行う方法が試みられている．画像データを対象とするデータ補完（Data Augmentation）の方法[11]，敵対生成ネットワーク（Generative Adversarial Networks, GAN）による方法[12]，などがある．GAN が生成するデータは，主として正常系テスティングを想定したもので，想定範囲内の網羅性向上を達成する．

［参考文献］

[1] P. Ammann and J. Offutt: Introduction to Software Testing, Cambridge University Press 2008.
[2] E.T. Barr, M. Harman, P. McMinn, M. Shahbaz, and S. Yoo: The Oracle Problem in Software Testing: A Survey, IEEE TSE, 41 (5), pp.507-525, 2015.
[3] S. Elbaum and D.S. Rosenblum: Known Unknowns - Testing in the Presence of Uncertainty, In Proc. 22nd FSE, pp.833-836, 2014.
[4] E.J. Weyuker: On Testing Non-testable Programs, Computer Journal, 25 (4), pp.465-470, 1982.
[5] B.P. Miller, L. Fredricksen, and B. So: An Empirical Study of the Reliability of UNIX Utilities, Comm. ACM, vol.33, no.12, pp.32-44, 1990.
[6] T.Y. Chen, S.C. Chung, and S.M. Yiu: Metamorphic Testing - A New Approach for Generating Next Test Cases, HKUST-CS98-01, The Hong Kong University of Science and Technology, 1998.
[7] 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, vol.51, no.1, Article No.4, pp. 1-27, 2018.
[8] 中島 震: 機械学習ソフトウェア・テスティングの技術動向，電子情報通信学会システム数理と応用研究会報告，2020.
[9] 中島 震: データセット多様性のソフトウェア・テスティング，コンピュータ・ソフトウェア，35(2), pp.26-32, 2018.
[10] S. Nakajima and T.Y. Chen: Generating Biased Dataset for Metamorphic Testing of Machine Learning Programs,IFIP-ICTSS 2019, pp.56-64, 2019.
[11] A. Krizhevsky, I. Sutskever, and G.E. Hinton: Imagenet Classification with Deep Convolutional Neural Networks, In Adv. NIPS 2012, pp.1097-1105, 2012.
[12] I.J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio: Generative Adversarial Nets, In Adv. NIPS 2014, pp.2672-2680, 2014.

7.5.2　安定性の評価と向上に関する諸技術

図 16 に示す安定性の評価と向上に関する技術を、6.6.3 の確認要求事項のレベルを参照

しながら説明する。

図 16　安定性の評価・向上に関連する技術（技術を適用する工程とレベル）

7.5.2.1　交差検証（cross validation）

交差検証は、データセットを訓練用、バリデーション用、テスト用に分割することによって、訓練時のデータセットへの過剰な適合を抑える訓練法であり、広く利用されている[Kohavi 1995]。例えば、K-分割交差検証では、訓練用のデータセットを K 分割し、1/K のデータセットをバリデーション用、残りの(K-1)/K を訓練用として、訓練用とバリデーション用のデータセットを入れ替えながら訓練する。Lv1 での適用が推奨される技術である。

7.5.2.2　正則化（regularization）

正則化手法とは、訓練時に学習するパラメータが過剰に大きくなることを抑える方法である[Nowlan 1992]。例えば、訓練時に最小化する損失関数に正則化の項（学習パラメータの絶対値とともに大きくなる項）を追加することによって、訓練用データセットへの過剰な適合（過学習）を抑えられる。また、最近ではドロップアウトという正則化手法も使われている[Srivastava 2014]。ドロップアウトでは、訓練中にランダムにニューロンを訓練対象から除外することによって、複数の機械学習モデルを同時に訓練する場合と同様の効果を得られる。安定性が必要な場合、レベル 1 での適用が推奨される技術である。

7.5.2.3　敵対的データ生成（adversarial example generation）

敵対的データは、判定を誤るように摂動を加えたデータである。機械学習モデルでは、人間には認識できないようなわずかな摂動でも誤推論することが知られている。そのような敵対的データを生成する様々な手法が提案されている[Pei 2017, Carlini 2017]。敵対的デー

タを生成するために必要な摂動が大きいほど安定していることを意味するため、その摂動の大きさは安定性の評価にも利用されている。このような敵対的データを生成するための実用的なツールはまだ整備されていないが、将来的にはレベル 2 の評価に適用可能な技術である。

7.5.2.4　最大安全半径（maximum safe radius）

最大安全半径とは、元データとその敵対的データの距離の最小値である。7.5.2.3 の敵対的データ生成では近傍の敵対的データを探索するが、最大安全半径では近傍（最大安全半径の超球の内側）に敵対的データが存在しないことを保証できる。最大安全半径を正確に求めるために形式手法（ソルバー）を用いた手法が提案されている[Katz 2017, Tjeng 2019]。しかし、正確な最大安全半径の計算コストは非常に高いため、計算可能なネットワークのサイズ（ニューロン数）に制限がある。そこで、最大安全半径よりも小さい安全半径の近似計算法も提案されている[Weng 2018, Boopathy 2019, Wu 2019]。また、100%ではないが、確率的に安全性を保証する半径を近似的に計算する手法も提案されている[Weng 2019]。これらはまだ研究段階の技術であるが、最大安全半径の超球の内側には敵対的データは存在しないことを保証できるため、将来的にはレベル 3 での適用が期待される技術である。

7.5.2.5　敵対的訓練/ロバスト訓練（adversarial training / robust training）

敵対的訓練は誤推論しそうな近傍データを探索しながら訓練を行う手法である[Madry 2019]。通常の訓練と比較して、機械学習モデルの敵対的データに対する耐性を向上させることができる。このような訓練を行うための実用的なツールはまだ整備されていないが、将来的にはレベル 2 に適用可能な技術である。

ロバスト訓練は近傍に敵対的データが存在しないように訓練を行う手法である[Wong 2018]。最大安全半径の近似計算法もセットで与えており、訓練用データセットの各データの近傍に敵対的データが存在しないことを保証できる。まだ研究段階であるが、将来的にはレベル 3 での適用が期待される技術である。

7.5.2.6　ランダムスムージング（randomized smoothing）

ランダムスムージングは、ランダムノイズを付加したデータをニューラルネットに入力

したときの出力の期待値を推論結果とする手法であり、ランダムスムージングによって安定性が向上することが報告されている[Liu 2018]。入力データの近傍に対するランダムスムージングの推論結果の正しさを保証するためのランダムノイズの付加方法も提案されている[Lecuyer 2019, Cohen 2019]。出力の期待値を平均値で近似するために複数回の推論が必要であることや、ランダムノイズを大きくすると安定性は向上するが正確性は低下するトレードオフがあることなどを考慮する必要があるが、推論結果の安定性を保証するために、レベル 3 で適用が期待される技術である。

7.5.2.7　敵対的データ検知（adversarial example detection）

運用時に敵対的データを検知する手法が提案されている[Xu 2018, Ma 2019]。まだ研究段階であるが、機械学習モデルに潜在する敵対的データを防御するために、将来的にはレベル 2 での適用が期待される技術である。

[参考文献]

- [Kohavi 1995] 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.
- [Nowlan 1992] Steven Nowlan and Geoffrey Hinton, Simplifying neural networks by soft weight-sharing. Neural Computation, 4(4), 1992.
- [Srivastava 2014] 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(Jun), pp.1929－1958, 2014.
- [Pei 2017] 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.
- [Carlini 2017] Nicholas Carlini and David Wagner, Towards Evaluating the Robustness of Neural Networks, IEEE Symposium on Security and Privacy (SP), pp.39-57, 2017.
- [katz 2017] Guy Katz, Clark Barrett, David Dill, Kyle Julian, and Mykel Kochenderfer, Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks, International Conference on Computer-Aided Verification (CAV), 2017.

- [Tjeng 2019] Vincent Tjeng, Kai Xiao, and Russ Tedrake, Evaluating robust- ness of neural networks with mixed integer programming, International Conference on Learning Representations, 2019.
- [Weng 2018] 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, The 35th International Conference on Machine Learning, PMLR 80, pp.5276-5285, 2018.
- [Boopathy 2019] 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, The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI 2019), pp.3240-3247, 2019.
- [Wu 2019] Min Wu, Matthew Wicker, Wenjie Ruan, Xiaowei Huang, and Marta Kwiatkowska, A Game-Based Approximate Verification of Deep Neural Networks with Provable Guarantees, Theoretical Computer Science (2019), https://doi.org/10.1016/j.tcs.2019.05.046
- [Weng 2019] 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, The 36th International Conference on Machine Learning (ICML 2019), PMLR vol. 97, pp.6727-6736, 2019.
- [Madry 2018] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu, Towards Deep Learning Models Resistant to Adversarial Attacks, The Sixth International Conference on Learning Representations (ICLR 2018). 2018.
- [Wong 2018] 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.
- [Liu 2018] X. Liu, M. Cheng, H. Zhang, and C. Hsieh, Towards robust neural networks via random self-ensemble, The European Conference on Computer Vision (ECCV 2018), 2018
- [Lecuyer 2019] 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.

・ [Cohen 2019] 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.
・ [Xu 2018] Weilin Xu, David Evans, and Yanjun Qi, Feature Squeezing: Detecting Adversarial Examples in Deep Neural Networks, Network and Distributed Systems Security Symposium (NDSS), 2018.
・ [Ma 2019] Shiqing Ma, Yingqi Liu, Guanhong Tao, Wen-Chuan Lee, and Xiangyu Zhang, NIC: Detecting Adversarial Samples with Neural Network Invariant Checking, Network and Distributed Systems Security Symposium (NDSS), 2019.

7.6 （欠番）

訓練済み学習モデルの安定性に関する技術については、前 7.5 節において正確性と一括して記述した。

本節は、6 章および 1.7 節と、本章の各節番の対応のため、欠番とする。

7.7 プログラムの健全性

7.7.1 基本的な考え方

プログラムの健全性は、通常のソフトウェアでも重要な項目であり、特にオープンソースを含む多数のライブラリを混用して用いる機械学習 AI の実装においては品質の管理が困難になることが想定される。オープンソースソフトウェアは無保証が基本であり、最終的な利用者との関係では、誤動作の差異の責任の所在だけでなく、潜在的な誤りの発見や監視、場合によっては修正までが、オープンソース利用者である開発者・運用者の責務になると考えられる。

さらに、機械学習モデルの構築においては、バグ（プログラムの誤り）の存在を訓練プロセスが織り込んでしまい、テストなどで表面的にバグの影響が現れないまま、品質の劣化が確認されることも判明している [Nakajima 2020]。

一方で、通常のソフトウェアとの関係では、ライブラリや基盤ソフトなどの構成管理や品

質のモニタリングは特にセキュリティ関係で重要視されており、そのための基盤やサービスが充実しつつある。そのサービスのもつデータの中には、限定的ではあるが、機械学習特有のライブラリなどの情報が含まれているものもある。機械学習応用においても、これらの存在する基盤は活用する価値があると考えられる。

7.7.2　オープンソースソフトウェアの品質管理

各事業者はオープンソースライブラリをどの程度信用し、その品質を自らメンテナンスするかについて、考えておくことが望ましい。

必要な品質のレベルと関係して、必要な場合には、品質の担保されたサポート付きのライブラリや、自社での品質検査プロセスを経たソフトウェアを使うようなことも想定される。

7.7.3　構成管理とバグ情報の追跡

ソフトウェア要素の構成管理については、例えばセキュリティ分野でシステムの構成要素を列挙するための共通 ID としての Common Platform Enumeration (CPE) [Mitre:CPE,IPA:CPE] があり、これらをベースとして、利用しているソフトウェア部品のバージョンを管理し更新などの情報を抽出する商業製品なども存在している。これと関連する脆弱性情報リスト Common Vulnerability Enumeration (CVE) [Mitre:CVE,IPA:CVE] には、セキュリティ脆弱性に直結しないバグが列挙に含まれていないが、少なくとも CPE に関係したツールは、ライブラリおよび基盤ソフトウェアの最新版の追跡に有益である可能性が高い。

7.7.4　テスティングによる具体的な確認の可能性

また、従来のソフトウェアと異なり機械学習要素においては、構成するソフトウェアにバグがある場合、そのバグが実際の機械学習結果に直接の影響を及ぼすとは限らない。特に、学習訓練のフィードバックループの中にバグを含むソフトウェアが置かれている場合、訓練済み学習モデルがそのバグの振る舞いを「覚えてしまう」ことにより、結果的に訓練が一見してうまく行ったように見える場合が存在する。このような場合において、7.5.1.2 節に掲げたメタモルフィックテスト技術を用いた統計的な振る舞いの分析により、ソフトウェア自身のバグによる潜在的な品質劣化を見付けられることがあることが報告されている。

7.7.5　ソフトウェア更新と性能・動作への悪影響の可能性

一方で、ソフトウェアライブラリの開発者が性能や動作の改善を意図した更新を行った場合でも、実際の複雑な構成のソフトウェアにおいては、却って動作が意図しないものとなる場合も多い。機械学習システムの構成によっては、バグの振る舞いが訓練過程において順応・学習されてしまうこともあり、ソフトウェア構成部品を更新した際には、動作及び性能の再確認が欠かせない。場合によっては訓練をやり直す、または脆弱性などの影響を考慮した上で古い版を使い続けるなどの判断も必要である。

具体的な判断はその状況に依存するため一概にガイドラインで推奨はできないが、このような場合に「きちんと品質を管理している」と主張するためには、その判断の経緯をきちんと記録しておき、説明可能にしておくことも重要となる。

7.7.6　参考文献

[Mitre:CPE] MITRE, Common Platform Enumerations. http://cpe.mitre.org/

[IPA:CPE] 独立行政法人 情報処理推進機構 セキュリティセンター, 共通プラットフォーム一覧 CPE 概説. https://www.ipa.go.jp/security/vuln/CPE.html

[Mitre:CVE] MITRE, Common Vulnerability Enumerations. https://cve.mitre.org/

[IPA:CVE] 独立行政法人 情報処理推進機構 セキュリティセンター, 共通脆弱性識別子 CVE 概説. https://www.ipa.go.jp/security/vuln/CVE.html

[Nakajima2020] 中島震, 訓練済み機械学習モデル歪みの定量指標, 電子情報通信学会ソフトウェアサイエンス研究会、2020.

7.8　運用時品質の維持性

本節では、運用開始時点で充足されていた内部品質を、運用期間中を通じて維持するための技術について述べる。運用時に発生しうる外部環境の変化に対し、運用開始時点で実現されていた内部品質を維持するためには、機械学習要素を変化に対して追従させる必要があり、その運用形態は 6.8.1 節に述べたとおり、必要に応じたタイミングで開発環境に戻ってデプロイしなおすという一括処理的な更新を行うパターンと、運用環境にて随時または高頻度に自動的に更新を行うパターンの 2 つがある。前者のパターンでは、いつ更新を行う

べきかの判断するためのモニタリングと、更新処理が主要素となり、後者のパターンでは、自動更新処理が主要素となり、実現にはオンライン学習［Shalev-Shwartz 2012］などが採用される。自動更新を行う場合にも、自動更新が正常稼働しているかのモニタリングと、正常稼働状態を逸脱した場合の対処としての更新処理は必要となる。

ここでは、いずれのパターンでも必要な、モニタリング技術について紹介し、特にコンセプトドリフト［Gama 2014］と呼ばれる、データ分布が時間経過に伴い変化する現象を検知する技術について紹介する。そして、訓練済み機械学習モデルの更新処理の中核となる再学習のための技術と、そこで用いられる追加の学習データの作成技術について紹介する。

7.8.1　モニタリング

運用時においては、外部環境の変化などの様々な要因によって、新たに取得した入力データと出力（推論）結果との関係が、学習に用いた訓練データの同関係から変化している場合がある。そのような入出力関係が変化したデータに対し、設計・開発時に学習させた訓練済み機械学習モデルを運用時においても使用し続けた場合、パフォーマンス（精度）が低下し、重大な損害が生じる恐れがある。それゆえ、運用開始時点で充足されていた品質を、運用期間を通じて維持しているかを調べるために機械学習システムや機械学習要素の振る舞いを継続的にモニタリングする必要がある。

運用時のモニタリングタスクとしては、以下の 4 つが挙げられる。

- 精度モニタリング
- KPI モニタリング
- モデル出力モニタリング
- 入力データモニタリング

「精度モニタリング」は、訓練済み機械学習モデルの精度を直接測定する。精度モニタリングは、精度の計算に必要な訓練済み機械学習モデルの推論結果に対する正解収集方法に従って、いくつかのパターンに分かれている。即ち、推論のあと、(1) 一定期間後に自動で正解が手に入る場合、(2) 自動で正解が手に入らず、人力でラベル付けを行う必要がある場合、そして、(3) 人力のラベル付けがコストに合わない、またはラベル付けが不可能な場合である。それゆえ、精度モニタリングを適切に行うためには、上記の正解収集方法の場合分けに従って、適切なモニタリング手法を選択する必要がある。またアプリケーションによっては、モデルの単なる精度だけではなく、コンバージョン率などユーザ利益に即した KPI の観点での監視、即ち「KPI モニタリング」が重要

視される場合がある。そのような場合においては、モデルの精度と KPI の一貫性にも留意してモニタリングを行うことが必要である。

続いて、「モデル出力モニタリング」および「入力データモニタリング」は、それぞれ訓練済み機械学習モデルによる推論結果またはその入力データの監視を表しており、その監視方式は、人力監視（全数、サンプリング）、自動監視（アラート条件が既知）、フィルタリング（アラートの可能性が高い条件が既知）に分類される。特に、モデル出力モニタリングにおいては、人力監視において、医療診断など出力系に専門家の確認が推論毎に必ず入る場合と、ある一定期間後にまとめて事後確認する場合に細分化される。同様に、フィルタリングによる監視においても、偽陽性は許容できるが、偽陰性は許容できないなど、様々な条件が付随する場合がある。加えて、入力データモニタリングにおいて、フィルタリングによって監視する場合にアラートを出すか、入力を捨てるかは、アプリケーション依存となる。

7.8.2　コンセプトドリフト検知手法

コンセプトドリフトは運用時に訓練済み機械学習モデルが精度低下を引き起こす主な原因の 1 つであり、近年、様々な監視（検知）手法が提案されている。コンセプトドリフト検知手法は、運用時に取得したデータに関する正解ラベルの使用の有無に従って、表 4 のように分類される [Sethi 2017]。

表 4　コンセプトドリフト検知手法の分類と特徴

正解ラベルの使用の有無	手法	特徴
ラベルあり検知（教師あり検知）	Sequential analysis	Accuracy などの絶対値の監視
	Statistical Process Control	エラー率の増減の監視（予兆発見による早期検出）
	Window based distribution monitoring	学習時と運用時の分布のずれの監視
ラベルなし検知（教師なし検知）	Novelty detection / clustering method	クラスタリングによる簡易検知
	Multivariate distribution monitoring	データの分布に対して統計的仮説検定などを適用して検知

	Model dependent monitoring	モデルに依存する出力。確信度(confidence)スコアなどを利用

特に近年においては、ラベルなし検知手法に関する研究が盛んに行われており、例えば文献［Faria 2013］では、k-means 法などのクラスタリング手法に基づいたマルチクラス問題における未知クラス検出アルゴリズム（MINAS）が提案されている。また、文献［Reis 2016］では、サンプルが同じ分布から発生しているか否かを判断するノンパラメトリック検定手法であるコルモゴロフ－スミルノフ(Kolmogorov-Smirnov, KS)検定を、逐次化することで計算量を改善したインクリメンタル KS 検定のアルゴリズムが提案されている。さらに、文献［Lindstrom 2011]では、訓練済み機械学習モデルの出力に付随する確信度スコア（confidence）を用いることで、正解ラベルを用いずにデータの変化を検知する手法（CDBD）が提案されている。

7.8.3　再学習

前述のモニタリングによってデータ分布の変化や訓練済み機械学習モデルの精度低下が検知された場合、直近のデータを訓練用データに追加または既存のデータと差し替えたデータセットを用いて機械学習モデルを再訓練する必要がある。この再学習に関する研究も盛んに行われており、例えば、文献［Tian 2018］では、既存の機械学習フレームワークに対し、モデル更新の適切なタイミングを自動で判断するためのプラットフォーム設計方法が提供されている。また、破滅的忘却（catastrophic forgetting）と呼ばれるニューラルネットワークの再学習による従来タスクの忘却を軽減するために、「既存のモデル」と「新たなタスクを学習させたモデル」の両パラメータのバランスをとったパラメータを、再学習後のモデルに適用する手法も提案されている［Lee 2017]。

7.8.4　追加の学習データ作成

正解ラベルが自動収集できないケースは実用上よくみられるが、この場合は、運用時に新たに取得したデータに対して手動で正解ラベル付けを行う必要がありコストが高い。この問題に対し、GUI (Graphical User Interface）を備えたソフトウェアを利用

してラベル付け労力を軽減する方法［Vostrikov 2019］や、アクティブ・ラーニング(能動学習)により、ラベル付けを行うデータ数を減らし再学習のコストを削減する研究［Settles 2009］などが、提案されている。

[参考文献]

- [Shalev-Shwartz 2012] S. Shalev-Shwartz, Online learning and online convex optimization, Foundations and Trends® in Machine Learning, 2012, 4.2, pp. 107-194.
- [Gama 2014] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, A survey on concept drift adaptation, ACM computing surveys (CSUR), 2014, 46.4, pp. 1-37.
- [Sethi 2017] T. S. Sethi, M. Kantardzic, On the reliable detection of concept drift from streaming unlabeled data, Expert Systems with Applications, 2017, 82, pp. 77-99.
- [Faria 2013] E. R. Faria, J. Gama, and A. C. Carvalho, Novelty detection algorithm for data streams multi class problems, In Proc. of the 28th annual ACM symposium on applied computing, 2013. pp. 795-800.
- [Reis 2016] 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 Int. Conf. on Knowledge Discovery and Data Mining, 2016, pp. 1545-1554.
- [Lindstrom 2011] P. Lindstrom, B. M. Namee, and S. J. Delany, Drift detection using uncertainty distribution divergence In Proc. of the 2011 IEEE 11th Int. Conf. on Data Mining Workshops, 2011, pp. 604-608.
- [Tian 2018] H. Tian, M. Yu, and W. Wang, Continuum: A platform for cost aware, low latency continual learning, In Proc. of the ACM Symposium on Cloud Computing, 2018. pp. 26-40.
- [Lee 2017] 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, 2017. pp. 4652-4662.
- [Vostrikov 2019] A. Vostrikov, S. Chernyshev, Training sample generation software. I. Czarnowski, R. Howlett, L. Jain (eds), Intelligent Decision Technologies 2019. Smart Innovation, Systems and Technologies, vol 143. Springer, Singapore, 2019.
- [Settles 2009] B. Settles, Active learning literature survey, University of Wisconsin-Madison Department of Computer Sciences, 2009.

8　（参考）関連する文書類に関する情報

本章の内容は参考（informative）である。

8.1　他のガイドライン類との相互関係

8.1.1　経済産業省の AI 契約ガイドライン

経済産業省が発表した「AI・データの利用に関する契約ガイドライン」[16] は、受発注・準委託などの関係により連携・分担して機械学習 AI などを含むシステムを開発する際における事業者間の契約に関する留意点をまとめたものである。

同契約ガイドラインは主に開発の事業者間での契約の在り方や責任の所在などを明確にするものである一方で、本ガイドラインは最終システムの利用者に対してサービス提供者が提供する品質の在り方を整理したものである。本ガイドラインの立場からは、本ガイドラインの規定する製品・サービスの利用時品質は、契約ガイドラインに役割として掲げられた開発のステークホルダ全員が共同してシステムに対して作り込み、製品・サービスの利用者に対して提供するものである。この品質を具体的にどのように事業者間で分担して実現するか、その際の契約や責任の分担をどのようにするかは、基本的には本ガイドラインの直接の対象外であり、契約ガイドラインなどに基づきステークホルダ間で合意をすれば良いものである。一方で、そのステークホルダ間の分担の際に着目し情報交換する品質上の技術的事項については、本ガイドラインがその検討の土台となり得る。一例としての役割分担の可能性については、5.2 節にも若干の分析を行った。

なお、当該契約ガイドラインにおいては、開発プロセスにおける受発注関係において「非ウォーターフォール型モデルが適合する」とされている。一方で、同ガイドラインの 46 ページにおいても③「開発」のほかに④「追加学習段階」などの前後段階のプロセスに着目していることから、本ガイドラインではシステムの企画から運用後廃棄までの広範囲をシステムライフサイクルプロセスの全体として捉え、モデルとしてはウォーターフォール型との混合モデルを基本として、25 ページの「図 5: 混合型機械学習ライフサイクルプロセス

[16] 経済産業省. AI・データの利用に関する契約ガイドライン. 2018 年 6 月. https://www.meti.go.jp/press/2018/06/20180615001/20180615001-3.pdf .

の概念図」として整理した。契約ガイドラインの推奨する「探索的段階型」の開発プロセスとの関係では、図 2 の中央の開発フェーズと、契約ガイドラインの「開発段階」が基本的に対応する。

8.1.2　QA4AI ガイドラインとの関係

AI プロダクト品質保証コンソーシアムは、2019 年 5 月に「AI プロダクト品質保証ガイドライン 2019.05 版」[17] を公開している。同ガイドラインは、「AI プロダクトの品質保証に対する共通の指針を与える」ものとして、AI プロダクトの品質保証において考慮すべき軸として

A)　「Data Integrity」
B)　「Model Robustness」
C)　「System Quality」
D)　「Process Agility」
E)　「Customer Expectation」

の 5 つの軸を提案し、それぞれの軸に対するチェックリストや具体的な品質管理技術のリストなどを提示している。

本ガイドラインとの関係では、これらの品質保証軸のうち A) B) C) の 3 つは、本ガイドラインの 1.7 節（16 ページ）に掲げる内部品質に対応していると考えられる。従って、同ガイドラインが具体的に列挙する技術も、7 章で提示する内部品質特性毎の品質管理手法と対応し、相互に補完しつつ実際に品質を実現するエンジニアに対する示唆を与えるものと位置づけられる。QA4AI ガイドライン中で 3 項目のチェックリストに掲げられた項目と、具体的な対応関係については、本ガイドライン 1.7 節に掲げる内部品質の対応関係については、表 5 に整理する。

また D) E) の保証軸は、本ガイドラインでは明確な品質管理軸として設定していないが、本ガイドラインが想定する適用プロセス（5.1 節）やそれを含む顧客とのビジネスプロセスなどを通じて実現されるものと考えられる。

全体として、QA4AI コンソーシアムのガイドラインは、実際に機械学習 AI を作るエンジニアにとって機械学習要素の内部品質を改善し作り込むための技術の可能性を見つけ出すための参考書（リファレンス）として有益である一方、本ガイドラインは機械学習 AI を含

[17] AI プロダクト品質保証コンソーシアム.「AI プロダクト品質保証ガイドライン 2019.05 版」. 2019 年 5 月. http://qa4ai.jp/download/ .

むシステム全体を企画開発する企業などが、システム全体の利用時品質をライフサイクルプロセス全体を通じて確保するための必要事項を網羅的に分析し可能な限り列挙することを意図しているもので、相互に補完関係にあると考えられる。

表 5: QA4AI ガイドライン（2019 年 5 月版）との関係の分析

・ QA4AI ガイドライン側の丸数字項番は、同ガイドライン 2.2 節での出現順に通し番号を振ったものである。

本ガイドラインの内部品質特性	QA4AI ガイドラインのチェックリスト
1.7.1 要求分析完全性	2.2.1 Data Integrity ②求める母集団のサンプルか、実際のデータを利用しているか、不必要なデータが含まれていないか、含むべきでない母集団のデータと混ざっていないか ⑤データは複雑すぎないか、単純すぎないか、必要な要素を適切に含んだサンプルか、ラベルは妥当か 2.2.2 Model Robustness ⑲数理的多様性、意味的多様性、社会的文化的多様性などを考慮し、十分に多様なデータで検証を行ったか
1.7.2 データセット完全性	2.2.1 Data Integrity ①量は充分か、コストは適正か、意味のある量か、「かさ増し」しても大丈夫か ②（再掲）
1.7.3 データセット被覆性	2.2.1 Data Integrity ④偏りやバイアス、汚染は無いか、自分たちが考えている「偏りを発生させるもの」だけでよいか ⑥データ内の性質（多重共線性など）は適切に考慮されているか
1.7.4 データセットの均一性	2.2.1 Data Integrity ④（再掲）
1.7.5 モデルの正確性	2.2.1 Data Integrity ③データに関する要求事項を満たしているか、データに関する制約に反していないことを監視しているか ⑦それぞれのデータは常識的な値か、外れ値は本当に外れているデータか、欠損に意味はないか、外れ値や

	欠損値の扱いは適切か ⑨学習用データと検証用データは独立しているか 2.2.2 Model Robustness ⑫正答率、適合率、再現率、 F 値 といった精度は妥当か ⑬汎化性能は確保されているか ⑭(AUROC といった)モデルのよさを表す指標は充分か ⑮学習は適切に進行したか、局所最適に陥っていないか ⑯適切なアルゴリズムやハイパーパラメータかどうかの検討は行ったか ㉒目標指標の計測が難しい場合、計測できるメトリクスとの関連は妥当か
1.7.6 モデルの安定性	2.2.2 Model Robustness ⑬⑮⑯ ⑰十分に交差検証などを行ったか ⑱ノイズに対して頑健か
1.7.7 プログラムの健全性	2.2.1 Data Integrity ⑪学習用プログラムやデータ生成プログラムの不具合によってデータの意味が毀損されないか
1.7.8 運用時品質の維持	2.2.1 Data Integrity ⑩オンライン学習を行う場合、その影響を適切に考慮しているか 2.2.2 Model Robustness ⑳デグレードは許容可能な範囲か、デグレードの影響範囲を把握できているか、学習は再現可能か、 学習時のふるまいと提供時のふるまいに齟齬はないか ㉑モデルが陳腐化していないか、実データに対する予測品質が劣化していかないか 2.2.3 System Quality ㉔性能などシステム全体のふるまいが劣化していかないか

外部品質に対応する項目	2.2.3 System Quality ㉓価値は適切に提供されているか ㉕システムを全体として、および意味のある単位で評価を行ったか ㉖発生しうる品質事故の致命度は許容できる程度に低く抑えられているか ㉗品質事故を引き起こしうる事象の発生頻度は低いと見積もることができるか、事象の発生頻度や事象の網羅性、環境統制性の検討は充分か ㉘システムの事故到達度・安全機能・耐攻撃性は充分か ㉙AI の寄与度を抑えられているか、システムが依存する他の（AI の、もしくは非 AI の）システムの変更は迅速かつ適切に反映できるか、不具合の影響を充分低く抑えられるか ㉚保証性、説明可能性、納得性は充分か
対応する特性がない項目	2.2.1 Data Integrity ⑧所有権や著作権・知的財産権、機密性、プライバシーは適切に考慮されているか

8.2　AI の品質に関する国際的取り組みとの関係

現在、AI の品質については、以下に述べるようなさまざまな国際的な取組がなされている。本ガイドラインの検討においては、これらの取り組みを踏まえて活用できる部分を取り入れる。また、本ガイドラインにより得られた優れた知見は国際標準化へ積極的に提案していく。

8.2.1　品質、安全性

ISO/IEC JTC 1/SC 42/WG 3 では品質や安全性に関するアドホック検討が立ちあげられ、

特に AI の品質特性、品質保証技術について、既存標準（ISO/IEC 25000（SQuaRE), ISO/IEC/IEEE 29119-4 （試験技術）の他、機能安全（IEC 61508, ISO 26262)）などの既存標準とのギャップ分析が実施されている。SC42 ではこれ以外にもデータ品質などについて、様々な議論が立ち上がりつつある。

8.2.2　透明性（transparence）

EU においては、High-Level Expert Group on Artificial Intelligence（AI-HLEG）が透明性への要求条件をまとめており、EU 加盟国を中心に国際的な影響力を持つと考えられる。透明性を担保するためのチェックリストを提示し（2019 年 4 月)、実際に企業との実証実験を通じて検証する作業が行われている（第 2 版が 2019 末の予定)。

一方 IEEE では現在、IEEE P7001（transparency of autonomous systems）を検討しており、用語の定義や考え方において今後の標準化に一定の影響力を持つ可能性がある。5 種類の関係者（ユーザ、事故調査委員会他）に対し 6 種の透明性レベル（0〜5）を規定しており(数字が大きくなると必ずしも基準が厳しくなるものではない)、パイロット認証プロジェクト ECPAIS にて適合性認証の実証が進められている。

ISO では、既に述べた ISO/IEC JTC 1/SC 42 の WG 3（trustworthiness）において、文書 TR24028 にて透明性に関する用語や概念を記載している。

8.2.3　公平性（バイアス）

EU AI-HLEG において、バイアス含む AI の ELSI 問題に関するハイレベルな原則をまとめている。

また前述した IEEE P7003（algorithmic bias）においては、アルゴリズムの開発において「負のバイアス」（人種や性別など法的に禁じられている差別、法的ではない差別を共に想定）を特定し、システムの立案から運用にいたるライフサイクルでバイアスを許容範囲に抑え込む方法論を規定しており、適合性認証を実現するためのパイロットプロジェクト ECPAIS(Ethics Certification Program for Autonomous and Intelligent Systems）での実証実験が進められている。

ISO/IEC JTC 1/SC 42/WG 3（trustworthiness）においても、TR 24028（Overview of trustworthiness in Artificial Intelligence), TR 24027 (Bias in AI systems and AI aided decision making）などの文書の作成作業が現在進められている。

8.2.4 その他の倫理的品質

IEEE P7000 シリーズにてプライバシーや Nudge（行動の誘導）他の検討が、また ISO/IEC JTC1/SC42 ではガバナンス他の検討が行われている。

9　（参考）分析に関する情報

本章では、主に 1.7 節および 6 章に掲げた内部品質の特性軸を導き出すに至る分析の過程を参考情報として整理する。

本章の内容は参考（informative）である。

9.1　リスク回避性及び AI パフォーマンスに対する内部品質特性軸の分析

まず初めに、リスク回避性の外部品質軸について、品質の劣化モードの特定を行った。この品質劣化は、「機械学習要素による個々の推論結果（極端に言えば、ニューラルネットワークの結果のベクトル値）が［正しくない・思わしくない・望ましくない］」ということに起因する。そこで抽象的な機械学習要素を想定し、機械学習要素が「望ましくない」答えを出す可能性について、2 分木的かつ抽象的な故障モードの分析を、Fault Tree Analysis (FTA) に類似した考え方に基づいて行った。その分析結果を図 17 に示す。

この分析はあくまで故障モードの網羅的な原因分解を目的としたものであり、実際に判断に失敗したケースについて、これらのどれが原因であったかについては、いわゆるオラクルの視点がなければ原因が特定できないことがある点は注意が必要である。しかし一方で、対策の観点からは、例えオラクルの視点がなくても、全ての故障原因に対して対策を行ってしまえば、全ての故障モードの可能性を減らすことができると言うことができる。もちろん実際にはこれらの各項を完璧に実現することは不可能であるし、また訓練入力データに含まれる誤差や、超多次元空間データにおける数学的課題などの諸々のギャップを意図的に無視しているが、全体として品質向上プロセスの方向性としてこれらの項に着目すれば、ギャップがあっても品質の向上に十分寄与することを意図している。この図の性質 1〜7 を、故障原因から「故障しないための特性」へと反転させたものが、内部品質の 6.1〜6.8（6.4 を除く）に対応している。

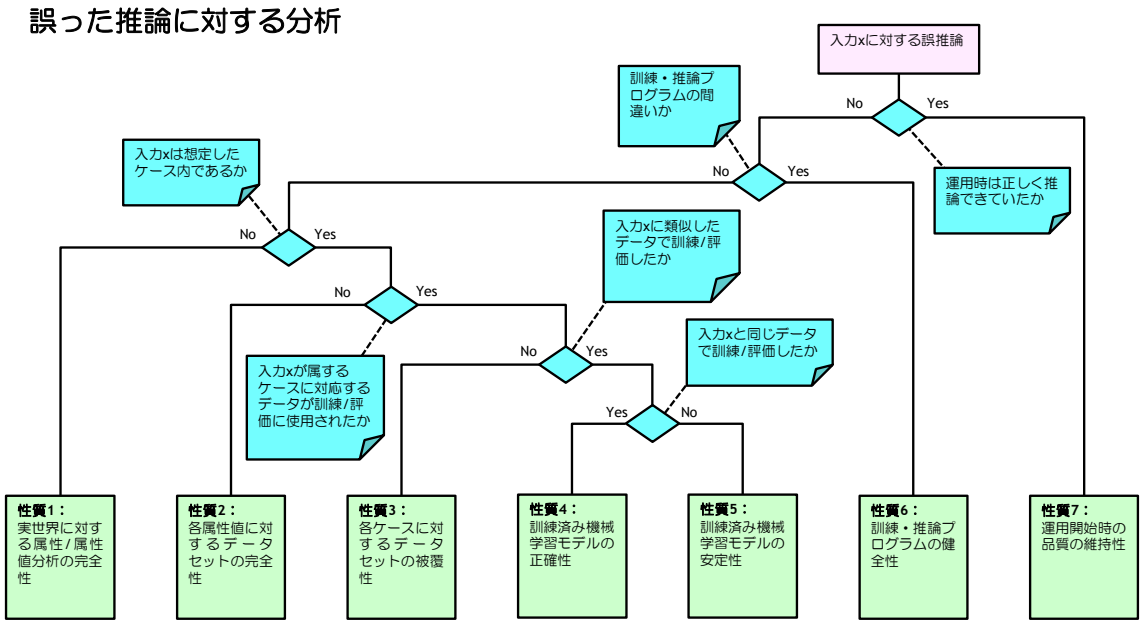

図 17　機械学習に対する故障モード解析例

9.2　AI パフォーマンスに対する品質管理軸の分析

前節の「リスク回避性」に対する内部品質管理軸を踏まえて、続いて AI パフォーマンスに対しても同様の分析を行った。

AI パフォーマンスも極論としては「機械学習要素による個々の推論結果（極端に言えば、ニューラルネットワークの結果のベクトル値）が［正しくない・思わしくない・望ましくない］」ということにその劣化が起因し、リスク回避性との差異はその個々のケースの重み付けの差異による全体の評価関数の違いと考えられる。このことから、基本的には同じ内部品質特性軸をそのまま利用できると考えられる。

ただし、リスク回避性においては、それぞれの想定リスクケースに対して対策としての学習データを十分に割り当てることに重点を置いており、学習データ全体としてのバランスを意図的に考慮していない。これは、特に発生頻度が稀ながら重篤なリスクケースに対して、均一に学習データをサンプリングしたのでは十分な学習が得られないことを想定しているが、AI パフォーマンスの観点からは、このような「意図的に偏って強化した学習データ」は全体性能の劣化を及ぼすことがあることが既に知られている。このことから、AI パフォーマンスを主に想定した内部品質軸として、6.4 に掲げる「データの均一性」を追加した。この結果が、6 章に掲げた 8 つの内部特性である。

9.3　公平性に対する品質管理軸の検討

公平性については、社会的にも学術的にもまだまだ検討の初期段階にあり、現時点では具体的な品質向上手法については十分な知見が得られていないと考えられる。

このことから現時点では、品質を確認検査する観点から、「公平性の種類」として考えられる以下の 5 項を、暫定的な品質特性軸として提示することにする。本項については、今後の研究の進展や国際規格などの動向を踏まえて随時見直しを行う。

- 結果が差別的でない
  - 判断プロセス（訓練済み学習モデル）に差別的な要素がない
  - 結果の分布が統計的に等しい
- 構築プロセスが差別的でない
  - 訓練用データに差別的な要素がない
  - 訓練用データが統計的に同等である
  - 訓練用データが分布として（公平と思われる）実世界に対応している

また、性別あるいは人種など、「何に対して公平であるべきか」の分析については、リスク回避性および AI パフォーマンスにおける要件分析を流用し、既存の内部品質の各項目に帰着させることにした。

10　図表

本章の内容は参考（informative）である。

10.1　外部品質特性と内部品質特性の対応表

表中の数値・記号は、それぞれ該当する節の「品質レベルごとの要求事項」に掲げられたチェック項目のレベルを示す。記号「+」は、追加的な検討を今後行うことを示す。太字の項目は、特に注意を要する項目を示す。

AISL	0	0.1	0.2	1	2	3	4
6.1 要求分析の十分性		1	2	3	+	+	+
6.2 データ設計の十分性		1	2	3	+	+	+
6.3 データセットの被覆性		1	2	3	+	+	+
6.4 データセットの均一性		S1	S2	S2	+	+	+
6.5 モデルの正確性		1	2	3	+	+	+
6.6 モデルの安全性		1	2	3	+	+	+
6.7 プログラムの健全性		1	2	3	+	+	+
6.8 初期品質の維持性		1	2	3	+	+	+

AIPL・AIFL	PL 0	PL 1	PL 2	FL 0	FL 1	FL 2
6.1 要求分析の十分性		1	2		1	2
6.2 データ設計の十分性		1	2		1	2
6.3 データセットの被覆性		1	2		1	2
6.4 データセットの均一性		E1	E2		**E2**	E2
6.5 モデルの正確性		1	2		1	2
6.6 モデルの安全性		1	2		1	2
6.7 プログラムの健全性		1	2		1	2
6.8 初期品質の維持性		1	2		**2**	**3**

産総研 人工知能研究センター・サイバーフィジカルセキュリティ研究センター

AI 品質マネジメント検討委員会

メンバリスト

中島 震	国立情報学研究所（委員長）
妹尾　義樹	産総研（副委員長）
岡本　球夫	パナソニック
土屋　哲	富士通
小林　健一	富士通研究所
佐藤　直人	日立製作所
小川　秀人	日立製作所
福島　真太朗	トヨタ自動車
本橋　洋介	日本電気
江川　尚志	日本電気
大岩　寛	産総研
鈴木　知道	東京理科大学
桑島　洋	デンソー
中島　裕生	テクマトリックス
浜谷　千波	アドソル日進

（敬称略）

機械学習品質マネジメントガイドライン　執筆者リスト

- 全体構成・執筆・編集　大岩 寛（産業技術総合研究所）
- 5.2 節　浜谷 千波（アドソル日進）
- 7.5 節　中島 震（国立情報学研究所）
- 7.5.2 節　磯部 祥尚（産業技術総合研究所）
- 7.8 節　小林　健一，大川　佳寛（富士通研究所）
- 8.2 節　江川 尚志（日本電気）の寄稿による
- 内容検討・監修　AI 品質マネジメント検討委員会 各委員
  AI 品質マネジメント検討委員会
  ガイドライン詳細検討タスクフォース メンバー

ガイドライン詳細検討タスクフォース　メンバーリスト

大岩　寛	産総研
中島　震	国立情報学研究所
岡本　球夫	パナソニック
土屋　哲	富士通
小林　健一	富士通研究所
福島　真太朗	トヨタ自動車
本橋　洋介	日本電気
江川　尚志	日本電気
桑島　洋	デンソー
中島　裕生	テクマトリックス
浜谷　千波	アドソル日進
妹尾　義樹	産総研
磯部　祥尚	産総研
川本　裕輔	産総研

（敬称略）