MLOps を成功させるための計画

# AWS 規範ガイダンス

# AWS 規範ガイダンス: MLOps を成功させるための計画

# Table of Contents

# MLOps を成功させるための計画

Bruno™、Amazon Web Services (AWS )

2021 年 12 月 (ドキュメント履歴 )

機械学習 (ML) ソリューションを本番環境にデプロイすると、標準ソフトウェア開発プロジェクトでは発生しない多くの課題が生じます。ML ソリューションは、そもそも正しく行うには、より複雑で手間がかかります。また、通常、変動の大きい環境にも存在します。この環境では、さまざまな予想外の理由でデータ分散が時間の経過とともに大きく逸脱します。

これらの問題は、多くの ML 実務者がソフトウェアエンジニアリングのバックグラウンドから来ていないという事実によってさらに悪化するため、テスト可能なコードの記述、コンポーネントのモジュール化、バージョン管理の効果的な使用など、この業界のベストプラクティスに慣れていない可能性があります。これらの課題は技術的負債を生み、ソリューションは ML チームにとって複合効果によって、時間の経過とともにより複雑になり、維持が困難になります。

このガイドでは、ML プロジェクトとワークロードにおけるこれらの課題を軽減するのに役立つ ML オペレーション (MLOps) のベストプラクティスを列挙します。

MLOps はクロスカットの懸念事項であるため、これらの問題はデプロイとモニタリングのプロセスだけでなく、モデルライフサイクル全体に影響します。このガイドでは、MLOps のベストプラクティスを 4 つの主要な領域にまとめています。

- [データ]
- トレーニング
- デプロイメント
- モニタリング

## ターゲットを絞ったビジネス成果

ML モデルを本番環境にデプロイすることは、これらのリソースを存続期間中 (場合によっては数年) 維持するために継続的な労力と専有チームを必要とするタスクです。ML モデルはビジネスデータから大きな価値を引き出すことができますが、コストは高くなります。コストを最小限に抑えるには、企業はソフトウェア開発とデータサイエンスのベストプラクティスに従う必要があります。しばらくするとモデルが予期せず動作するデータドリフトなど、ML システムのニュアンスに注意する必要が

あります。これらの懸念を認識することで、企業は短期的および長期的に安全かつ俊敏にビジネス目標を達成できます。

ML モデルにはいくつかの種類があり、ターゲットとする業界にはさまざまなタイプの ML タスクやビジネス上の問題があるため、モデルや業界ごとに異なる一連の懸念を考慮する必要があります。このガイドで説明するプラクティスは、モデルやビジネスに固有のものではありませんが、デプロイ時間を改善し、生産性を高め、ガバナンスとセキュリティを強化するために、幅広いモデルや業界に適用されます。

モデルを本稼働環境に移行するのは、データサイエンティスト、機械学習エンジニア、データエンジニア、ソフトウェアエンジニアを必要とする学際的なタスクです。ML チームを構築するときは、これらのスキルと背景をターゲットにすることをお勧めします。

# データ

DevOps は、ソフトウェアの運用化を扱うソフトウェアエンジニアリングプラクティスです。DevOps の一般的な要素は、バージョン管理されたコード、継続的インテグレーションと継続的デリバリー (CI/CD) パイプライン、ユニットテスト、再現可能なコードビルドとデプロイであり、すべてコードが含まれます。ML モデルはコードとデータの積であるため、データはコードと同じ基準を満たす必要があります。MLOps は、データ品質を維持する方法、データ内のエッジケースを特定する方法、データを保護する方法、データをより保守しやすくする方法など、データ関連の問題に対処する必要があります。

トピック



# ラベリング

## 明確なラベル付け手順を提供する

データセットにはあいまいなサンプルが含まれており、データセット全体でラベル付けに一貫性がない可能性があります。たとえば、犬を含むイメージにラベルを付けるタスクを考えてみましょう。一部のサンプルには、動物の垣間のみが含まれている場合があります。これらは正または負のラベルでマークする必要がありますか? このタイプの問題は、明確で目標的な指示をラベラーに提供することで解決できます。

## 過半数投票を使用する

ここで、音声の多い音声を含むspeech-to-textデータセットに、now and go、shoe and two、cry and high、right and write など、音声的に類似または同一である単語をラベル付けする問題を考えてみましょう。この場合、ラベラーはこれらのサンプルに一貫性のないラベルを付ける可能性があります。

ラベリングの高度の正確性を維持するために、一般的なアプローチは多数決を使用することです。この場合、同じデータサンプルが複数のワーカーに渡され、その結果が集計されます。この方法とそのより高度なバリエーションについては、Machine AWS Machine Learning ブログのブログ記事Amazon SageMaker AI Ground Truth で群衆の知恵を使用して、データをより正確に注釈付けする」を参照してください。

# 分割とデータ漏洩

データ漏洩は、モデルが本番環境にあり、予測リクエストを受信している推論中にモデルがデータを取得したときに発生します。たとえば、トレーニングに使用されたデータサンプルや、モデルが本番環境にデプロイされたときに使用できない情報など、アクセスすべきではないデータ漏洩が発生します。

モデルが誤ってトレーニングデータでテストされた場合、データ漏洩によってオーバーフィットが発生する可能性があります。オーバーフィットは、モデルが見えないデータに対してうまく一般化されないことを意味します。このセクションでは、データ漏洩やオーバーフィットを回避するためのベストプラクティスについて説明します。

## データを少なくとも 3 つのセットに分割する

データ漏洩の一般的な原因の 1 つは、トレーニング中にデータを不適切に分割 (分割) することです。たとえば、データサイエンティストは、テストに使用されたデータについてモデルを意図的または無意識にトレーニングした可能性があります。このような状況では、オーバーフィットによって引き起こされる非常に高い成功メトリクスが見られることがあります。この問題を解決するには、データを training、、 validationの 3 つ以上のセットに分割する必要がありますtesting。

この方法でデータを分割することで、 validationセットを使用して、学習プロセス (ハイパーパラメータ) を制御するために使用されるパラメータを選択および調整できます。望ましい結果に達した場合、または改善の頂点に達した場合は、testingセットに対して評価を実行します。testing セットのパフォーマンスメトリクスは、他のセットのメトリクスと似ている必要があります。これは、セット間でディストリビューションの不一致がなく、モデルが本番環境でうまく一般化されることが予想されることを示します。

## 層別化分割アルゴリズムを使用する

小さなデータセットtestingでデータを training、validation、 に分割する場合、または非常に不均衡なデータを使用する場合は、必ず階層分割アルゴリズムを使用してください。層別化は、各分割に、各分割のクラスのほぼ同じ数または分布が含まれることを保証します。scikit-learn ML ライブラリはすでに階層化を実装しており、Apache Spark も実装しています。

サンプルサイズについては、統計的に有意な結論に達することができるように、検証セットとテストセットに評価に十分なデータがあることを確認してください。たとえば、比較的小さなデータセット (100 万個未満のサンプル) の一般的な分割サイズは、、、および で 70%training、15%validation、15% ですtesting。非常に大規模なデータセット (100 万個

を超えるサンプル) では、90%、5%、5% を使用して利用可能なトレーニングデータを最大化できます。

ユースケースによっては、本番稼働用データが収集期間中に急激で急激な分散の変化を経験していた可能性があるため、データを追加のセットに分割すると便利です。たとえば、食料品店の商品の需要予測モデルを構築するためのデータ収集プロセスを考えてみましょう。データサイエンスチームが 2019 年中にtrainingデータを収集し、2020 年 1 月から 2020 年 3 月までtestingのデータを収集した場合、モデルはおそらくtestingセットに対して適切にスコア付けされます。ただし、モデルを本番環境にデプロイすると、特定のアイテムのコンシューマーパターンは COVID-19 の世界的大混乱のために既に大幅に変更されており、モデルの結果は低下します。このシナリオでは、モデル承認の追加の保護として別のセット ( などrecent_testing) を追加するのが理にかなっています。この追加により、ディストリビューションの不一致が原因ですぐにパフォーマンスが低下するモデルを本番環境で承認できなくなる可能性があります。

場合によっては、少数集団に関連するデータなど、特定のタイプのサンプルを含む追加のvalidationまたは testingセットを作成する場合があります。これらのデータサンプルは正しくするために重要ですが、データセット全体では適切に表現されていない可能性があります。これらのデータサブセットはスライスと呼ばれます。

国全体のデータでトレーニングされ、ターゲット変数のドメイン全体を均等に考慮するようにバランスが取れたクレジット分析用の ML モデルの場合を考えてみましょう。さらに、このモデルにCity機能がある可能性があることも考慮してください。このモデルを使用する銀行がビジネスを特定の都市に拡大する場合、そのリージョンでのモデルのパフォーマンスに関心がある可能性があります。したがって、承認パイプラインは、国全体のテストデータに基づいてモデルの品質を評価するだけでなく、特定の都市スライスのテストデータも評価する必要があります。

データサイエンティストが新しいモデルに取り組むとき、モデルの検証段階で少数派のスライスを統合することで、モデルの機能を簡単に評価し、エッジケースを考慮できます。

## ランダム分割を実行するときは、重複したサンプルを検討してください。

もう 1 つの、あまり一般的ではない漏洩の原因は、重複したサンプルが多すぎる可能性のあるデータセットにあります。この場合、データをサブセットに分割しても、異なるサブセットに共通のサンプルがある可能性があります。重複の数によっては、オーバーフィットが一般化と誤解される可能性があります。

## 本番環境で推論を受信するときに使用できない可能性のある機能を検討する

データ漏洩は、推論が呼び出された時点で、実稼働環境で利用できない機能を使用してモデルがトレーニングされた場合にも発生します。多くの場合、モデルは履歴データに基づいて構築されるため、このデータは、ある時点で存在しなかった追加の列または値で強化される可能性があります。過去 6 か月間に顧客が銀行に対して行ったローンの数を追跡する機能を持つクレジット承認モデルの場合を考えてみましょう。このモデルがデプロイされ、銀行に 6 か月の履歴がない新規顧客のクレジット承認に使用されると、データ漏洩のリスクがあります。

Amazon SageMaker AI Feature Store は、この問題を解決するのに役立ちます。タイムトラベルクエリを使用してモデルをより正確にテストできます。タイムトラベルクエリを使用すると、特定の時点でデータを表示できます。

## 特徴量ストア

SageMaker AI Feature Store を使用すると、コンポーネントの境界 (ストレージと使用状況など) が切り離されるため、チームの生産性が向上します。また、組織内のさまざまなデータサイエンスチーム間で機能の再利用性を提供します。

### タイムトラベルクエリを使用する

Feature Store のタイムトラベル機能は、モデル構築を再現し、より強力なガバナンスプラクティスをサポートします。これは、Git などのバージョン管理ツールがコードを評価する方法と同様に、組織がデータ系統を評価する場合に便利です。タイムトラベルクエリは、組織がコンプライアンスチェックのために正確なデータを提供するのにも役立ちます。詳細については、 AWS Machine Learning ブログのAmazon SageMaker AI Feature Store の主な機能を理解する」を参照してください。

### IAM ロールの使用

Feature Store は、チームの生産性やイノベーションに影響を与えることなく、セキュリティの向上にも役立ちます。 AWS Identity and Access Management (IAM) ロールを使用して、特定のユーザーまたはグループの特定の機能へのきめ細かなアクセスを許可または制限できます。

たとえば、次のポリシーは Feature Store の機密機能へのアクセスを制限します。

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Deny",
            "Action": "*",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket--usw2-az1--x-s3/12345678910/
sagemaker/us-east-2/offline-store/doctor-appointments"
        }
    ]
}
```

Feature Store を使用したデータセキュリティと暗号化の詳細については、SageMaker AI ドキュメントの「セキュリティとアクセスコントロール」を参照してください。

## ユニットテストを使用する

データサイエンティストが一部のデータに基づいてモデルを作成する場合、データの分布について仮定することも、データプロパティを完全に理解するために徹底的な分析を実行することもあります。これらのモデルがデプロイされると、最終的には古くなります。データセットが古くなると、データサイエンティスト、ML エンジニア、および (場合によっては) 自動システムは、オンラインまたはオフラインストアから取得される新しいデータでモデルを再トレーニングします。

ただし、この新しいデータの分散が変更され、現在のアルゴリズムのパフォーマンスに影響する可能性があります。これらのタイプの問題をチェックする自動化された方法は、ソフトウェアエンジニアリングからユニットテストの概念を借用することです。テストする一般的な問題には、欠損値の割合、カテゴリ変数の基数、仮説テスト統計 (t テスト) などのフレームワークを使用して、実際の値列が予想される分布に準拠しているかどうかなどがあります。また、データスキーマを検証して、データスキーマが変更されておらず、無効な入力機能がサイレントに生成されないようにすることもできます。

ユニットテストでは、ML プロジェクトの一部として実行する正確なアサーションを計画できるように、データとドメインを理解する必要があります。詳細については、 AWS ビッグデータブログのPyDeequ を使用した大規模なデータ品質のテスト」を参照してください。

# トレーニング

MLOps は ML ライフサイクルの運用に関心があります。したがって、技術的負債を発生させることなく、ビジネスニーズを満たし、長期的にうまく機能する実用的なモデルの作成に向けて、データサイエンティストやデータエンジニアの作業を容易にする必要があります。

このセクションのベストプラクティスに従って、モデルトレーニングの課題に対処します。

トピック

- ベースラインモデルを作成する
- データ中心のアプローチとエラー分析を使用する
- 高速反復のためにモデルを設計する
- ML 実験を追跡する
- トレーニングジョブのトラブルシューティング

## ベースラインモデルを作成する

実務者が ML ソリューションでビジネス上の問題に直面する場合、通常、最初の目的はstate-of-the-artアルゴリズムを使用することです。state-of-the-artアルゴリズムがタイムテストされていない可能性が高いため、この手法はリスクが高くなります。さらに、state-of-the-artアルゴリズムは、多くの場合、より複雑で十分に理解されていないため、より単純な代替モデルよりもわずかにしか改善されない可能性があります。比較的迅速に検証とデプロイを行い、プロジェクトの利害関係者の信頼を得ることができるベースラインモデルを作成することをお勧めします。

ベースラインを作成するときは、可能な限りメトリクスのパフォーマンスを評価することをお勧めします。ベースラインモデルのパフォーマンスを他の自動化システムまたは手動システムと比較して、その成功を保証し、モデルの実装またはプロジェクトを中長期的に配信できることを確認します。

ベースラインモデルは、ML エンジニアとさらに検証して、モデルが推論時間、データの移行が予想される頻度、モデルがこのような場合に簡単に再トレーニングできるかどうか、デプロイ方法など、プロジェクトで確立された非機能的な要件を満たしていること、ソリューションのコストに影響を与えることを確認する必要があります。これらの質問に対して学際的な視点を持てば、成功し、長時間実行されるモデルを開発している可能性が高まります。

データサイエンティストは、ベースラインモデルにできるだけ多くの機能を追加する傾向があるかもしれません。これにより、希望する結果を予測するためのモデルの能力が向上しますが、これらの機能の一部は増分メトリクスの改善のみを生成する可能性があります。多くの機能、特に高い相関性を

持つ機能は冗長である可能性があります。機能の追加が多すぎると、より多くのコンピューティングリソースと調整が必要になるため、コストが増加します。データドリフトの可能性が高まったり、より速く発生したりするため、特徴量が多すぎるとモデルのday-to-dayにも影響します。

2 つの入力特徴の相関性は高いが、因果性がある特徴は 1 つだけであるモデルを考えてみましょう。例えば、ローンがデフォルトかどうかを予測するモデルには、顧客の年齢や収入などの入力機能があり、高い相関関係がある可能性がありますが、ローンの付与または拒否には収入のみを使用する必要があります。これら 2 つの機能でトレーニングされたモデルは、年齢などの因果性を持たない機能に依存して予測出力を生成している可能性があります。本番環境に移行した後、モデルがトレーニングセットに含まれる平均期間よりも年長または年少の顧客に対して推論リクエストを受信した場合、パフォーマンスが低下する可能性があります。

さらに、個々の機能ごとに本番環境で分散シフトが発生し、モデルが予期せず動作する可能性があります。これらの理由から、モデルに搭載されている機能が多いほど、ドリフトや古さに関してより脆弱になります。

データサイエンティストは相関測定と Shapley 値を使用して、どの特徴量が予測に十分な値を追加し、保持する必要があるかを判断する必要があります。このような複雑なモデルがあると、モデルがモデル化された環境を変更するフィードバックループが発生する可能性が高くなります。例としては、モデルのレコメンデーションが原因でコンシューマーの動作が変化するレコメンデーションシステムがあります。モデル間で動作するフィードバックループはあまり一般的ではありません。例えば、映画を推奨するレコメンデーションシステムと、本を推奨する別のシステムを考えてみましょう。両方のモデルが同じコンシューマーセットをターゲットにしている場合、互いに影響します。

開発するモデルごとに、これらのダイナミクスに影響する可能性のある要因を検討し、本番環境でモニタリングするメトリクスを把握します。

## データ中心のアプローチとエラー分析を使用する

シンプルなモデルを使用する場合、ML チームはデータ自体の改善と、モデル中心のアプローチではなくデータ中心のアプローチを取ることに集中できます。プロジェクトで、画像、テキスト、オーディオなどの人間が評価できる形式 (ラベルに効率的にマッピングすることが難しい構造化データと比較して) などの非構造化データを使用している場合は、モデルのパフォーマンスを向上させるには、エラー分析を実行することをお勧めします。

エラー分析では、検証セットでモデルを評価し、最も一般的なエラーを確認します。これにより、モデルが正しく機能しなくなる可能性のある類似データサンプルの潜在的なグループを特定できます。エラー分析を実行するには、予測エラーが高い推論を一覧表示したり、あるクラスのサンプルが別のクラスからのものであると予測されたエラーをランク付けしたりできます。

# 高速反復のためにモデルを設計する

データサイエンティストがベストプラクティスに従うと、概念実証や再トレーニング中に、新しいアルゴリズムを試したり、さまざまな機能を簡単かつ迅速に組み合わせたりすることができます。この実験は、本番環境での成功に役立ちます。ベストプラクティスは、ベースラインモデルに基づいて構築し、少し複雑なアルゴリズムを採用し、トレーニングと検証セットのパフォーマンスをモニタリングしながら新しい機能を繰り返し追加して、実際の動作と予想される動作を比較することです。このトレーニングフレームワークは、予測能力の最適なバランスを提供し、技術的負債のフットプリントを小さくしてモデルをできるだけシンプルに保つのに役立ちます。

高速反復では、データサイエンティストは、特定のデータに使用する最適なモデルを決定するために、異なるモデル実装を交換する必要があります。大規模なチーム、短い期限、その他のプロジェクト管理関連の物流がある場合、方法がないと、迅速なイテレーションが困難になる可能性があります。

ソフトウェアエンジニアリングでは、Liskov 置換の原則はソフトウェアコンポーネント間のインタラクションを設計するためのメカニズムです。この原則では、クライアントアプリケーションや実装を中断することなく、インターフェイスの 1 つの実装を別の実装に置き換えることができる必要があります。ML システムのトレーニングコードを記述するときは、この原則を採用して境界を確立し、コードをカプセル化できるため、アルゴリズムを簡単に置き換え、新しいアルゴリズムをより効果的に試すことができます。

例えば、次のコードでは、新しいクラスの実装を追加するだけで、新しい実験を追加できます。

```
from abc import ABC, abstractmethod

from pandas import DataFrame


class ExperimentRunner(object):

    def __init__(self, *experiments):
        self.experiments = experiments

    def run(self, df: DataFrame) -> None:
        for experiment in self.experiments:
            result = experiment.run(df)
            print(f'Experiment "{experiment.name}" gave result {result}')


class Experiment(ABC):
```

```
    @abstractmethod
    def run(self, df: DataFrame) -> float:
        pass

    @property
    @abstractmethod
    def name(self) -> str:
        pass


class Experiment1(Experiment):

    def run(self, df: DataFrame) -> float:
        print('performing experiment 1')
        return 0

    def name(self) -> str:
        return 'experiment 1'


class Experiment2(Experiment):

    def run(self, df: DataFrame) -> float:
        print('performing experiment 2')
        return 0

    def name(self) -> str:
        return 'experiment 2'


class Experiment3(Experiment):

    def run(self, df: DataFrame) -> float:
        print('performing experiment 3')
        return 0

    def name(self) -> str:
        return 'experiment 3'


if __name__ == '__main__':
    runner = ExperimentRunner(*[
        Experiment1(),
```

```
        Experiment2(),
        Experiment3()
    ])
    df = ...
    runner.run(df)
```

## ML 実験を追跡する

多数の実験を行う場合は、観察した改善が実装された変更または機会の成果であるかどうかを判断することが重要です。Amazon SageMaker AI Experiments を使用すると、簡単に実験を作成し、メタデータを関連付けて追跡、比較、評価を行うことができます。

モデル構築プロセスのランダム性を減らすことは、同じコードとデータを考慮して、出力モデルの推論をより確実に予測できるため、デバッグ、トラブルシューティング、ガバナンスの改善に役立ちます。

ランダムな重みの初期化、並列コンピューティングの同期、内部 GPU の複雑さ、および同様の非決定的要因のために、トレーニングコードを完全に再現可能にすることはできません。ただし、ランダムシードを適切に設定して、各トレーニング実行が同じポイントから開始され、同様に動作するようにすると、結果の予測可能性が大幅に向上します。

## トレーニングジョブのトラブルシューティング

場合によっては、データサイエンティストにとって非常に単純なベースラインモデルでも適合することが難しい場合があります。この場合、複雑な関数により適合できるアルゴリズムが必要であると判断することがあります。良いテストは、データセットのごく一部 (例: 約 10 個のサンプル) のベースラインを使用して、アルゴリズムがこのサンプルに過剰適合していることを確認することです。これにより、データやコードの問題を除外できます。

複雑なシナリオをデバッグするためのもう 1 つの便利なツールはAmazon SageMaker AI デバッガーです。これは、最適なコンピューティング使用量など、アルゴリズムの正確性やインフラストラクチャに関連する問題をキャプチャできます。

# デプロイ

ソフトウェアエンジニアリングでは、コードが予期せず動作したり、予期しないユーザーの動作によってソフトウェアが破損したり、予期しないエッジケースが見つかったりする可能性があるため、コードを本番環境に配置するにはデューデリジェンスが必要です。ソフトウェアエンジニアと DevOps エンジニアは通常、ユニットテストとロールバック戦略を使用して、これらのリスクを軽減します。ML では、実際の環境がドリフトすることが予想されるため、モデルを本番環境に配置するにはさらに計画が必要です。多くの場合、モデルは改善しようとしている実際のビジネスメトリクスのプロキシであるメトリクスで検証されます。

これらの課題に対処するには、このセクションのベストプラクティスに従ってください。

トピック

- デプロイサイクルを自動化する
- デプロイ戦略を選択する
- 推論要件を検討する

## デプロイサイクルを自動化する

ヒューマンエラーを防ぎ、ビルドチェックが一貫して実行されるように、トレーニングとデプロイのプロセスを完全に自動化する必要があります。ユーザーには、本番環境への書き込みアクセス許可があってはなりません。

Amazon SageMaker AI Pipelines と は、ML プロジェクトの CI/CD パイプラインの作成AWS CodePipelineに役立ちます。CI/CD パイプラインを使用する利点の 1 つは、データの取り込み、モデルのトレーニング、モニタリングの実行に使用されるすべてのコードが、Git などのツールを使用してバージョン管理できることです。場合によっては、同じアルゴリズムとハイパーパラメータを使用してモデルを再トレーニングする必要がありますが、データは異なります。正しいバージョンのアルゴリズムを使用していることを確認する唯一の方法は、ソースコントロールとタグを使用することです。SageMaker AI が提供するデフォルトのプロジェクトテンプレートを MLOps プラクティスの開始点として使用できます。

モデルをデプロイする CI/CD パイプラインを作成するときは、ビルドアーティファクトにビルド識別子、コードバージョンまたはコミット、データバージョンをタグ付けしてください。この手法は、デプロイの問題のトラブルシューティングに役立ちます。また、規制の厳しいフィールドで予測を行うモデルにもタグ付けが必要になることがあります。ML モデルに関連する正確なデータ、コード、ビルド、チェック、承認を逆算して特定できるので、ガバナンスを大幅に改善できます。

CI/CD パイプラインのジョブの一部は、構築しているものに対してテストを実行することです。データユニットテストは、データが特徴量ストアによって取り込まれる前に行われることが予想されますが、パイプラインは引き続き特定のモデルの入力と出力に対してテストを実行し、主要なメトリクスをチェックする責任があります。このようなチェックの一例としては、固定検証セットで新しいモデルを検証し、確立されたしきい値を使用してそのパフォーマンスが以前のモデルと類似していることを確認することがあります。パフォーマンスが予想よりも大幅に低い場合、ビルドは失敗し、モデルは本番環境に移行しません。

CI/CD パイプラインの広範な使用はプルリクエストもサポートしているため、人為的ミスを防ぐのに役立ちます。プルリクエストを使用する場合、コードの変更はすべて、本番環境に移行する前に、少なくとも 1 人の他のチームメンバーによってレビューおよび承認される必要があります。プルリクエストは、ビジネスルールに準拠していないコードを特定し、チーム内で知識を広めるのにも役立ちます。

# デプロイ戦略を選択する

MLOps デプロイ戦略には、ブルー/グリーン、カナリア、シャドウ、A/B テストが含まれます。

## Blue/Green

ブルー/グリーンデプロイは、ソフトウェア開発で非常に一般的です。このモードでは、開発中も 2 つのシステムが実行され続けます。ブルーは古い環境 (この場合は置き換えられるモデル) で、グリーンは本番環境に向かう新しくリリースされたモデルです。古いシステムは稼働状態であるため、変更は最小限のダウンタイムで簡単にロールバックできます。SageMaker のコンテキストにおける Blue/Green デプロイの詳細については、 AWS Machine Learning ブログのブログ記事「 AWS CodePipeline と を使用して Amazon SageMaker AI エンドポイントを安全にデプロイおよびモニタリング AWS CodeDeployする」を参照してください。

## Canary

Canary デプロイは、2 つのモデルを一緒に実行し続けるという点で Blue/Green デプロイに似ています。ただし、Canary デプロイでは、すべてのトラフィックが最終的に新しいモデルに移行するまで、新しいモデルがユーザーに段階的にロールアウトされます。ブルー/グリーンデプロイと同様に、新しい (および潜在的に障害のある) モデルは最初のロールアウト中に注意深くモニタリングされ、問題が発生した場合にロールバックできるため、リスクは軽減されます。SageMaker AI では、InitialVariantWeight API を使用して初期トラフィック分散を指定できます。

## シャドウ

シャドウデプロイを使用して、モデルを本番環境に安全に導入できます。このモードでは、新しいモデルは古いモデルやビジネスプロセスと連携して動作し、決定に影響を与えることなく推論を実行します。このモードは、モデルを本番環境に昇格させる前の最終チェックやより高い忠実度実験に役立ちます。

シャドウモードは、ユーザーの推論フィードバックを必要としない場合に便利です。エラー分析を実行し、新しいモデルを古いモデルと比較することで予測の品質を評価できます。また、出力分布をモニタリングして、期待どおりに動作することを確認できます。SageMaker AI でシャドウデプロイを行う方法については、 AWS Machine Learning ブログのブログ記事「Deploy shadow ML models in Amazon SageMaker AI」を参照してください。

## A/B テスト

ML 実務者が環境でモデルを開発する場合、最適化するメトリクスは、重要なビジネスメトリクスのプロキシであることがよくあります。これにより、新しいモデルが収益やクリックスルー率などのビジネス成果を実際に向上させ、ユーザーからの苦情の数を減らすかどうかを判断するのが困難になります。

ビジネス目標ができるだけ多くの製品を販売することである e コマースウェブサイトの場合を考えてみましょう。レビューチームは、売上と顧客満足度が情報に基づいた正確なレビューと直接相関していることを知っています。チームメンバーは、売上を改善するために新しいレビューランキングアルゴリズムを提案する場合があります。A/B テストを使用すると、古いアルゴリズムと新しいアルゴリズムを異なる類似しているユーザーグループにロールアウトし、結果をモニタリングして、新しいモデルから予測を受け取ったユーザーが購入する可能性が高くなります。

A/B テストは、モデルの古さとドリフトによるビジネスへの影響を測定するのにも役立ちます。チームは、新しいモデルをある程度の繰り返しで本番環境に投入し、各モデルで A/B テストを実行し、経過時間とパフォーマンスのグラフを作成できます。これにより、チームは本番稼働用データのデータドリフトの変動を把握できます。

SageMaker AI で A/B テストを実行する方法の詳細については、 AWS Machine Learning ブログのブログ記事Amazon SageMaker AI を使用して本番環境で ML モデルをテストする」を参照してください。

# 推論要件を検討する

SageMaker AI を使用すると、基盤となるインフラストラクチャを選択して、さまざまな方法でモデルをデプロイできます。これらの推論呼び出し機能は、さまざまなユースケースとコストプロファイルをサポートします。オプションには、次のセクションで説明するように、リアルタイム推論、非同期推論、バッチ変換が含まれます。

## リアルタイム推論

リアルタイム推論は、リアルタイム、インタラクティブ、低レイテンシーの要件がある推論ワークロードに最適です。SageMaker AI ホスティングサービスにモデルをデプロイし、推論に使用できるエンドポイントを取得できます。これらのエンドポイントはフルマネージド型で、自動スケーリングをサポートしており (Amazon SageMaker AI モデルを自動的にスケーリングする」を参照 )、複数のアベイラビリティーゾーンにデプロイできます。

Apache MXNet、PyTorch、または TensorFlow で構築された深層学習モデルがある場合は、Amazon SageMaker AI Elastic Inference (EI) を使用することもできます。EI を使用すると、任意の SageMaker AI インスタンスに小数 GPUs をアタッチして推論を高速化できます。クライアントインスタンスを選択してアプリケーションを実行し、EI アクセラレーターをアタッチして、推論のニーズに合わせて適切な量の GPU アクセラレーションを使用できます。

もう 1 つのオプションは、多数のモデルをデプロイするためのスケーラブルで費用対効果の高いソリューションを提供するマルチモデルエンドポイントを使用することです。これらのエンドポイントは、複数のモデルをホストするために有効になっている共有サービングコンテナを使用します。マルチモデルエンドポイントは、単一モデルエンドポイントの使用と比較してエンドポイントの使用率を向上させることで、ホスティングコストを削減します。また、SageMaker AI がメモリへのモデルのロードとトラフィックパターンに基づくスケーリングを管理するため、デプロイのオーバーヘッドも削減されます。

SageMaker AI に ML モデルをデプロイするためのその他のベストプラクティスについては、SageMaker AI ドキュメントの「デプロイのベストプラクティス」を参照してください。

## 非同期推論

Amazon SageMaker AI 非同期推論は、受信リクエストをキューに入れ、非同期的に処理する SageMaker AI の機能です。このオプションは、最大 1 GB の大きなペイロードサイズ、長い処理時間、ほぼリアルタイムのレイテンシー要件を持つリクエストに最適です。非同期推論を使用すると、処理するリクエストがない場合にインスタンス数を自動的にゼロにスケーリングすることでコストを節約できるため、エンドポイントがリクエストを処理している場合にのみ料金が発生します。

# バッチ変換

以下を実行する場合は、バッチ変換を使用します。

- データセットを前処理して、トレーニングや推論を妨げるノイズやバイアスをデータセットから取り除く場合。
- 大規模なデータセットから推論を取得する場合。
- 永続的なエンドポイントが不要なときに推論を実行する場合。
- 入力レコードを推論に関連付けて結果の解釈に役立てる場合。

# モニタリング

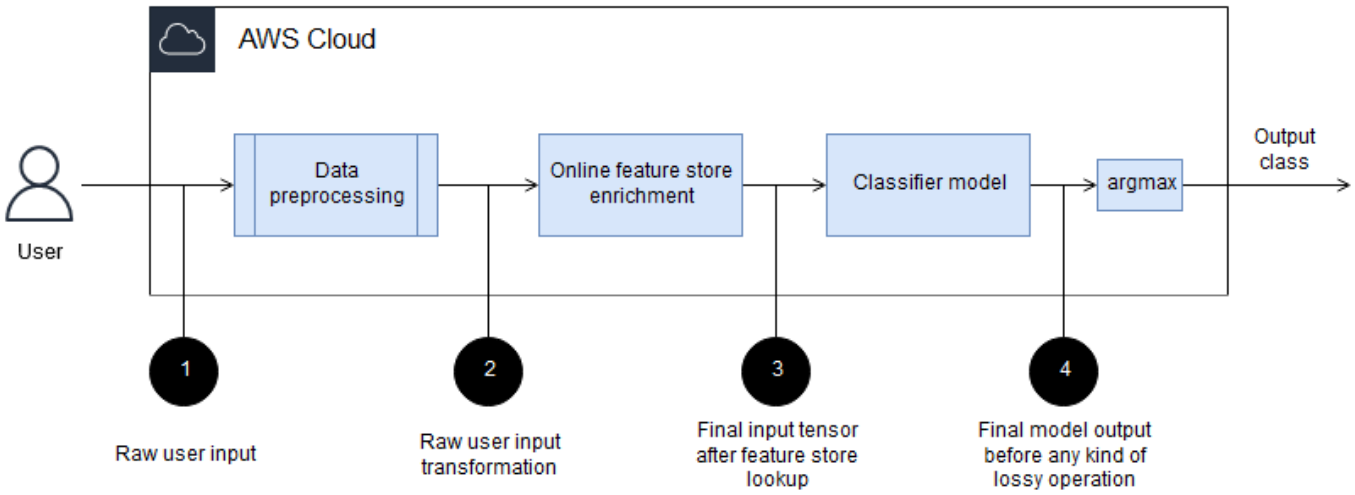
モデルが既に本番稼働中でビジネス価値を提供している場合は、継続的なチェックを実行して、モデルを再トレーニングまたはアクションを実行する必要があるタイミングを特定します。

モニタリングチームは、環境のデータ動作をよりよく理解し、データドリフトの頻度、レート、および突然性を特定するために、事後対応的ではなく積極的に行動する必要があります。チームは、トレーニングセット、検証セット、およびその他のエッジケーススライスで過小評価されている可能性のあるデータ内の新しいエッジケースを特定する必要があります。サービス品質 (QoS) メトリクスを保存し、問題が発生したときにアラームを使用してすぐにアクションを実行し、現在のデータセットを取り込んで修正する戦略を定義する必要があります。これらのプラクティスは、モデルのリクエストとレスポンスのログ記録から始まり、トラブルシューティングや追加のインサイトのリファレンスを提供します。

理想的には、処理中にデータ変換をいくつかの主要なステージに記録する必要があります。

- あらゆる種類の前処理前
- 任意の種類の特徴量ストアエンリッチメントの後
- モデルのすべての主要ステージの後
- など、モデル出力であらゆる種類の損失関数の前に `argmax`

次の図は、これらのステージを示しています。



SageMaker AI Model Monitor を使用して、入力データと出力データを自動的にキャプチャし、Amazon Simple Storage Service (Amazon S3) に保存できます。カスタムサービングコンテナにログを追加することで、他のタイプの中間ログ記録を実装できます。

モデルからデータをログに記録すると、分散ドリフトをモニタリングできます。場合によっては、推論の直後にグラウンドトゥルース (正しくラベル付けされたデータ) を取得できます。一般的な例は、ユーザーに表示する最も関連性の高い広告を予測するモデルです。ユーザーがページを離れるとすぐに、広告をクリックしたかどうかを判断できます。ユーザーが広告をクリックした場合は、その情報をログに記録することができます。この簡単な例では、トレーニングとデプロイの両方で測定できる精度や F1 などのメトリクスを使用することで、モデルの成功度を簡単に定量化できます。データにラベルを付けたシナリオの詳細については、SageMaker AI ドキュメントの「モデル品質のモニタリング」を参照してください。ただし、これらの単純なシナリオはまれです。これは、モデルが実際のビジネス成果のプロキシにすぎない数学的に便利なメトリクスを最適化するように設計されることがよくあるためです。このような場合のベストプラクティスは、モデルが本番環境にデプロイされたときにビジネス成果をモニタリングすることです。

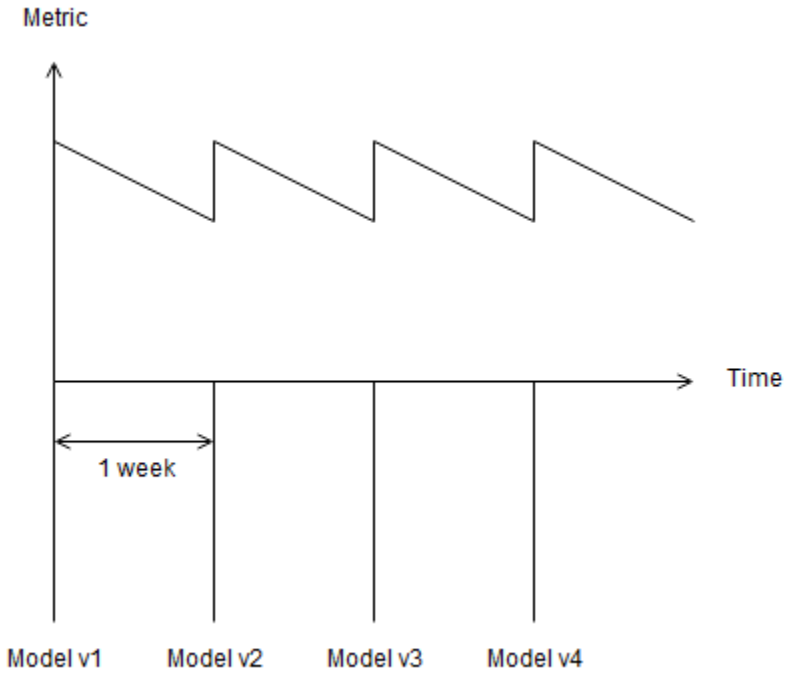
レビューランキングモデルの場合を考えてみましょう。ML モデルの定義されたビジネス成果が、最も関連性の高い有用なレビューをウェブページの上部に表示する場合は、「役に立ちましたか？」などのボタンを追加することで、モデルの成功を測定できます。各レビューの 。このボタンのクリック率を測定することは、モデルが本番環境でどの程度うまく機能しているかを測定するのに役立つビジネス成果の尺度になる可能性があります。

SageMaker AI で入力ラベルまたは出力ラベルのドリフトをモニタリングするには、SageMaker AI Model Monitor のデータ品質機能を使用して、入力と出力の両方をモニタリングできます。カスタムコンテナを構築することで、SageMaker AI Model Monitor に独自のロジックを実装することもできます。

モデルが開発時とランタイムの両方で受け取るデータをモニタリングすることは重要です。エンジニアは、スキーマの変更だけでなく、ディストリビューションの不一致についてもデータをモニタリングする必要があります。スキーマの変更の検出は簡単で、一連のルールで実装できますが、特にアラームを発生させるタイミングを定量化するためのしきい値を定義する必要があるため、分散の不一致は多くの場合、より困難です。モニタリング対象のディストリビューションがわかっている場合、多くの場合、最も簡単な方法はディストリビューションのパラメータをモニタリングすることです。正規分布の場合は、平均値と標準偏差になります。欠損値の割合、最大値、最小値など、他の主要なメトリクスも役立ちます。

また、トレーニングデータと推論データをサンプリングし、その分布を比較する継続的なモニタリングジョブを作成することもできます。これらのジョブは、モデル入力とモデル出力の両方で作成し、

データを時間に対してプロットして、突然または段階的なドリフトを視覚化できます。これは次の図に示されています。




データ分散が大幅に変化する頻度、レート、突然など、データのドリフトプロファイルをよりよく理解するには、新しいモデルバージョンを継続的にデプロイし、パフォーマンスをモニタリングすることをお勧めします。例えば、チームが毎週新しいモデルをデプロイし、モデルのパフォーマンスが毎回大幅に向上することがわかった場合、少なくとも 1 週間未満で新しいモデルを配信する必要があると判断できます。

# 次のステップとリソース

このガイドでは、本番環境に導入する機械学習モデルのライフサイクルを計画する際のいくつかの考慮事項について説明します。データ、トレーニング、デプロイ、モニタリングの 4 つの領域における課題とベストプラクティスについて説明し、関連するその他のリソースも含まれています。

AWS は Well-Architected フレームワークを提供します。これは、クラウドアーキテクトが、さまざまなアプリケーション、ワークロード、テクノロジードメイン向けに、安全で高性能、耐障害性、効率的なインフラストラクチャを構築するのに役立ちます。詳細については、 AWS Well-Architected が提供する Machine Learning Lens を参照してください。

# リソース

Amazon SageMaker AI ドキュメント

- Amazon SageMaker AI Feature Store
- Feature Store のセキュリティとアクセスコントロール
- Shapley 値
- Amazon SageMaker AI デバッガー
- Amazon SageMaker AI パイプライン
- Amazon SageMaker AI のデフォルトプロジェクトテンプレート
- SageMaker AI リアルタイム推論
- Amazon SageMaker AI モデルを自動的にスケーリングする
- Amazon SageMaker AI 非同期推論
- SageMaker AI Model Monitor

AWS デベロッパーツール

- AWS CodePipeline

AWS ブログ投稿

- Amazon SageMaker AI Feature Store の主な機能について
- PyDeequ による大規模なデータ品質のテスト

- Amazon SageMaker AI 実験
- CodePipeline と を使用して Amazon SageMaker エンドポイントを安全にデプロイおよびモニタリングする AWS CodeDeploy
- Amazon SageMaker AI にシャドウ ML モデルをデプロイする
- Amazon SageMaker AI を使用して本番環境で ML モデルを A/B テストする

# ドキュメント履歴

以下の表は、本ガイドの重要な変更点について説明したものです。今後の更新に関する通知を受け取る場合は、RSS フィード をサブスクライブできます。

| 変更 | 説明 | 日付 |
| --- | --- | --- |
| 初版発行 | — | 2021 年 12 月 20 日 |

# AWS 規範ガイダンスの用語集

以下は、 AWS 規範ガイダンスによって提供される戦略、ガイド、パターンで一般的に使用される用語です。エントリを提案するには、用語集の最後のフィードバックの提供リンクを使用します。

## 数字

7 Rs

アプリケーションをクラウドに移行するための 7 つの一般的な移行戦略。これらの戦略は、ガートナーが 2011 年に特定した 5 Rs に基づいて構築され、以下で構成されています。

- リファクタリング/アーキテクチャの再設計 — クラウドネイティブ特徴を最大限に活用して、俊敏性、パフォーマンス、スケーラビリティを向上させ、アプリケーションを移動させ、アーキテクチャを変更します。これには、通常、オペレーティングシステムとデータベースの移植が含まれます。例: オンプレミスの Oracle データベースを Amazon Aurora PostgreSQL 互換エディションに移行する。
- リプラットフォーム (リフトアンドリシェイプ) — アプリケーションをクラウドに移行し、クラウド機能を活用するための最適化レベルを導入します。例: お客様のオンプレミスの Oracle データベースを AWS クラウドの Oracle 用の Amazon Relational Database Service (Amazon RDS) に移行する。
- 再購入 (ドロップアンドショップ) — 通常、従来のライセンスから SaaS モデルに移行して、別の製品に切り替えます。例: 顧客関係管理 (CRM) システムを Salesforce.com に移行する。
- リホスト (リフトアンドシフト) — クラウド機能を活用するための変更を加えずに、アプリケーションをクラウドに移行します。例: お客様のオンプレミスの Oracle データベースを AWS クラウドの EC2 インスタンス上の Oracle に移行する。
- 再配置 (ハイパーバイザーレベルのリフトアンドシフト) – 新しいハードウェアを購入したり、アプリケーションを書き換えたり、既存の運用を変更したりすることなく、インフラストラクチャをクラウドに移行できます。オンプレミスプラットフォームから同じプラットフォームのクラウドサービスにサーバーを移行します。例： Microsoft Hyper-Vアプリケーションを に移行します AWS。
- 保持 (再アクセス) — アプリケーションをお客様のソース環境で保持します。これには、主要なリファクタリングを必要とするアプリケーションや、お客様がその作業を後日まで延期したいアプリケーション、およびそれらを移行するためのビジネス上の正当性がないため、お客様が保持するレガシーアプリケーションなどがあります。
- 廃止 — お客様のソース環境で不要になったアプリケーションを停止または削除します。

# A

ABAC

「属性ベースのアクセス制御」をご覧ください。

抽象化されたサービス

「マネージドユーザー」をご覧ください。

ACID

「原子性、一貫性、分離性、耐久性 (ACID)」をご覧ください。

アクティブ/アクティブ移行

(双方向レプリケーションツールまたは二重書き込み操作を使用して) ソースデータベースとターゲットデータベースを同期させ、移行中に両方のデータベースが接続アプリケーションからのトランザクションを処理するデータベース移行方法。この方法では、1 回限りのカットオーバーの必要がなく、管理された小規模なバッチで移行できます。アクティブ/パッシブ移行よりも柔軟な方法ですが、さらに多くの作業が必要となります。

アクティブ/パッシブ移行

ソースデータベースとターゲットデータベースを同期させながら、データがターゲットデータベースにレプリケートされている間、接続しているアプリケーションからのトランザクションをソースデータベースのみで処理するデータベース移行方法。移行中、ターゲットデータベースはトランザクションを受け付けません。

集計関数

複数行に処理を行い、グループ全体を対象に単一の戻り値を計算する SQL 関数。集計関数の例としては、SUM や MAX などがあります。

AI

「人工知能」をご覧ください。

AIOps

「AI オペレーション」をご覧ください。

匿名化

データセット内の個人情報を完全に削除するプロセス。匿名化は個人のプライバシー保護に役立ちます。匿名化されたデータは、もはや個人データとは見なされません。

アンチパターン

繰り返し起こる問題に対して頻繁に用いられる解決策で、その解決策が逆効果であったり、効果がなかったり、代替案よりも効果が低かったりするもの。

アプリケーション制御

マルウェアからシステムを保護するために、承認されたアプリケーションのみを使用できるようにするセキュリティアプローチ。

アプリケーションポートフォリオ

アプリケーションの構築と維持にかかるコスト、およびそのビジネス価値を含む、組織が使用する各アプリケーションに関する詳細情報の集まり。この情報は、ポートフォリオの検出と分析プロセス の重要な要素であり、移行、モダナイズ、最適化するアプリケーションを特定し、優先順位を付けるのに役立ちます。

人工知能 (AI)

コンピューティングテクノロジーを使用し、学習、問題の解決、パターンの認識など、通常は人間に関連づけられる認知機能の実行に特化したコンピュータサイエンスの分野。詳細については、「人工知能 (AI) とは何ですか?」をご覧ください。

AI オペレーション (AIOps)

機械学習技術を使用して運用上の問題を解決し、運用上のインシデントと人の介入を減らし、サービス品質を向上させるプロセス。 AWS 移行戦略での AIOps の使用方法については、オペレーション統合ガイド を参照してください。

非対称暗号化

暗号化用のパブリックキーと復号用のプライベートキーから成る 1 組のキーを使用した、暗号化のアルゴリズム。パブリックキーは復号には使用されないため共有しても問題ありませんが、プライベートキーの利用は厳しく制限する必要があります。

原子性、一貫性、分離性、耐久性 (ACID)

エラー、停電、その他の問題が発生した場合でも、データベースのデータ有効性と運用上の信頼性を保証する一連のソフトウェアプロパティ。

属性ベースのアクセス制御 (ABAC)

部署、役職、チーム名など、ユーザーの属性に基づいてアクセス許可をきめ細かく設定する方法。詳細については、 AWS Identity and Access Management (IAM) ドキュメントの「 の ABAC AWS」を参照してください。

信頼できるデータソース

最も信頼性のある情報源とされるデータのプライマリーバージョンを保存する場所。匿名化、編集、仮名化など、データを処理または変更する目的で、信頼できるデータソースから他の場所にデータをコピーすることができます。

アベイラビリティーゾーン (AZ)

他のアベイラビリティーゾーンの障害から AWS リージョン 隔離され、同じリージョン内の他のアベイラビリティーゾーンへの低コストで低レイテンシーのネットワーク接続を提供する 内の別の場所。

AWS クラウド導入フレームワーク (AWS CAF)

組織がクラウドへの移行を成功させるための効率的で効果的な計画を立て AWS るための、 のガイドラインとベストプラクティスのフレームワークです。 AWS CAF は、ビジネス、人材、ガバナンス、プラットフォーム、セキュリティ、運用という 6 つの重点分野にガイダンスを整理しています。ビジネス、人材、ガバナンスの観点では、ビジネススキルとプロセスに重点を置き、プラットフォーム、セキュリティ、オペレーションの視点は技術的なスキルとプロセスに焦点を当てています。例えば、人材の観点では、人事 (HR)、人材派遣機能、および人材管理を扱うステークホルダーを対象としています。この観点から、 AWS CAF は、クラウド導入を成功させるための組織の準備に役立つ人材開発、トレーニング、コミュニケーションに関するガイダンスを提供します。詳細については、AWS CAF ウェブサイトと AWS CAF のホワイトペーパー を参照してください。

AWS ワークロード認定フレームワーク (AWS WQF)

データベース移行ワークロードを評価し、移行戦略を推奨し、作業見積もりを提供するツール。AWS WQF は AWS Schema Conversion Tool (AWS SCT) に含まれています。データベーススキーマとコードオブジェクト、アプリケーションコード、依存関係、およびパフォーマンス特性を分析し、評価レポートを提供します。

# B

不正なボット

個人や組織に混乱や損害を与えることを目的としたボット。

BCP

「ビジネス継続性計画 (BCP)」をご覧ください。

動作グラフ

リソースの動作とインタラクションを経時的に示した、一元的なインタラクティブビュー。Amazon Detective の動作グラフを使用すると、失敗したログオンの試行、不審な API 呼び出し、その他同様のアクションを調べることができます。詳細については、Detective ドキュメントの「動作グラフのデータ」を参照してください。

ビッグエンディアンシステム

最上位バイトを最初に格納するシステム。「エンディアン性」もご覧ください。

二項分類

バイナリ結果 (2 つの可能なクラスのうちの 1 つ) を予測するプロセス。例えば、お客様の機械学習モデルで「この E メールはスパムですか、それともスパムではありませんか」などの問題を予測する必要があるかもしれません。または「この製品は書籍ですか、車ですか」などの問題を予測する必要があるかもしれません。

ブルームフィルター

要素がセットのメンバーであるかどうかをテストするために使用される、確率的でメモリ効率の高いデータ構造。

ブルー/グリーンデプロイ

それぞれが独立しているが、同一の環境を 2 つ作成するデプロイ戦略。現在のアプリケーションバージョンを 1 つの環境 (ブルー) で実行し、新しいアプリケーションバージョンを別の環境 (グリーン) で実行します。この戦略は、最小限の影響で迅速にロールバックするのに役立ちます。

ボット

インターネット経由で自動タスクを実行し、人間のアクティビティややり取りをシミュレートするソフトウェアアプリケーション。インターネット上の情報のインデックスを作成するウェブクローラーなど、一部のボットは有用または有益です。悪質なボットと呼ばれる他のボットの中には、個人や組織を混乱させたり、損害を与えたりすることを意図したものもあります。

ボットネット

マルウェアに感染しており、ボットハーダーまたはボットオペレーターと呼ばれる単一の当事者によって制御されているボットのネットワーク。ボットネットは、ボットとその影響力を拡大する仕組みとして、非常によく知られています。

ブランチ

コードリポジトリに含まれる領域。リポジトリに最初に作成するブランチは、メインブランチといいます。既存のブランチから新しいブランチを作成し、その新しいブランチで機能を開発した

り、バグを修正したりできます。機能を構築するために作成するブランチは、通常、機能ブランチと呼ばれます。機能をリリースする準備ができたら、機能ブランチをメインブランチに統合します。詳細については、「ブランチの概要」(GitHub ドキュメント) を参照してください。

ブレークグラスアクセス

例外的な状況では、承認されたプロセスを通じて、ユーザーが AWS アカウント 通常アクセス許可を持たない にすばやくアクセスできるようにします。詳細については、 AWS Well-Architected ガイダンスの「ブレークグラス手順の実装」インジケータを参照してください。

ブラウンフィールド戦略

環境の既存インフラストラクチャ。システムアーキテクチャにブラウンフィールド戦略を導入する場合、現在のシステムとインフラストラクチャの制約に基づいてアーキテクチャを設計します。既存のインフラストラクチャを拡張している場合は、ブラウンフィールド戦略とグリーンフィールド戦略を融合させることもできます。

バッファキャッシュ

アクセス頻度が最も高いデータが保存されるメモリ領域。

ビジネス能力

価値を生み出すためにビジネスが行うこと (営業、カスタマーサービス、マーケティングなど)。マイクロサービスのアーキテクチャと開発の決定は、ビジネス能力によって推進できます。詳細については、AWSでのコンテナ化されたマイクロサービスの実行ホワイトペーパーの「ビジネス機能を中心に組織化」セクションを参照してください。

ビジネス継続性計画 (BCP)

大規模移行など、中断を伴うイベントが運用に与える潜在的な影響に対処し、ビジネスを迅速に再開できるようにする計画。

# C

CAF

「AWS クラウド導入フレームワーク」を参照してください

カナリアデプロイ

エンドユーザーへのバージョンリリースを、時間をかけて段階的に行うこと。確信が持てたら新規バージョンをデプロイして、現在のバージョン全体を置き換えます。

CCoE

「Cloud Center of Excellence」を参照してください。

CDC

「変更データキャプチャ」を参照してください。

変更データキャプチャ (CDC)

データソース (データベーステーブルなど) の変更を追跡し、その変更に関するメタデータを記録するプロセス。CDC は、ターゲットシステムでの変更を監査またはレプリケートして同期を維持するなど、さまざまな目的に使用できます。

カオスエンジニアリング

障害や破壊的なイベントを意図的に導入して、システムの耐障害性をテストすること。AWS Fault Injection Service (AWS FIS) を使用して、 AWS ワークロードにストレスを与え、その応答を評価する実験を実行できます。

CI/CD

「継続的インテグレーションと継続的デリバリー」を参照してください。

分類

予測を生成するのに役立つ分類プロセス。分類問題の機械学習モデルは、離散値を予測します。離散値は、常に互いに区別されます。例えば、モデルがイメージ内に車があるかどうかを評価する必要がある場合があります。

クライアント側の暗号化

ターゲットがデータ AWS のサービス を受信する前のローカルでのデータの暗号化。

Cloud Center of Excellence (CCoE)

クラウドのベストプラクティスの作成、リソースの移動、移行のタイムラインの確立、大規模変革を通じて組織をリードするなど、組織全体のクラウド導入の取り組みを推進する学際的なチーム。詳細については、 AWS クラウド エンタープライズ戦略ブログの CCoE 投稿を参照してください。

クラウドコンピューティング

リモートデータストレージと IoT デバイス管理に通常使用されるクラウドテクノロジー。クラウドコンピューティングは、一般的に、エッジコンピューティングに接続されています。

クラウド運用モデル

IT 組織において、1 つ以上のクラウド環境を構築、成熟、最適化するために使用される運用モデル。詳細については、「クラウド運用モデルの構築」を参照してください。

導入のクラウドステージ

組織が、 AWS クラウドへの移行時に通常実行する 4 つの段階。

- プロジェクト — 概念実証と学習を目的として、クラウド関連のプロジェクトをいくつか実行する
- 基礎固め — お客様のクラウドの導入を拡大するための基礎的な投資 (ランディングゾーンの作成、CCoE の定義、運用モデルの確立など)
- 移行 — 個々のアプリケーションの移行
- 再発明 — 製品とサービスの最適化、クラウドでのイノベーション

これらのステージは、 AWS クラウド エンタープライズ戦略ブログのブログ記事「クラウドファーストへのジャーニー」と「導入のステージ」で Stephen Orban によって定義されました。移行戦略との関連性については、 AWS 「移行準備ガイド」を参照してください。

CMDB

「構成管理データベース (CMDB)」を参照してください。

コードリポジトリ

ソースコードやその他の資産 (ドキュメント、サンプル、スクリプトなど) が保存され、バージョン管理プロセスを通じて更新される場所。一般的なクラウドリポジトリには、GitHub や Bitbucket Cloud があります。コードの各バージョンはブランチと呼ばれます。マイクロサービスの構造では、各リポジトリは 1 つの機能専用です。1 つの CI/CD パイプラインで複数のリポジトリを使用できます。

コールドキャッシュ

空である、または、かなり空きがある、もしくは、古いデータや無関係なデータが含まれているバッファキャッシュ。データベースインスタンスはメインメモリまたはディスクから読み取る必要があり、バッファキャッシュから読み取るよりも時間がかかるため、パフォーマンスに影響します。

コールドデータ

めったにアクセスされず、通常は過去のデータです。この種類のデータをクエリする場合、通常は低速なクエリでも問題ありません。このデータを低パフォーマンスで安価なストレージ階層またはクラスに移動すると、コストを削減することができます。

コンピュータビジョン (CV)

機械学習を使用してデジタルイメージやビデオといった、ビジュアル形式の情報を分析および抽出する AI の分野。例えば、Amazon SageMaker AI では、CV 用の画像処理アルゴリズムを利用できます。

設定ドリフト

ワークロードにおいて、設定が想定した状態から変化すること。これによって、ワークロードが非準拠になる可能性がありますが、この状態は、徐々に生じ、意図的なものではありません。

構成管理データベース (CMDB)

データベースとその IT 環境 (ハードウェアとソフトウェアの両方のコンポーネントとその設定を含む) に関する情報を保存、管理するリポジトリ。通常、CMDB のデータは、移行のポートフォリオの検出と分析の段階で使用します。

コンフォーマンスパック

コンプライアンスチェックとセキュリティチェックをカスタマイズするためにアセンブルできる AWS Config ルールと修復アクションのコレクション。YAML テンプレートを使用して、コンフォーマンスパックを AWS アカウント および リージョンの単一のエンティティとしてデプロイすることも、組織全体にデプロイすることもできます。詳細については、 AWS Config ドキュメントの「コンフォーマンスパック」を参照してください。

継続的インテグレーションと継続的デリバリー (CI/CD)

ソフトウェアリリースプロセスのソース、ビルド、テスト、ステージング、本番の各ステージを自動化するプロセス。CI/CD は一般的にパイプラインと呼ばれます。プロセスの自動化、生産性の向上、コード品質の向上、配信の加速化を可能にします。詳細については、「継続的デリバリーの利点」を参照してください。CD は継続的デプロイ (Continuous Deployment) の略語でもあります。詳細については「継続的デリバリーと継続的なデプロイ」を参照してください。

CV

「コンピュータビジョン」を参照してください。

# D

保管中のデータ

ストレージ内にあるデータなど、常に自社のネットワーク内にあるデータ。

データ分類

ネットワーク内のデータを重要度と機密性に基づいて識別、分類するプロセス。データに適した保護および保持のコントロールを判断する際に役立つため、あらゆるサイバーセキュリティのリスク管理戦略において重要な要素です。データ分類は、 AWS Well-Architected フレームワークのセキュリティの柱のコンポーネントです。詳細については、「データ分類」を参照してください。

データドリフト

実稼働データと ML モデルのトレーニングに使用されたデータとの間に有意な差異が生じたり、入力データが時間の経過と共に有意に変化したりすることです。データドリフトは、ML モデル予測の全体的な品質、精度、公平性を低下させる可能性があります。

転送中のデータ

ネットワーク内 (ネットワークリソース間など) を活発に移動するデータ。

データメッシュ

非一元的で分散型のデータ所有権を持つとともに、一元的な管理およびガバナンスを行えるアーキテクチャフレームワーク。

データ最小化

厳密に必要なデータのみを収集し、処理するという原則。でデータ最小化を実践 AWS クラウドすることで、プライバシーリスク、コスト、分析のカーボンフットプリントを削減できます。

データ境界

AWS 環境内の一連の予防ガードレール。信頼された ID のみが、期待されるネットワークから信頼されたリソースにアクセスできるようにします。詳細については、「 でのデータ境界の構築 AWS」を参照してください。

データの前処理

raw データをお客様の機械学習モデルで簡単に解析できる形式に変換すること。データの前処理とは、特定の列または行を削除して、欠落している、矛盾している、または重複する値に対処することを意味します。

データ出所

データの生成、送信、保存の方法など、データのライフサイクル全体を通じてデータの出所と履歴を追跡するプロセス。

データ件名

データを収集、処理している個人。

データウェアハウス

分析などのビジネスインテリジェンスをサポートするデータ管理システム。データウェアハウスには、一般的に、大量の履歴データが含まれており、多くの場合、それらはクエリや分析に使用されます。

データベース定義言語 (DDL)

データベース内のテーブルやオブジェクトの構造を作成または変更するためのステートメントまたはコマンド。

データベース操作言語 (DML)

データベース内の情報を変更 (挿入、更新、削除) するためのステートメントまたはコマンド。

DDL

「データベース定義言語」を参照してください。

ディープアンサンブル

予測のために複数の深層学習モデルを組み合わせます。ディープアンサンブルを使用して、より正確な予測を取得したり、予測の不確実性を推定したりできます。

深層学習

人工ニューラルネットワークの複数層を使用して、入力データと対象のターゲット変数の間のマッピングを識別する機械学習サブフィールド。

多層防御

一連のセキュリティメカニズムとコントロールをコンピュータネットワーク全体に層状に重ねて、ネットワークとその内部にあるデータの機密性、整合性、可用性を保護する情報セキュリティの手法。この戦略を に採用するときは AWS、リソースの保護に役立つように、 AWS Organizations 構造の異なるレイヤーに複数のコントロールを追加します。たとえば、多層防御アプローチでは、多要素認証、ネットワークセグメンテーション、暗号化を組み合わせることができます。

委任管理者

では AWS Organizations、互換性のあるサービスが AWS メンバーアカウントを登録して組織のアカウントを管理し、そのサービスのアクセス許可を管理できます。このアカウントを、そのサービスの委任管理者と呼びます。詳細、および互換性のあるサービスの一覧は、 AWS

Organizations ドキュメントの「AWS Organizationsで使用できるサービス」を参照してください。

トラブルシューティング

アプリケーション、新機能、コードの修正をターゲットの環境で利用できるようにするプロセス。デプロイでは、コードベースに変更を施した後、アプリケーションの環境でそのコードベースを構築して実行します。

開発環境

「環境」を参照してください。

検出管理

イベントが発生したときに、検出、ログ記録、警告を行うように設計されたセキュリティコントロール。これらのコントロールは副次的な防衛手段であり、実行中の予防的コントロールをすり抜けたセキュリティイベントをユーザーに警告します。詳細については、「 AWSでのセキュリティコントロールの実装」の「検出的コントロール」を参照してください。

開発バリューストリームマッピング (DVSM)

ソフトウェア開発ライフサイクルのスピードと品質に悪影響を及ぼす制約を特定し、優先順位を付けるために使用されるプロセス。DVSM は、もともとリーンマニュファクチャリング・プラクティスのために設計されたバリューストリームマッピング・プロセスを拡張したものです。ソフトウェア開発プロセスを通じて価値を創造し、動かすために必要なステップとチームに焦点を当てています。

デジタルツイン

建物、工場、産業機器、生産ラインなど、現実世界のシステムを仮想的に表現したものです。デジタルツインは、予知保全、リモートモニタリング、生産最適化をサポートします。

ディメンションテーブル

スタースキーマにおいて、ファクトテーブルの定量データに関するデータ属性が含まれる小さいテーブル。ディメンションテーブルの属性は、通常、テキストフィールド、またはテキストのように扱える個別の数値で示されます。これらの属性は、一般的に、クエリの制約、フィルタリング、結果セットのラベル付けに使用されます。

ディザスタ

ワークロードまたはシステムが、導入されている主要な場所でのビジネス目標の達成を妨げるイベント。これらのイベントは、自然災害、技術的障害、または意図しない設定ミスやマルウェア攻撃などの人間の行動の結果である場合があります。

ディザスタリカバリ (DR)

ディザスタによるダウンタイムとデータ損失を最小限に抑えるための戦略とプロセス。詳細については、 AWS Well-Architected フレームワークの「 でのワークロードのディザスタリカバリ AWS: クラウドでのリカバリ」を参照してください。

DML

「データベース操作言語」を参照してください。

ドメイン駆動型設計

各コンポーネントが提供している変化を続けるドメイン、またはコアビジネス目標にコンポーネントを接続して、複雑なソフトウェアシステムを開発するアプローチ。この概念は、エリック・エヴァンスの著書、Domain-Driven Design: Tackling Complexity in the Heart of Software (ドメイン駆動設計:ソフトウェアの中心における複雑さへの取り組み) で紹介されています (ボストン: Addison-Wesley Professional、2003)。strangler fig パターンでドメイン駆動型設計を使用する方法の詳細については、「コンテナと Amazon API Gateway を使用して、従来の Microsoft ASP.NET (ASMX) ウェブサービスを段階的にモダナイズ」を参照してください。

DR

「ディザスタリカバリ」を参照してください。

ドリフト検出

ベースライン設定からの偏差を追跡します。たとえば、 AWS CloudFormation を使用してシステムリソースのドリフトを検出したり、 を使用して AWS Control Tower 、ガバナンス要件への準拠に影響するランディングゾーンの変更を検出したりできます。

DVSM

「開発バリューストリームマッピング」を参照してください。

# E

EDA

「探索的データ分析」を参照してください。

EDI

「電子データ交換」を参照してください。

エッジコンピューティング

IoT ネットワークのエッジにあるスマートデバイスの計算能力を高めるテクノロジー。クラウドコンピューティングと比較すると、エッジコンピューティングは通信レイテンシーを短縮し、応答時間を改善できます。

電子データ交換 (EDI)

組織間で行う、ビジネスドキュメントの自動交換。詳細については、「電子データ交換とは」を参照してください。

暗号化

人間が読み取り可能なプレーンテキストデータを暗号文に変換するコンピューティング処理。

暗号化キー

暗号化アルゴリズムが生成した、ランダム化されたビットからなる暗号文字列。キーの長さは決まっておらず、各キーは予測できないように、一意になるように設計されています。

エンディアン

コンピュータメモリにバイトが格納される順序。ビッグエンディアンシステムでは、最上位バイトが最初に格納されます。リトルエンディアンシステムでは、最下位バイトが最初に格納されます。

エンドポイント

「サービスエンドポイント」を参照してください。

エンドポイントサービス

仮想プライベートクラウド (VPC) 内でホストして、他のユーザーと共有できるサービス。を使用してエンドポイントサービスを作成し AWS PrivateLink 、他の AWS アカウント または AWS Identity and Access Management (IAM) プリンシパルにアクセス許可を付与できます。これらのアカウントまたはプリンシパルは、インターフェイス VPC エンドポイントを作成することで、エンドポイントサービスにプライベートに接続できます。詳細については、Amazon Virtual Private Cloud (Amazon VPC) ドキュメントの「エンドポイントサービスを作成する」を参照してください。

エンタープライズリソースプランニング (ERP)

エンタープライズの主要なビジネスプロセス (会計、MES、プロジェクト管理など) を自動化および管理するシステム。

エンベロープ暗号化

暗号化キーを、別の暗号化キーを使用して暗号化するプロセス。詳細については、 AWS Key Management Service (AWS KMS) ドキュメントの「エンベロープ暗号化」を参照してください。

環境

実行中のアプリケーションのインスタンス。クラウドコンピューティングにおける一般的な環境の種類は以下のとおりです。

- 開発環境 — アプリケーションのメンテナンスを担当するコアチームのみが使用できる、実行中のアプリケーションのインスタンス。開発環境は、上位の環境に昇格させる変更をテストするときに使用します。このタイプの環境は、テスト環境と呼ばれることもあります。
- 下位環境 — 初期ビルドやテストに使用される環境など、アプリケーションのすべての開発環境。
- 本番環境 — エンドユーザーがアクセスできる、実行中のアプリケーションのインスタンス。CI/CD パイプラインでは、本番環境が最後のデプロイ環境になります。
- 上位環境 — コア開発チーム以外のユーザーがアクセスできるすべての環境。これには、本番環境、本番前環境、ユーザー承認テスト環境などが含まれます。

エピック

アジャイル方法論で、お客様の作業の整理と優先順位付けに役立つ機能カテゴリ。エピックでは、要件と実装タスクの概要についてハイレベルな説明を提供します。例えば、 AWS CAF セキュリティエピックには、ID とアクセスの管理、検出コントロール、インフラストラクチャセキュリティ、データ保護、インシデント対応が含まれます。 AWS 移行戦略のエピックの詳細については、プログラム実装ガイドを参照してください。

ERP

「エンタープライズリソース計画」を参照してください。

探索的データ分析 (EDA)

データセットを分析してその主な特性を理解するプロセス。お客様は、データを収集または集計してから、パターンの検出、異常の検出、および前提条件のチェックのための初期調査を実行します。EDA は、統計の概要を計算し、データの可視化を作成することによって実行されます。

# F

ファクトテーブル

スタースキーマの中央にあるテーブル。ビジネスオペレーションに関する定量的データが保存されます。一般的に、ファクトテーブルは、2 種類の列で構成されます。1 つは測定値が含まれる列、もう 1 つはディメンションテーブルへの外部キーが含まれる列です。

フェイルファスト

開発ライフサイクルを短縮するために、頻繁かつ段階的にテストを行う哲学であり、アジャイルアプローチでは、この考え方がきわめて重要です。

障害分離境界

では AWS クラウド、障害の影響を制限し、ワークロードの耐障害性を高めるのに役立つアベイラビリティーゾーン AWS リージョン、コントロールプレーン、データプレーンなどの境界。詳細については、「AWS 障害分離境界」を参照してください。

機能ブランチ

「ブランチ」を参照してください。

特徴量

お客様が予測に使用する入力データ。例えば、製造コンテキストでは、特徴量は製造ラインから定期的にキャプチャされるイメージの可能性もあります。

特徴量重要度

モデルの予測に対する特徴量の重要性。これは通常、Shapley Additive Deskonations (SHAP) や積分勾配など、さまざまな手法で計算できる数値スコアで表されます。詳細については、「 を使用した機械学習モデルの解釈可能性 AWS」を参照してください。

機能変換

追加のソースによるデータのエンリッチ化、値のスケーリング、単一のデータフィールドからの複数の情報セットの抽出など、機械学習プロセスのデータを最適化すること。これにより、機械学習モデルはデータの恩恵を受けることができます。例えば、「2021-05-27 00:15:37」の日付を「2021 年」、「5 月」、「木」、「15」に分解すると、学習アルゴリズムがさまざまなデータコンポーネントに関連する微妙に異なるパターンを学習するのに役立ちます。

数ショットプロンプト

LLM に、タスクと望ましい出力を示す例を少数提示した後に、類似のタスクを実行させること。この手法は、プロンプトに記述された例 (ショット) からモデルが学習する「インコンテキスト学

習」の一種です。数ショットプロンプトは、特定のフォーマット、推論、専門知識が必要なタスクに効果的です。「ゼロショットプロンプト」も参照してください。

FGAC

「きめ細かなアクセス制御」を参照してください。

きめ細かなアクセス制御 (FGAC)

複数の条件を使用してアクセス要求を許可または拒否すること。

フラッシュカット移行

変更データのキャプチャによる継続的なデータ複製を利用して、段階的なアプローチではなく、可能な限り短時間でデータを移行するデータベース移行方法。目的はダウンタイムを最小限に抑えることです。

FM

「基盤モデル」を参照してください。

基盤モデル (FM)

大規模な深層学習ニューラルネットワークであり、一般化およびラベル付けされていないデータからなる大規模データセットでトレーニングされています。FM により、言語理解、テキストおよび画像生成、自然言語での会話といった、一般的な各種タスクを実行できます。詳細については、「基盤モデルとは何ですか?」を参照してください。

# G

生成 AI

AI モデルのサブセット。大量のデータでトレーニングされており、シンプルなテキストプロンプトを使用して、画像、動画、テキスト、オーディオなどの新しいコンテンツやアーティファクトを作成できます。詳細については、「生成 AI とは何ですか?」を参照してください。

ジオブロッキング

「地理的制限」を参照してください。

地理的制限 (ジオブロッキング)

特定の国のユーザーがコンテンツ配信にアクセスできないようにするための、Amazon CloudFront のオプション。アクセスを許可する国と禁止する国は、許可リストまたは禁止リスト

を使って指定します。詳細については、CloudFront ドキュメントの「コンテンツの地理的ディストリビューションの制限」を参照してください。

Gitflow ワークフロー

下位環境と上位環境が、ソースコードリポジトリでそれぞれ異なるブランチを使用する方法。Gitflow ワークフローは古いと見なされている方法であり、トランクベースのワークフローは推奨されている新しい方法です。

ゴールデンイメージ

システムまたはソフトウェアのスナップショットであり、システムまたはソフトウェアの新規インスタンスをデプロイするテンプレートとして使用されます。製造の例で言えば、ゴールデンイメージを使用すると、複数のデバイスにソフトウェアをプロビジョニングして、デバイス製造オペレーションの速度、スケーラビリティ、生産性を向上させることができます。

グリーンフィールド戦略

新しい環境に既存のインフラストラクチャが存在しないこと。システムアーキテクチャにグリーンフィールド戦略を導入する場合、既存のインフラストラクチャ (別名ブラウンフィールド) との互換性の制約を受けることなく、あらゆる新しいテクノロジーを選択できます。既存のインフラストラクチャを拡張している場合は、ブラウンフィールド戦略とグリーンフィールド戦略を融合させることもできます。

ガードレール

組織単位 (OU) 全般のリソース、ポリシー、コンプライアンスを管理するのに役立つ概略的なルール。予防ガードレールは、コンプライアンス基準に一致するようにポリシーを実施します。これらは、サービスコントロールポリシーと IAM アクセス許可の境界を使用して実装されます。検出ガードレールは、ポリシー違反やコンプライアンス上の問題を検出し、修復のためのアラートを発信します。これらは AWS Config、、 AWS Security Hub CSPM、Amazon GuardDuty、、 AWS Trusted Advisor Amazon Inspector、およびカスタム AWS Lambda チェックを使用して実装されます。

# H

HA

「高可用性」を参照してください。

異種混在データベースの移行

別のデータベースエンジンを使用するターゲットデータベースへお客様の出典データベースの移行 (例えば、Oracle から Amazon Aurora)。異種間移行は通常、アーキテクチャの再設計作業の一部であり、スキーマの変換は複雑なタスクになる可能性があります。AWS は、スキーマの変換に役立つ AWS SCTを提供します。

高可用性 (HA)

課題や災害が発生した場合に、介入なしにワークロードを継続的に運用できること。HA システムは、自動的にフェイルオーバーし、一貫して高品質のパフォーマンスを提供し、パフォーマンスへの影響を最小限に抑えながらさまざまな負荷や障害を処理するように設計されています。

ヒストリアンのモダナイゼーション

製造業のニーズによりよく応えるために、オペレーションテクノロジー (OT) システムをモダナイズし、アップグレードするためのアプローチ。ヒストリアンは、工場内のさまざまなソースからデータを収集して保存するために使用されるデータベースの一種です。

ホールドアウトデータ

機械学習モデルのトレーニング用データセットから保留される、ラベル付き履歴データの一部。ホールドアウトデータを使用すると、モデル予測をホールドアウトデータと比較して、モデルのパフォーマンスを評価できます。

同種データベースの移行

お客様の出典データベースを、同じデータベースエンジンを共有するターゲットデータベース (Microsoft SQL Server から Amazon RDS for SQL Server など) に移行する。同種間移行は、通常、リホストまたはリプラットフォーム化の作業の一部です。ネイティブデータベースユーティリティを使用して、スキーマを移行できます。

ホットデータ

リアルタイムデータや最近の翻訳データなど、頻繁にアクセスされるデータ。通常、このデータには高速なクエリ応答を提供する高性能なストレージ階層またはクラスが必要です。

ホットフィックス

本番環境の重大な問題を修正するために緊急で配布されるプログラム。緊急性が高いため、通常の DevOps のリリースワークフローからは外れた形で実施されます。

ハイパーケア期間

カットオーバー直後、移行したアプリケーションを移行チームがクラウドで管理、監視して問題に対処する期間。通常、この期間は 1～4 日です。ハイパーケア期間が終了すると、アプリケーションに対する責任は一般的に移行チームからクラウドオペレーションチームに移ります。

## I

IaC

「Infrastructure as Code」を参照してください。

ID ベースのポリシー

AWS クラウド 環境内のアクセス許可を定義する 1 つ以上の IAM プリンシパルにアタッチされたポリシー。

アイドル状態のアプリケーション

90 日間の平均的な CPU およびメモリ使用率が 5～20% のアプリケーション。移行プロジェクトでは、これらのアプリケーションを廃止するか、オンプレミスに保持するのが一般的です。

IIoT

「インダストリアル IoT」を参照してください。

イミュータブルインフラストラクチャ

既存インフラストラクチャの更新、パッチ適用、変更などを行わずに、本番環境ワークロードに使用する新規インフラストラクチャをデプロイするモデル。本質的に、イミュータブルインフラストラクチャは、ミュータブルインフラストラクチャよりも一貫性、信頼性、予測性に優れています。詳細については、 AWS Well-Architected フレームワークにある「イミュータブルインフラストラクチャを使用してデプロイする」のベストプラクティスを参照してください。

インバウンド (受信) VPC

AWS マルチアカウントアーキテクチャでは、アプリケーションの外部からネットワーク接続を受け入れ、検査し、ルーティングする VPC。AWS Security Reference Architecture では、アプリケーションとより広範なインターネット間の双方向のインターフェイスを保護するために、インバウンド、アウトバウンド、インスペクションの各 VPC を使用してネットワークアカウントを設定することを推奨しています。

増分移行

アプリケーションを 1 回ですべてカットオーバーするのではなく、小さい要素に分けて移行するカットオーバー戦略。例えば、最初は少数のマイクロサービスまたはユーザーのみを新しいシステムに移行する場合があります。すべてが正常に機能することを確認できたら、残りのマイクロサービスやユーザーを段階的に移行し、レガシーシステムを廃止できるようにします。この戦略により、大規模な移行に伴うリスクが軽減されます。

インダストリー 4.0

2016 年に Klaus Schwab 氏が提唱した用語で、接続、リアルタイムデータ、オートメーション、分析、AI/ML の進歩による、ビジネスプロセスのモダナイズを意味します。

インフラストラクチャ

アプリケーションの環境に含まれるすべてのリソースとアセット。

Infrastructure as Code (IaC)

アプリケーションのインフラストラクチャを一連の設定ファイルを使用してプロビジョニングし、管理するプロセス。IaC は、新しい環境を再現可能で信頼性が高く、一貫性のあるものにするため、インフラストラクチャを一元的に管理し、リソースを標準化し、スケールを迅速に行えるように設計されています。

インダストリアル IoT (IIoT)

製造、エネルギー、自動車、ヘルスケア、ライフサイエンス、農業などの産業部門におけるインターネットに接続されたセンサーやデバイスの使用。詳細については、「インダストリアル IoT (IIoT) デジタルトランスフォーメーション戦略の構築」を参照してください。

インスペクション VPC

AWS マルチアカウントアーキテクチャでは、VPC (同一または異なる 内 AWS リージョン)、インターネット、オンプレミスネットワーク間のネットワークトラフィックの検査を管理する一元化された VPCs。AWS Security Reference Architecture では、アプリケーションとより広範なインターネット間の双方向のインターフェイスを保護するために、インバウンド、アウトバウンド、インスペクションの各 VPC を使用してネットワークアカウントを設定することを推奨しています。

IoT

インターネットまたはローカル通信ネットワークを介して他のデバイスやシステムと通信する、センサーまたはプロセッサが組み込まれた接続済み物理オブジェクトのネットワーク。詳細については、「IoT とは」を参照してください。

解釈可能性

機械学習モデルの特性で、モデルの予測がその入力にどのように依存するかを人間が理解できる度合いを表します。詳細については、「 を使用した機械学習モデルの解釈可能性 AWS」を参照してください。

IoT

「IoT」を参照してください。

IT 情報ライブラリ (ITIL)

IT サービスを提供し、これらのサービスをビジネス要件に合わせるための一連のベストプラクティス。ITIL は ITSM の基盤を提供します。

IT サービス管理 (ITSM)

組織の IT サービスの設計、実装、管理、およびサポートに関連する活動。クラウドオペレーションと ITSM ツールの統合については、オペレーション統合ガイドを参照してください。

ITIL

「IT 情報ライブラリ」を参照してください。

ITSM

「IT サービス管理」を参照してください。

# L

ラベルベースアクセス制御 (LBAC)

強制アクセス制御 (MAC) の実装で、ユーザーとデータ自体にそれぞれセキュリティラベル値が明示的に割り当てられます。ユーザーセキュリティラベルとデータセキュリティラベルが交差する部分によって、ユーザーに表示される行と列が決まります。

ランディングゾーン

ランディングゾーンは、スケーラブルで安全な、適切に設計されたマルチアカウント AWS 環境です。これは、組織がセキュリティおよびインフラストラクチャ環境に自信を持ってワークロードとアプリケーションを迅速に起動してデプロイできる出発点です。ランディングゾーンの詳細については、「安全でスケーラブルなマルチアカウント AWS 環境のセットアップ」を参照してください。

大規模言語モデル (LLM)

大量のデータで事前トレーニングされた深層学習 AI モデル。LLM では、質問への回答、ドキュメントの要約、他言語へのテキスト翻訳、文を完成させるなど、さまざまなタスクを実行できます。詳細については、「大規模言語モデル (LLM) とは何ですか?」を参照してください。

大規模な移行

300 台以上のサーバの移行。

LBAC

「ラベルベースアクセス制御」を参照してください。

最小特権

タスクの実行には必要最低限の権限を付与するという、セキュリティのベストプラクティス。詳細については、IAM ドキュメントの「最小特権アクセス許可を適用する」を参照してください。

リフトアンドシフト

「7 Rs」を参照してください。

リトルエンディアンシステム

最下位バイトを最初に格納するシステム。「エンディアン性」もご覧ください。

LLM

「大規模言語モデル」を参照してください。

下位環境

「環境」を参照してください。

# M

機械学習 (ML)

パターン認識と学習にアルゴリズムと手法を使用する人工知能の一種。ML は、モノのインターネット (IoT) データなどの記録されたデータを分析して学習し、パターンに基づく統計モデルを生成します。詳細については、「機械学習」を参照してください。

メインブランチ

「ブランチ」を参照してください。

マルウェア

コンピュータのセキュリティやプライバシーを侵害するように設計されたソフトウェア。マルウェアは、コンピュータシステムの中断、機密情報の漏洩、不正アクセスを招く可能性があります。マルウェアの例には、ウイルス、ワーム、ランサムウェア、トロイの木馬、スパイウェア、キーロガーなどがあります。

マネージドサービス

AWS のサービス はインフラストラクチャレイヤー、オペレーティングシステム、プラットフォーム AWS を運用し、エンドポイントにアクセスしてデータを保存および取得します。マネージドサービスの例として、Amazon Simple Storage Service (Amazon S3) と Amazon DynamoDB が挙げられます。このサービスは、抽象化されたサービスとも呼ばれます。

製造実行システム (MES)

生産プロセスを追跡、モニタリング、文書化、制御するソフトウェアシステムであり、工場では、これによって、原材料から製品を完成させます。

MAP

「Migration Acceleration Program」を参照してください。

メカニズム

ツールを作成してその導入を推進し、導入結果を調べて調整を行うための包括的なプロセス。メカニズムとは、運用中にそれ自体を強化し改善するサイクルを意味します。詳細については、AWS 「 Well-Architected フレームワーク」の「メカニズムの構築」を参照してください。

メンバーアカウント

組織の一部である管理アカウント AWS アカウント 以外のすべて AWS Organizations。 アカウントが 組織のメンバーになることができるのは、一度に 1 つのみです。

MES

「製造実行システム」を参照してください。

Message Queuing Telemetry Transport (MQTT)

発行/サブスクライブのパターンに基づく、軽量のマシンツーマシン (M2M) 通信プロトコルであり、リソースに限りのある IoT デバイスに使用されます。

マイクロサービス

明確に定義された API を介して通信し、通常は小規模な自己完結型のチームが所有する、小規模で独立したサービスです。例えば、保険システムには、販売やマーケティングなどのビジネス

機能、または購買、請求、分析などのサブドメインにマッピングするマイクロサービスが含まれる場合があります。マイクロサービスの利点には、俊敏性、柔軟なスケーリング、容易なデプロイ、再利用可能なコード、回復力などがあります。詳細については、AWS「サーバーレスサービスを使用したマイクロサービスの統合」を参照してください。

マイクロサービスアーキテクチャ

各アプリケーションプロセスをマイクロサービスとして実行する独立したコンポーネントを使用してアプリケーションを構築するアプローチ。これらのマイクロサービスは、軽量 API を使用して、明確に定義されたインターフェイスを介して通信します。このアーキテクチャの各マイクロサービスは、アプリケーションの特定の機能に対する需要を満たすように更新、デプロイ、およびスケーリングできます。詳細については、「 でのマイクロサービスの実装 AWS」を参照してください。

Migration Acceleration Program (MAP)

組織がクラウドに移行するための強力な運用基盤を構築し、移行の初期コストを相殺するのに役立つコンサルティングサポート、トレーニング、サービスを提供する AWS プログラム。MAP には、組織的な方法でレガシー移行を実行するための移行方法論と、一般的な移行シナリオを自動化および高速化する一連のツールが含まれています。

大規模な移行

アプリケーションポートフォリオの大部分を次々にクラウドに移行し、各ウェーブでより多くのアプリケーションを高速に移動させるプロセス。この段階では、以前の段階から学んだベストプラクティスと教訓を使用して、移行ファクトリー チーム、ツール、プロセスのうち、オートメーションとアジャイルデリバリーによってワークロードの移行を合理化します。これは、AWS 移行戦略 の第 3 段階です。

移行ファクトリー

自動化された俊敏性のあるアプローチにより、ワークロードの移行を合理化する部門横断的なチーム。移行ファクトリーチームには、通常、運用、ビジネスアナリストおよび所有者、移行エンジニア、デベロッパー、およびスプリントで作業する DevOps プロフェッショナルが含まれます。エンタープライズアプリケーションポートフォリオの 20～50% は、ファクトリーのアプローチによって最適化できる反復パターンで構成されています。詳細については、このコンテンツセットの移行ファクトリーに関する解説とCloud Migration Factory ガイドを参照してください。

移行メタデータ

移行を完了するために必要なアプリケーションおよびサーバーに関する情報。移行パターンごとに、異なる一連の移行メタデータが必要です。移行メタデータの例としては、ターゲットサブネット、セキュリティグループ、 AWS アカウントなどがあります。

移行パターン

移行戦略、移行先、および使用する移行アプリケーションまたはサービスを詳述する、反復可能な移行タスク。例: AWS Application Migration Service を使用して Amazon EC2 への移行をリホストします。

Migration Portfolio Assessment (MPA)

オンラインツール。これによって、 AWS クラウドに移行するビジネスケースの検証に必要な情報を得られます。MPA は、詳細なポートフォリオ評価 (サーバーの適切なサイジング、価格設定、TCO 比較、移行コスト分析) および移行プラン (アプリケーションデータの分析とデータ収集、アプリケーションのグループ化、移行の優先順位付け、およびウェーブプランニング) を提供します。MPA ツール (ログインが必要) は、すべての AWS コンサルタントと APN パートナーコンサルタントが無料で利用できます。

移行準備状況評価 (MRA)

AWS CAF を使用して、組織のクラウド準備状況に関するインサイトを取得し、長所と短所を特定し、特定されたギャップを埋めるためのアクションプランを構築するプロセス。詳細については、移行準備状況ガイドを参照してください。MRA は、AWS 移行戦略の第一段階です。

移行戦略

ワークロードを AWS クラウドに移行するために使用するアプローチ。詳細については、この用語集の 7 Rs エントリと、「組織を動員して大規模な移行を加速する」を参照してください。

ML

「機械学習」を参照してください。

モダナイゼーション

古い (レガシーまたはモノリシック) アプリケーションとそのインフラストラクチャをクラウド内の俊敏で弾力性のある高可用性システムに変換して、コストを削減し、効率を高め、イノベーションを活用します。詳細については、「AWS クラウドでのアプリケーションのモダナイズ戦略」を参照してください。

モダナイゼーション準備状況評価

組織のアプリケーションのモダナイゼーションの準備状況を判断し、利点、リスク、依存関係を特定し、組織がこれらのアプリケーションの将来の状態をどの程度適切にサポートできるかを決定するのに役立つ評価。評価の結果として、ターゲットアーキテクチャのブループリント、モダナイゼーションプロセスの開発段階とマイルストーンを詳述したロードマップ、特定されたギャップに対処するためのアクションプランが得られます。詳細については、「AWS クラウドでのアプリケーションのモダナイゼーションの準備状況を評価する」を参照してください。

モノリシックアプリケーション (モノリス)

緊密に結合されたプロセスを持つ単一のサービスとして実行されるアプリケーション。モノリシックアプリケーションにはいくつかの欠点があります。1 つのアプリケーション機能エクスペリエンスの需要が急増する場合は、アーキテクチャ全体をスケーリングする必要があります。モノリシックアプリケーションの特徴を追加または改善することは、コードベースが大きくなると複雑になります。これらの問題に対処するには、マイクロサービスアーキテクチャを使用できます。詳細については、「モノリスをマイクロサービスに分解する」を参照してください。

MPA

「Migration Portfolio Assessment」を参照してください。

MQTT

「Message Queuing Telemetry Transport」を参照してください。

多クラス分類

複数のクラスの予測を生成するプロセス (2 つ以上の結果の 1 つを予測します)。例えば、機械学習モデルが、「この製品は書籍、自動車、電話のいずれですか?」 または、「このお客様にとって最も関心のある商品のカテゴリはどれですか?」と聞くかもしれません。

ミュータブルなインフラストラクチャ

本番ワークロードに使用する既存のインフラストラクチャを更新および変更するためのモデル。Well-Architected AWS フレームワークでは、一貫性、信頼性、予測可能性を向上させるために、イミュータブルインフラストラクチャの使用をベストプラクティスとして推奨しています。

# O

OAC

「オリジンアクセス制御」を参照してください。

OAI

「オリジンアクセスアイデンティティ」を参照してください。

OCM

「組織変更管理」を参照してください。

オフライン移行

移行プロセス中にソースワークロードを停止させる移行方法。この方法はダウンタイムが長くなるため、通常は重要ではない小規模なワークロードに使用されます。

OI

「オペレーション統合」を参照してください。

Ola

「オペレーショナルレベルアグリーメント」を参照してください。

オンライン移行

ソースワークロードをオフラインにせずにターゲットシステムにコピーする移行方法。ワークロードに接続されているアプリケーションは、移行中も動作し続けることができます。この方法はダウンタイムがゼロから最小限で済むため、通常は重要な本番稼働環境のワークロードに使用されます。

OPC-UA

「Open Process Communications - Unified Architecture」を参照してください。

Open Process Communications - Unified Architecture (OPC-UA)

産業オートメーション用のマシンツーマシン (M2M) 通信プロトコル。OPC-UA により、相互運用の際に、データ暗号化、認証、認可の各スキームを標準化できます。

オペレーショナルレベルアグリーメント (OLA)

サービスレベルアグリーメント (SLA) をサポートするために、どの機能的 IT グループが互いに提供することを約束するかを明確にする契約。

運用準備状況レビュー (ORR)

質問と関連するベストプラクティスのチェックリスト。インシデントや起こり得る障害を理解、評価、防止したり、その範囲を縮小したりする際に役立ちます。詳細については、 AWS Well-Architected フレームワークの「Operational Readiness Reviews (ORR)」を参照してください。

運用テクノロジー (OT)

産業オペレーション、機器、インフラストラクチャを制御するために物理環境と連携させるハードウェアおよびソフトウェアシステム。製造分野では、Industry 4.0 への変革を進める上で、OT と情報技術 (IT) システムの統合に焦点が当てられています。

オペレーション統合 (OI)

クラウドでオペレーションをモダナイズするプロセスには、準備計画、オートメーション、統合が含まれます。詳細については、オペレーション統合ガイドを参照してください。

組織の証跡

組織 AWS アカウント 内のすべての のすべてのイベント AWS CloudTrail を ログに記録する によって作成された証跡 AWS Organizations。証跡は、組織に含まれている各 AWS アカウント に作成され、各アカウントのアクティビティを追跡します。詳細については、CloudTrail ドキュメントの「組織の証跡の作成」を参照してください。

組織変更管理 (OCM)

人材、文化、リーダーシップの観点から、主要な破壊的なビジネス変革を管理するためのフレームワーク。OCM は、変化の導入を加速し、移行問題に対処し、文化や組織の変化を推進することで、組織が新しいシステムと戦略の準備と移行するのを支援します。 AWS 移行戦略では、クラウド導入プロジェクトに必要な変化のスピードにより、このフレームワークは人材アクセラレーションと呼ばれます。詳細については、OCM ガイドを参照してください。

オリジンアクセス制御 (OAC)

Amazon Simple Storage Service (Amazon S3) コンテンツを保護するための、CloudFront のアクセス制限の強化オプション。OAC は AWS リージョン、すべての S3 バケット、 AWS KMS (SSE-KMS) によるサーバー側の暗号化、S3 バケットへの動的 PUT および DELETEリクエストをサポートします。

オリジンアクセスアイデンティティ (OAI)

CloudFront の、Amazon S3 コンテンツを保護するためのアクセス制限オプション。OAI を使用すると、CloudFront が、Amazon S3 に認証可能なプリンシパルを作成します。認証されたプリンシパルは、S3 バケット内のコンテンツに、特定の CloudFront ディストリビューションを介してのみアクセスできます。OAC も併せて参照してください。OAC では、より詳細な、強化されたアクセス制御が可能です。

ORR

「運用準備状況レビュー」を参照してください。

OT

「運用テクノロジー」を参照してください。

アウトバウンド (送信) VPC

AWS マルチアカウントアーキテクチャでは、アプリケーション内から開始されたネットワーク接続を処理する VPC。AWS Security Reference Architecture では、アプリケーションとより広範なインターネット間の双方向のインターフェイスを保護するために、インバウンド、アウトバウンド、インスペクションの各 VPC を使用してネットワークアカウントを設定することを推奨しています。

# P

アクセス許可の境界

ユーザーまたはロールが使用できるアクセス許可の上限を設定する、IAM プリンシパルにアタッチされる IAM 管理ポリシー。詳細については、IAM ドキュメントのアクセス許可の境界を参照してください。

個人を特定できる情報 (PII)

直接閲覧した場合、または他の関連データと組み合わせた場合に、個人の身元を合理的に推測するために使用できる情報。PII の例には、氏名、住所、連絡先情報などがあります。

PII

「個人を特定できる情報」を参照してください。

プレイブック

クラウドでのコアオペレーション機能の提供など、移行に関連する作業を取り込む、事前定義された一連のステップ。プレイブックは、スクリプト、自動ランブック、またはお客様のモダナイズされた環境を運用するために必要なプロセスや手順の要約などの形式をとることができます。

PLC

「プログラマブルロジックコントローラー」を参照してください。

PLM

「製品ライフサイクル管理」を参照してください。

ポリシー

次の操作を可能にするオブジェクト: アクセス許可を定義する (ID ベースのポリシーを参照)。アクセス条件を指定する (リソースベースのポリシーを参照)。 AWS Organizations の組織における全アカウントにアクセス許可の上限を定義する (サービスコントロールポリシーを参照)。

多言語の永続性

データアクセスパターンやその他の要件に基づいて、マイクロサービスのデータストレージテクノロジーを個別に選択します。マイクロサービスが同じデータストレージテクノロジーを使用している場合、実装上の問題が発生したり、パフォーマンスが低下する可能性があります。マイクロサービスは、要件に最も適合したデータストアを使用すると、より簡単に実装でき、パフォーマンスとスケーラビリティが向上します。

ポートフォリオ評価

移行を計画するために、アプリケーションポートフォリオの検出、分析、優先順位付けを行うプロセス。詳細については、「移行の準備状況の評価」を参照してください。

述語

`true` または `false` を返すためのクエリ条件。一般的に、`WHERE` 句に記述されます。

述語プッシュダウン

データベースクエリを最適化する手法。これによって、転送前にクエリ内のデータをフィルタリングします。この手法を取ると、リレーショナルデータベースから取得し処理する必要のあるデータの量が減少するため、クエリのパフォーマンスが向上します。

予防的コントロール

イベントの発生を防ぐように設計されたセキュリティコントロール。このコントロールは、ネットワークへの不正アクセスや好ましくない変更を防ぐ最前線の防御です。詳細については、「AWSでのセキュリティコントロールの実装」の「予防的コントロール」を参照してください。

プリンシパル

アクションを実行し AWS 、リソースにアクセスできる のエンティティ。このエンティティは通常、、IAM AWS アカウントロール、または ユーザーのルートユーザーです。詳細については、IAM ドキュメントの「ロールに関する用語と概念」にあるプリンシパルを参照してください。

プライバシーバイデザイン

開発プロセス全体を通してプライバシーが考慮されているシステムエンジニアリングのアプローチ。

プライベートホストゾーン

1 つ以上の VPC 内のドメインとそのサブドメインへの DNS クエリに対し、Amazon Route 53 がどのように応答するかに関する情報を保持するコンテナ。詳細については、Route 53 ドキュメントの「プライベートホストゾーンの使用」を参照してください。

プロアクティブコントロール

非準拠リソースのデプロイ防止を目的としたセキュリティコントロール。このコントロールにより、プロビジョニング前にリソースをスキャンします。コントロールに準拠していないリソースは、プロビジョニングされません。詳細については、 AWS Control Tower ドキュメントの「コントロールリファレンスガイド」および「 セキュリティコントロールの実装」の「プロアクティブコントロール」を参照してください。 AWS

製品ライフサイクル管理 (PLM)

製品の設計、開発、発売から、成長、成熟、衰退、廃棄に至る、製品のライフサイクル全体を通してデータとプロセスを管理すること。

本番環境

「環境」を参照してください。

プログラマブルロジックコントローラー (PLC)

製造分野で使用される、信頼性と適応性に優れたコンピュータであり、これによって、マシンをモニタリングするとともに、製造プロセスを自動化します。

プロンプトチェイニング

1 つの LLM プロンプトによる出力を次のプロンプトの入力に使用して、より良いレスポンスを生成します。この手法を使用すると、複雑なタスクをサブタスクに分割したり、事前レスポンスを繰り返し改良または拡張したりできます。これによって、モデルのレスポンスの精度と関連性が向上し、粒度の高いパーソナライズされた結果を得られます。

仮名化

データセット内の個人識別子をプレースホルダー値に置き換えるプロセス。仮名化は個人のプライバシー保護に役立ちます。仮名化されたデータは、依然として個人データとみなされます。

発行/サブスクライブ (pub/sub)

マイクロサービス間の非同期通信を可能にするパターン。これにより、スケーラビリティと応答性を向上させます。例えば、マイクロサービスベースの MES の場合、マイクロサービスは、他のマイクロサービスがサブスクライブ可能なチャネルにイベントメッセージを発行できます。このシステムでは、発行サービスの変更なしに、新規マイクロサービスを追加できます。

# Q

クエリプラン

手順などの一連のステップであり、SQL リレーショナルデータベースシステムのデータにアクセスするために使用されます。

クエリプランのリグレッション

データベースサービスのオプティマイザーが、データベース環境に特定の変更が加えられる前に選択されたプランよりも最適性の低いプランを選択すること。これは、統計、制限事項、環境設定、クエリパラメータのバインディングの変更、およびデータベースエンジンの更新などが原因である可能性があります。

# R

RACI マトリックス

「実行責任者、説明責任者、協業先、報告先 (RACI)」を参照してください。

RAG

「検索拡張生成」を参照してください。

ランサムウェア

決済が完了するまでコンピュータシステムまたはデータへのアクセスをブロックするように設計された、悪意のあるソフトウェア。

RASCI マトリックス

「実行責任者、説明責任者、協業先、報告先 (RACI)」を参照してください。

RCAC

「行と列のアクセス制御」を参照してください。

リードレプリカ

読み取り専用に使用されるデータベースのコピー。クエリをリードレプリカにルーティングして、プライマリデータベースへの負荷を軽減できます。

リアーキテクト

「7 Rs」を参照してください。

目標復旧時点 (RPO)

最後のデータリカバリポイントからの最大許容時間です。これにより、最後の回復時点からサービスが中断されるまでの間に許容できるデータ損失の程度が決まります。

目標復旧時間 (RTO)

サービスの中断から復旧までの最大許容遅延時間。

リファクタリング

「7 Rs」を参照してください。

リージョン

地理的エリア内の AWS リソースのコレクション。各 AWS リージョン は、耐障害性、安定性、耐障害性を提供するために、他の から分離され、独立しています。詳細については、「アカウントが使用できる AWS リージョン を指定する」を参照してください。

リグレッション

数値を予測する機械学習手法。例えば、「この家はどれくらいの値段で売れるでしょうか?」という問題を解決するために、機械学習モデルは、線形回帰モデルを使用して、この家に関する既知の事実 (平方フィートなど) に基づいて家の販売価格を予測できます。

リホスト

「7 Rs」を参照してください。

リリース

デプロイプロセスで、変更を本番環境に昇格させること。

再配置

「7 Rs」を参照してください。

リプラットフォーム

「7 Rs」を参照してください。

再購入

「7 Rs」を参照してください。

回復性

中断に抵抗または中断から回復するアプリケーションの機能。 AWS クラウドでの回復力を計画する際には、一般的に、高可用性とディザスタリカバリが考慮されます。詳細については、「AWS クラウド の耐障害性」を参照してください。

リソースベースのポリシー

Amazon S3 バケット、エンドポイント、暗号化キーなどのリソースにアタッチされたポリシー。このタイプのポリシーは、アクセスが許可されているプリンシパル、サポートされているアクション、その他の満たすべき条件を指定します。

実行責任者、説明責任者、協業先、報告先 (RACI) に基づくマトリックス

移行活動とクラウド運用に関わるすべての関係者の役割と責任を定義したマトリックス。マトリックスの名前は、マトリックスで定義されている責任の種類、すなわち責任 (R) 、説明責任 (A) 、協議 (C) 、情報提供 (I) に由来します。サポート (S) タイプはオプションです。サポートが含まれる場合は RASCI マトリックスと呼ばれ、含まれない場合は RACI マトリックスと呼ばれます。

レスポンシブコントロール

有害事象やセキュリティベースラインからの逸脱について、修復を促すように設計されたセキュリティコントロール。詳細については、「 AWSでのセキュリティコントロールの実装」の「レスポンシブコントロール」を参照してください。

保持

「7 Rs」を参照してください。

廃止

「7 Rs」を参照してください。

検索拡張生成 (RAG)

生成 AI の技術。これにより、LLM では、レスポンスの生成前に、トレーニングデータソースの外部にある信頼できるデータソースが参照されます。例えば、RAG モデルによって、組織のナレッジベースまたはカスタムデータのセマンティック検索を実行できる場合があります。細については、「RAG (検索拡張生成) とは何ですか?」を参照してください。

ローテーション

定期的にシークレット情報を更新して、攻撃者が認証情報にアクセスするのをより困難にするプロセス。

行と列のアクセス制御 (RCAC)

アクセスルールが定義された、基本的で柔軟な SQL 表現の使用。RCAC は行権限と列マスクで構成されています。

RPO

「目標復旧時点」を参照してください。

RTO

「目標復旧時間」を参照してください。

ランブック

特定のタスクを実行するために必要な手動または自動化された一連の手順。これらは通常、エラー率の高い反復操作や手順を合理化するために構築されています。

# S

SAML 2.0

多くの ID プロバイダー (IdP) が使用しているオープンスタンダード。この機能を使用すると、フェデレーティッドシングルサインオン (SSO) が有効になるため、ユーザーは組織内のすべてのユーザーを IAM で作成しなくても、 AWS マネジメントコンソール にログインしたり AWS 、API オペレーションを呼び出すことができます。SAML 2.0 ベースのフェデレーションの詳細については、IAM ドキュメントの「SAML 2.0 ベースのフェデレーションについて」を参照してください。

SCADA

「監視制御とデータ取得」を参照してください。

SCP

「サービスコントロールポリシー」を参照してください。

シークレット

暗号化された形式で保存する AWS Secrets Managerパスワードやユーザー認証情報などの機密情報または制限付き情報。シークレット値とそのメタデータで構成されます。シークレット値には、バイナリ、1 つの文字列、複数の文字列を指定できます。詳細については、Secrets Manager ドキュメントの「Secrets Manager シークレットの概要」を参照してください。

セキュリティバイデザイン

開発プロセス全体を通してセキュリティが考慮されているシステムエンジニアリングのアプローチ。

セキュリティコントロール

脅威アクターによるセキュリティ脆弱性の悪用を防止、検出、軽減するための、技術上または管理上のガードレール。セキュリティコントロールには、主に 4 つの種類があります。4 つとは、予防、検出、レスポンシブ、プロアクティブです。

セキュリティ強化

アタックサーフェスを狭めて攻撃への耐性を高めるプロセス。このプロセスには、不要になったリソースの削除、最小特権を付与するセキュリティのベストプラクティスの実装、設定ファイル内の不要な機能の無効化、といったアクションが含まれています。

Security Information and Event Management (SIEM) システム

セキュリティ情報管理 (SIM) とセキュリティイベント管理 (SEM) のシステムを組み合わせたツールとサービス。SIEM システムは、サーバー、ネットワーク、デバイス、その他ソースからデータを収集、モニタリング、分析して、脅威やセキュリティ違反を検出し、アラートを発信します。

セキュリティレスポンスの自動化

セキュリティイベントへの自動レスポンスまたは自動修復を目的として、事前定義およびプログラムされたアクション。これらの自動化は、セキュリティのベストプラクティスを実装するのに役立つ検出的または応答的な AWS セキュリティコントロールとして機能します。自動レスポンスアクションの例には、VPC セキュリティグループの変更、Amazon EC2 インスタンスへのパッチ適用、認証情報の更新などがあります。

サーバー側の暗号化

送信先で、それ AWS のサービス を受け取る によるデータの暗号化。

サービスコントロールポリシー (SCP)

AWS Organizationsの組織内の、すべてのアカウントのアクセス許可を一元的に管理するポリシー。SCP は、管理者がユーザーまたはロールに委任するアクションに、ガードレールを定義したり、アクションの制限を設定したりします。SCP は、許可リストまたは拒否リストとして、許可または禁止するサービスやアクションを指定する際に使用できます。詳細については、 AWS Organizations ドキュメントの「サービスコントロールポリシー」を参照してください。

サービスエンドポイント

のエントリポイントの URL AWS のサービス。ターゲットサービスにプログラムで接続するには、エンドポイントを使用します。詳細については、「AWS 全般のリファレンス」の「AWS のサービス エンドポイント」を参照してください。

サービスレベルアグリーメント (SLA)

サービスのアップタイムやパフォーマンスなど、IT チームがお客様に提供すると約束したものを明示した合意書。

サービスレベルインジケータ (SLI)

エラー率、可用性、スループットといった、サービスパフォーマンス面の指標。

サービスレベル目標 (SLO)

サービスレベルインジケータによって測定され、サービスの状態を表すターゲットメトリクス。

責任共有モデル

クラウドのセキュリティとコンプライアンス AWS について と共有する責任を説明するモデル。AWS はクラウドのセキュリティを担当しますが、 はクラウドのセキュリティを担当します。詳細については、「責任共有モデル」を参照してください。

SIEM

「Security Information and Event Management システム」を参照してください。

単一障害点 (SPOF)

特定のアプリケーションを構成する単一の重要なコンポーネントで発生し、システム稼働に支障をきたす可能性のある障害。

SLA

「サービスレベルアグリーメント」を参照してください。

SLI

「サービスレベルインジケータ」を参照してください。

SLO

「サービスレベルの目標」を参照してください。

スプリットアンドシードモデル

モダナイゼーションプロジェクトのスケーリングと加速のためのパターン。新機能と製品リリースが定義されると、コアチームは解放されて新しい製品チームを作成します。これにより、お客様の組織の能力とサービスの拡張、デベロッパーの生産性の向上、迅速なイノベーションのサポートに役立ちます。詳細については、「AWS クラウドでのアプリケーションをモダナイズするための段階的アプローチ」を参照してください。

SPOF

「単一障害点」を参照してください。

スタースキーマ

データベースの編成構造を意味し、1 つの大きいファクトテーブルにトランザクションデータまたは測定データが保存され、1 つ以上の小さいディメンションテーブルにデータ属性が保存されます。この構造は、データウェアハウスやビジネスインテリジェンスを用途とするように設計されています。

strangler fig パターン

レガシーシステムが廃止されるまで、システム機能を段階的に書き換えて置き換えることにより、モノリシックシステムをモダナイズするアプローチ。このパターンは、宿主の樹木から根を成長させ、最終的にその宿主を包み込み、宿主に取って代わるイチジクのつるを例えています。そのパターンは、モノリシックシステムを書き換えるときのリスクを管理する方法として Martin Fowler により提唱されました。このパターンの適用方法の例については、「コンテナと Amazon API Gateway を使用して、従来の Microsoft ASP.NET (ASMX) ウェブサービスを段階的にモダナイズ」を参照してください。

サブネット

VPC 内の IP アドレスの範囲。サブネットは、1 つのアベイラビリティーゾーンに存在する必要があります。

監視制御とデータ取得 (SCADA)

製造分野において、ハードウェアとソフトウェアを使用して物理アセットと本番運用をモニタリングするシステム。

対称暗号化

データの暗号化と復号に同じキーを使用する暗号化のアルゴリズム。

合成テスト

ユーザーとのやり取りをシミュレートして、起こり得る問題を検出したり、パフォーマンスをモニタリングしたりすることで、システムをテストします。Amazon CloudWatch Synthetics を使用すると、こうしたテストを作成できます。

システムプロンプト

コンテキスト、指示、ガイドラインなどを提示して、LLM に動作を指示する手法。システムプロンプトは、コンテキストを設定して、ユーザーとやり取りするルールを確立するのに有用です。

# T

タグ

AWS リソースを整理するためのメタデータとして機能するキーと値のペア。タグは、 リソースの管理、識別、整理、検索、フィルタリングに役立ちます。詳細については、「AWS リソースのタグ付け」を参照してください。

ターゲット変数

監督された機械学習でお客様が予測しようとしている値。これは、結果変数のことも指します。例えば、製造設定では、ターゲット変数が製品の欠陥である可能性があります。

タスクリスト

ランブックの進行状況を追跡するために使用されるツール。タスクリストには、ランブックの概要と完了する必要のある一般的なタスクのリストが含まれています。各一般的なタスクには、推定所要時間、所有者、進捗状況が含まれています。

テスト環境

「環境」を参照してください。

トレーニング

お客様の機械学習モデルに学習するデータを提供すること。トレーニングデータには正しい答えが含まれている必要があります。学習アルゴリズムは入力データ属性をターゲット (お客様が予測したい答え) にマッピングするトレーニングデータのパターンを検出します。これらのパターンをキャプチャする機械学習モデルを出力します。そして、お客様が機械学習モデルを使用して、ターゲットがわからない新しいデータでターゲットを予測できます。

トランジットゲートウェイ

VPC とオンプレミスネットワークを相互接続するために使用できる、ネットワークの中継ハブ。詳細については、 AWS Transit Gateway ドキュメントの「トランジットゲートウェイとは」を参照してください。

トランクベースのワークフロー

デベロッパーが機能ブランチで機能をローカルにビルドしてテストし、その変更をメインブランチにマージするアプローチ。メインブランチはその後、開発環境、本番前環境、本番環境に合わせて順次構築されます。

信頼されたアクセス

ユーザーに代わって AWS Organizations およびそのアカウントで組織内でタスクを実行するために指定したサービスにアクセス許可を付与します。信頼されたサービスは、サービスにリンクされたロールを必要なときに各アカウントに作成し、ユーザーに代わって管理タスクを実行します。詳細については、 ドキュメントの「Using AWS Organizations with other AWS services AWS Organizations 」を参照してください。

チューニング

機械学習モデルの精度を向上させるために、お客様のトレーニングプロセスの側面を変更する。例えば、お客様が機械学習モデルをトレーニングするには、ラベル付けセットを生成し、ラベルを追加します。これらのステップを、異なる設定で複数回繰り返して、モデルを最適化します。

ツーピザチーム

2 枚のピザを分け合えることができるくらい小さな DevOps チーム。ツーピザチームの規模では、ソフトウェア開発におけるコラボレーションに最適な機会が確保されます。

# U

不確実性

予測機械学習モデルの信頼性を損なう可能性がある、不正確、不完全、または未知の情報を指す概念。不確実性には、次の 2 つのタイプがあります。認識論的不確実性は、限られた、不完全なデータによって引き起こされ、弁論的不確実性 は、データに固有のノイズとランダム性によって引き起こされます。

未分化なタスク

ヘビーリフティングとも呼ばれ、アプリケーションの作成と運用には必要だが、エンドユーザーに直接的な価値をもたらさなかったり、競争上の優位性をもたらしたりしない作業です。未分化なタスクの例としては、調達、メンテナンス、キャパシティプランニングなどがあります。

上位環境

「環境」を参照してください。

# V

バキューミング

ストレージを再利用してパフォーマンスを向上させるために、増分更新後にクリーンアップを行うデータベースのメンテナンス操作。

バージョンコントロール

リポジトリ内のソースコードへの変更など、変更を追跡するプロセスとツール。

VPC ピアリング

プライベート IP アドレスを使用してトラフィックをルーティングできる、2 つの VPC 間の接続。詳細については、Amazon VPC ドキュメントの「VPC ピア機能とは」を参照してください。

脆弱性

システムのセキュリティを脅かすソフトウェアまたはハードウェアの欠陥。

# W

ウォームキャッシュ

頻繁にアクセスされる最新の関連データを含むバッファキャッシュ。データベースインスタンスはバッファキャッシュから、メインメモリまたはディスクからよりも短い時間で読み取りを行うことができます。

ウォームデータ

アクセス頻度の低いデータ。この種類のデータをクエリする場合、通常は適度に遅いクエリでも問題ありません。

ウィンドウ関数

現在のレコードに何らかの形で関連している行のグループに計算を実行する SQL 関数。ウィンドウ関数は、移動平均を計算したり、現在の行の相対位置に基づいて他の行の値にアクセスするといったタスクの処理に役立ちます。

ワークロード

ビジネス価値をもたらすリソースとコード (顧客向けアプリケーションやバックエンドプロセスなど) の総称。

ワークストリーム

特定のタスクセットを担当する移行プロジェクト内の機能グループ。各ワークストリームは独立していますが、プロジェクト内の他のワークストリームをサポートしています。たとえば、ポートフォリオワークストリームは、アプリケーションの優先順位付け、ウェーブ計画、および移行メタデータの収集を担当します。ポートフォリオワークストリームは、これらの設備を移行ワークストリームで実現し、サーバーとアプリケーションを移行します。

WORM

「Write-Once-Read-Many」を参照してください。

WQF

「AWS ワークロード資格フレームワーク」を参照してください

Write-Once-Read-Many (WORM)

データを 1 回のみ書き込むことで、データの削除や変更を防ぐストレージモデル。承認済みユーザーは、必要な回数だけデータを読み取ることができますが、変更することはできません。このデータストレージインフラストラクチャは、イミュータブルと見なされます。

# Z

ゼロデイエクスプロイト

ゼロデイ脆弱性を悪用した攻撃 (一般的にマルウェアによる)。

ゼロデイ脆弱性

実稼働システムにおける未解決の欠陥または脆弱性。脅威アクターは、このような脆弱性を利用してシステムを攻撃する可能性があります。開発者は、よく攻撃の結果で脆弱性に気付きます。

ゼロショットプロンプト

LLM にタスク実行の手順は提示するが、実行のガイドとして役立つ例 (ショット) は提示しない方法。LLM は、事前トレーニング済みの知識を使用してタスクを処理する必要があります。ゼロショットプロンプトの有効性は、タスクの複雑さとプロンプトの品質によって異なります。「数ショットプロンプト」も参照してください。

ゾンビアプリケーション

平均 CPU およびメモリ使用率が 5% 未満のアプリケーション。移行プロジェクトでは、これらのアプリケーションを廃止するのが一般的です。

翻訳は機械翻訳により提供されています。提供された翻訳内容と英語版の間で齟齬、不一致または矛盾がある場合、英語版が優先します。