

Azure Machine Learning Design Principle

機械学習の開発・運用管理の基本ガイドライン

2021-June v1.1 released

日本マイクロソフト株式会社

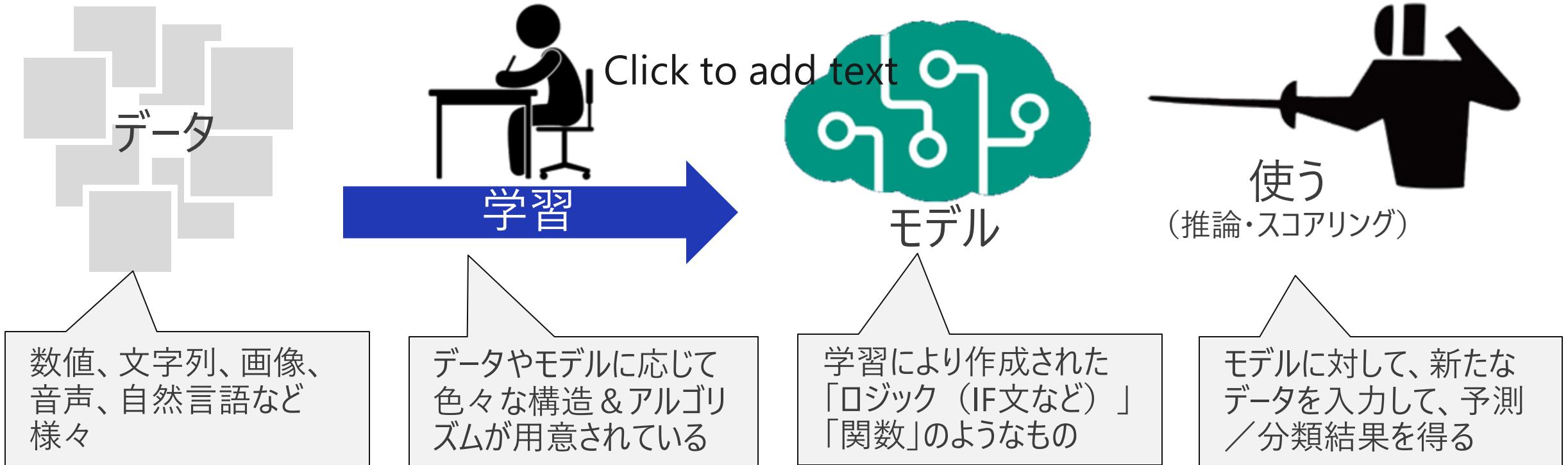
- Machine Learning Introduction
- Workspace
- MLOps
- Security
- Compute
- Deploy & Inference
- Responsible AI

Machine Learning Introduction



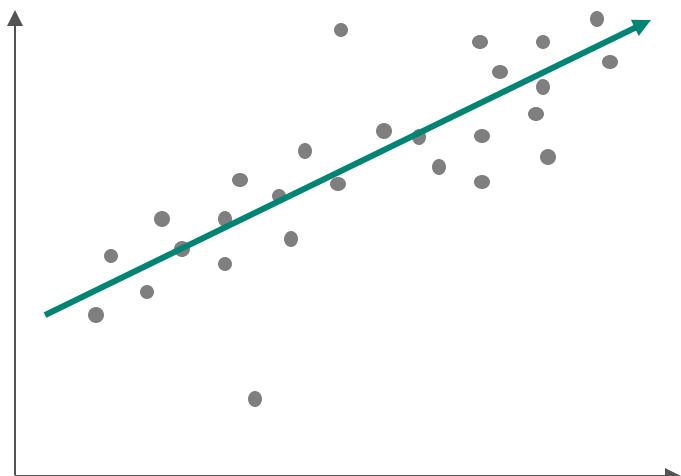
機械学習とは？

「データ」を 「学習」して 「モデル」を作り
それを使って**「回帰」・「分類」・「クラスタリング」**などを行うこと



機械学習のタスク

回帰



$$F(x) = mx + b$$

m と b をデータから学習

分類



イヌ
0.78

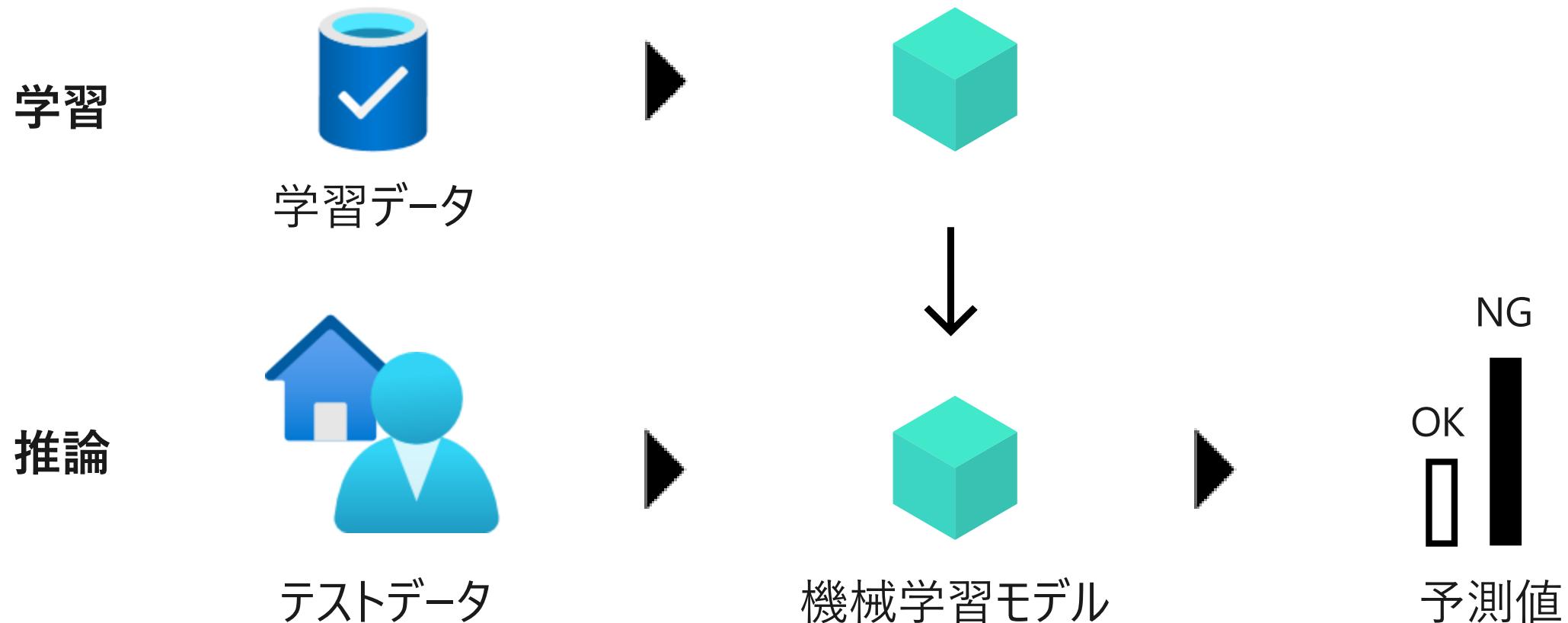
ネコ
0.04

トラ
0.001

クラスタリング

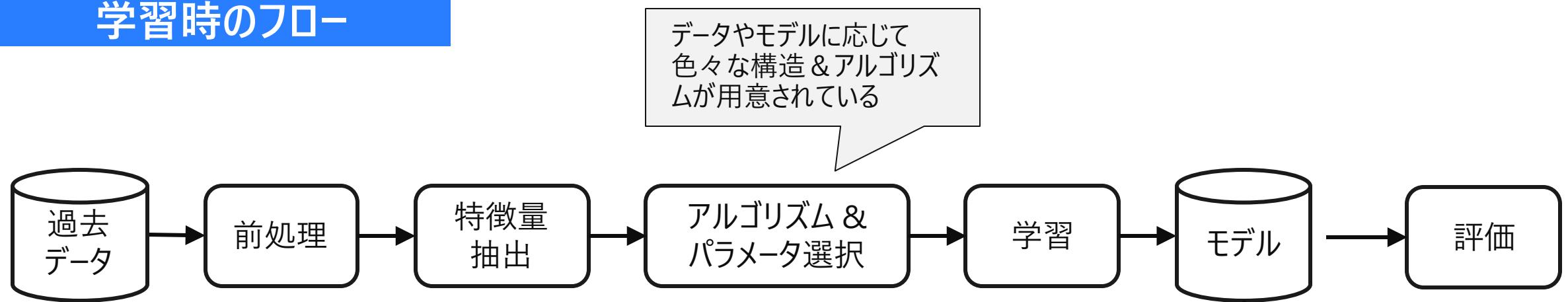
- 顧客のセグメント分け
- 記事の分類
- 類似した項目の検出

機械学習のプロセス (概要)

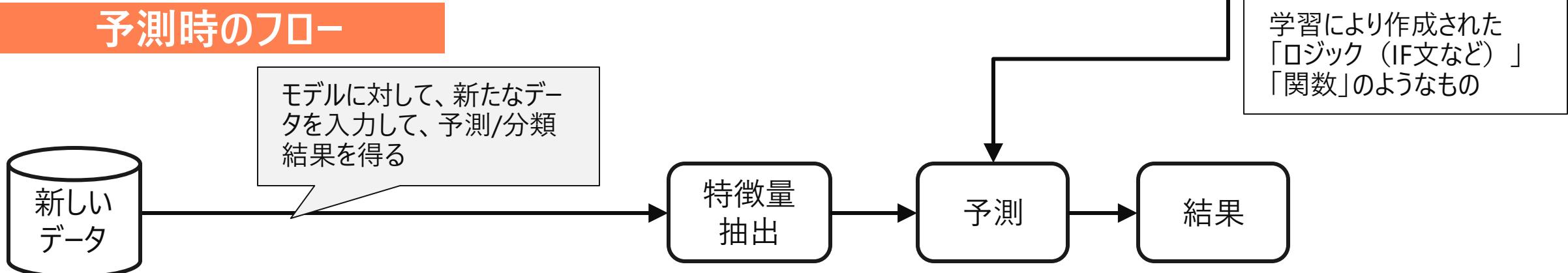


機械学習のプロセス (詳細)

学習時のフロー



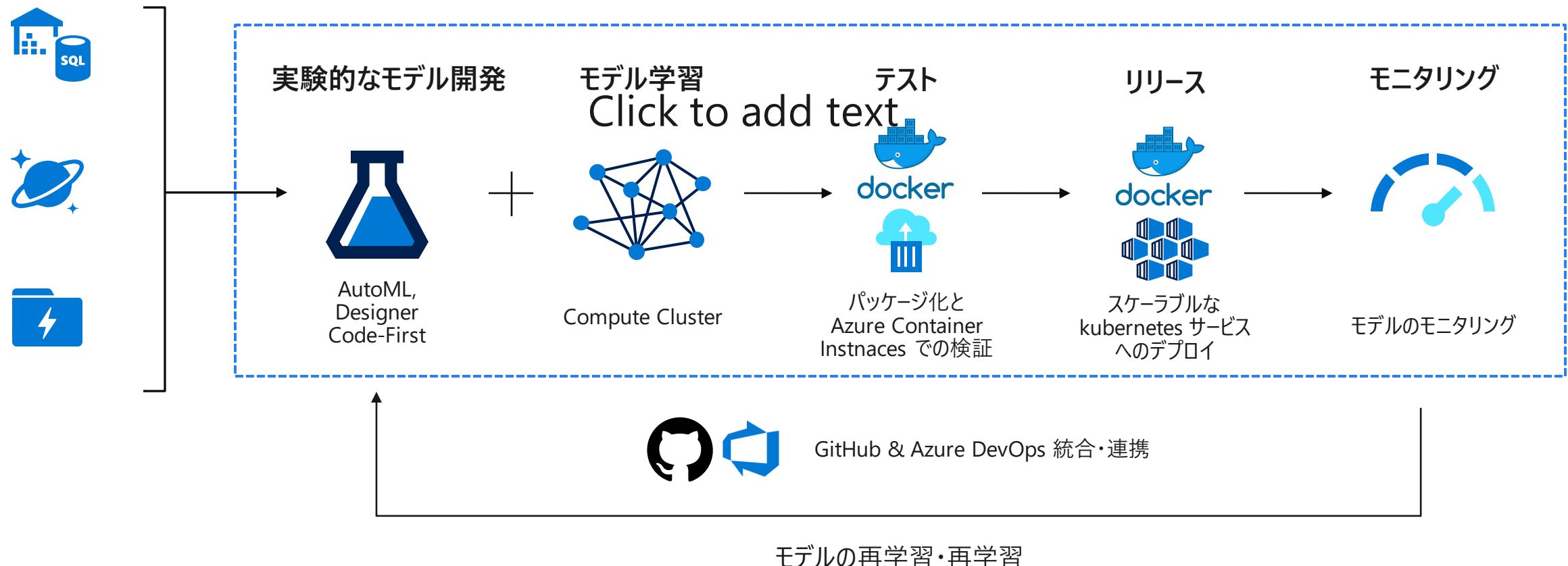
予測時のフロー



Azure Machine Learning ライフサイクル

Azure Machine Learning における機械学習の開発・運用管理のライフサイクル

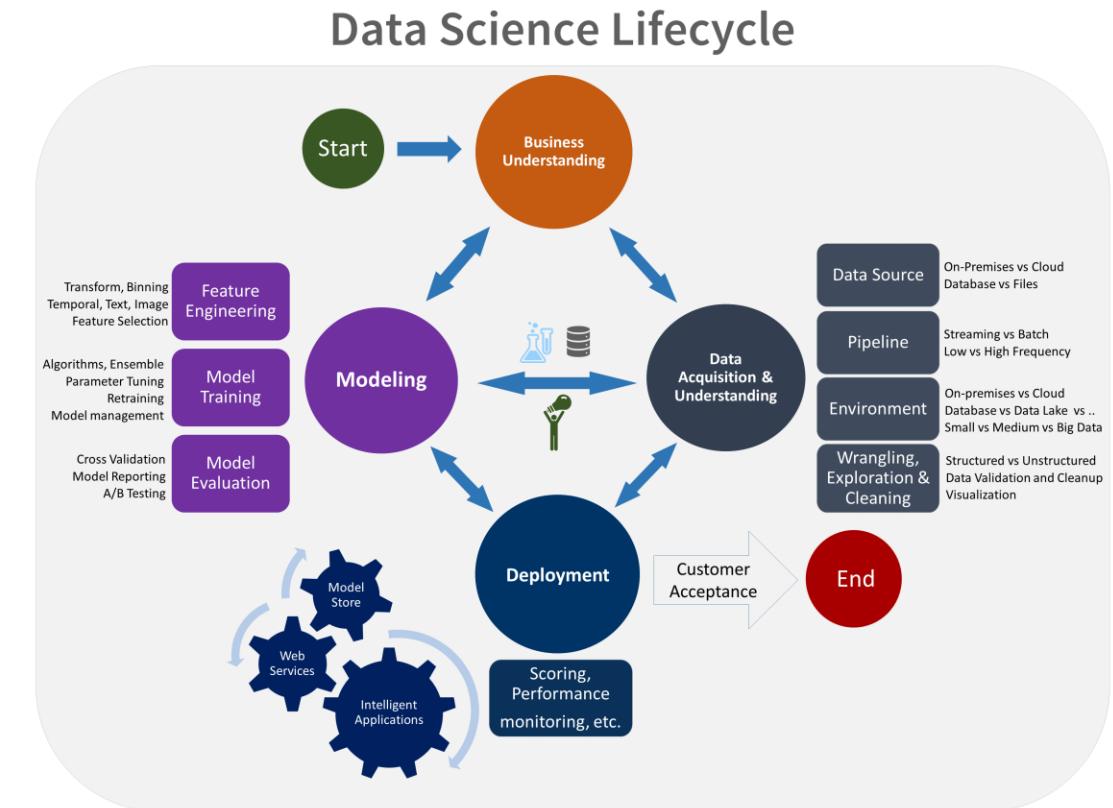
機械学習のライフサイクル



Team Data Science Process (TDSP)

Team Data Science Process (TDSP) には、データサイエンス プロジェクトの開発を体系化するライフサイクルが用意されています。

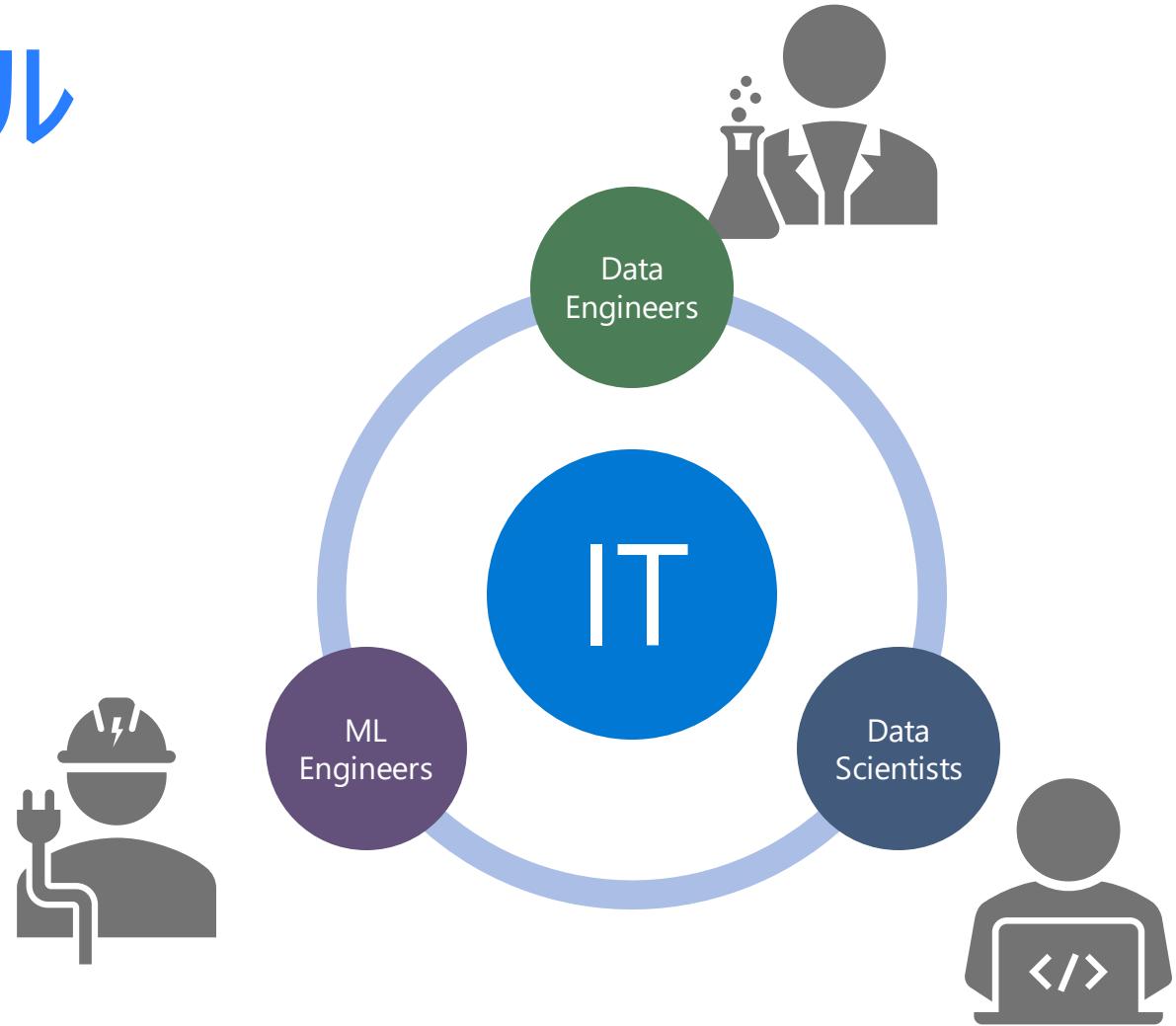
ライフサイクルは、プロジェクトで通常（多くの場合に繰り返し）実行される主要なステージのアウトラインを示します。



[What is the Team Data Science Process? | Microsoft Docs](#)

機械学習の関わるロール

- Data Scientist
- Data Engineer
- MLOps Engineer



※ 各ロールが明確に定義されることはありますが、
|Data Scientist がこの 3 つのロールをカバーしていることもあります。

ロールの定義

Data Scientist



- ✓ 機械学習・統計解析などのテクノロジーを用いてビジネス課題を対応する
- ✓ モデル学習・検証のためのデータ加工を行う
- ✓ モデルを学習し改善する
- ✓ 再利用・再現可能なパイプラインを構築する

Data Engineer



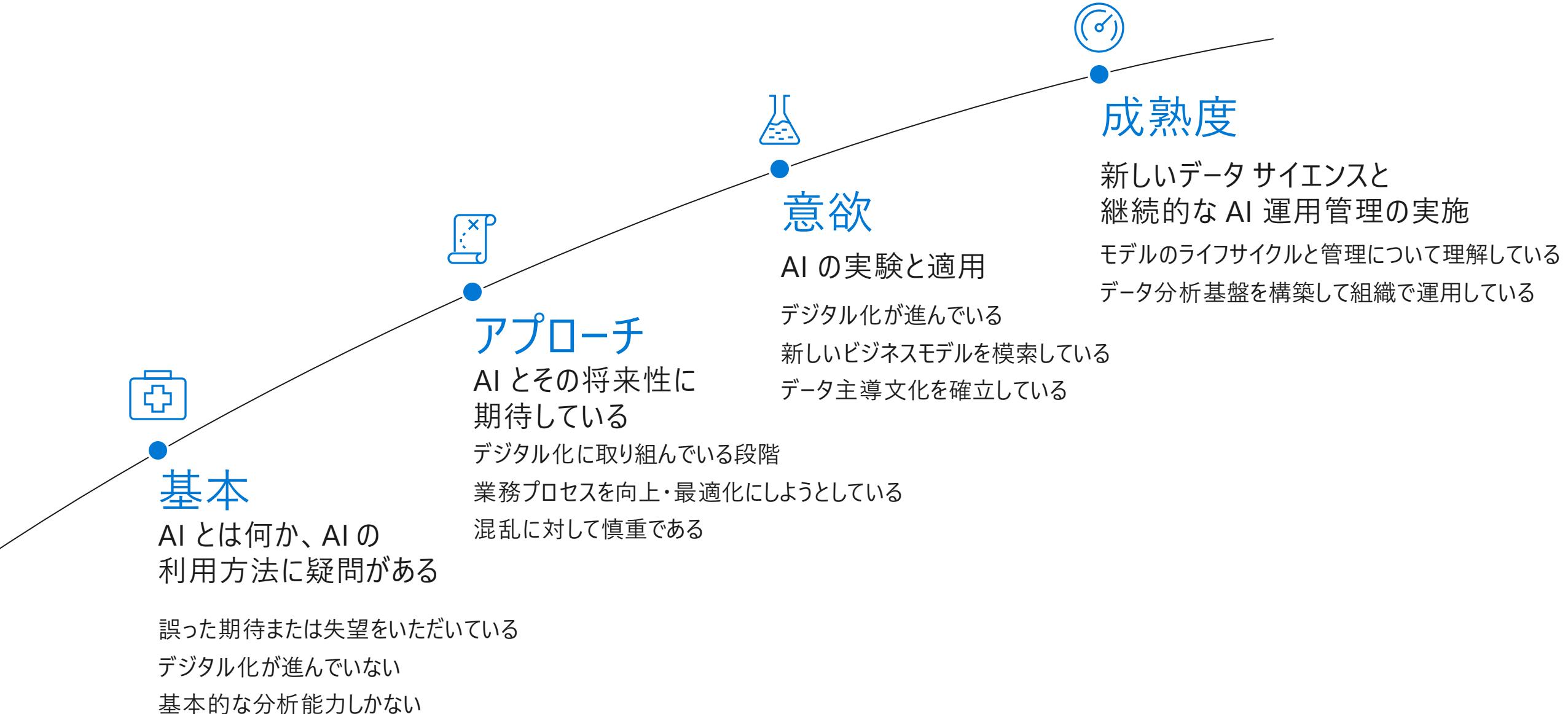
- ✓ データソースからデータを抽出し、データサイエンティストが利用できるように準備する
- ✓ データ操作・理解できるようにクレンジングを行う
- ✓ 最新データを取得するデータパイプラインを構築する

MLOps Engineer



- ✓ 機械学習のライフサイクルを自動化するパイプラインを構築する
- ✓ 推論環境におけるモデルのエラーやデータドリフトをモニタリングする
- ✓ 推論結果を連携しているアプリケーションやエンドユーザーに返す

企業・組織における成熟度ロードマップ

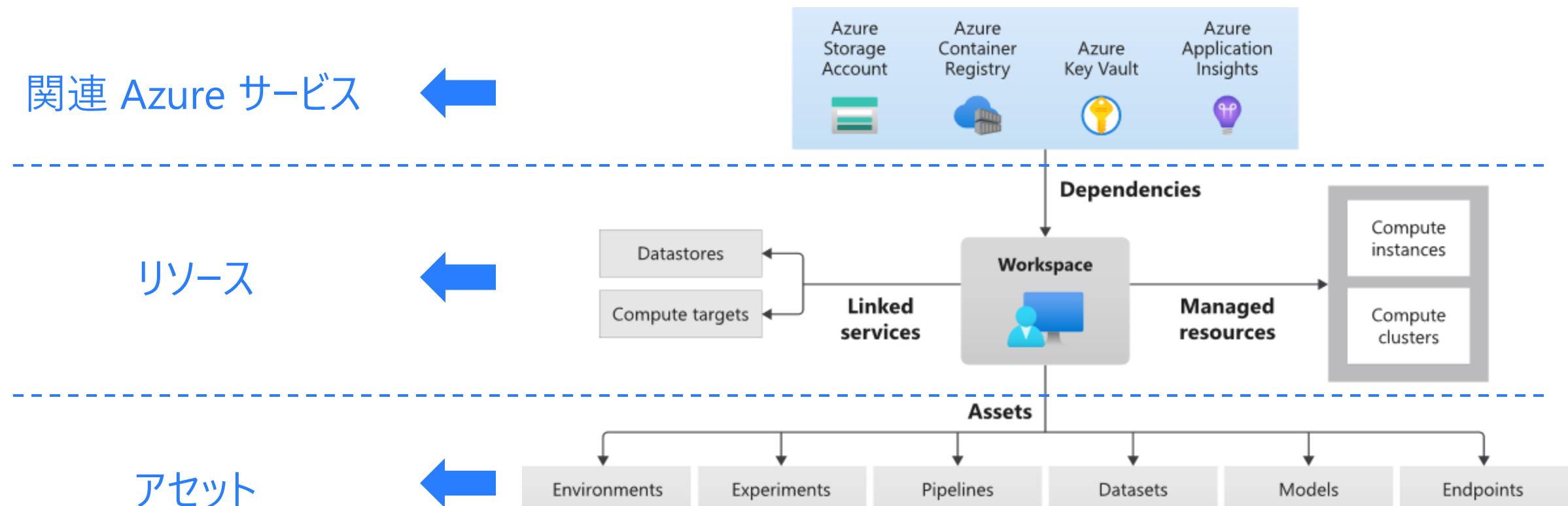


Workspace



Azure Machine Learning Workspace とは？

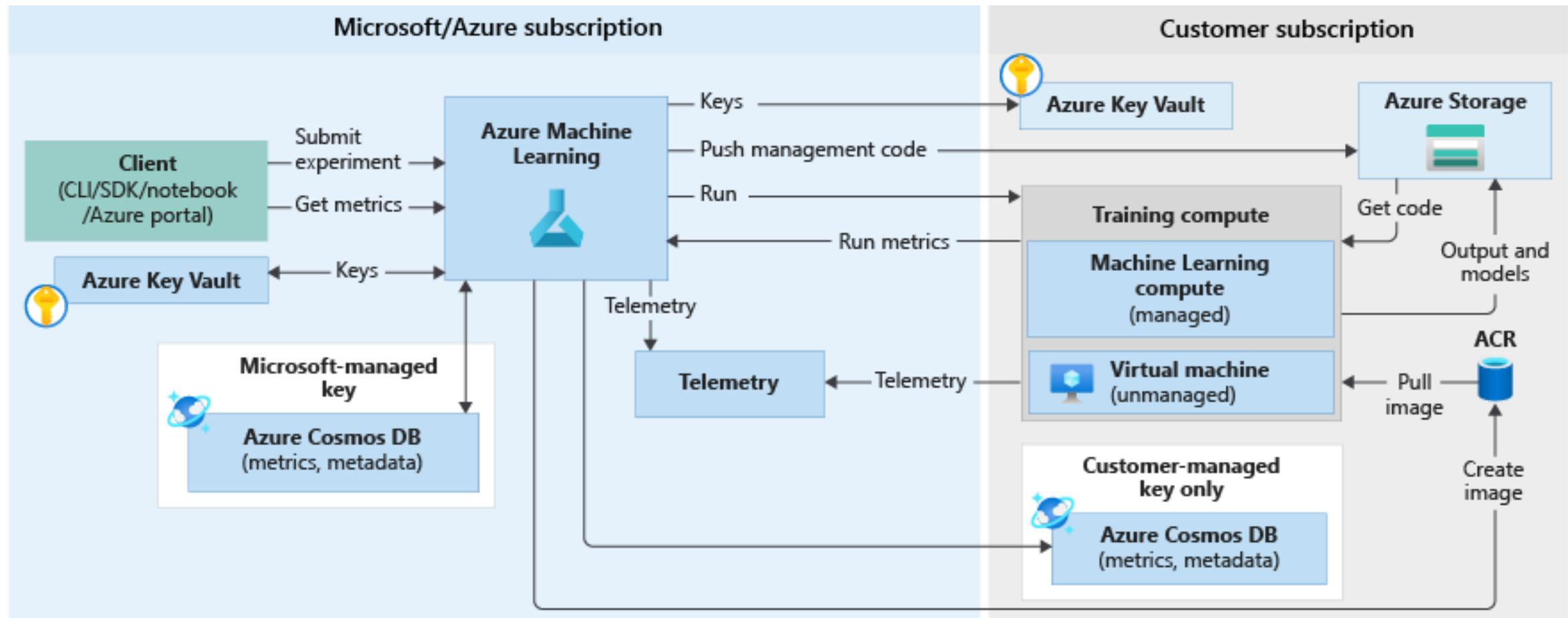
Azure Machine Learning Workspae は、最上位のリソースであり、あらゆるリソースを操作・運用管理するための一元的な環境を提供します。



Azure Machine Learning 構成の理解

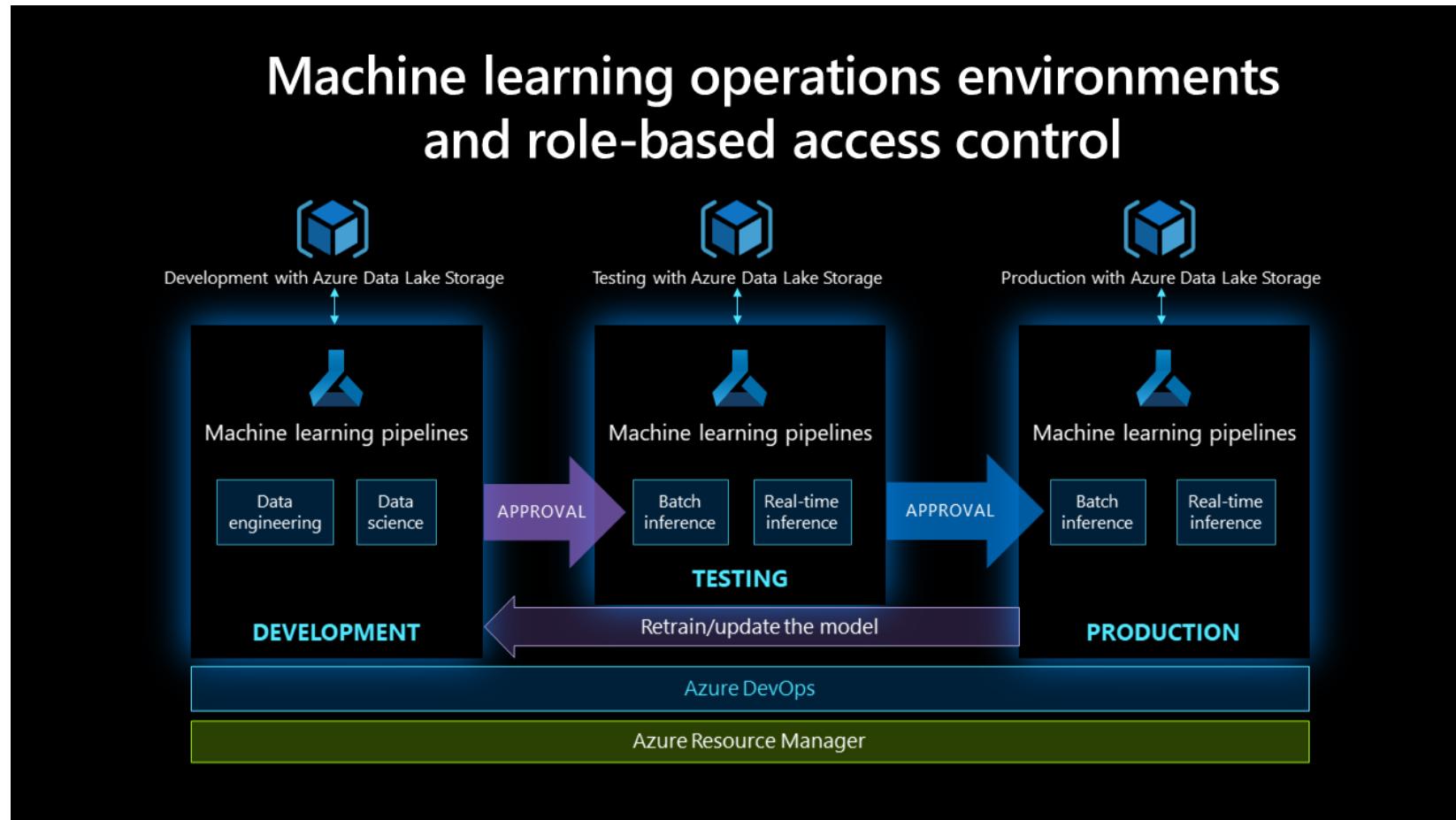
Microsoft 管理の Azure 環境

お客様管理の Azure 環境



ワークフローに応じて Workspace を構築する

開発環境、テスト環境、本番環境とワークフローに応じて Workspace を構築することを推奨。

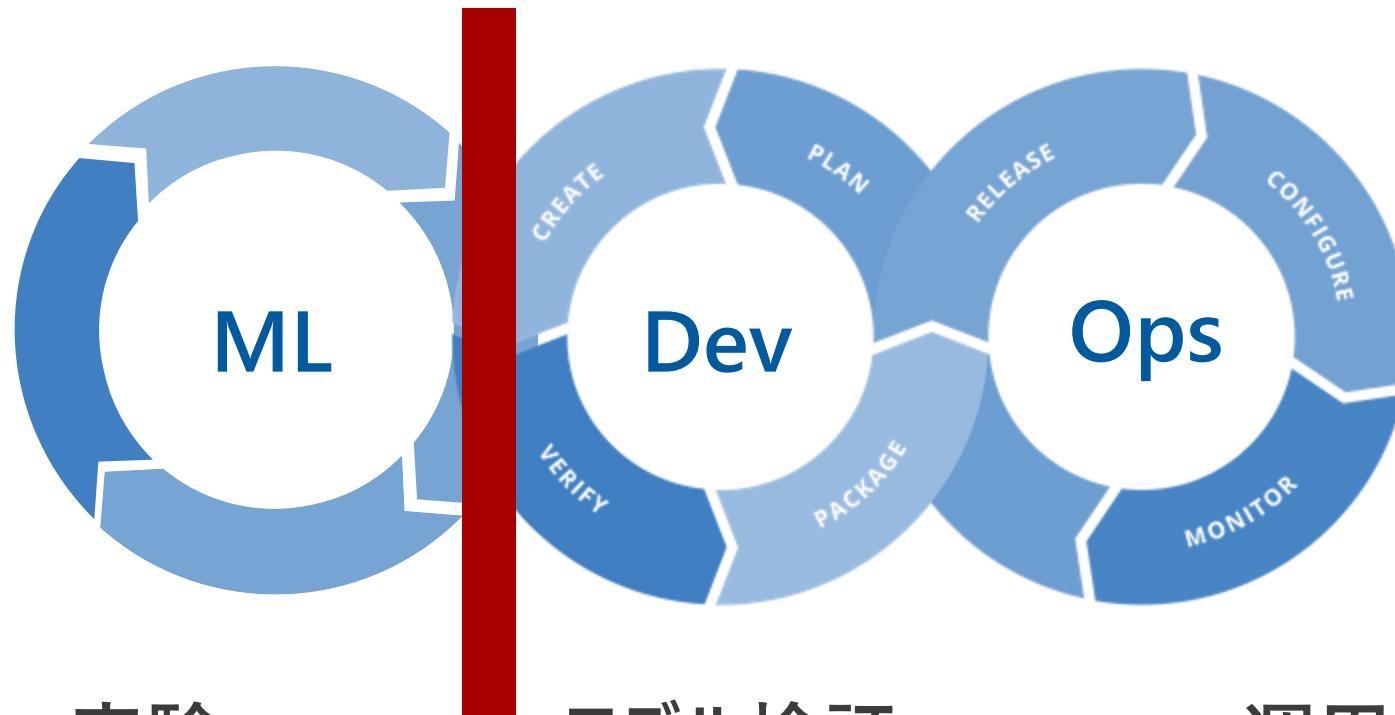


MLOps



Machine Learning Operations = MLOps

機械学習ライフサイクルの自動化と加速



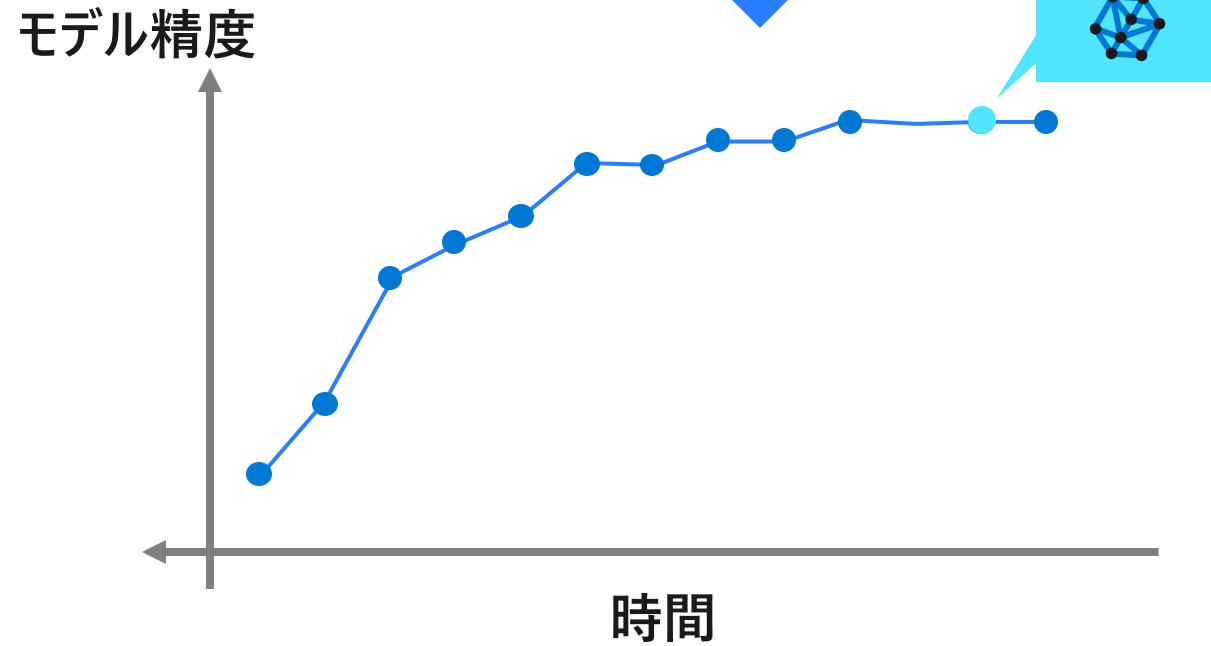
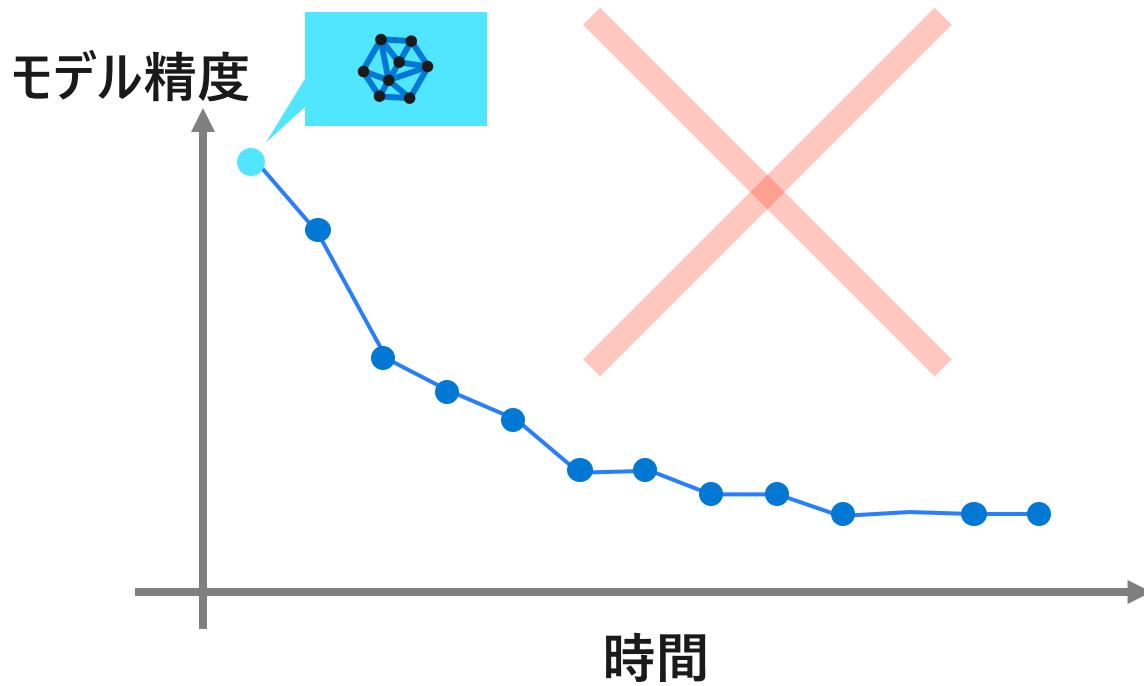
ビジネス要件の確認
データ準備
モデル学習 (PoC)

Code & Data テスト
モデル学習 (再現)
モデルのパッケージ化

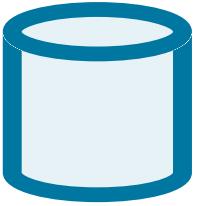
デプロイ (Cloud, Edge)
データ、モデル、システムの監視
モデルの再学習・再作成

モデル精度変化

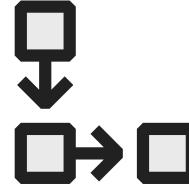
通常モデルの精度は劣化するため、
継続的な改善をしていく必要がある。



DevOps との違い



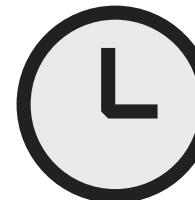
Data/Model のバージョン管理とコードのバージョン管理は異なり、データのスキーマの変化やコンテンツの変更を管理する必要がある。



コードやデータの変更があるたびにデジタル監査証跡の要件が変更される。



機械学習モデルの再利用は、一般的なソフトウェアの再利用とは異なります。入力データやシナリオに基づいてモデルを調整する必要がある。



機械学習モデルは時間の経過とともに減衰する傾向にあり、オンデマンドでモデルを再学習し本番環境に適用させていく必要がある。

用語の整理

データのバージョン管理

- ✓ スキーマと実際のデータは時間の経過とともに変わる可能性があるため、コードのバージョン管理とデータセットのバージョン管理が必要です。これにより、データを再現し、データを他のチームメンバーにも見えるようにし、ユースケースを監査するのに役立ちます。

モデルの追跡

- ✓ モデルのファイルは、多くの場合、ストレージ、バージョン管理、タグ付け機能が識別されるモデルレジストリに格納されます。これらのレジストリは、ソースコード、そのパラメーター、およびモデルのトレーニングに使用された対応するデータを識別する必要があり、そのすべてがモデルが作成された場所を示します。

デジタル監査証跡

- ✓ モデルのファイルは、多くの場合、ストレージ、バージョン管理、タグ付け機能が識別されるモデルレジストリに格納されます。これらのレジストリは、ソースコード、そのパラメーター、およびモデルのトレーニングに使用された対応するデータを識別する必要があり、そのすべてがモデルが作成された場所を示します。

汎化

- ✓ モデルは再利用のためのコードとは異なり、入力データやシナリオに基づいてモデルを調整する必要があります。新しいデータを新しいシナリオで使用するために、モデルの微調整が必要になる場合があります。

モデルの再学習

- ✓ モデルのパフォーマンスは時間の経過とともに低下する可能性があるため、モデルの有用性を維持するためには、モデルを再トレーニングすることが重要です。

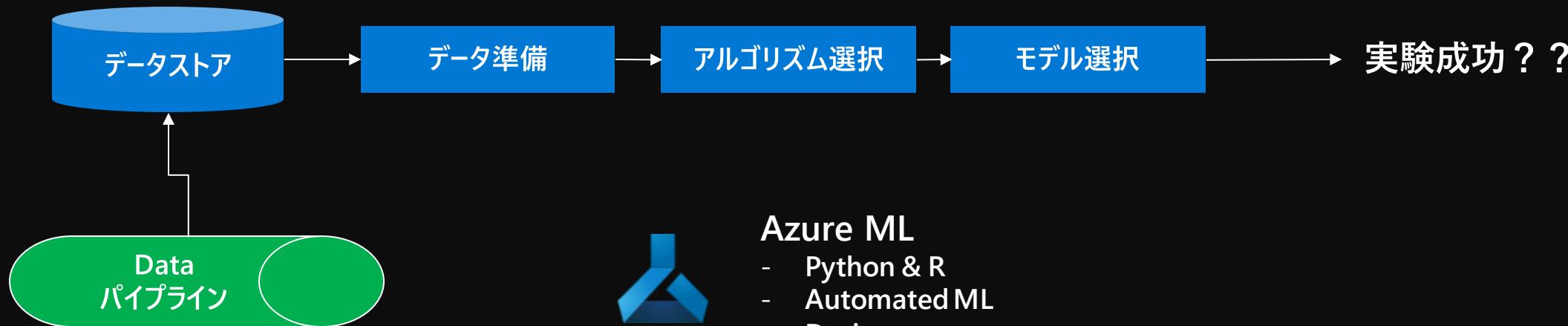
Maturity Level	People	Model Creation	Model Release	Application Integration	Technology
Level 1 - No MLOps	<ul style="list-style-type: none"> Data Scientists - silo'd, not in regular comms with larger team Data Engineers - silo'd (if exists), not in regular comms with larger team Software Engineers - Silo'd, receive model "over the wall" 	<ul style="list-style-type: none"> Data pipeline gathers data automatically Compute may or may not be managed Experiments are not predictably tracked End result may be a single file manually handed off (model), with inputs/outputs 	<ul style="list-style-type: none"> Manual process Scoring script may be manually created well after experiments, likely version controlled Is handed off to Software Engineers 	<ul style="list-style-type: none"> Basic integration tests exist for the model Heavily reliant on Data Scientist expertise to implement model Releases are automated Application code has unit tests 	<ul style="list-style-type: none"> Automated Builds Automated Tests for Application code Manual model training No centralized tracking of model performance
Level 2 - Automated Training	<ul style="list-style-type: none"> Data Scientists - Working directly with Data Engineers to convert experimentation code into repeatable scripts/jobs Data Engineers - Working with Data Scientists Software Engineers - Silo'd, receive model "over the wall" 	<ul style="list-style-type: none"> Data pipeline gathers data automatically Compute is managed Experiment results are tracked Both training code and resulting models are version controlled 	<ul style="list-style-type: none"> Manual Release Scoring Script is version controlled with tests Release is managed by Software engineering team 	<ul style="list-style-type: none"> Basic integration tests exist for the model Heavily reliant on Data Scientist expertise to implement model Application code has unit tests 	<ul style="list-style-type: none"> Automated Builds Automated Tests for Application code Automated model training Centralized tracking of model training performance Model Management
Level 3 - Automated Model Deployment	<ul style="list-style-type: none"> Data Scientists - Working directly with Data Engineers to convert experimentation code into repeatable scripts/jobs Data Engineers - Working with Data Scientists and Software Engineers to manage inputs/outputs Software Engineers - Working with Data Engineers to automate model integration into application code 	<ul style="list-style-type: none"> Data pipeline gathers data automatically Compute is managed Experiment results are tracked Both training code and resulting models are version controlled 	<ul style="list-style-type: none"> Automatic Release Scoring Script is version controlled with tests Release is managed by CI/CD pipeline 	<ul style="list-style-type: none"> Unit and Integration tests for each model release Less reliant on Data Scientist expertise to implement model Application code has unit/integration tests 	<ul style="list-style-type: none"> Automated Builds Integrated A/B testing of model performance for deployment Automated Tests for All code Automated model training Centralized tracking of model training performance Model Management
Level 4 - Automated Retraining (full MLOps)	<ul style="list-style-type: none"> Data Scientists - Working directly with Data Engineers to convert experimentation code into repeatable scripts/jobs. Working with Software Engineers to identify markers for retraining Data Engineers - Working with Data Scientists and Software Engineers to manage inputs/outputs Software Engineers - Working with Data Engineers to automate model integration into application code. Implementing metrics gathering post-deployment 	<ul style="list-style-type: none"> Data pipeline gathers data automatically Retraining triggered automatically based on production metrics Compute is managed Experiment results are tracked Both training code and resulting models are version controlled 	<ul style="list-style-type: none"> Automatic Release Scoring Script is version controlled with tests Release is managed by CI/CD pipeline 	<ul style="list-style-type: none"> Unit and Integration tests for each model release Less reliant on Data Scientist expertise to implement model Application code has unit/integration tests 	<ul style="list-style-type: none"> Automated Builds Integrated A/B testing of model performance for deployment Automated Tests for All code Automated model training and testing Centralized tracking of model training performance Model Management Verbose, centralized metrics from deployed model

Level 1 – 機械学習の実験

対話的、探索的にモデルを構築し、方針を定めていく



Data Scientist



Azure ML

- Python & R
- Automated ML
- Designer

Level 2 – 再現可能なモデル学習

コード、データ、モデルなどの資産管理 & 再現性の確保



Data Scientist



紐付けを行う ...

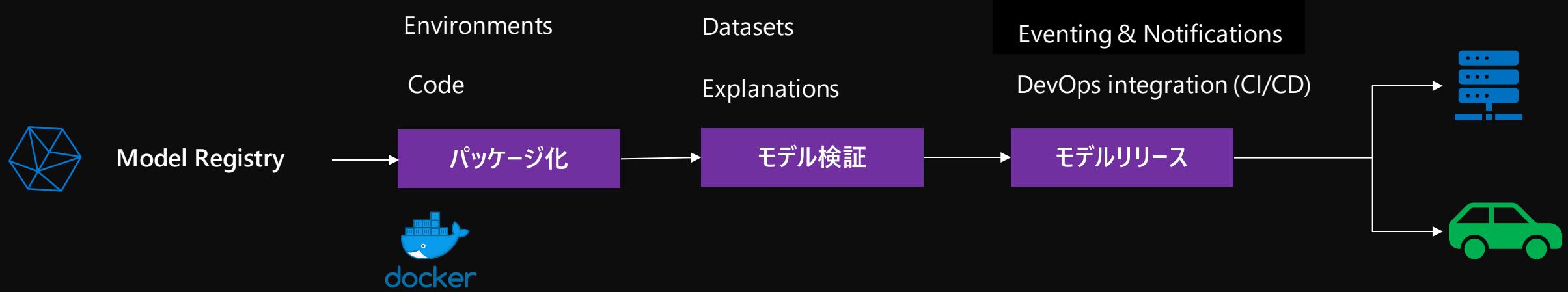
- データセット
- 環境情報
- コード
- ログ / メトリック
- 出力ファイル



実験管理 (Experiment)

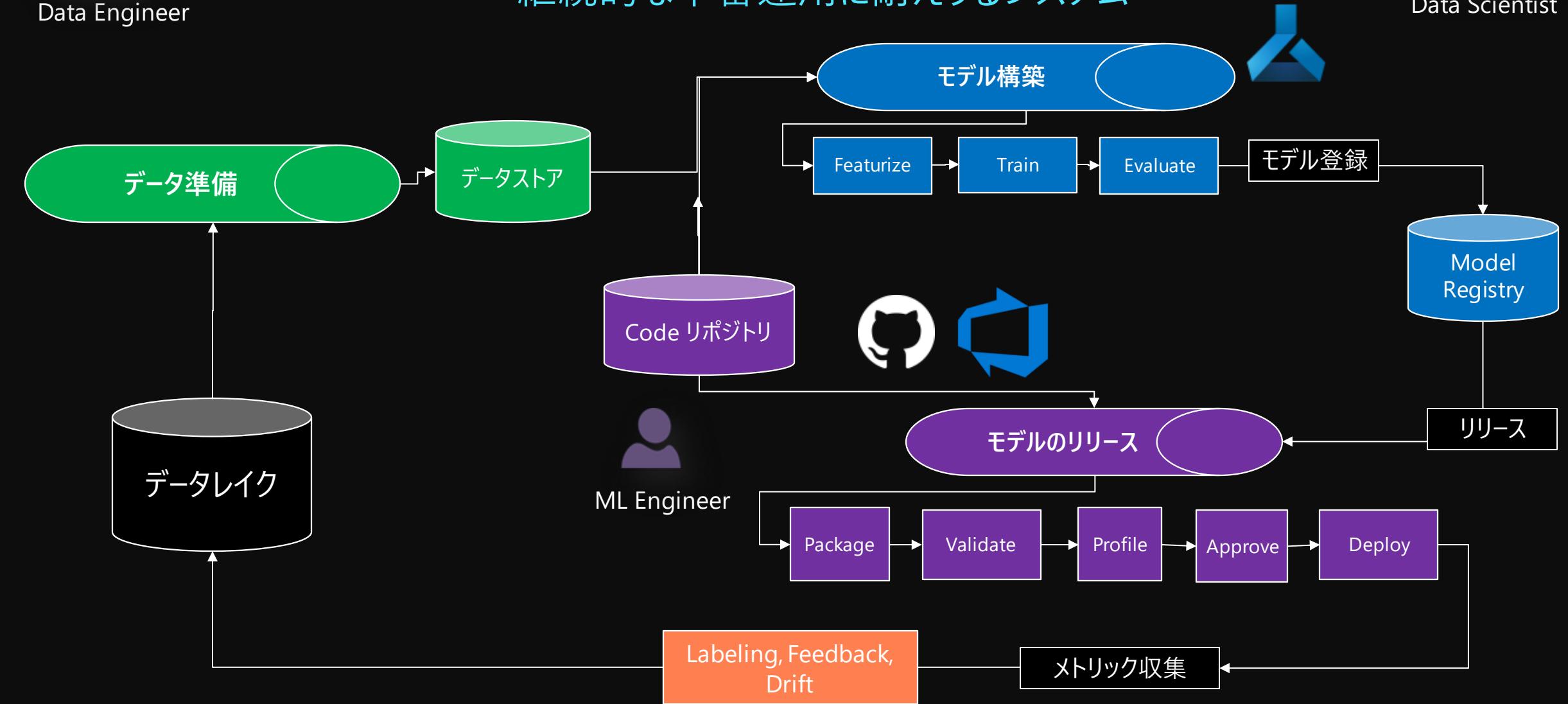
Level 3 – モデルの管理と運用

パッケージ化、検証、デプロイ



Level 4 : E2E の機械学習システム

継続的な本番運用に耐えうるシステム



5つのプラクティス

1 再現可能なモデル開発

Github / Azure DevOps でのコード管理と
Azure Machine Learning Pipeline の構築

2 自動化パイプライン

Azure Machine Learning Pipeline と Github Actions /
Azure Pipelines によるライフサイクルの自動化

3 監査証跡の取得

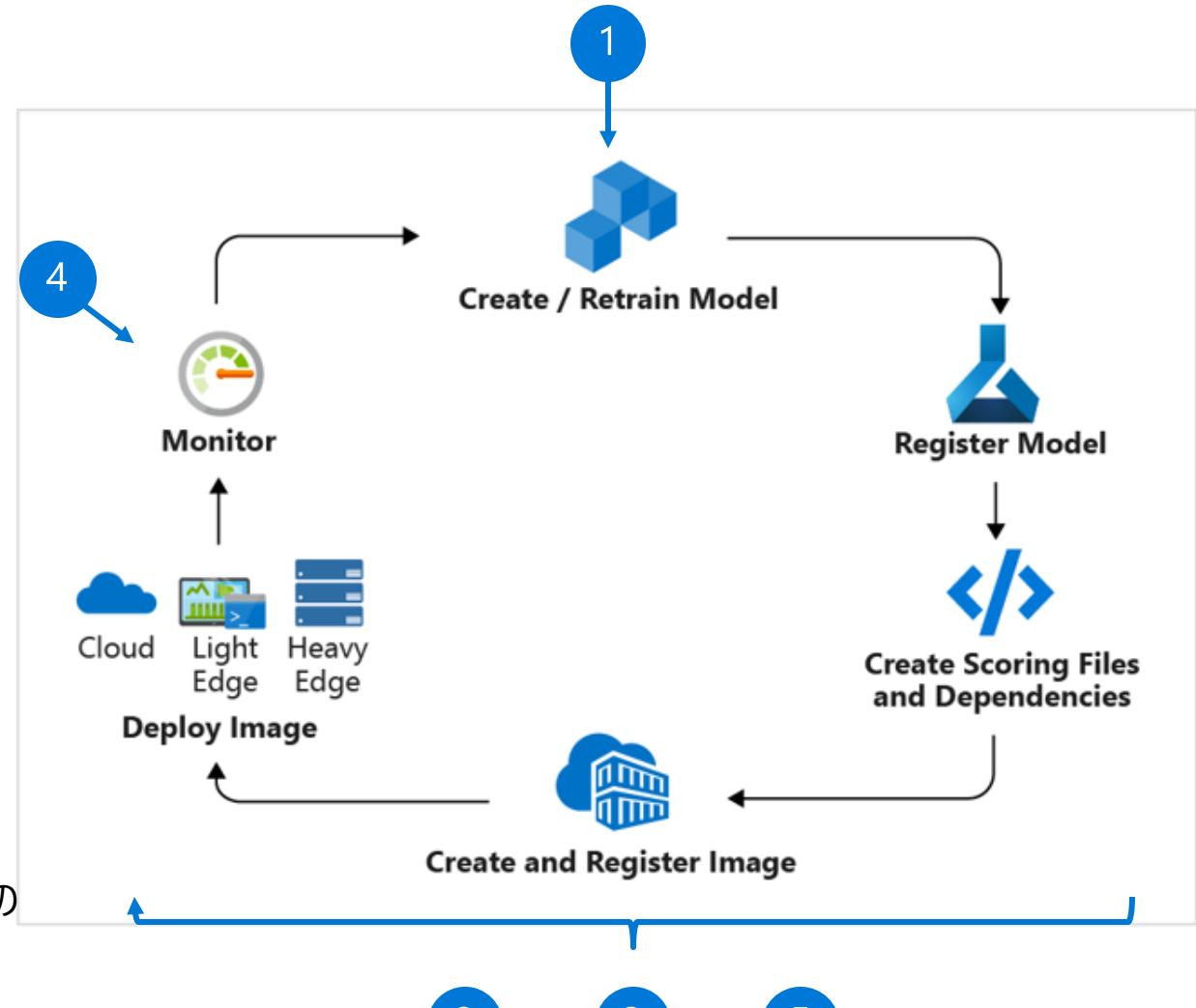
Azure Machine Learning Experiment / MLflow による
実験メトリック、ログ、データ、実行環境、モデル、エンドポイント
のアセット管理と関係性の記録

4 モニタリング

推論環境でのデータ収集とドリフト検知、システムパフォーマンスの
監視、ユーザ動線・利用率の変化 etc

5 通知、アラート

Azure Event Grid によるアラート、トリガー実行、他システム連携



①再現可能なモデル開発

- ・機械学習とプログラミングの違い
 - ・機械学習はデータに大きく依存する
- ・注意ポイント
 - ・Data
 - ・データの傾向が時間の経過によって変化
 - ・アプリの仕様変更などによるデータ性質の変化
 - ・Output
 - ・ユーザ要件に望ましい Output が変化
 - ・ノイズによって正確な Output のインプットが困難
 - ・Computer
 - ・特に Python パッケージのバージョン管理は困難
 - ・Program
 - ・Program (=モデル) からだけでは同じものは作れない

Traditional Programming



Machine Learning

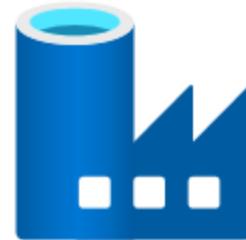


②自動化パイプライン

機械学習ライフサイクルの自動化のメリット

- ・工数削減
- ・再現性の確保 (人依存からの脱却)

データオーケストレーション
データ前処理



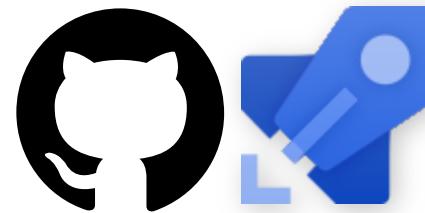
Azure Data Factory
Pipeline

モデル学習
バッチ推論



Azure Machine Learning
Pipeline

機械学習ワークフローの実行
(CI/CD)



GitHub Actions
Azure Pipelines

③監査証跡の取得



ソースコード

- ・ コードにおける変化履歴や構成変化をトラック



データセット

- ・ データのバージョン管理、スキーマ管理



モデル

- ・ モデルのバージョン管理
- ・ 学習コードや推論コード、環境情報との紐付け



実験トラッキング

- ・ 実験のメトリック管理
- ・ 関連アセットの紐付け(コード、データ、モデル etc)

Author
 Notebooks
 Automated ML
 Designer
Assets
 Datasets
 Experiments
 Pipelines
 Models
 Endpoints
Manage
 Compute
 Datastores
 Data Labeling
 Linked Services

アセット

- **Datasets** - データの登録と管理
- **Experiments** - 実験記録
- **Pipelines** - 学習・推論のパイプライン
- **Models** - モデル管理
- **Endpoints**
 - **real-time** : リアルタイム型の推論環境
 - **batch** : バッチ型の推論環境
- **Environment**

環境

- **Compute** - 学習・推論の計算環境
- **Datastores** - データソースの設定

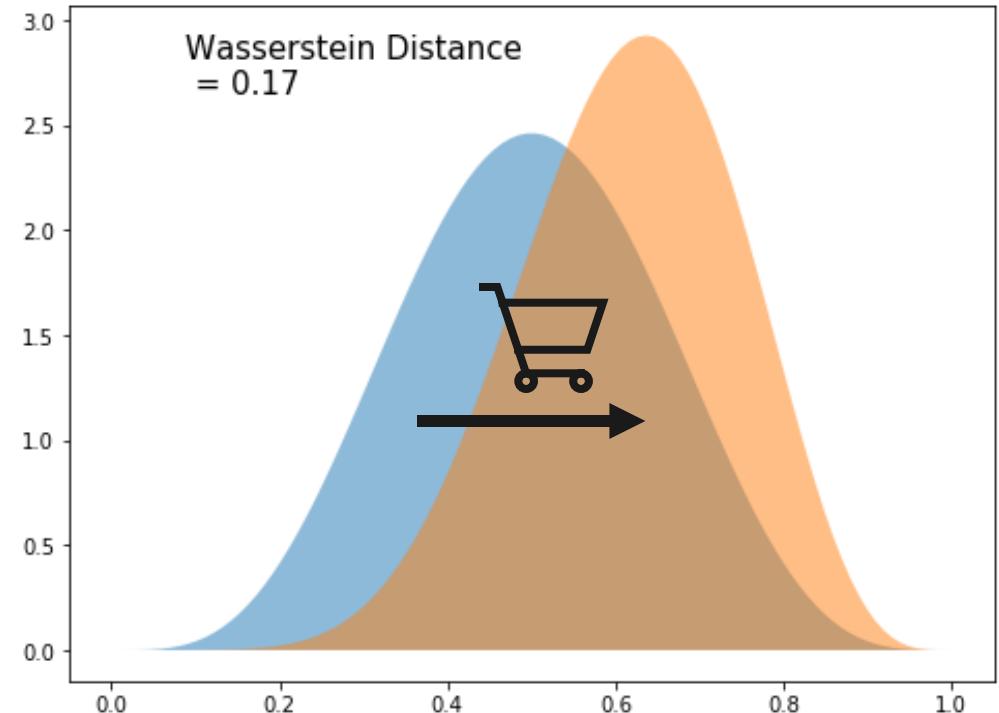
④モニタリング

- ・パフォーマンスが劣化する原因

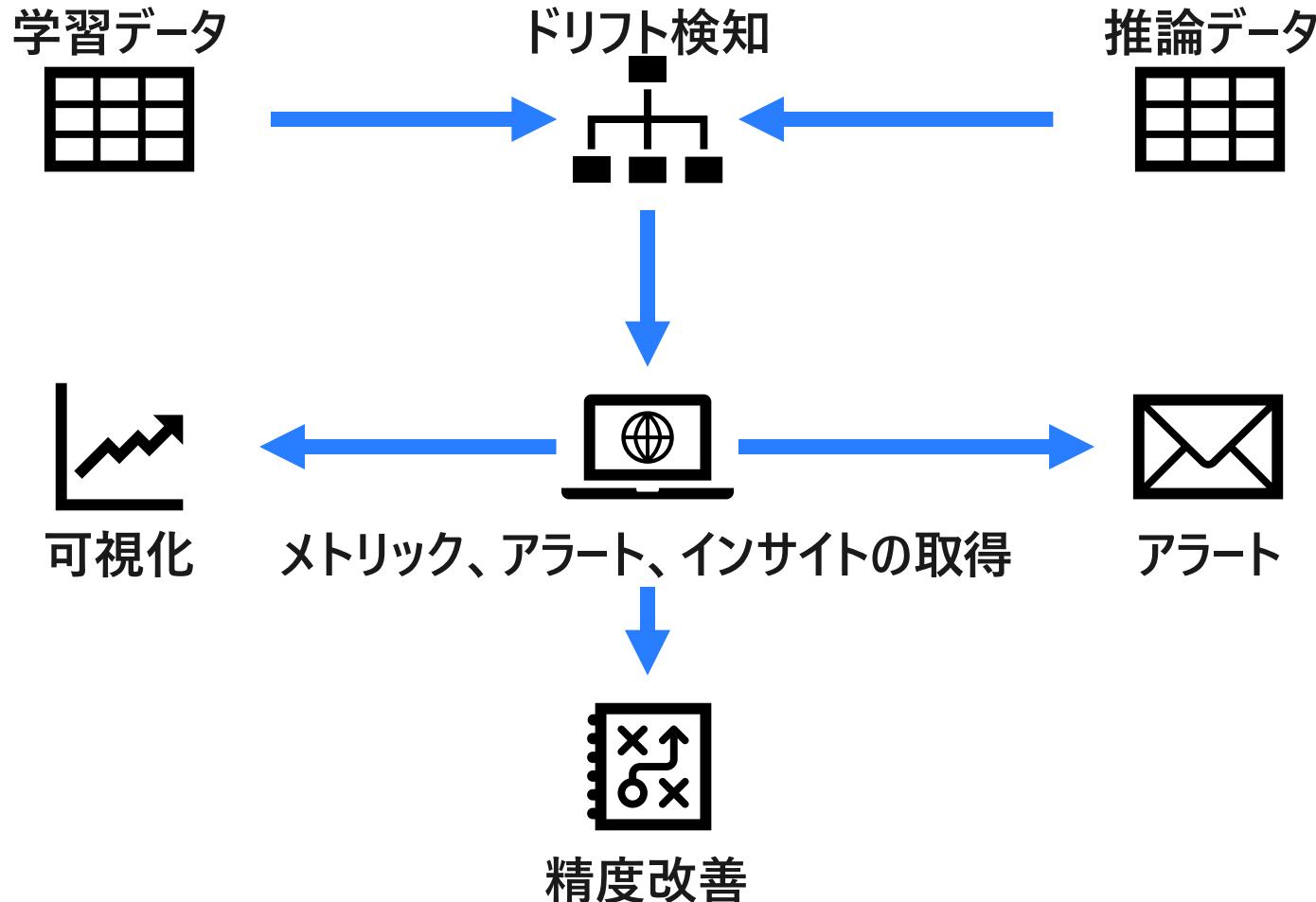
- ・最新バージョンのモデルの精度が悪い
- ・システムによるデータ仕様の変更
- ・人間の行動パターンの変化
- ・アプリケーションへの期待値の変化

- ・モニタリングする

- ・データが取れるとき
 - ・予測結果と正解データを比較して、精度を算出する
 - ・モデルへのインプットデータの分布などの特徴をみて、母集団の変化を検知する
- ・データが取れないとき
 - ・アプリケーションにおけるユーザの動作の変化を検知する
 - ・利用ユーザにインタビューする



Azure ML Data Drift の仕組み



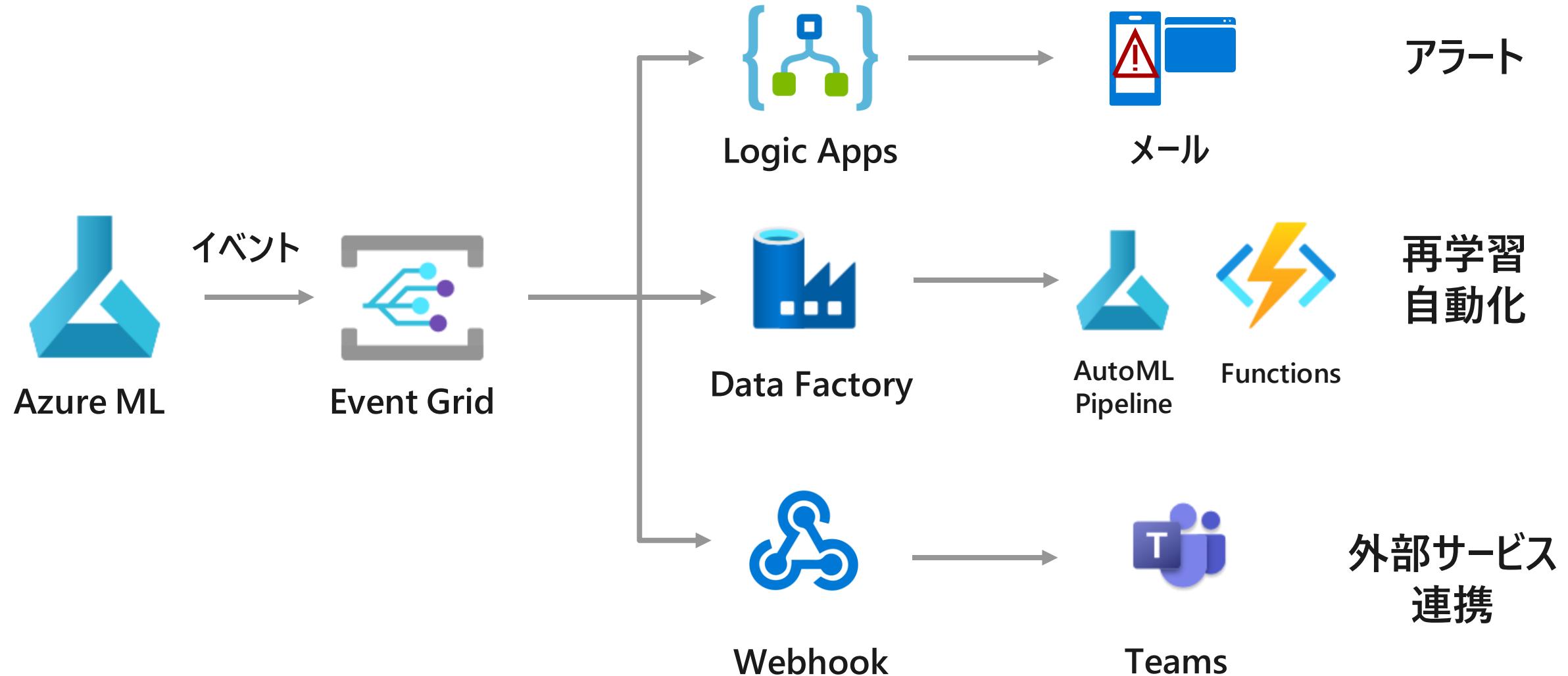
数値系の特徴量

- ワッサーライン距離
- 平均値
- 最小値
- 最大値

カテゴリーの特徴量

- ユークリッド距離
- ユニークな値の数

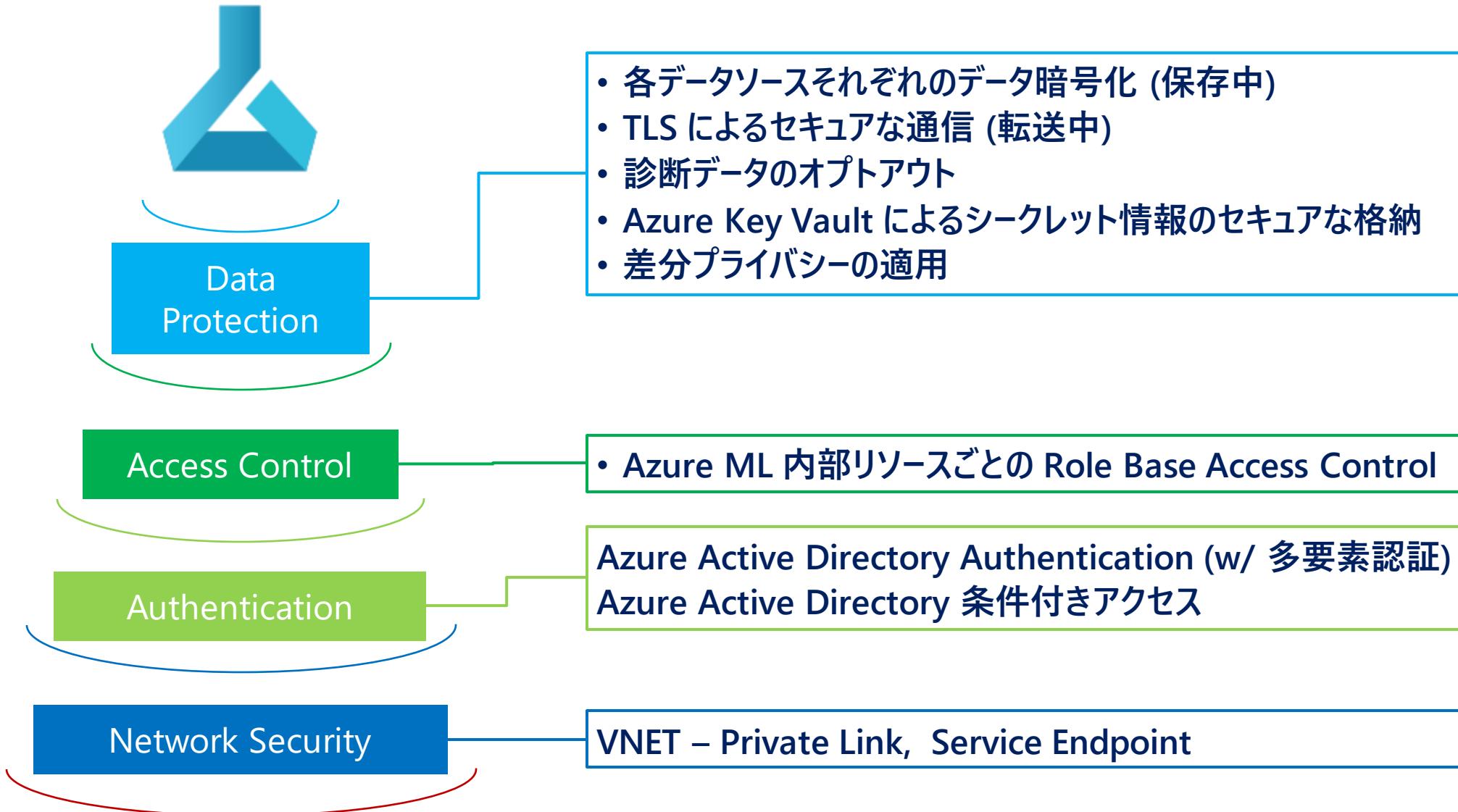
⑤通知・アラート



Security



セキュリティのプラクティス



セキュリティ考慮事項

ネットワーク

- ✓仮想ネットワークによる環境の保護

脆弱性管理

- ✓リソースの脆弱性の評価

データの復旧

- ✓データ & キーのバックアップ

ログとモニタリング

- ✓ログの一元管理、監査ログ有効化、
ログ監視

アセット・資産管理

- ✓リソース管理、ポリシー、機能制限

インシデント対応

- ✓ガイドの作成、通知、対応自動化

認証・認可

- ✓カスタムロール、パスワード変更、SSO、
多要素認証、PAW

セキュリティ保護構成

- ✓ポリシー、OS 設定、構成の git 管理、
シークレット管理、

ペネトレーションテスト

- ✓ペネトレーションテストによる脆弱性の
調査

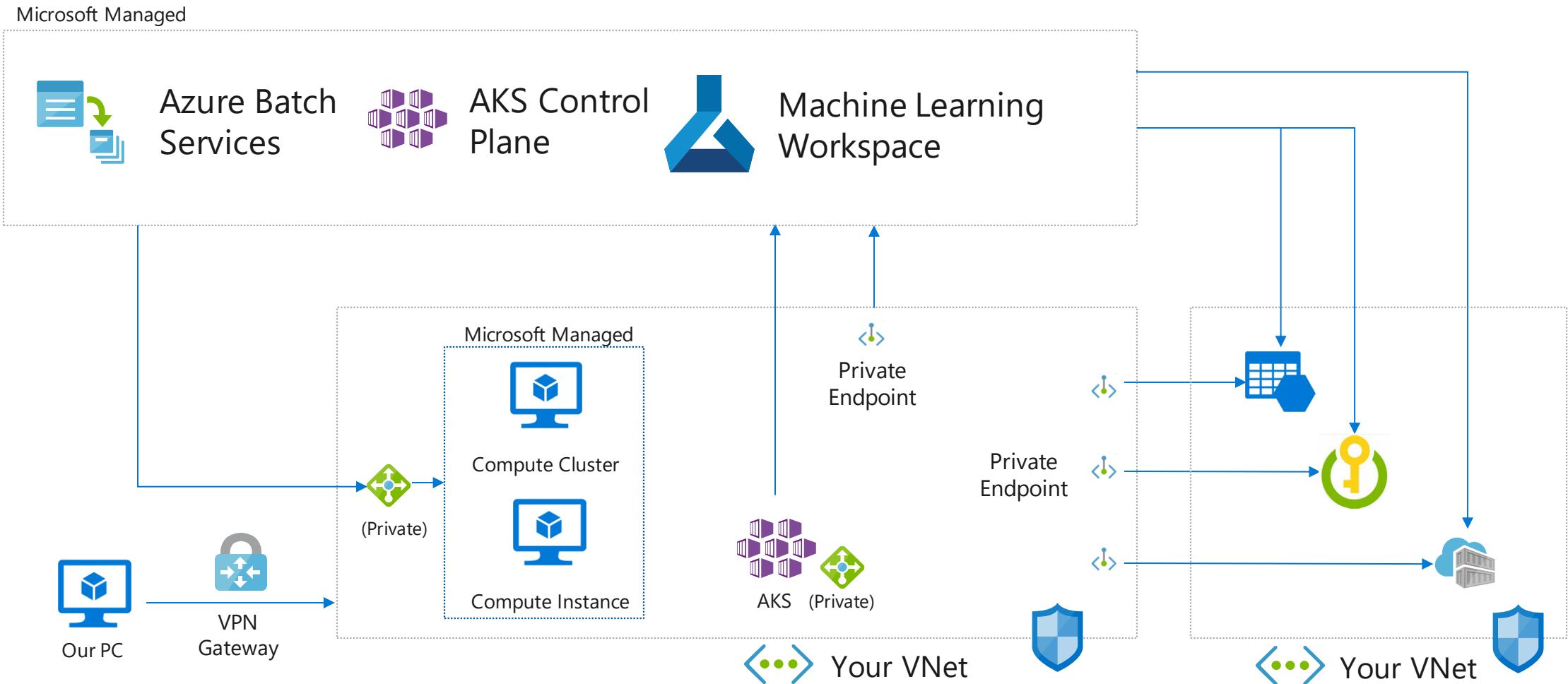
データ保護

- ✓タグ付与、データカタログ、通信暗号
化、アクセス制御、独自キー暗号化

マルウェア防御

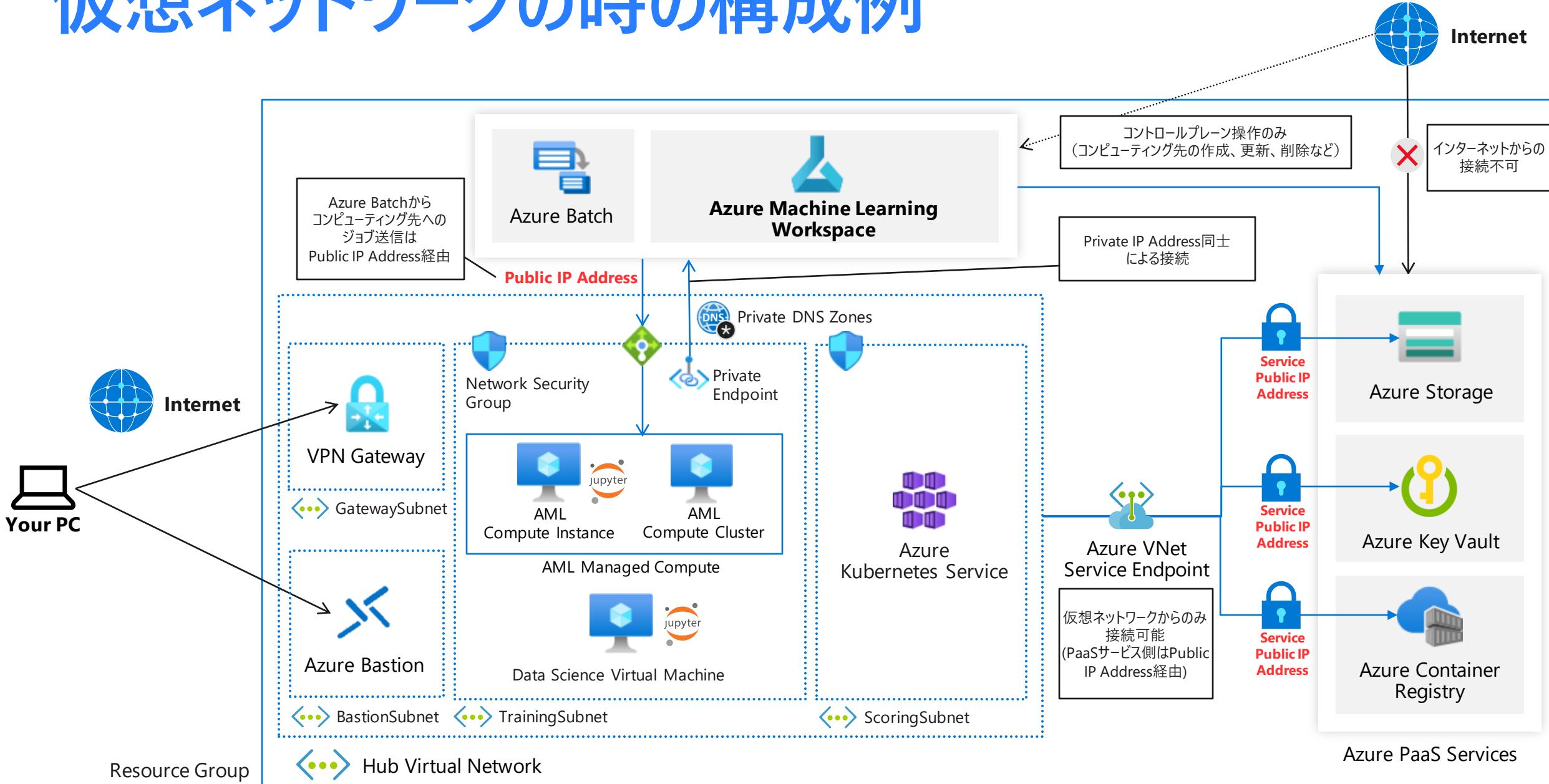
- ✓マルウェア対策ソフト、スキャン

ネットワーク



仮想ネットワークの時の構成例

Network Security



仮想ネットワークを考慮した構築

概要

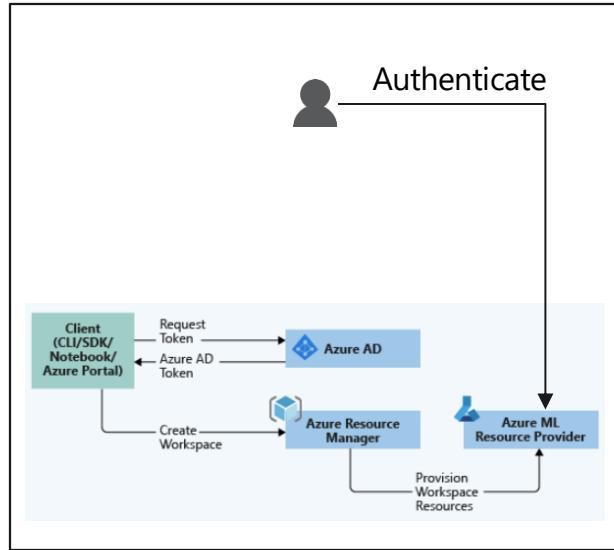
仮想ネットワークの分離とプライバシーの概要

Azure Machine Learning はコンポーネントが多岐に渡るため、考慮事項があります。必ずドキュメントを参照して確認してください。

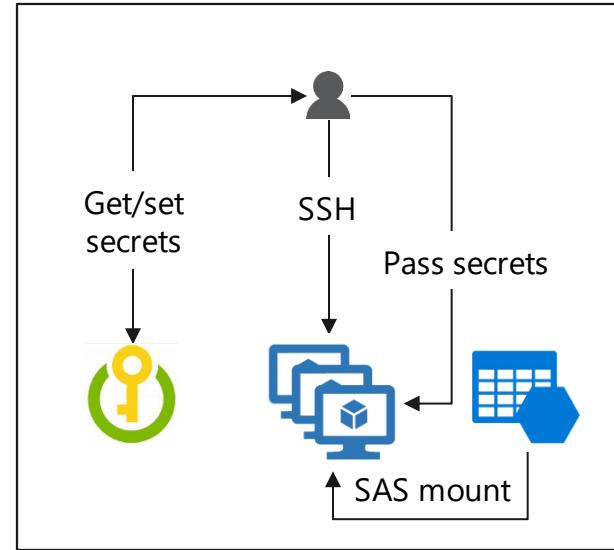
ステップ

1. 仮想ネットワークを使用して Azure Machine Learning ワークスペースをセキュリティで保護する
 - [Azure Machine Learning ワークスペース用に Azure Private Link を構成する](#)
 - [Azure Key Vault をセキュリティで保護する](#)
 - [Azure のストレージアカウントをサービスエンドポイントで保護する](#)
2. 仮想ネットワークを使用して Azure Machine Learning トレーニング環境をセキュリティで保護する
 - [必須ポート](#)
 - [仮想ネットワークからのアウトバウンド接続を制限する](#)
 - [仮想ネットワーク内にコンピューティングクラスターを作成する](#)
 - [ネットワークポートを構成する](#)
 - (オプション) [コンピューティング インスタンス ノートブック内のデータにアクセスする](#)
 - コーディング環境として[Azure Machine Learning コンピューティングインスタンス](#)を利用する場合
3. 仮想ネットワークを使用して Azure Machine Learning 推論環境をセキュリティで保護する
4. Azure 仮想ネットワークで Azure Machine Learning Studio を使用する
GUI機能を利用する場合

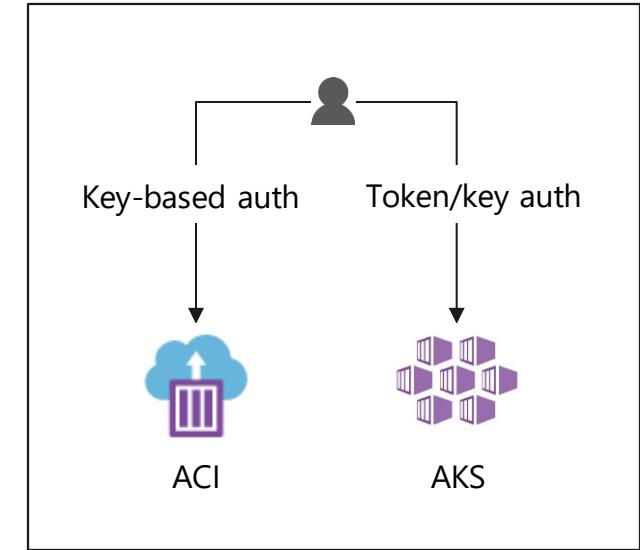
認証・認可



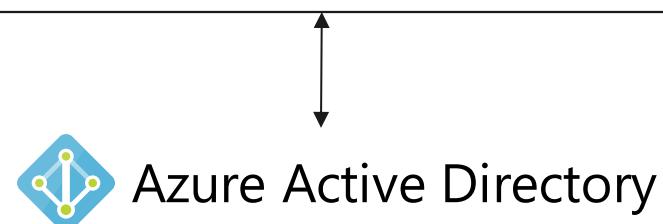
Create Workspace and Authenticate



Training



Inference

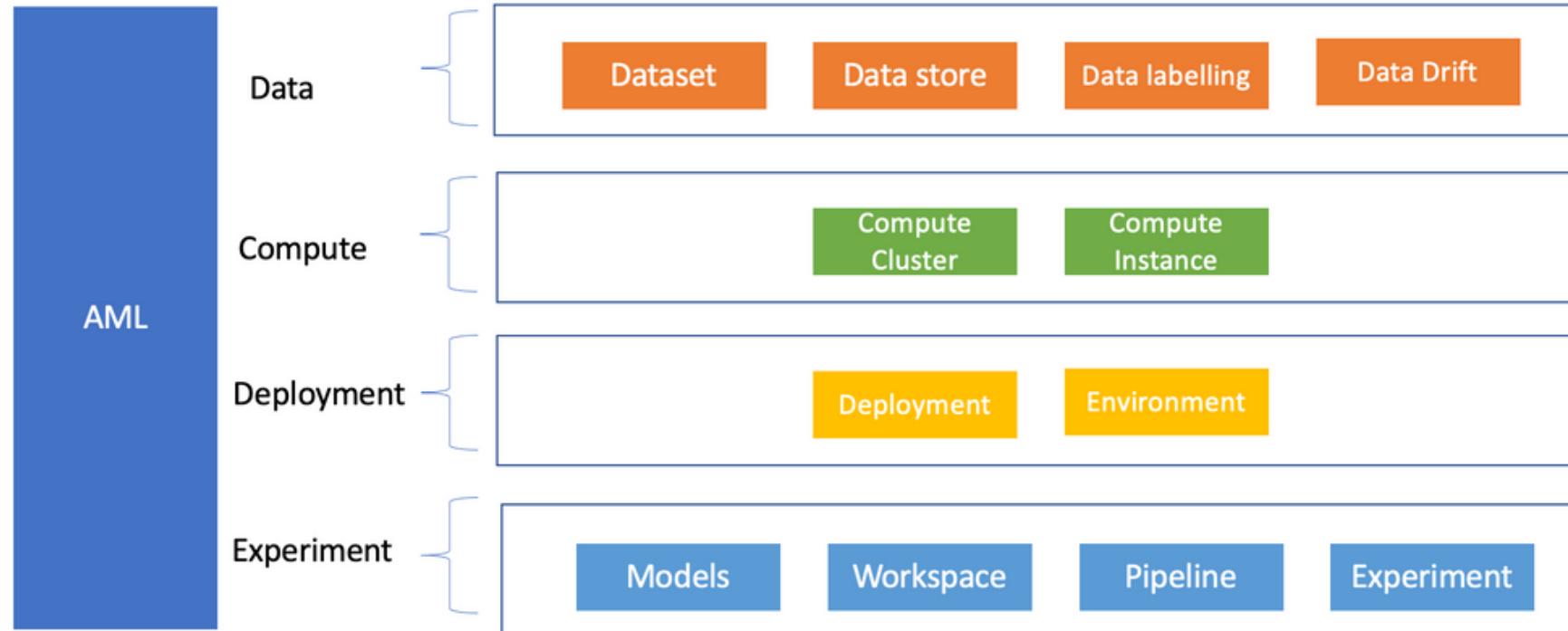


多要素認証
条件付きアクセス

アクセス制御

Azure Machine Learning のコンポーネントごとのアクセス制御の実現

利用可能なコンポーネントの例



利用可能な Azure リソースプロバイダーの操作の一覧は、
[Azure リソース プロバイダーの操作 | Microsoft Docs](#) を参照ください。

画面イメージ

管理者から見える画面

Microsoft Azure Machine Learning

ml-workspace > Compute

リソース作成権限あり

Compute instances Compute clusters Inference clusters

+ New Refresh Delete View quota

Name	Provisioning state	Virtual machine size
k80cluster	Succeeded (0 nodes)	STANDARD_NC6

New Home Author Notebooks Automated ML (preview) Designer (preview) Assets Datasets Experiments Pipelines Models Endpoints Manage Compute

データサイエンティストから見える画面

Microsoft Azure Machine Learning

ml-workspace > Compute

Compute

Compute instances Compute clusters Inference clusters Attached compute

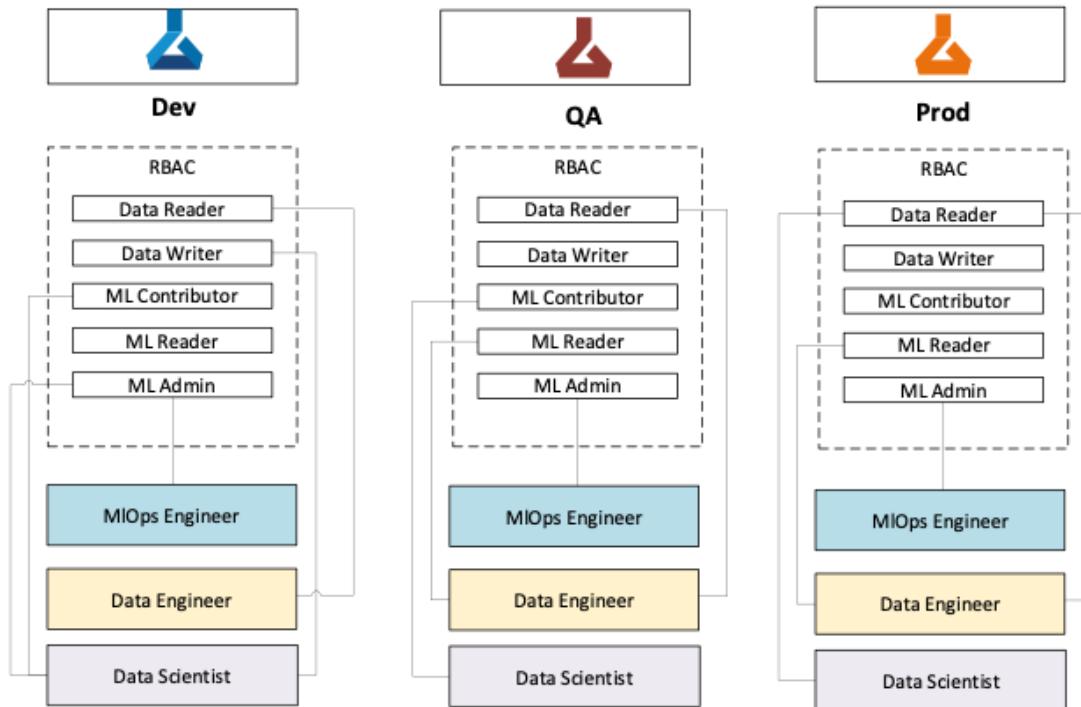
Refresh View quota

Name	Provisioning state	Virtual machine size
k80cluster	Succeeded (0 nodes)	STANDARD_NC6

Assets Datasets Experiments Pipelines Models Endpoints Manage Compute Datastores

各環境におけるロール設定

本番環境の信頼性・セキュリティを高めるために、ワークフローに応じてワークスペースを分割し、それぞれで適切な権限設定を行う。



	Dev	QA	Prod
Data Engineer	😊	😊	😊
Data Scientist	😊	😊	😢
MLOps Engineer	😊	😊	😊

😊 Most permitted permissions

😊 Medium restricted permissions

😢 Highly restricted permissions

診断データのオプトアウト

Azure Machine Learning では、デフォルトで設定されている Microsoft マネージドな Key だけでなく、お客様管理 Key (CMK) の両方を使用して、転送中/保存中の暗号化をサポートしています。

The screenshot shows the Azure portal interface with a dashboard titled 'ignite2020Demo'. The dashboard features a central card with the text 'Cosmos DB がユーザの Subscription 内で立ち上がる(課金対象)' and a list of resources below it. To the right, there is a diagram illustrating key management.

Diagram Labels:

- お客様管理 Key (Customer-managed Key)
- マイクロソフト 管理 Key (Microsoft-managed Key)

Resource List:

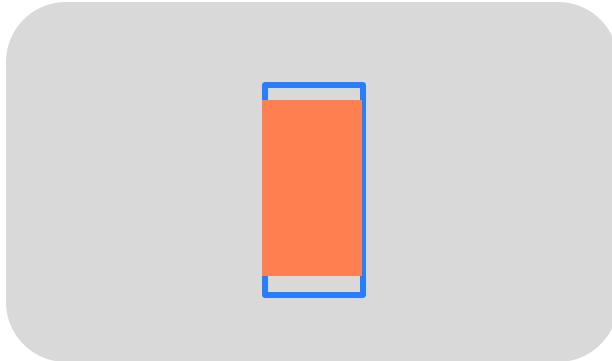
Resource Type	Name	Region
Container registry	ignite2020demoacr	eastus2euap
Application Insights	ignite2020demoai	South Central US
Key vault	ignite2020demokv	eastus2euap
Storage account	ignite2020demosa	eastus2euap
Virtual network	ignite2020demovnet	eastus2euap
Azure Cosmos DB account	cosmosdb895056896	eastus2euap
Search service	ignite2020de787886a4	
Storage account	sa1814799360	
Virtual network	vnet	

Document Reference: ドキュメント: [Data encryption with Azure Machine Learning](#)

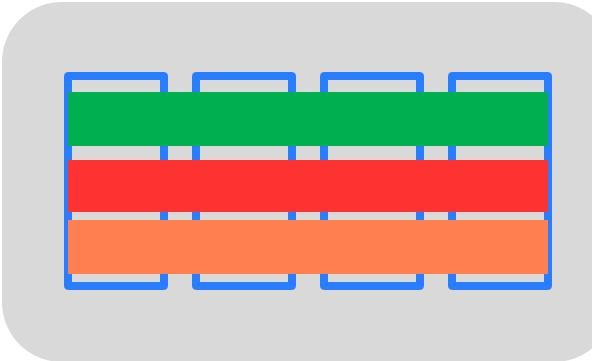
Compute



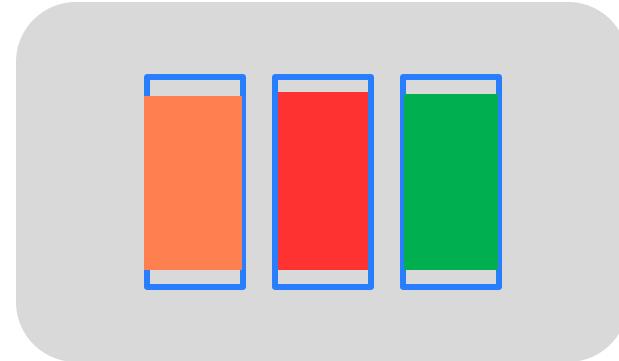
計算環境の種類



シングル



分散処理

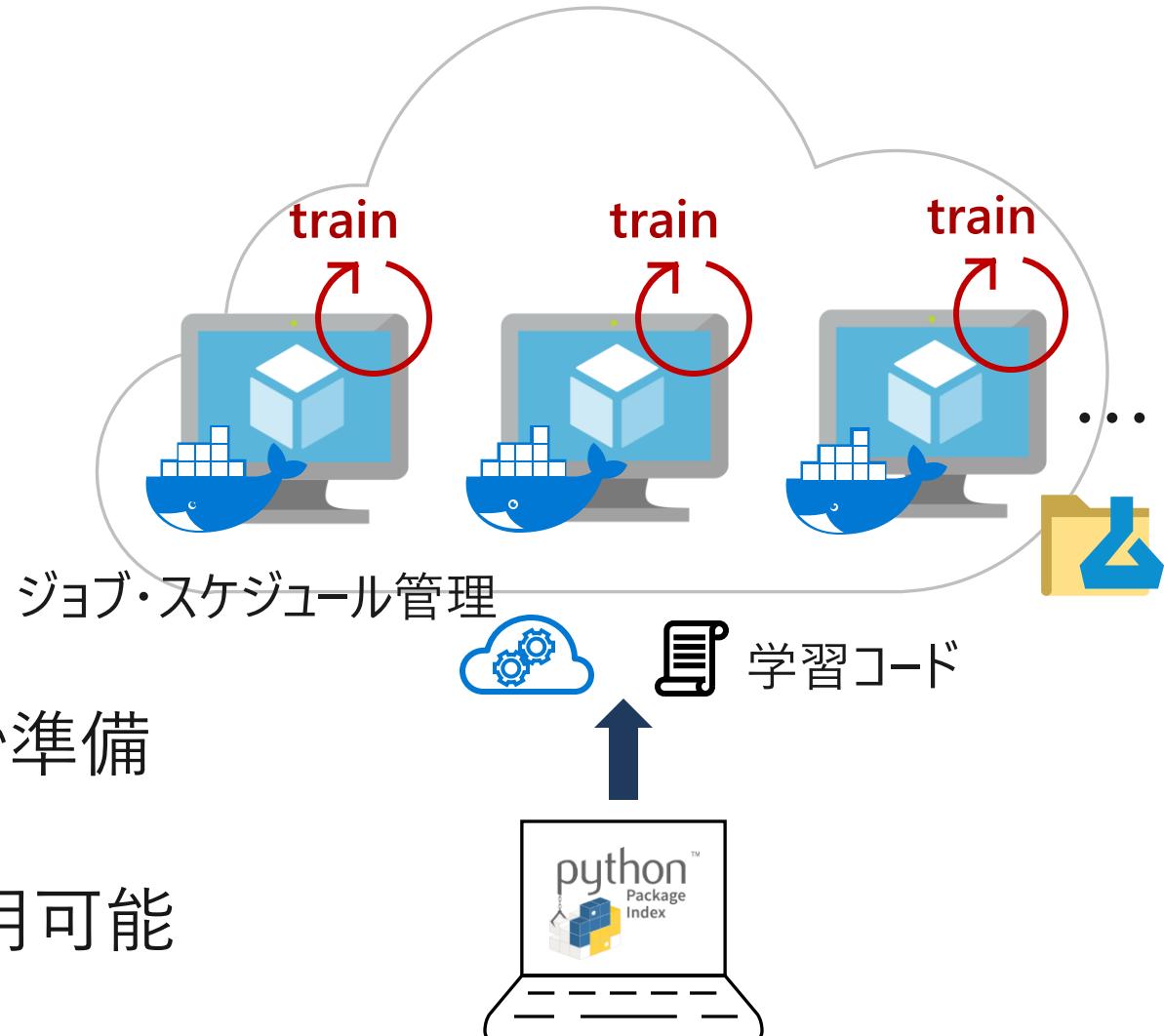


並列処理

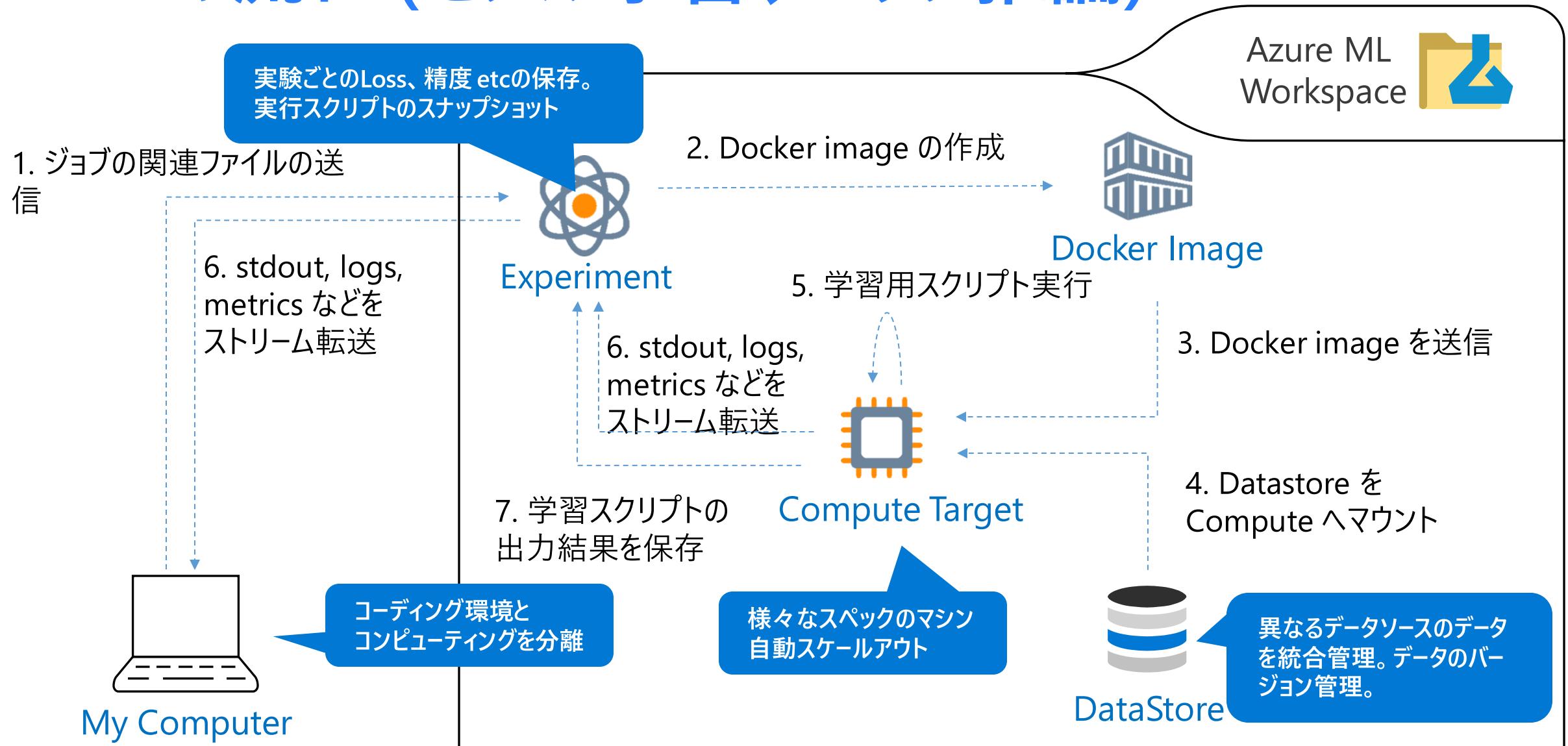
Azure Machine Learning Compute Cluster

モデル学習・推論のためのクラウドネイティブなクラスター環境

- 様々なスペックのVMを選択・起動
- 自動スケールアウト・ダウン
- ジョブ管理、スケジュール管理
- Job に必要なライブラリ・データを自動で準備
- 低優先度オプション : 80% 割引で利用可能



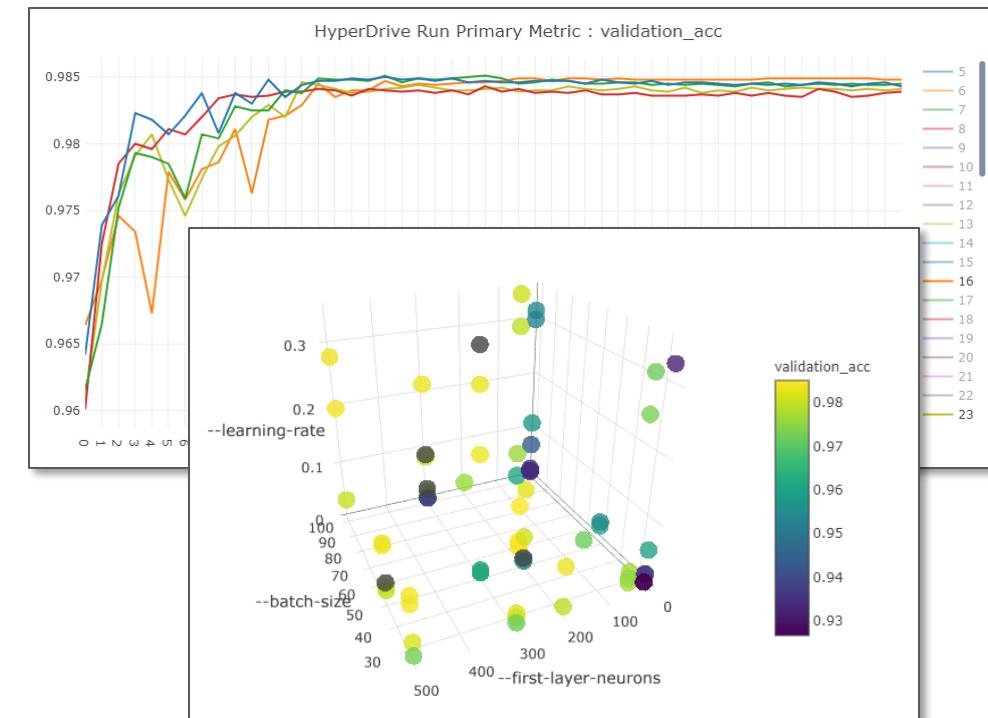
Job の流れ (モデル学習、バッチ推論)



ハイパーパラメータチューニング Hyperdrive

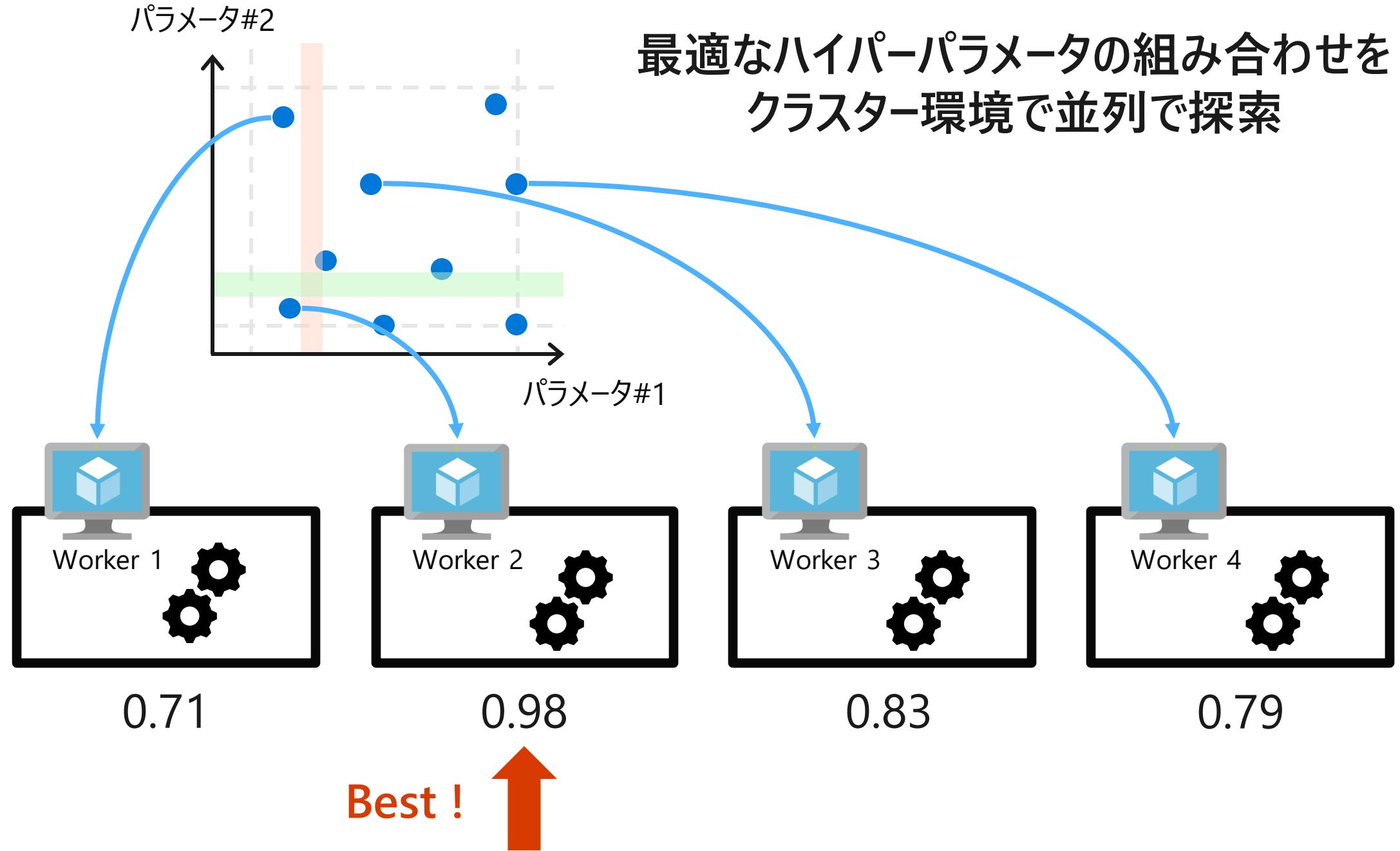
分散環境で並列実行することで高速化を実現

- メジャーなチューニング手法を提供
 - Grid Search
 - Random Search
 - Bayesian Optimization
- 早期終了条件の指定が可能
- 実行結果の可視化
 - Azure ML Experimentとの統合
- Compute Cluster上で並列実行

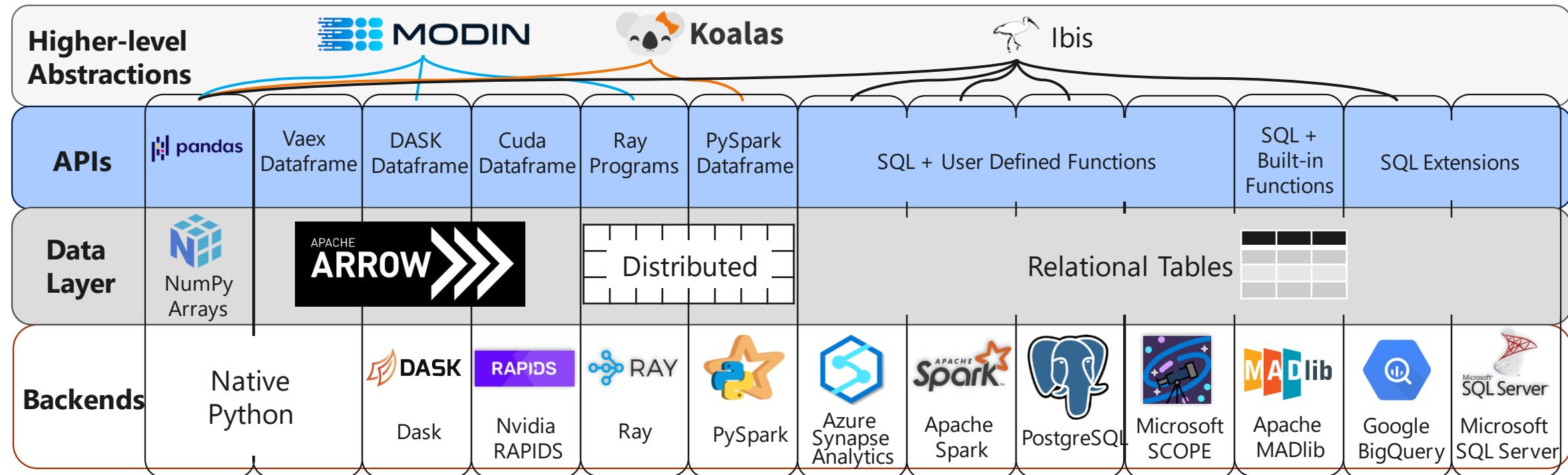


hyperdrive 可視化の例

最適なハイパーパラメータの組み合わせを
クラスター環境で並列で探索



大規模ワークフローに対応するテクノロジー



Deploy & Inference



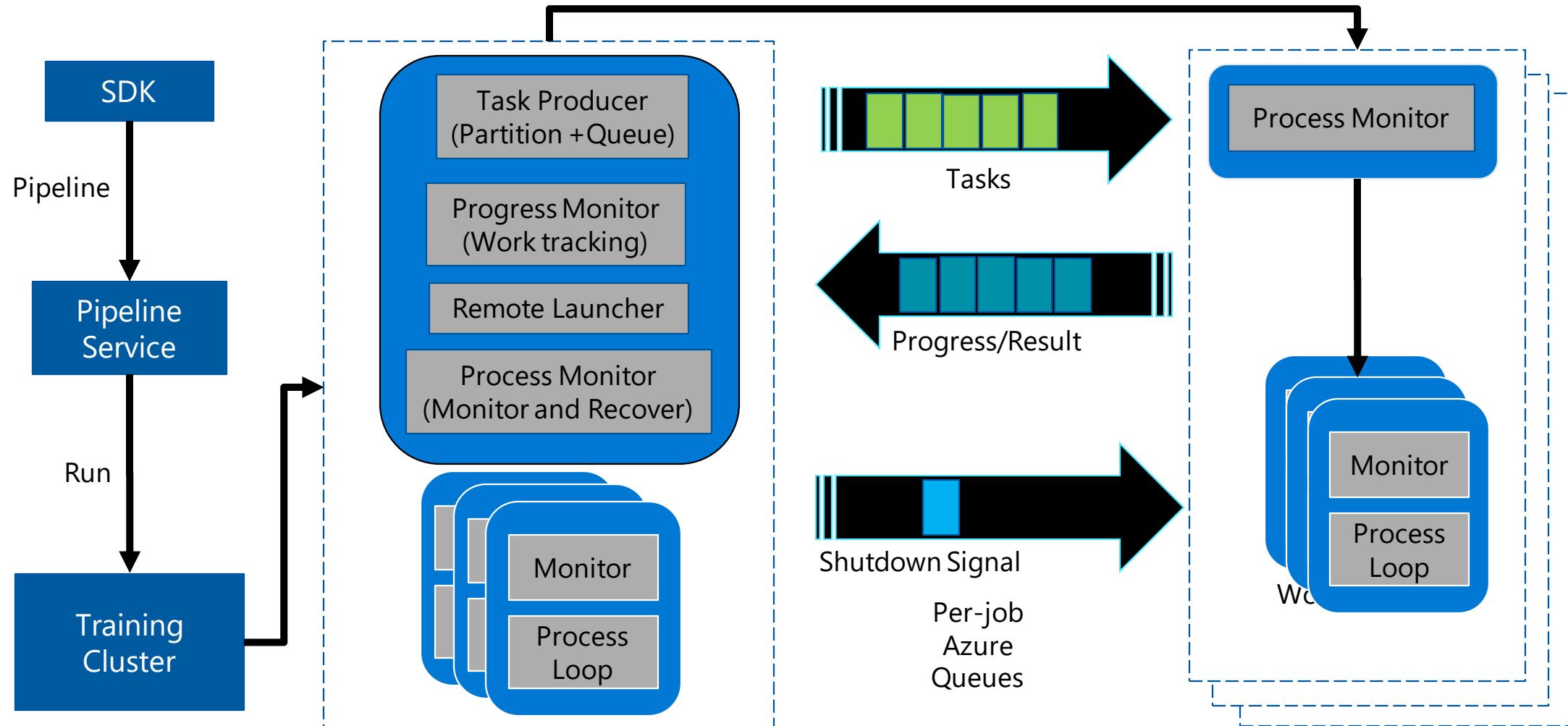
デプロイメントの選択肢

要件に応じて適切な推論環境を使い分ける

Compute Target	Real-time or Batch	本番環境	GPU	分散・並列	オンプレミス	用途
Local Computer	Both		✓		✓	テスト・デバッグ
Azure Container Instances	Real-time					テスト・デバッグ
Azure Kubernetes Service	Real-time	✓	✓	✓		リアルタイム推論
Azure ML Compute Cluster	Batch	✓		✓		バッチ推論
Azure Synapse Analytics Spark Pool	Batch	✓		✓		バッチ推論
Azure IoT Edge	Realtime	✓			✓	IoT エッジ推論

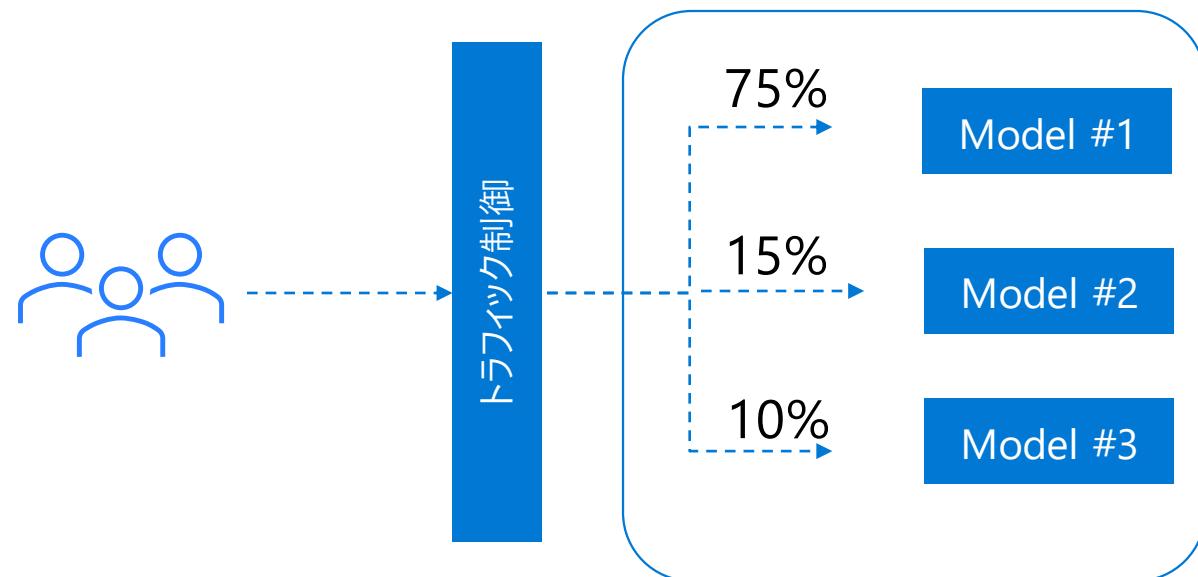
バッチ推論を高速に実行する ParallelRunStep

ジョブを並列に実行して高速にモデル学習・推論を実行



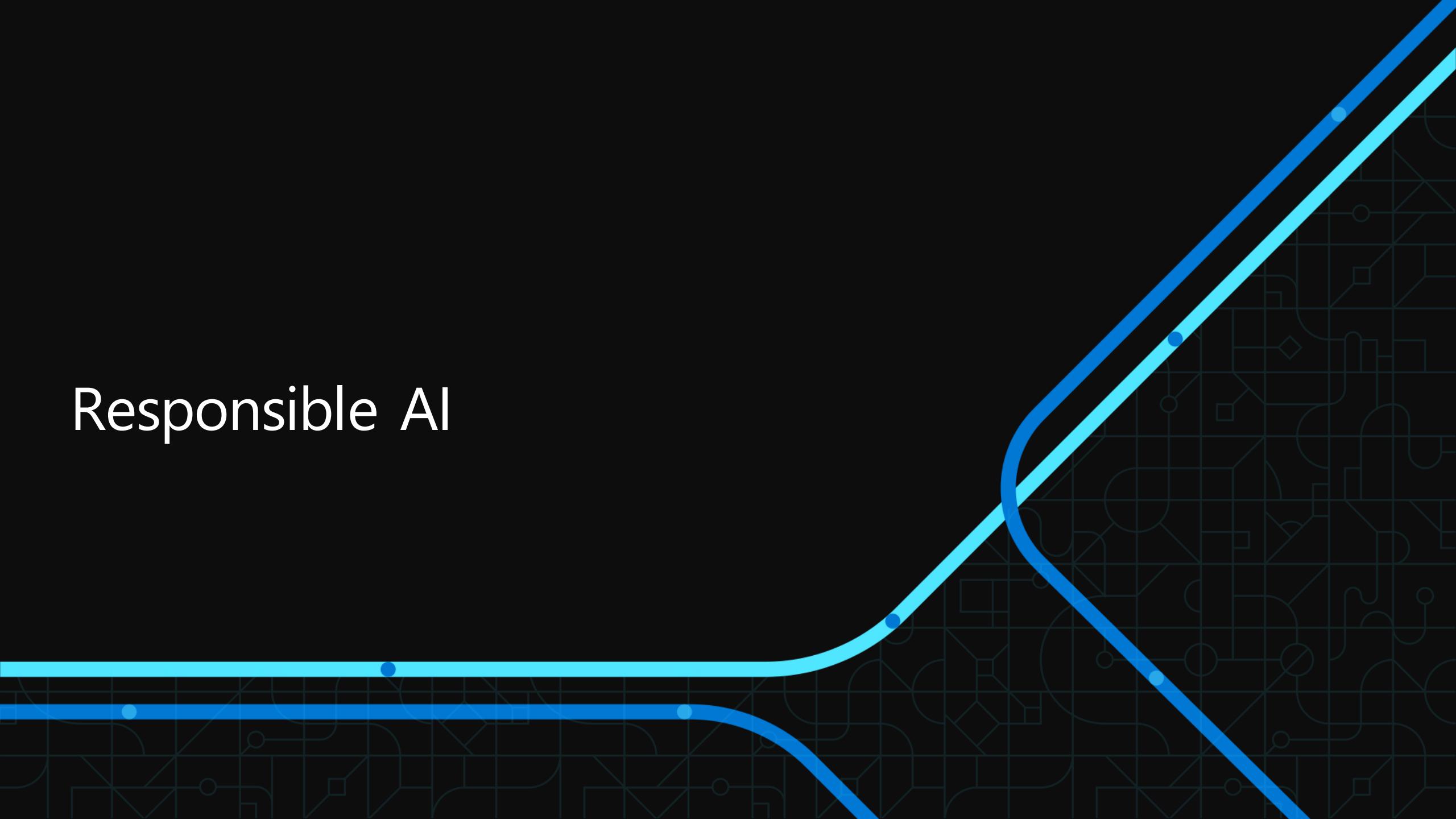
推論環境のモデル更改

Controlled Rollout (Preview) を用いることで、最大 6 つのモデルのバージョンを保持し、各モデルへのトラフィックを制御できます。業務への影響を最小限に抑えることができます。



※ 一般的なアプリケーションとして Blue/Green や Canary Deployment を利用しても問題ございません。

Responsible AI



責任のある AI - 基本原則



Fairness

AI システムはすべての関係者を公平に扱う必要があり、好ましくない固定観念や偏見を増長させてはならない



Inclusiveness

AI システムは、能力にかかわらずあらゆる人間の能力を強化し、フィードバックのチャネルを提供することによって人間とつながるべきである



Reliability & safety

AI システムは、最悪のシナリオにおいても安全に動作するよう設計すべきである



Privacy & security

AI システムは、データを悪用から保護し、プライバシー権を保証すべきである



Transparency

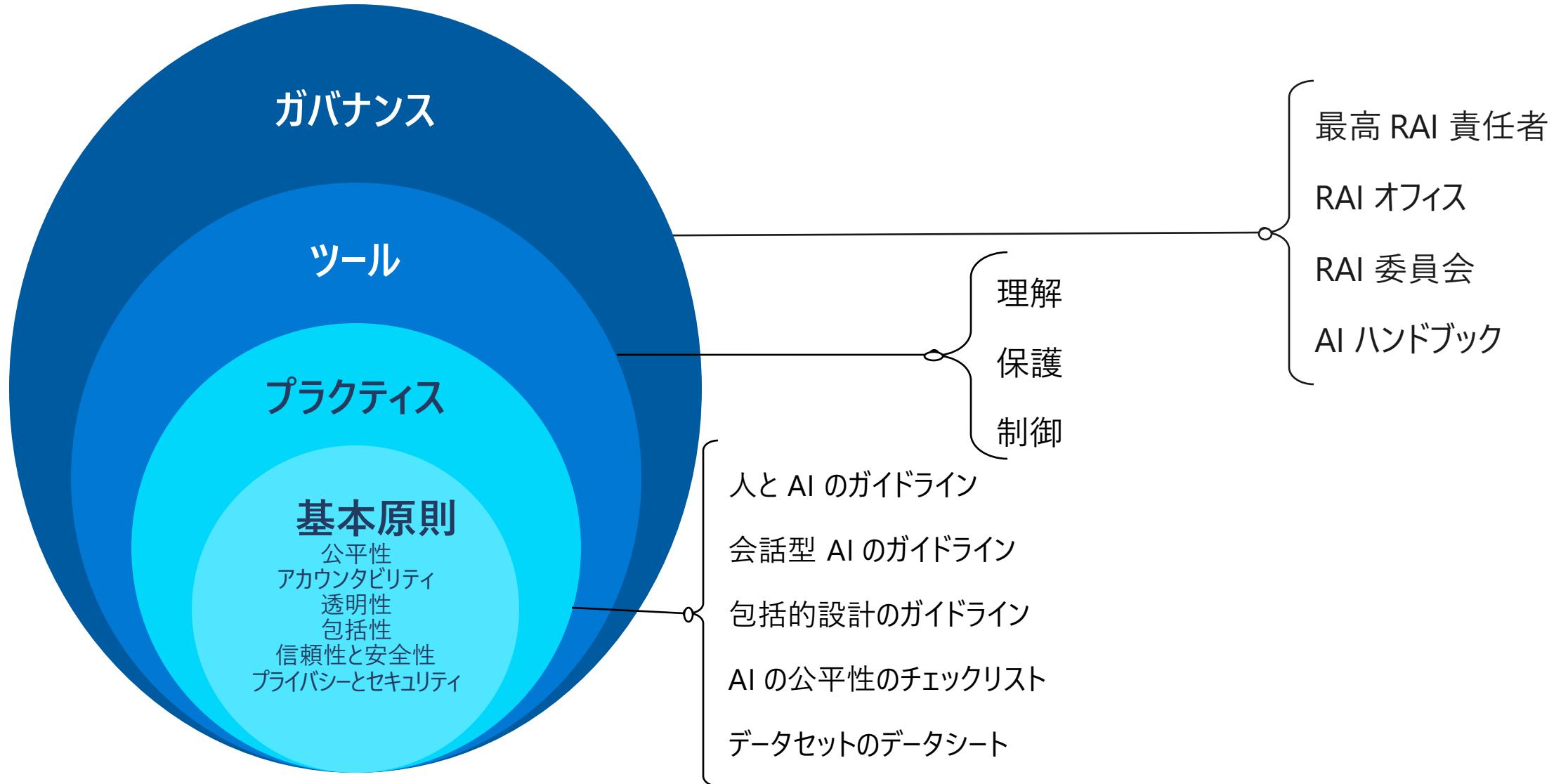
AI システムとその出力は、ステークホルダーにとって理解可能なものであるべきである



Accountability

AI システムを設計、デプロイする人物は、システムの運用方法に責任を負う必要がある

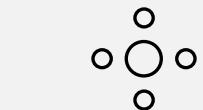
ガバナンスフレームワーク



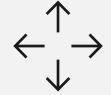
責任のある AI - ツール



理解



Interpret ML



データ ドリフト



制御



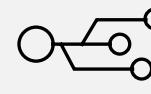
Homomorphic
暗号



Presidio



差分
プライバシー



機密 ML



保護



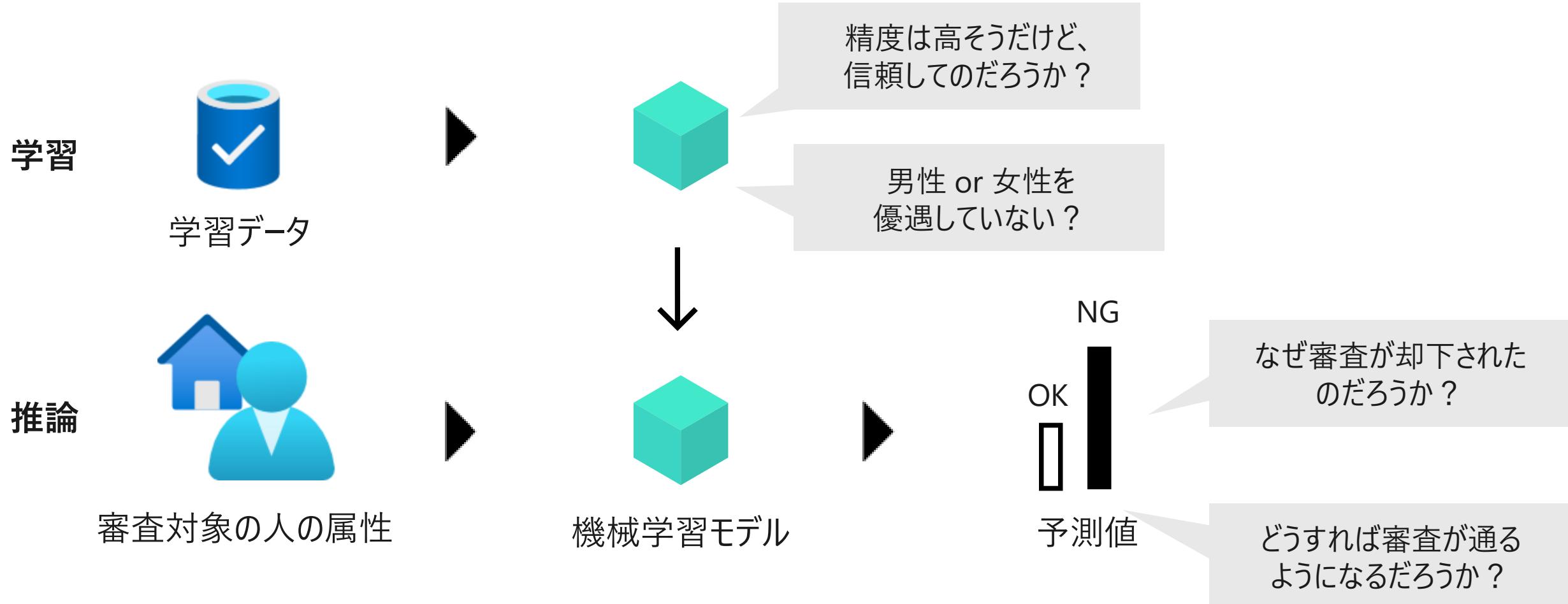
MLOps



RBAC

アプローチ：モデルの説明性・解釈性

住宅ローンの審査での機械学習の利用を考える





機械学習モデル解釈・説明のための包括的なフレームワーク



Glass-box models

決定木

ルールリスト

線形回帰・ロジスティック回帰

一般化加法モデル

...

比較的「解釈しやすいモデル」を使用して、
モデルの説明可能性を確保するアプローチ



Black-box explanations

SHAP

LIME

Partial Dependence

Sensitivity Analysis

...

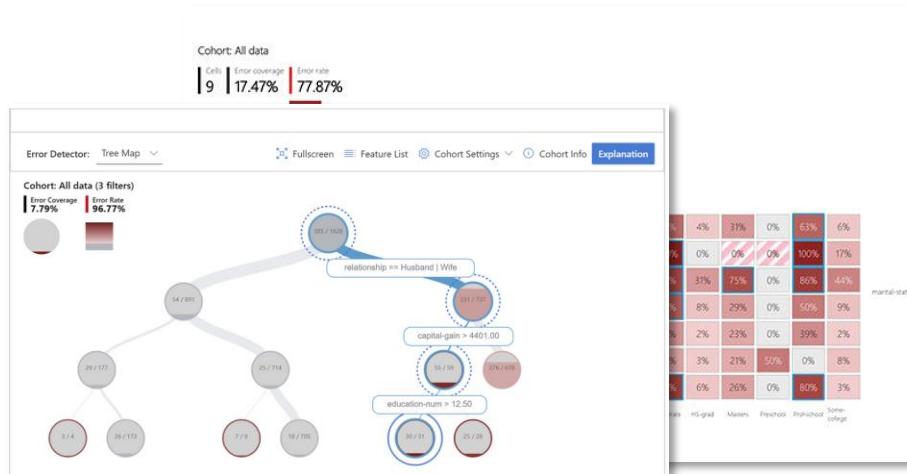
任意のモデルに対し、説明性の高いモデルによる
近似で説明性を付与するか、入力データの予測へ
の影響から推定的に説明性を付与するアプローチ

<https://interpret.ml/>

① Error Analysis

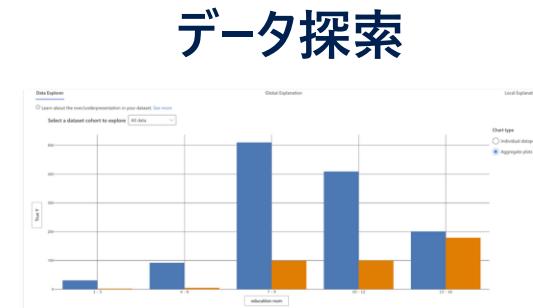
集計されたモデル精度指標では捉えられないモデル誤差の傾向分析

① Identification
誤差が大きいコホートを特定する



木構造で各条件下におけるエラー率・カバレッジを表示

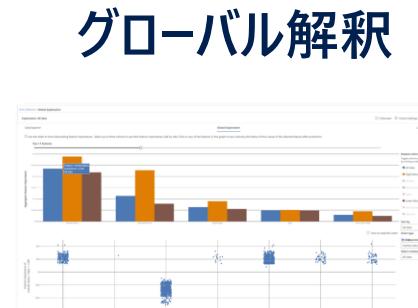
② Diagnostics
対象のコホートを比較し深掘り分析する



データ探索



ローカル解釈



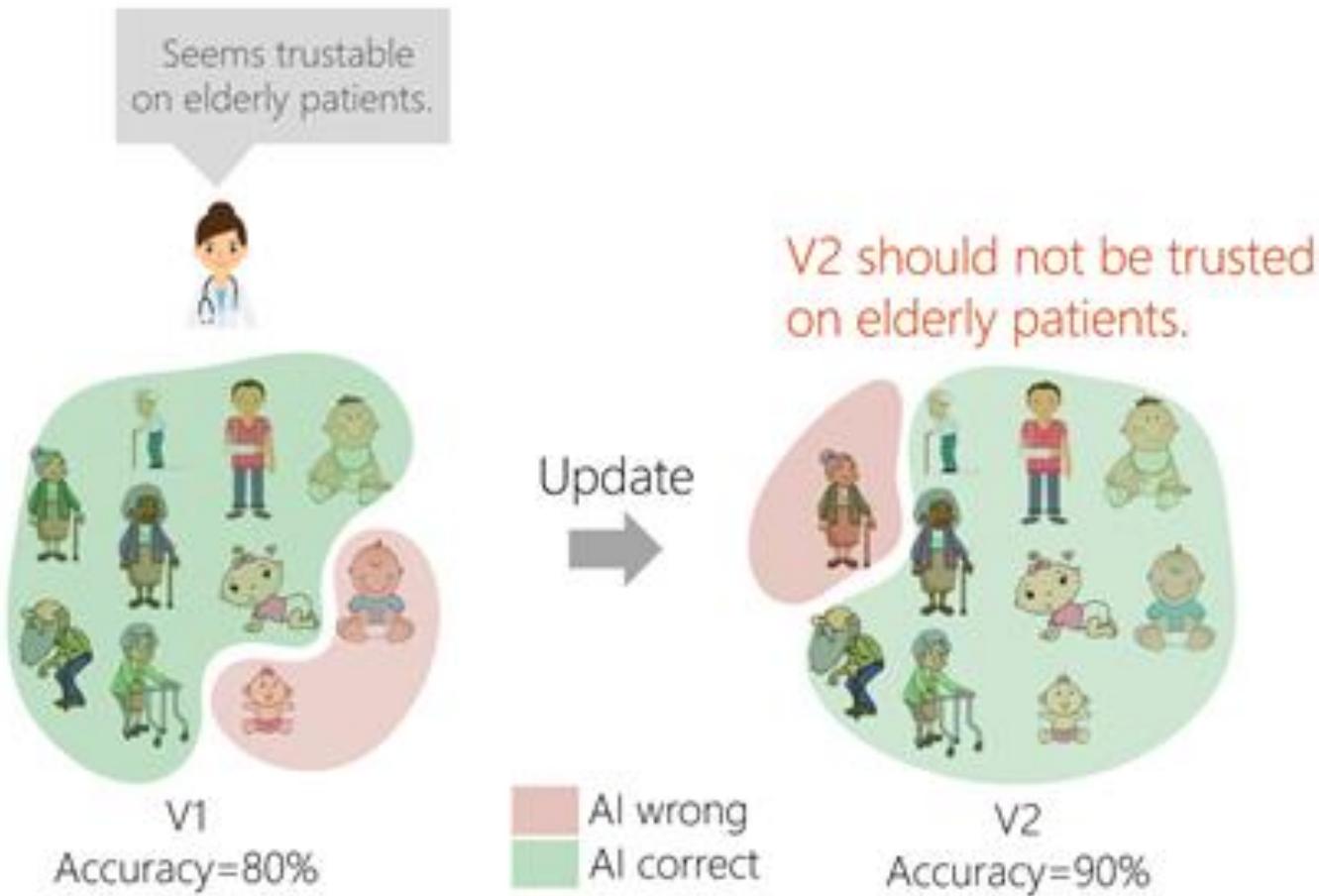
what-if 分析

Learn More : [Error Analysis](#)

アプローチ：後方互換性

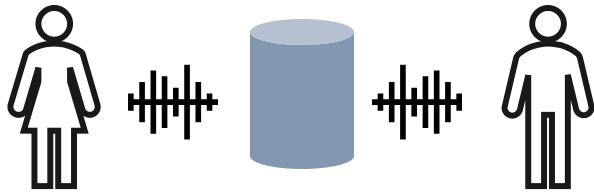
モデルの再学習による互換性の考慮

例：Accuracy が改善した
病理診断モデルにアップデートしたが
高齢者の方の診断を誤ることが多くなった



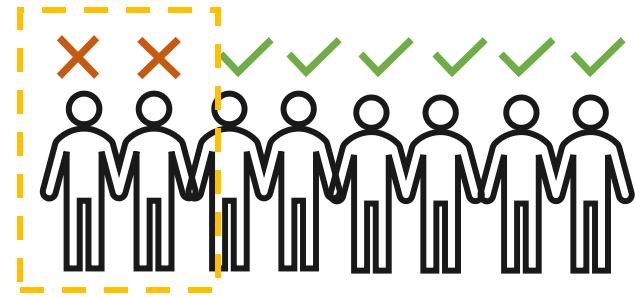
アプローチ：機械学習モデルの不公平性

機械学習モデル不公平性とは人種、性別、年齢の違いによって
ネガティブな影響を与えること



文字起こしシステムは、女性よりも
男性の方が精度が高いかもしれません。

Quality-of-service harms
サービス品質の害



ローンの申し込み審査において、
他のグループよりも白人男性を優先するかもしれません。

Allocation harms
割り当ての害

センシティブな機械学習のユースケースにおいては、
機械学習モデルの公平性の評価と対策を行う必要がある。



①

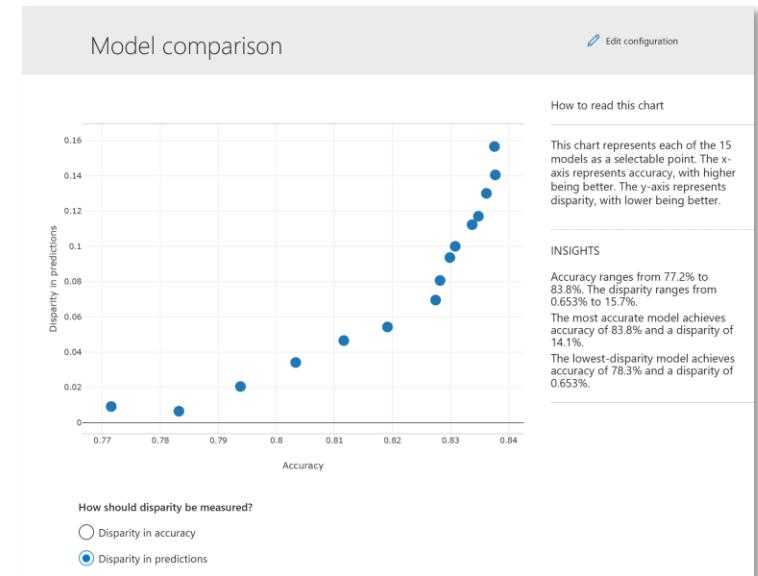
公平性の評価：

公平性を評価する一般的なメトリックとダッシュボードを利用した Sensitive Feature の評価

②

不公平性の軽減：

最先端のアルゴリズムによって分類・回帰モデルの不公平性を軽減



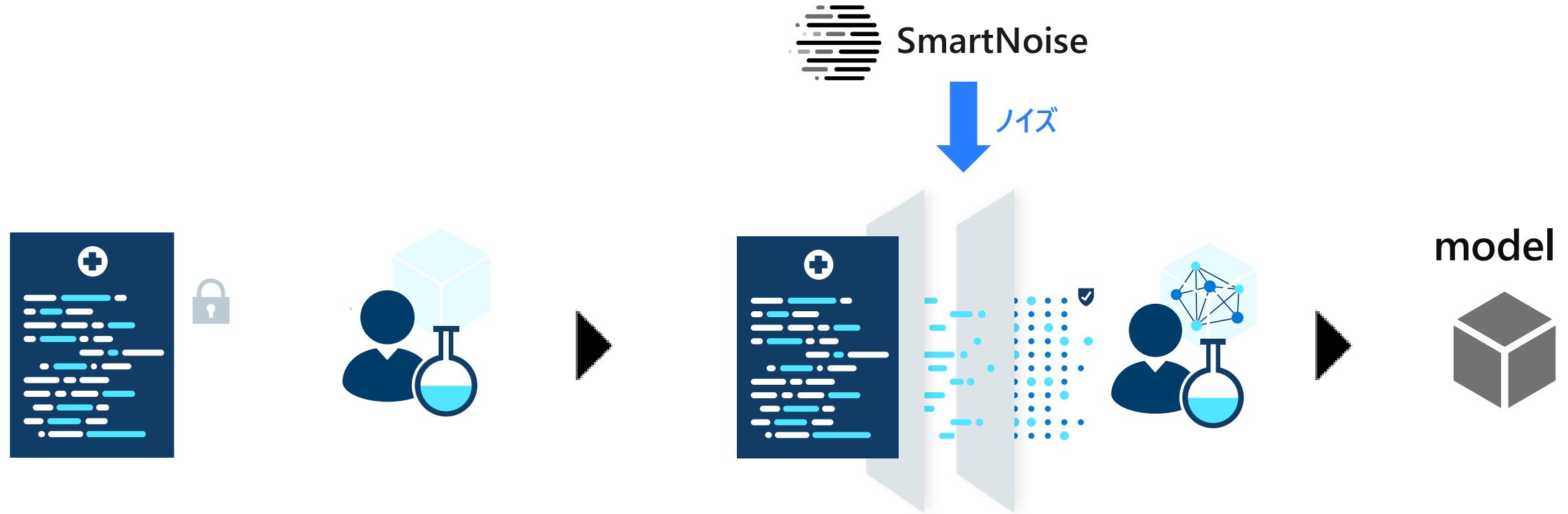
モデルのフォーマット：
scikit-learn, TensorFlow, PyTorch,
Keras などに対応

メトリック：
15以上の一般的なグループを対象にした公平性メトリック

モデルの種類：
クラス分類、回帰

* 詳細は [Fairlearn](#) を参照

アプローチ：差分プライバシー

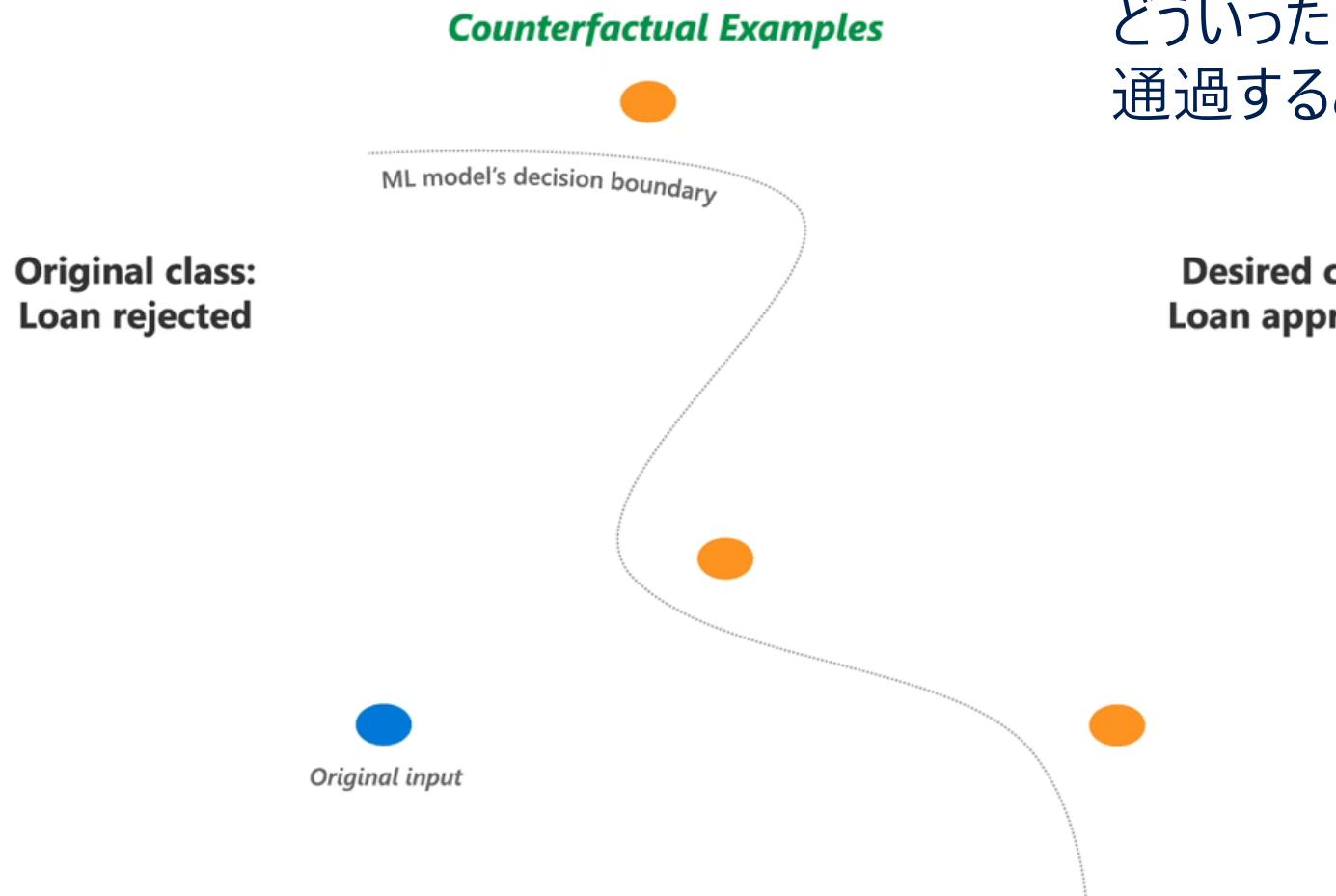


患者データのプライバシーの規制により
データを直接みれない

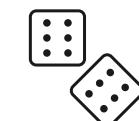
SmartNoise を用いてデータに
“統計的ノイズ”を入れて個人情報の特定を防ぐ

アプローチ：反実仮想サンプル

反実仮想サンプルによる機械学習モデルの解釈



住宅ローンモデルがローンを却下した場合、
どういった入力データであれば、ローン審査を
通過することができたのかを教えてくれる。



DiCE [interpretml/DiCE](#)
反実仮想によるモデル解釈

ガイドライン

- ・人間と AI のインタラクションのガイドライン
- ・会話 AI のガイドライン
- ・インクルーシブデザインのガイドライン
- ・AI 公平性チェックリスト (checklist)
- ・データセット用データシートのテンプレート (template)

詳細情報：責任のある AI のリソース



Microsoft AI

