aws

ユーザーガイド

# Amazon Fraud Detector

Version latest

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

# Amazon Fraud Detector: ユーザーガイド

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon の商標およびトレードドレスはAmazon 以外の製品およびサービスに使用することはできません。また、お客様に誤解を与える可能性がある形式で、または Amazon の信用を損なう形式で使用することもできません。Amazon が所有していないその他のすべての商標は Amazon との提携、関連、支援関係の有無にかかわらず、それら該当する所有者の資産です。

# Table of Contents

Amazon Fraud Detector とは ..... 1
利点 ..... 1
主要概念と用語 ..... 3
Amazon Fraud Detector の仕組み ..... 6
Amazon Fraud Detector による不正の検出 ..... 7
Amazon Fraud Detector へのアクセス ..... 9
可用性 ..... 9
インターフェイス ..... 9
料金 ..... 10
Amazon Fraud Detector のセットアップ ..... 11
にサインアップする AWS ..... 11
にサインアップする AWS アカウント ..... 11
管理アクセスを持つユーザーを作成する ..... 12
Amazon Fraud Detector インターフェイスにアクセスするためのアクセス許可を設定する ..... 13
を使用して Amazon Fraud Detector にアクセスするためのインターフェイスを設定する ..... 15
Amazon Fraud Detector コンソールにアクセスする ..... 15
セットアップ AWS CLI ..... 15
AWS SDK のセットアップ ..... 16
Amazon Fraud Detector の使用を開始する ..... 17
サンプルデータセットを取得してアップロードする ..... 17
チュートリアル: Amazon Fraud Detector コンソールの使用を開始する ..... 19
パート A: Amazon Fraud Detector モデルの構築、トレーニング、デプロイ ..... 19
パート B: 不正予測を生成する ..... 24
チュートリアル: の使用を開始する AWS SDK for Python (Boto3) ..... 29
前提条件 ..... 29
はじめに ..... 29
(オプション) Jupyter (iPython) ノートブックを使用した Amazon Fraud Detector API について詳しく知る ..... 39
次のステップ ..... 39
イベントデータセット ..... 40
イベントデータセット構造 ..... 41
データモデルエクスプローラーを使用してイベントデータセットの要件を取得する ..... 42
データモデルエクスプローラー ..... 42
イベントデータの収集 ..... 43

データセットの検証 ..... 49
データセットストレージ ..... 50
イベントタイプ ..... 52
イベントタイプを作成する ..... 52
Amazon Fraud Detector コンソールでイベントタイプを作成する ..... 53
を使用してイベントタイプを作成する AWS SDK for Python (Boto3) ..... 54
イベントまたはイベントタイプの削除 ..... 55
イベントデータストレージ ..... 57
Amazon S3 を使用してイベントデータを外部に保存する ..... 58
CSV ファイルを作成する ..... 58
イベントデータを Amazon S3 バケットにアップロードする ..... 61
Amazon Fraud Detector を使用してイベントデータを内部に保存する ..... 62
ストレージ用のイベントデータを準備する ..... 63
バッチインポートを使用してイベントデータを保存する ..... 64
GetEventPredictions API オペレーションを使用してイベントデータを保存する ..... 77
SendEvent API オペレーションを使用してイベントデータを保存する ..... 78
保存されたイベントデータの詳細を取得する ..... 79
保存されたイベントデータセットのメトリクスを表示する ..... 80
イベントのオーケストレーション ..... 81
イベントオーケストレーションの設定 ..... 82
Amazon Fraud Detector でイベントオーケストレーションを有効にする ..... 83
Amazon Fraud Detector コンソールでイベントオーケストレーションを有効にする ..... 83
を使用してイベントオーケストレーションを有効にする AWS SDK for Python (Boto3) ..... 84
Amazon Fraud Detector でイベントオーケストレーションを無効にする ..... 84
Amazon Fraud Detector コンソールでイベントオーケストレーションを無効にする ..... 84
を使用してイベントオーケストレーションを無効にする AWS SDK for Python (Boto3) ..... 85
モデル ..... 86
モデルタイプの選択 ..... 86
オンライン不正インサイト ..... 86
トランザクション不正インサイト ..... 88
アカウント乗っ取りに関するインサイト ..... 91
モデルの構築 ..... 96
を使用してモデルをトレーニングおよびデプロイする AWS SDK for Python (Boto3) ..... 96
モデルスコア ..... 98
モデルパフォーマンスメトリクス ..... 99
モデル変数の重要度 ..... 101

モデル変数重要度値の使用 ... 103
モデル変数重要度値の評価 ... 104
モデル変数の重要度ランキングの表示 ... 104
モデル変数重要度値の計算方法の理解 ... 104
SageMaker AI モデルのインポート ... 105
を使用して SageMaker AI モデルをインポートする AWS SDK for Python (Boto3) ... 105
モデルまたはモデルバージョンの削除 ... 107
ディテクター ... 109
ディテクターの作成 ... 109
Amazon Fraud Detector コンソールでディテクターを作成する ... 109
を使用してディテクターを作成する AWS SDK for Python (Boto3) ... 113
ディテクターバージョンの作成 ... 113
ルール実行モード ... 113
を使用してディテクターバージョンを作成する AWS SDK for Python (Boto3) ... 114
ディテクター、ディテクターバージョン、ルールバージョンの削除 ... 115
リソース ... 117
変数 ... 117
データ型 ... 117
デフォルト値 ... 118
変数タイプ ... 118
変数エンリッチメント ... 141
変数の作成 ... 148
変数の削除 ... 150
ラベル ... 151
ラベルの作成 ... 151
ラベルの更新 ... 152
Amazon Fraud Detector に保存されているイベントデータのイベントラベルの更新 ... 153
ラベルの削除 ... 153
ルール ... 154
ルール言語リファレンス ... 155
ルールを作成する ... 160
更新ルール ... 162
Lists ... 163
リストを作成する ... 164
リストにエントリを追加する ... 166
変数タイプをリストに割り当てる ... 167

リストを削除する ..... 168
リストからエントリを削除する ..... 168
リストからすべてのエントリを削除する ..... 169
結果 ..... 170
結果の作成 ..... 170
結果の削除 ..... 172
エンティティ ..... 172
エンティティタイプの作成 ..... 173
エンティティタイプの削除 ..... 174
を使用して リソースを管理する AWS CloudFormation ..... 175
Amazon Fraud Detector テンプレートの作成 ..... 175
Amazon Fraud Detector スタックの管理 ..... 175
Amazon Fraud Detector CloudFormation パラメータの理解 ..... 176
Amazon Fraud Detector リソースのサンプル AWS CloudFormation テンプレート ..... 177
の詳細 AWS CloudFormation ..... 178
不正予測 ..... 179
リアルタイム予測 ..... 180
リアルタイム不正予測の仕組み ..... 180
リアルタイムの不正予測の取得 ..... 181
バッチ予測 ..... 182
バッチ予測の仕組み ..... 182
入力および出力ファイル ..... 183
バッチ予測の取得 ..... 183
IAM ロールに関するガイダンス ..... 185
を使用してバッチ不正予測を取得する AWS SDK for Python (Boto3) ..... 185
予測説明 ..... 186
予測説明の表示 ..... 188
予測説明の計算方法の理解 ..... 189
セキュリティ ..... 191
データ保護 ..... 192
保管中の暗号化 ..... 193
転送中の暗号化 ..... 193
キーの管理 ..... 193
VPC エンドポイントAWS PrivateLink ..... 195
オプトアウト ..... 197
Identity and Access Management ..... 198

対象者 ..... 198
アイデンティティを使用した認証 ..... 199
ポリシーを使用したアクセスの管理 ..... 202
Amazon Fraud Detector で IAM が機能する仕組み ..... 205
アイデンティティベースのポリシーの例 ..... 210
混乱した代理の防止 ..... 218
トラブルシューティング ..... 220
Amazon Fraud Detector のモニタリング ..... 223
コンプライアンス検証 ..... 223
耐障害性 ..... 224
インフラストラクチャセキュリティ ..... 225
Amazon Fraud Detector のモニタリング ..... 226
CloudWatch によるモニタリング ..... 226
Amazon Fraud Detector の CloudWatch メトリクスの使用。 ..... 227
Amazon Fraud Detector メトリクス ..... 229
を使用した Amazon Fraud Detector API コールのログ記録 AWS CloudTrail ..... 233
CloudTrail での Amazon Fraud Detector 情報 ..... 233
Amazon Fraud Detector ログファイルエントリの概要 ..... 234
トラブルシューティング ..... 236
トレーニングデータに関する問題のトラブルシューティング ..... 236
指定されたデータセットの不安定な不正率 ..... 237
不十分なデータ ..... 237
異なる EVENT_LABEL 値がない ..... 240
EVENT_TIMESTAMP 値が欠落しているか正しくない ..... 241
データが取り込まれない ..... 242
変数が不十分 ..... 243
変数タイプが欠落しているか、正しくない ..... 243
欠落している変数値 ..... 244
一意な変数値が不十分 ..... 244
変数式が誤っている ..... 245
一意のエンティティが不十分 ..... 246
クォータ ..... 248
Amazon Fraud Detector モデル ..... 248
Amazon Fraud Detector ディテクター/変数/結果/ルール ..... 248
Amazon Fraud Detector API ..... 249
ドキュメント履歴 ..... 250

..... ccliv

# Amazon Fraud Detector とは

Amazon Fraud Detector は、オンラインでの不正行為の可能性の検出を自動化するフルマネージド型の不正検出サービスです。これらのアクティビティには、不正なトランザクションやフェイクアカウントの作成が含まれます。Amazon Fraud Detector は、機械学習を使用してデータを分析することで機能します。これは、Amazon での 20 年以上にわたる不正検出の実績ある専門知識を基に構築されています。

Amazon Fraud Detector を使用して、カスタマイズされた不正検出モデルを構築し、モデルの不正評価を解釈する決定ロジックを追加し、可能な不正評価ごとに合格または送信などの結果をレビューに割り当てることができます。Amazon Fraud Detector では、不正行為を検出するために機械学習の専門知識は必要ありません。

開始するには、組織で収集した不正データを収集して準備します。次に、Amazon Fraud Detector はこのデータを使用して、ユーザーに代わってカスタム不正検出モデルをトレーニング、テスト、デプロイします。このプロセスの一環として、Amazon Fraud Detector は、 から不正パターンを学習した機械学習モデル AWS と Amazon 独自の不正専門知識を使用して、不正データを評価し、モデルスコアとモデルパフォーマンスデータを生成します。モデルのスコアを解釈し、各不正評価の処理方法の結果を割り当てるように決定ロジックを設定します。

## 利点

Amazon Fraud Detector には以下の利点があります。これらの利点により、不正管理システムの構築と維持に従来必要な時間とリソースを費やすことなく、不正を迅速に検出できます。

不正モデルの自動作成

Amazon Fraud Detector の不正検出モデルは、特定のビジネスニーズに合わせてカスタマイズされた完全に自動化された機械学習モデルです。Amazon Fraud Detector モデルを使用すると、新しいアカウントの作成、オンライン支払い、ゲストチェックアウトなどのオンライントランザクションで潜在的な不正を特定できます。

不正モデルは自動化されたプロセスによって作成されるため、モデルの作成とトレーニングに関連する多くのステップは省略できます。これらのステップには、データの検証とエンリッチメント、特徴量エンジニアリング、アルゴリズムの選択、ハイパーパラメータの調整、モデルのデプロイが含まれます。

Amazon Fraud Detector を使用して不正検出モデルを作成するには、会社の過去の不正データセットをアップロードし、モデルタイプのみを選択します。次に、Amazon Fraud Detector はユースケース

に最適な不正検出アルゴリズムを自動的に検出し、モデルを作成します。不正検出モデルを作成するために、コーディングを知っている必要や機械学習の専門知識を持っている必要はありません。

進化して学習する不正モデル

不正検出モデルは、変化する不正の状況に対応するために絶えず進化する必要があります。Amazon Fraud Detector は、アカウント経過時間、前回のアクティビティからの時間、アクティビティ数などの情報を計算することで、これを自動的に行います。その結果、頻繁に取引を行う信頼された顧客と、不正行為者に典型的な試みが続くことの違いをモデルが学習します。これにより、再トレーニングセッション間でモデルのパフォーマンスをより長く維持できます。

不正モデルのパフォーマンスの視覚化

指定したデータを使用してモデルがトレーニングされると、Amazon Fraud Detector はモデルのパフォーマンスを検証します。また、パフォーマンスを評価するためのビジュアルツールも提供します。トレーニングするモデルごとに、モデルのパフォーマンススコア、スコア分布グラフ、混同行列、しきい値テーブル、およびモデルのパフォーマンスへの影響によってランク付けされたすべての入力を確認できます。これらのパフォーマンスツールを使用して、モデルのパフォーマンスと、モデルのパフォーマンスを高めている入力を確認できます。必要に応じて、モデルを微調整して全体的なパフォーマンスを向上させることができます。

不正予測

Amazon Fraud Detector は、組織のビジネス活動の不正予測を生成します。不正予測は、不正リスクのビジネスアクティビティの評価です。Amazon Fraud Detector は、アクティビティに関連付けられたデータを含む予測ロジックを使用して予測を生成します。このデータは、不正検出モデルの作成時に指定しました。1 つのアクティビティの不正予測をリアルタイムで取得したり、一連のアクティビティの不正予測をオフラインにしたりできます。

不正予測の説明の視覚化

Amazon Fraud Detector は、不正予測プロセスの一環として予測の説明を生成します。予測の説明は、モデルのトレーニングに使用された各データ要素がモデルの不正予測スコアにどのように影響したかについてのインサイトを提供します。予測の説明は、テーブルやグラフなどのビジュアルツールを使用して提供されます。これらのツールを使用して、各データ要素が予測スコアにどの程度影響しているかを視覚的に識別できます。次に、この情報を使用して、データセット全体の不正パターンを分析し、バイアスがあれば検出できます。最後に、予測の説明を使用して、手動の不正調査プロセス中にトップリスク指標を特定することもできます。これにより、誤検出予測につながる根本原因を絞り込むことができます。

ルールベースのアクション

不正検出モデルのトレーニングが完了したら、ルールを追加して、評価されたデータに対してデータの承認、レビューのためのデータの送信、データの収集などのアクションを実行できます。ルールは、不正予測中にデータを解釈する方法を Amazon Fraud Detector に指示する条件です。例えば、疑わしい顧客アカウントのレビュー対象にフラグを付けるルールを作成できます。検出されたモデルスコアが事前に決定されたしきい値よりも大きく、アカウント支払いの認可コード (AUTH_CODE) が有効でない場合に、このルールを開始するように設定できます。

# 主要概念と用語

以下は、Amazon Fraud Detector で使用される主要な概念と用語のリストです。

イベント

イベントは、不正リスクについて評価される組織のビジネスアクティビティです。Amazon Fraud Detector は、イベントの不正予測を生成します。

ラベル

ラベルは、1 つのイベントを不正または正当として分類します。ラベルは、Amazon Fraud Detector で機械学習モデルをトレーニングするために使用されます。

エンティティ

エンティティは、イベントを実行しているユーザーを表します。イベントを実行した特定のエンティティを示すために、会社の不正データの一部としてエンティティ ID を指定します。

イベントタイプ

イベントタイプは、Amazon Fraud Detector に送信されるイベントの構造を定義します。これには、イベントの一部として送信されるデータ、イベントを実行するエンティティ (顧客など )、イベントを分類するラベルが含まれます。イベントタイプの例には、オンライン支払いトランザクション、アカウント登録、認証などがあります。

エンティティタイプ

エンティティタイプは、エンティティを分類します。分類の例には、顧客、マーチャント、アカウントなどがあります。

イベントデータセット

イベントデータセットは、特定のビジネスアクティビティまたはイベントの会社の履歴データです。例えば、会社のイベントがオンラインアカウント登録である場合があります。1 つのイベ

ント (登録) からのデータには、関連付けられた IP アドレス、E メールアドレス、請求先アドレス、イベントタイムスタンプが含まれる場合があります。Amazon Fraud Detector にイベントデータセットを提供して、不正検出モデルを作成およびトレーニングします。

モデル

モデルは機械学習アルゴリズムの出力です。これらのアルゴリズムはコードに実装され、指定したイベントデータで実行されます。

モデルタイプ

モデルタイプは、モデルトレーニング中に使用されるアルゴリズム、エンリッチメント、および特徴変換を定義します。また、モデルをトレーニングするためのデータ要件も定義します。これらの定義は、特定のタイプの不正に対してモデルを最適化するために機能します。モデルの作成時に使用するモデルタイプを指定します。

モデルトレーニング

モデルトレーニングは、提供されたイベントデータセットを使用して、不正なイベントを予測できるモデルを作成するプロセスです。モデルトレーニングプロセスのすべてのステップは完全に自動化されています。これらのステップには、データ検証、データ変換、特徴量エンジニアリング、アルゴリズムの選択、モデルの最適化が含まれます。

モデルスコア

モデルスコアは、会社の過去の不正データの評価結果です。モデルトレーニングプロセス中、Amazon Fraud Detector はデータセットの不正行為を評価し、0 から 1000 までのスコアを生成します。このスコアでは、0 は低い不正リスクを表し、1000 は最も高い不正リスクを表します。スコア自体は誤検出率 (FPR) に直接関係しています。

モデルバージョン

モデルバージョンは、モデルのトレーニングからの出力です。

モデルのデプロイ

モデルデプロイは、モデルバージョンをアクティブ化し、不正予測を生成できるようにするプロセスです。

Amazon SageMaker AI モデルエンドポイント

Amazon Fraud Detector を使用してモデルを構築するだけでなく、オプションで Amazon Fraud Detector の評価で SageMaker AI がホストするモデルエンドポイントを使用することもできます。

SageMaker AI でモデルを構築する方法の詳細については、「 でモデルをトレーニング Amazon SageMaker AIする」を参照してください。

ディテクター

ディテクターには、不正について評価する特定のイベントのモデルやルールなどの検出ロジックが含まれています。モデルバージョンを使用してディテクターを作成します。

ディテクターバージョン

ディテクターは複数のバージョンを持つことができ、各バージョンのステータスは `Draft`、`Active`、または `Inactive` になります。一度に 1 つのディテクターバージョンのみが `Active` ステータスになることができます。

変数

変数は、不正予測で使用するイベントに関連付けられたデータ要素を表します。変数は、不正予測の一部としてイベントとともに送信することも、Amazon Fraud Detector モデルの出力や など、派生させることもできます Amazon SageMaker AI。

ルール

ルールは、不正予測時に変数値の解釈方法を Amazon Fraud Detector に指示する条件です。ルールは、1 つ以上の変数、論理式、および 1 つ以上の結果で構成されています。ルールで使用される変数は、ディテクターが評価するイベントデータセットの一部である必要があります。さらに、各ディテクターには少なくとも 1 つのルールが関連付けられている必要があります。

結果

これは、不正予測の結果または出力です。不正予測で使用される各ルールは、1 つ以上の結果を指定する必要があります。

不正予測

不正予測は、1 つのイベントまたは一連のイベントに対する不正の評価です。Amazon Fraud Detector は、ルールに基づいてモデルスコアと結果を同期的に提供することで、単一のオンラインイベントの不正予測をリアルタイムで生成します。Amazon Fraud Detector は、一連のイベントの不正予測をオフラインで生成します。予測を使用して、オフラインproof-of-conceptを実行したり、不正リスクを時間単位、日単位、または週単位で遡及的に評価したりできます。

不正予測の説明

不正予測の説明は、各変数がモデルの不正予測スコアにどのように影響したかについてのインサイトを提供します。各変数がリスクスコアにどのように影響するかについて、大きさ (0～5 の範囲、5 が最高) と方向 (スコアを上下に動かす) の観点から情報を提供します。

# Amazon Fraud Detector の仕組み

Amazon Fraud Detector は、ビジネスにおける潜在的な不正なオンラインアクティビティを検出するようにカスタマイズされた機械学習モデルを構築します。開始するには、ビジネスユースケースを指定します。ビジネスユースケースに応じて、Amazon Fraud Detector は不正検出モデルの作成に使用するモデルタイプを推奨します。さらに、ビジネスの履歴データの一部として提供する必要があるデータ要素に関するインサイトも提供します。Amazon Fraud Detector は、履歴データセットを使用して、カスタマイズされたモデルを自動的に作成してトレーニングします。

自動モデルトレーニングプロセスには、特定のビジネスユースケースの不正を検出する機械学習アルゴリズムの選択、提供したデータの検証、モデルのパフォーマンスを向上させるためのデータ操作の実行が含まれます。モデルをトレーニングすると、Amazon Fraud Detector はモデルスコアやその他のモデルパフォーマンスメトリクスを生成します。スコアとパフォーマンスメトリクスを使用して、モデルのパフォーマンスを評価できます。必要に応じて、トレーニング用に指定したデータセットからデータ要素を追加または削除し、モデルを再トレーニングしてモデルスコアを向上させることができます。

モデルを作成、トレーニング、アクティブ化したら、ビジネスによって生成されたデータの解釈方法をモデルに指示するルールとも呼ばれる決定ロジックを設定し、各アクティビティの解釈方法の結果を割り当てる必要があります。結果は、アクティビティの承認やレビューなどのアクションを表すことも、高リスク、中リスク、低リスクなどのアクティビティのリスクレベルを表すこともできます。

ディテクターは、モデルと関連するルールを保持するコンテナです。ディテクターを作成、テストし、本番環境にデプロイする必要があります。

本番環境にデプロイされたディテクターは、ビジネスアプリケーションに不正検出機能を提供します。不正評価を実行するために、モデルはビジネスアクティビティから受信したすべてのデータをビジネスの履歴データと比較し、その高度な機械学習アルゴリズムを作成したルールを使用して結果を分析し、結果を割り当てます。Amazon Fraud Detector を使用すると、1 つのビジネスアクティビティからのデータをリアルタイムで評価することも、複数のビジネスアクティビティからのデータをオフラインで評価することもできます。

アクティビティの 1 つとしてオンライン資金の送金を行う企業があるとします。Amazon Fraud Detector を使用して、資金送金の不正なリクエストをリアルタイムで検出したい。開始するには、まず Amazon Fraud Detector に過去の資金送金リクエストのデータを提供する必要があります。Amazon Fraud Detector は、このデータを使用して、資金振替の不正なリクエストを検出するようにカスタマイズされたモデルを作成およびトレーニングします。次に、モデルを追加し、データを解釈するようにモデルのルールを設定して、ディテクターを作成します。オンライン資金振替アク

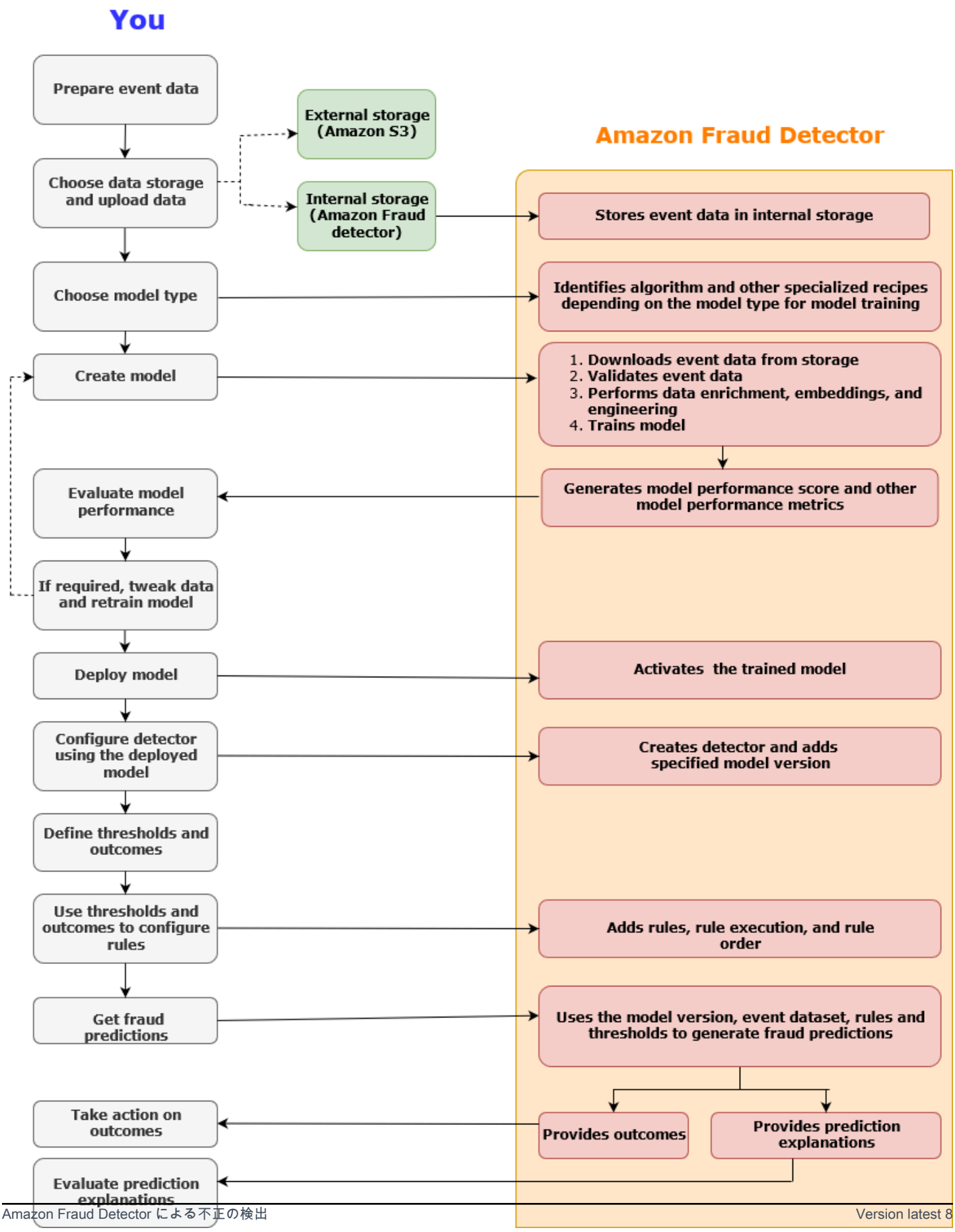
ティビティのルールの例としては、資金振替のリクエストが xyz@example.com の E メールアドレスから送信されている場合は、レビューリクエストを送信できます。ビジネスの本番環境では、資金移管のリクエストが入ると、モデルはリクエストに付随したデータを分析し、ルールを使用して結果を割り当てます。その後、割り当てられた結果に応じて、リクエストに対してアクションを実行できます。

Amazon Fraud Detector は、トレーニングデータセット、モデル、ディテクター、ルール、結果などのコンポーネントを使用して、ビジネスに不正評価ロジックを提供します。

Amazon Fraud Detector を使用して不正を検出するために使用するワークフローについては、「」を参照してください。 Amazon Fraud Detector による不正の検出

# Amazon Fraud Detector による不正の検出

このセクションでは、Amazon Fraud Detector で不正を検出するための一般的なワークフローについて説明します。また、これらのタスクを実行する方法についてもまとめています。次の図は、Amazon Fraud Detector で不正を検出するためのワークフローの概要を示しています。


You
Prepare event data
Choose data storage and upload data
External storage (Amazon S3)
Internal storage (Amazon Fraud detector)
Choose model type
Create model
Evaluate model performance
If required, tweak data and retrain model
Deploy model
Configure detector using the deployed model
Define thresholds and outcomes
Use thresholds and outcomes to configure rules
Get fraud predictions
Take action on outcomes
Evaluate prediction explanations
Amazon Fraud Detector
Stores event data in internal storage
Identifies algorithm and other specialized recipes depending on the model type for model training
1. Downloads event data from storage
2. Validates event data
3. Performs data enrichment, embeddings, and engineering
4. Trains model
Generates model performance score and other model performance metrics
Activates the trained model
Creates detector and adds specified model version
Adds rules, rule execution, and rule order
Uses the model version, event dataset, rules and thresholds to generate fraud predictions
Provides outcomes
Provides prediction explanations

不正検出は継続的なプロセスです。モデルをデプロイしたら、予測の説明に基づいてそのパフォーマンススコアとメトリクスを必ず評価してください。これにより、トップリスク指標を特定し、誤検出につながる根本原因を絞り込み、データセット全体の不正パターンを分析し、存在する場合はバイアスを検出できます。予測の精度を高めるために、データセットを微調整して新規または改訂されたデータを含めることができます。その後、更新されたデータセットを使用してモデルを再トレーニングできます。より多くのデータが利用可能になったら、精度を高めるためにモデルの再トレーニングを続けます。

# Amazon Fraud Detector へのアクセス

Amazon Fraud Detector は複数の で利用 AWS リージョン でき、 AWS インターフェイスを使用してアクセスできます。

## 可用性

Amazon Fraud Detector は、米国東部 (バージニア北部 ) 、米国東部 (オハイオ ) 、米国西部 (オレゴン ) 、欧州 (アイルランド ) 、アジアパシフィック (シンガポール ) 、アジアパシフィック (シドニー) で利用できます AWS リージョン。

## インターフェイス

次のいずれかのインターフェイスを使用して、不正検出モデルとディテクターを作成、トレーニング、デプロイ、テスト、実行、管理できます。

AWS Management Console - Amazon Fraud Detector は、ウェブベースのユーザーインターフェイスである Amazon Fraud Detector コンソールを提供します。にサインアップすると AWS アカウント、Amazon Fraud Detector コンソールにアクセスできます。詳細については、「Amazon Fraud Detector のセットアップ」を参照してください。

AWS Command Line Interface （ AWS CLI ） - コマンドラインシェルのコマンドを使用して AWS のサービス、Amazon Fraud Detector を含むさまざまな のセットとやり取りするために使用できるインターフェイスを提供します。Amazon Fraud Detector の AWS CLI コマンドは、Amazon Fraud Detector コンソールで提供される機能と同等の機能を実装します。

AWS SDK - 言語固有の APIs を提供し、署名の計算、リクエストの再試行処理、エラー処理など、接続の詳細の多くを管理します。詳細については、「構築するツール AWS」ページに移動し、SDK セクションまで下にスクロールし、プラス記号 (+) を選択してセクションを展開します。

AWS CloudFormation - Amazon Fraud Detector のリソースとプロパティの定義に使用できるテンプレートを提供します。詳細については、「 AWS CloudFormation ユーザーガイド」の「Amazon Fraud Detector リソースタイプのリファレンス」を参照してください。

# 料金

Amazon Fraud Detector では、使用した分に対してのみ料金が発生します。最低料金や前払いの義務はありません。モデルのトレーニングとホストに使用されたコンピューティング時間、使用するストレージの量、および行った不正予測の量に基づいて課金されます。詳細については、「Amazon Fraud Detector の料金」を参照してください。

# Amazon Fraud Detector のセットアップ

Amazon Fraud Detector を使用するには、まず Amazon Web Services (AWS) アカウントが必要です。次に、すべてのインターフェイス AWS アカウント へのアクセスを許可するアクセス許可を設定する必要があります。後で Amazon Fraud Detector リソースの作成を開始するときに、Amazon Fraud Detector がユーザーに代わってタスクを実行し、ユーザーが所有するリソースにアクセスすることを許可するアクセス許可を付与する必要があります。

Amazon Fraud Detector を使用するためのセットアップを行うには、このセクションの以下のタスクを実行します。

- にサインアップします AWS。
- が Amazon Fraud Detector インターフェイス AWS アカウント にアクセスできるようにするアクセス許可を設定します。
- Amazon Fraud Detector へのアクセスに使用するインターフェイスを設定します。

これらの手順を完了したら、「Amazon Fraud Detector の使用を開始する」を参照して Amazon Fraud Detector の開始方法を続けてください。

## にサインアップする AWS

Amazon Web Services (AWS) にサインアップすると AWS、 AWS アカウント は Amazon Fraud Detector を含む のすべてのサービスに自動的にサインアップされます。サービスを実際に使用した分の料金のみが請求されます。が既にある場合は AWS アカウント、次のタスクに進みます。

### にサインアップする AWS アカウント

がない場合は AWS アカウント、次の手順を実行して作成します。

にサインアップするには AWS アカウント

1. https://portal.aws.amazon.com/billing/signup を開きます。
2. オンラインの手順に従います。

   サインアップ手順の一部では、電話またはテキストメッセージを受信し、電話のキーパッドに検証コードを入力します。

にサインアップすると AWS アカウント、 AWS アカウントのルートユーザー が作成されます。ルートユーザーには、アカウントのすべての AWS のサービス とリソースへのアクセス権があります。セキュリティベストプラクティスとして、ユーザーに管理アクセス権を割り当て、ルートユーザーアクセスが必要なタスクの実行にはルートユーザーのみを使用するようにしてください。

AWS サインアッププロセスが完了すると、 から確認メールが送信されます。https://aws.amazon.com/ の [マイアカウント] をクリックして、いつでもアカウントの現在のアクティビティを表示し、アカウントを管理することができます。

## 管理アクセスを持つユーザーを作成する

にサインアップしたら AWS アカウント、日常的なタスクにルートユーザーを使用しないように AWS アカウントのルートユーザー、 を保護し AWS IAM Identity Center、 を有効にして管理ユーザーを作成します。

を保護する AWS アカウントのルートユーザー

1. ルートユーザーを選択し、 AWS アカウント E メールアドレスを入力して、アカウント所有者AWS Management Consoleとして にサインインします。次のページでパスワードを入力します。

   ルートユーザーを使用してサインインする方法については、AWS サインイン ユーザーガイドのルートユーザーとしてサインインするを参照してください。
2. ルートユーザーの多要素認証 (MFA) を有効にします。

   手順については、IAM ユーザーガイドの AWS アカウント 「ルートユーザー (コンソール) の仮想 MFA デバイスを有効にする」を参照してください。

管理アクセスを持つユーザーを作成する

1. IAM アイデンティティセンターを有効にします。

   手順については、「AWS IAM Identity Center ユーザーガイド」の「AWS IAM Identity Centerの有効化」を参照してください。
2. IAM アイデンティティセンターで、ユーザーに管理アクセスを付与します。

を ID ソース IAM アイデンティティセンターディレクトリ として使用する方法のチュートリアルについては、AWS IAM Identity Center 「 ユーザーガイド」の「デフォルトを使用してユーザーアクセスを設定する IAM アイデンティティセンターディレクトリ」を参照してください。

管理アクセス権を持つユーザーとしてサインインする

- IAM アイデンティティセンターのユーザーとしてサインインするには、IAM アイデンティティセンターのユーザーの作成時に E メールアドレスに送信されたサインイン URL を使用します。

  IAM Identity Center ユーザーを使用してサインインする方法については、AWS サインイン 「 ユーザーガイド」の AWS 「 アクセスポータルにサインインする」を参照してください。

追加のユーザーにアクセス権を割り当てる

1. IAM アイデンティティセンターで、最小特権のアクセス許可を適用するというベストプラクティスに従ったアクセス許可セットを作成します。

   手順については、「AWS IAM Identity Center ユーザーガイド」の「権限設定を作成する」を参照してください。
2. グループにユーザーを割り当て、そのグループにシングルサインオンアクセス権を割り当てます。

   手順については、「AWS IAM Identity Center ユーザーガイド」の「グループの結合」を参照してください。

# Amazon Fraud Detector インターフェイスにアクセスするためのアクセス許可を設定する

Amazon Fraud Detector を使用するには、Amazon Fraud Detector コンソールと API オペレーションにアクセスするためのアクセス許可を設定します。

セキュリティのベストプラクティスに従って、Amazon Fraud Detector オペレーションに制限されたアクセスと必要なアクセス許可を持つ AWS Identity and Access Management (IAM) ユーザーを作成します。必要に応じて他のアクセス許可を追加できます。

次のポリシーは、Amazon Fraud Detector を使用するために必要なアクセス許可を示します。

- `AmazonFraudDetectorFullAccessPolicy`

  次のアクションを実行できます。

  - Amazon Fraud Detector リソースへのアクセス
  - SageMaker AI のすべてのモデルエンドポイントを一覧表示して記述する
  - アカウント内のすべての IAM ロールを一覧表示する
  - Amazon S3 バケットをすべて一覧表示する
  - IAM パスロールが Amazon Fraud Detector にロールを渡すことを許可する
- `AmazonS3FullAccess`

  へのフルアクセスを許可します Amazon Simple Storage Service。これは、トレーニングデータセットを Amazon S3 にアップロードする必要がある場合に必要です。

IAM ユーザーを作成し、必要なアクセス許可を割り当てる方法について説明します。

ユーザーを作成し、必要なアクセス許可を割り当てるには

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/iam/ で IAM コンソールを開きます。
2. ナビゲーションペインで、[Users] (ユーザー)、[Add user] (ユーザーを追加する) の順に選択します。
3. [User name] (ユーザー名) に「**`AmazonFraudDetectorUser`**」と入力します。
4. AWS マネジメントコンソールのアクセスチェックボックスを選択し、ユーザーパスワードを設定します。
5. ( オプション) デフォルトでは、 は新しいユーザーが初めてサインインするときに新しいパスワードを作成 AWS する必要があります。[User must create a new password at next sign-in] (ユーザーは次回のサインイン時に新しいパスワードを作成する必要がある) の隣にあるチェックボックスのチェックを外して、新しいユーザーがサインインしてからパスワードをリセットできるようにできます。
6. [次へ: アクセス許可] を選択します。
7. [グループの作成] を選択します。
8. [グループ名] に「**`AmazonFraudDetectorGroup`**」と入力します。
9. ポリシーのリストで、AmazonFraudDetectorFullAccessPolicy と AmazonS3FullAccess のチェックボックスをオンにします。[グループの作成] を選択してください。

10. グループのリストで、新しいグループのチェックボックスをオンにします。リストにグループが表示されない場合は、更新を選択します。
11. [Next: Tags] (次へ: タグ) を選択します。
12. (オプション) タグをキーバリューペアとしてアタッチして、メタデータをユーザーに追加します。IAM でタグを使用する方法については、「IAM ユーザーとロールのタグ付け」を参照してください。
13. [次へ: 確認] を選択して、新しいユーザーのユーザーの詳細とアクセス許可の概要を確認します。続行する準備ができたら、[Create user (ユーザーの作成)] を選択します。

# を使用して Amazon Fraud Detector にアクセスするためのインターフェイスを設定する

Amazon Fraud Detector コンソール AWS CLI、または AWS SDK を使用して Amazon Fraud Detector にアクセスできます。これらを使用する前に、まず AWS CLI と AWS SDK をセットアップします。

## Amazon Fraud Detector コンソールにアクセスする

Amazon Fraud Detector コンソールやその他の AWS サービスには、 からアクセスできます AWS Management Console。では AWS アカウント、 へのアクセスが許可されます AWS Management Console。

Amazon Fraud Detector コンソールにアクセスするには、

1. に移動https://console.aws.amazon.com/し、 にサインインします AWS アカウント。
2. Amazon Fraud Detector に移動します。

Amazon Fraud Detector コンソールを使用すると、モデルと、ディテクター、変数、イベント、エンティティ、ラベル、結果などの不正検出リソースを作成および管理できます。予測を生成し、モデルのパフォーマンスと予測を評価できます。

## セットアップ AWS CLI

AWS Command Line Interface （ AWS CLI) を使用して Amazon Fraud Detector を操作するには、コマンドラインシェルでコマンドを実行します。最小限の設定で、 を使用して、ターミナルのコマン

ドプロンプトから Amazon Fraud Detector コンソールが提供する機能と同様の機能でコマンド AWS CLI を実行できます。

をセットアップするには AWS CLI

AWS CLIをダウンロードして設定します。手順については、 AWS Command Line Interface ユーザーガイドの以下のトピックを参照してください。

- AWS コマンドラインインターフェイスのセットアップ
- AWS コマンドラインインターフェイスの設定

Amazon Fraud Detector コマンドの詳細については、「使用可能なコマンド」を参照してください。

## AWS SDK のセットアップ

AWS SDKs を使用して、不正検出リソースを作成および管理するためのコードを記述したり、不正予測を取得したりできます。 AWS SDKs、JavaScript および Python (Boto3) の Amazon Fraud Detector をサポートしています。

をセットアップするには AWS SDK for Python (Boto3)

AWS SDK for Python (Boto3) を使用して、 AWS サービスを作成、設定、管理できます。Boto をインストールする方法については、AWS 「 SDK for Python (Boto3)」を参照してください。Boto3 SDK バージョン 1.14.29 以降を使用していることを確認します。

インストール後 AWS SDK for Python (Boto3)、次の Python の例を実行して、環境が正しく設定されていることを確認します。正しく設定されている場合、レスポンスにはディテクターのリストが含まれます。ディテクターが作成されていない場合、リストは空です。

```
import boto3
fraudDetector = boto3.client('frauddetector')

response = fraudDetector.get_detectors()
print(response)
```

Java 用 AWS SDKs をセットアップするには

をインストールしてロードする方法については AWS SDK for JavaScript、「 SDK for JavaScript のセットアップ」を参照してください。

# Amazon Fraud Detector の使用を開始する

開始する前に、「」のステップを読みAmazon Fraud Detector による不正の検出、完了していることを確認してくださいAmazon Fraud Detector のセットアップ。

このセクションの実践的なチュートリアルを使用して、Amazon Fraud Detector を使用して不正検出モデルを構築、トレーニング、デプロイする方法について説明します。このチュートリアルでは、機械学習モデルを使用して、新しいアカウント登録が不正であるかどうかを予測する不正アナリストの役割を引き受けます。モデルは、アカウント登録のデータを使用してトレーニングする必要があります。Amazon Fraud Detector は、このチュートリアルのアカウント登録データセットの例を提供します。チュートリアルを開始する前に、サンプルデータセットをアップロードする必要があります。

Amazon Fraud Detector の使用を開始するには、次のいずれかのインターフェイスを使用します。チュートリアルを開始する前に、次の手順に従ってください。 サンプルデータセットを取得してアップロードする

- チュートリアル: Amazon Fraud Detector コンソールの使用を開始する
- チュートリアル: の使用を開始する AWS SDK for Python (Boto3)

## サンプルデータセットを取得してアップロードする

このチュートリアルで使用するデータセットの例は、オンラインアカウント登録の詳細を示しています。データセットは、UTF-8 形式のカンマ区切り値 (CSV) を使用するテキストファイルにあります。CSV データセットファイルの最初の行には、ヘッダーが含まれています。ヘッダー行の後に複数のデータ行が続きます。これらの各行は、1 つのアカウント登録のデータ要素で構成されます。データはわかりやすいようにラベル付けされています。データセット内の列は、アカウント登録が不正かどうかを示します。

サンプルデータセットを取得してアップロードするには

1. サンプルに移動します。

   オンラインアカウント登録データがあるデータファイルは、register_data_20K_minimum.csv と registration_data_20K_full.csv の 2 つです。ファイル`registration_data_20K_minimum`には、ip_address と email_address の 2 つの変数のみが含まれます。ファイルには他の変数`registration_data_20K_full`が含まれています。これらの変数はイベントごとに使用され、billing_address、phone_number、user_agent が含まれます。両方のデータファイルには、次の 2 つの必須フィールドも含まれています。

- EVENT_TIMESTAMP — いつイベントが発生したかを定義します
- EVENT_LABEL - イベントを不正または正当として分類します

このチュートリアルでは、2 つのファイルのいずれかを使用できます。使用するデータファイルをダウンロードします。

2. Amazon Simple Storage Service (Amazon S3) バケットを作成します。

   このステップでは、データセットを保存する外部ストレージを作成します。この外部ストレージは Amazon S3 バケットです。Amazon S3 の詳細については、Amazon S3とは」を参照してください。

   a. にサインイン AWS Management Console し、https://console.aws.amazon.com/s3/ で Amazon S3 コンソールを開きます。

   b. [Buckets] (バケット) で、[Create bucket] (バケットの作成) を選択します。

   c. [バケット名] に、バケットの名前を入力します。コンソールでバケットの命名規則に従い、グローバルに一意の名前を指定してください。バケットの目的を説明する名前を使用することをお勧めします。

   d. でAWS リージョン、バケットを作成する AWS リージョン を選択します。選択したリージョンは Amazon Fraud Detector をサポートしている必要があります。レイテンシーを減らすには、地理的な場所に最も AWS リージョン 近い を選択します。Amazon Fraud Detector をサポートするリージョンのリストについては、「 グローバルインフラストラクチャガイド」の「リージョン表」を参照してください。

   e. このチュートリアルでは、オブジェクト所有権、パブリックアクセスブロックのバケット設定、バケットバージョニング、タグのデフォルト設定のままにします。

   f. デフォルトの暗号化では、このチュートリアルで Disable を選択します。

   g. バケット設定を確認し、バケットの作成を選択します。

3. サンプルデータファイルを Amazon S3 バケットにアップロードします。

   バケットを作成したら、先ほどダウンロードしたサンプルファイルの 1 つを、先ほど作成した Amazon S3 バケットにアップロードします。

   a. バケットには、バケット名が一覧表示されます。バケットを選択します。

   b. [アップロード] を選択します。

   c. [Files and folders] (ファイルとフォルダ) で、[Add files] (ファイルを追加) を選択します。

d. コンピュータにダウンロードしたサンプルデータファイルの 1 つを選択し、開くを選択します。

e. Destination、Permissions、Properties のデフォルト設定のままにします。

f. 設定を確認し、アップロードを選択します。

g. サンプルデータファイルは Amazon S3 バケットにアップロードされます。バケットの場所を書き留めます。オブジェクトで、アップロードしたサンプルデータファイルを選択します。

h. オブジェクトの概要で、S3 URI の下に場所をコピーします。これは、サンプルデータファイルの Amazon S3 の場所です。後で使用します。さらに、S3 バケットの Amazon リソースネーム (ARN) をコピーして保存することもできます。

# チュートリアル: Amazon Fraud Detector コンソールの使用を開始する

このチュートリアルは 2 つの部分で構成されています。最初のパートでは、不正検出モデルを構築、トレーニング、デプロイする方法について説明します。2 番目のパートでは、モデルを使用して不正予測をリアルタイムで生成する方法について説明します。モデルは、S3 バケットにアップロードしたサンプルデータファイルを使用してトレーニングされます。このチュートリアルを完了すると、以下のアクションが完了します。

- Amazon Fraud Detector モデルの構築とトレーニング
- リアルタイムの不正予測を生成する

> **Important**
>
> 続行する前に、「」の手順に従っていることを確認してください。 サンプルデータセットを取得してアップロードする

## パート A: Amazon Fraud Detector モデルの構築、トレーニング、デプロイ

パート A では、ビジネスユースケースの定義、イベントの定義、モデルの構築、モデルのトレーニング、モデルのパフォーマンスの評価、モデルのデプロイを行います。

ステップ 1: ビジネスユースケースを選択する

- このステップでは、データモデルエクスプローラーを使用して、ビジネスユースケースを Amazon Fraud Detector でサポートされている不正検出モデルタイプと一致させます。データモデルエクスプローラーは、Amazon Fraud Detector コンソールと統合されたツールで、ビジネスユースケースの不正検出モデルの作成とトレーニングに使用するモデルタイプを推奨します。データモデルエクスプローラーは、データセットに含める必要がある必須、推奨、およびオプションのデータ要素に関するインサイトも提供します。データセットは、不正検出モデルの作成とトレーニングに使用されます。

  このチュートリアルでは、ビジネスユースケースは新しいアカウント登録です。ビジネスユースケースを指定すると、データモデルエクスプローラーは不正検出モデルを作成するためのモデルタイプを推奨し、データセットの作成に必要なデータ要素のリストも提供します。新しいアカウント登録のデータを含むサンプルデータセットを既にアップロードしているため、新しいデータセットを作成する必要はありません。

  a. AWS マネジメントコンソールを開き、アカウントにサインインします。Amazon Fraud Detector に移動します。

  b. 左側のナビゲーションペインで、データモデルエクスプローラーを選択します。

  c. データモデルエクスプローラーページの「ビジネスユースケース」で、「新規アカウントの不正」を選択します。

  d. Amazon Fraud Detector は、選択したビジネスユースケースの不正検出モデルを作成するために使用する推奨モデルタイプを表示します。モデルタイプは、Amazon Fraud Detector が不正検出モデルのトレーニングに使用するアルゴリズム、エンリッチメント、および変換を定義します。

     推奨されるモデルタイプを書き留めます。これは、後でモデルを作成するときに必要になります。

  e. データモデルのインサイトペインには、不正検出モデルの作成とトレーニングに必要な必須および推奨のデータ要素に関するインサイトが表示されます。

     ダウンロードしたサンプルデータセットを見て、必須データ要素と推奨データ要素がすべて表にリストされていることを確認します。

     後で特定のビジネスユースケースのモデルを作成するときに、提供されたインサイトを使用してデータセットを作成します。

ステップ 2: イベントタイプを作成する

- このステップでは、不正を評価するビジネスアクティビティ (イベント) を定義します。イベントを定義するには、データセット内の変数、イベントを開始するエンティティ、およびイベントを分類するラベルを設定します。このチュートリアルでは、アカウント登録イベントを定義します。

  a. AWS マネジメントコンソールを開き、アカウントにサインインします。Amazon Fraud Detector に移動します。

  b. 左側のナビゲーションペインで [イベント] を選択します。

  c. イベントタイプページで、作成を選択します。

  d. イベントタイプの詳細で、イベントタイプ名`sample_registration`として を入力し、オプションでイベントの説明を入力します。

  e. Entity で、Create entity を選択します。

  f. エンティティの作成ページで、エンティティタイプ名`sample_customer`として を入力します。必要に応じて、エンティティタイプの説明を入力します。

  g. [エンティティの作成] を選択します。

  h. イベント変数 で、このイベントの変数を定義する方法を選択する で、トレーニングデータセットから変数を選択 を選択します。

  i. IAM ロール で、IAM ロールの作成 を選択します。

  j. 「IAM ロールの作成」ページで、サンプルデータをアップロードした S3 バケットの名前を入力し、「ロールの作成」を選択します。

  k. データの場所で、サンプルデータへのパスを入力します。これは、サンプルデータをアップロードした後に保存した`S3 URI`パスです。パスは のようになります`S3://your-bucket-name/example dataset filename.csv`。

  l. [アップロード] を選択します。

     Amazon Fraud Detector は、サンプルデータファイルからヘッダーを抽出し、変数タイプにマッピングします。マッピングがコンソールに表示されます。

  m. ラベル - オプションで、ラベル で新しいラベルの作成 を選択します。

  n. ラベルの作成ページで、名前`fraud`として を入力します。このラベルは、サンプルデータセット内の不正なアカウント登録を表す値に対応します。

  o. [ラベルの作成] を選択します。

p. 2 番目のラベルを作成し、名前`legit`として を入力します。このラベルは、サンプルデータセット内の正当なアカウント登録を表す値に対応します。

q. [イベントタイプの作成] を選択します。

### ステップ 3: モデルを作成する

1. モデルページで、モデルの追加を選択し、モデルの作成を選択します。
2. ステップ 1 – モデルの詳細を定義するには、モデル名`sample_fraud_detection_model`として を入力します。必要に応じて、モデルの説明を追加します。
3. [モデルタイプ] では、オンライン不正インサイトモデルを選択します。
4. イベントタイプで sample_registration を選択します。これは、ステップ 1 で作成したイベントタイプです。
5. 履歴イベントデータで、

   a. イベントデータソースで、S3 に保存されているイベントデータを選択します。

   b. IAM ロールで、ステップ 1 で作成したロールを選択します。

   c. トレーニングデータの場所で、サンプルデータファイルへの S3 URI パスを入力します。
6. [Next (次へ)] を選択します。

### ステップ 4: モデルをトレーニングする

1. [モデル入力] で、すべてのチェックボックスをオンのままにします。デフォルトでは、Amazon Fraud Detector は履歴イベントデータセットのすべての変数をモデル入力として使用します。
2. ラベル分類では、不正ラベルはサンプルデータセット内の不正イベントを表す値に対応するため、不正ラベルでは不正を選択します。正当なラベルの場合は、このラベルがサンプルデータセット内の正当なイベントを表す値に対応するため、正当を選択します。
3. ラベルなしイベントの処理では、このデータセット例のラベルなしイベントを無視するをデフォルトの選択のままにします。
4. [Next (次へ)] を選択します。
5. 確認後、[モデルの作成とトレーニング] を選択します。Amazon Fraud Detector はモデルを作成し、モデルの新しいバージョンのトレーニングを開始します。

   モデルバージョンでは、ステータス列にモデルトレーニングのステータスが表示されます。サンプルデータセットを使用するモデルトレーニングの完了には約 45 分かかります。モデルトレーニングが完了すると、ステータスはデプロイ準備完了に変わります。

### ステップ 5: モデルのパフォーマンスを確認する

Amazon Fraud Detector を使用する上で重要なステップは、モデルスコアとパフォーマンスメトリクスを使用してモデルの精度を評価することです。モデルトレーニングが完了すると、Amazon Fraud Detector はモデルのトレーニングに使用されなかったデータの 15% を使用してモデルのパフォーマンスを検証し、モデルのパフォーマンススコアやその他のパフォーマンスメトリクスを生成します。

1. モデルのパフォーマンスを表示するには、
   a. Amazon Fraud Detector コンソールの左のナビゲーションペインで、[モデル] を選択します。
   b. モデルページで、トレーニングしたモデル (sample_fraud_detection_model) を選択し、1.0 を選択します。これは、Amazon Fraud Detector がモデルに対して作成したバージョンです。
2. Amazon Fraud Detector がこのモデルに対して生成したモデルパフォーマンスの総合スコアとその他のすべてのメトリクスを確認します。

   このページのモデルパフォーマンススコアとパフォーマンスメトリクスの詳細については、モデルスコア「」および「」を参照してくださいモデルパフォーマンスメトリクス。

   トレーニング済みのすべての Amazon Fraud Detector モデルには、このチュートリアルでモデルについて確認したパフォーマンスメトリクスと同様の実際の不正検出パフォーマンスメトリクスがあることが期待できます。

### ステップ 6: モデルをデプロイする

トレーニング済みモデルのパフォーマンスメトリクスを確認し、不正予測を生成する準備ができたら、モデルをデプロイできます。

1. Amazon Fraud Detector コンソールの左側のナビゲーションペインで、モデルを選択します。
2. モデルページで sample_fraud_detection_model を選択し、デプロイする特定のモデルバージョンを選択します。このチュートリアルでは、1.0 を選択します。
3. [モデルバージョン] ページで、[アクション] を選択してから、モデルバージョンのデプロイ] をクリックします。
4. モデルバージョンでは、ステータスにデプロイのステータスが表示されます。デプロイが完了すると、ステータスはアクティブに変わります。これは、モデルバージョンがアクティブ化され、不正予測を生成できることを示します。「」に進みパート B: 不正予測を生成する、不正予測を生成するステップを完了します。

# パート B: 不正予測を生成する

不正予測は、ビジネスアクティビティ (イベント) に対する不正の評価です。Amazon Fraud Detector は、ディテクターを使用して不正予測を生成します。ディテクターには、不正について評価する特定のイベントのモデルやルールなどの検出ロジックが含まれています。検出ロジックは、ルールを使用して、モデルに関連付けられたデータの解釈方法を Amazon Fraud Detector に指示します。このチュートリアルでは、前にアップロードしたアカウント登録サンプルデータセットを使用して、アカウント登録イベントを評価します。

パート A では、モデルの作成、トレーニング、およびデプロイを行いました。パート B では、`sample_registration`イベントタイプのディテクターを構築し、デプロイされたモデルを追加し、ルールとルール実行順序を作成してから、不正予測の生成に使用するディテクターのバージョンを作成してアクティブ化します。

ステップ 1: ディテクターを構築する

ディテクターを作成するには

1. Amazon Fraud Detector コンソールの左のナビゲーションペインで、[ディテクター] をクリックします。
2. [ディテクターの作成] を選択します。
3. ディテクターの詳細の定義ページで、ディテクター名`sample_detector`に を入力します。必要に応じて、 などのディテクターの説明を入力します`my sample fraud detector`。
4. イベントタイプ で sample_registration を選択します。これは、このチュートリアルのパート A で作成したイベントです。
5. [Next (次へ)] を選択します。

ステップ 2: モデルを追加する

このチュートリアルのパート A を完了した場合は、ディテクターに追加できる Amazon Fraud Detector モデルが既にある可能性があります。モデルをまだ作成していない場合は、パート A に進み、モデルを作成、トレーニング、デプロイするステップを完了してからパート B に進みます。

1. モデルの追加 - オプションで、モデルの追加を選択します。
2. モデルの追加ページのモデルの選択で、前にデプロイした Amazon Fraud Detector モデル名を選択します。バージョンの選択 で、デプロイされたモデルのモデルバージョンを選択します。
3. [モデルの追加] を選択します。

4. [Next (次へ)] を選択します。

### ステップ 3: ルールを追加する

ルールは、不正予測を評価するときにモデルパフォーマンススコアを解釈する方法を Amazon Fraud Detector に指示する条件です。このチュートリアルでは、、、high_fraud_risk、medium_fraud_riskの 3 つのルールを作成しますlow_fraud_risk。

1. 「ルールの追加」ページの「ルールの定義high_fraud_risk」で、ルール名に「説明 - オプション」で、ルールの説明**This rule captures events with a high ML model score**として「」と入力します。
2. [式] で、Amazon Fraud Detector 簡易ルール式言語を使用して、次のルール式を入力します。

   ```
   $sample_fraud_detection_model_insightscore > 900
   ```
3. [結果] では、[新しい結果の作成] を選択します。結果は、不正予測の結果であり、評価中にルールが一致した場合に返されます。
4. [新しい結果の作成] には、結果名として「verify_customer」と入力します。必要に応じて説明に説明を入力します。
5. [結果の保存] を選択します。
6. [ルールの追加] を選択して、ルール検証チェッカーを実行し、ルールを保存します。作成後、Amazon Fraud Detector はルールをディテクターで使用できるようにします。
7. [別のルールの追加] を選択してから、[ルールの作成] タブをクリックします。
8. このプロセスをさらに 2 回繰り返して、次のルールの詳細を使用して medium_fraud_risk と low_fraud_risk ルールを作成します。

   - medium_fraud_risk

     ルール名: medium_fraud_risk

     結果: review

     式:

     ```
     $sample_fraud_detection_model_insightscore <= 900 and

     $sample_fraud_detection_model_insightscore > 700
     ```
   - low_fraud_risk

ルール名: low_fraud_risk

結果: approve

式:

$sample_fraud_detection_model_insightscore <= 700

これらの値は、このチュートリアルで使用される例です。独自のディテクターのルールを作成するときは、モデルとユースケースに適した値を使用します。

9. すべてのルールを追加したら、[次へ] を選択します。

ルールの作成と記述の詳細については、「ルール」および「ルール言語リファレンス」を参照してください。

ステップ 4: ルールの実行とルールの順序を設定する

ディテクターに含まれるルールのルール実行モードは、定義したすべてのルールが評価されるか、最初に一致したルールでルール評価が停止されるかを決定します。また、ルールの順序によって、ルールを実行する順序が決まります。

デフォルトのルール実行モードは FIRST_MATCHED です。

最初の一致

最初の一致のルール実行モードは、定義されたルールの順序に基づいて、最初の一致ルールの結果を返します。FIRST_MATCHED を指定した場合、Amazon Fraud Detector はルールを順番に評価し、最初に一致したルールで停止します。Amazon Fraud Detector は、その 1 つのルールの結果を示します。

ルールを実行する順序は、結果として生じる不正予測の結果に影響を与える可能性があります。ルールを作成したら、以下の手順に従って、ルールの順序を変更して必要な順序で実行します。

high_fraud_risk ルールがルールリストの上部にない場合は、順序を選択し、1 を選択します。これにより、high_fraud_risk が一番上に移動します。

このプロセスを繰り返して、medium_fraud_risk ルールが 2 番目の位置に来て、low_fraud_risk ルールが 3 番目の位置に来るようにします。

すべての一致

すべての一致ルール実行モードは、ルールの順序に関係なく、一致したすべてのルールの結果を返します。ALL_MATCHED を指定した場合、Amazon Fraud Detector はすべてのルールを評価し、一致したすべてのルールの結果を返します。

このチュートリアルFIRST_MATCHEDで を選択し、次へ を選択します。

ステップ 5: ディテクターバージョンを確認して作成する

ディテクターバージョンは、不正予測の生成に使用される特定のモデルとルールを定義します。

1. 確認と作成ページで、設定したディテクターの詳細、モデル、ルールを確認します。何らかの変更を加える必要がある場合は、対応するセクションの隣にある [編集] をクリックします。
2. [ディテクターの作成] を選択します。ディテクターの作成後、ディテクターの最初のバージョンが Draft ステータスで Detector バージョンテーブルに表示されます。

   ドラフトバージョンを使用して、ディテクターをテストします。

ステップ 6: ディテクターバージョンをテストしてアクティブ化する

Amazon Fraud Detector コンソールでは、テストの実行機能でモックデータを使用してディテクターのロジックをテストできます。このチュートリアルでは、サンプルデータセットのアカウント登録データを使用できます。

1. [ディテクターバージョンの詳細] ページの下部にある [テストの実行] までスクロールします。
2. イベントメタデータには、イベントが発生した時刻のタイムスタンプを入力し、イベントを実行するエンティティの一意の識別子を入力します。このチュートリアルでは、タイムスタンプの日付ピッカーから日付を選択し、エンティティ ID に「1234」と入力します。
3. イベント変数に、テストする変数値を入力します。このチュートリアルでは、 フィールドip_addressと email_addressフィールドのみが必要です。これは、Amazon Fraud Detector モデルのトレーニングに使用される入力であるためです。次のサンプル値を使用できます。これは、推奨される変数名を使用したことを前提としています。

   - ip_address: 205.251.233.178
   - email_address: johndoe@exampledomain.com
4. [テストの実行] を選択します。

5. Amazon Fraud Detector は、ルール実行モードに基づいて不正予測の結果を返します。ルール実行モードが の場合FIRST_MATCHED、返される結果は一致した最初のルールに対応します。最初のルールは、優先順位が最も高いルールです。true と評価された場合に一致します。ルール実行モードが の場合ALL_MATCHED、返される結果は一致したすべてのルールに対応します。つまり、それらはすべて true と評価されます。Amazon Fraud Detector は、ディテクターに追加されたモデルのモデルスコアも返します。

   入力を変更し、いくつかのテストを実行して、さまざまな結果を確認できます。テストにはサンプルデータセットの ip_address 値と email_address 値を使用し、結果が期待どおりであるかどうかを確認できます。

6. ディテクターの動作に満足したら、 から Draftに昇格しますActive。これにより、ディテクターをリアルタイムの不正検出に使用できます。

   [ディテクターバージョンの詳細] ページで、[アクション]、[発行]、[バージョンの発行] の順に選択します。これにより、ディテクターのステータスがドラフトからアクティブに変更されます。

   この時点で、モデルと関連するディテクターロジックは、Amazon Fraud Detector GetEventPrediction API を使用してオンラインアクティビティをリアルタイムで評価できます。CSV 入力ファイルと CreateBatchPredictionJob API を使用して、イベントをオフラインで評価することもできます。不正予測の詳細については、「」を参照してください。 不正予測

このチュートリアルを完了すると、次のことが実行されました。

- イベントデータセットの例を Amazon S3 にアップロードしました。
- サンプルデータセットを使用して Amazon Fraud Detector 不正検出モデルを作成し、トレーニングしました。
- Amazon Fraud Detector が生成したモデルパフォーマンススコアおよびその他のパフォーマンスメトリクスを表示しました。
- 不正検出モデルをデプロイしました。
- ディテクターを作成し、デプロイされたモデルを追加しました。
- ディテクターにルール、ルール実行順序、結果を追加しました。
- 異なる入力を提供し、ルールとルール実行順序が期待どおりに機能するかどうかを確認することで、ディテクターをテストしました。
- ディテクターを公開してアクティブ化しました。

# チュートリアル: の使用を開始する AWS SDK for Python (Boto3)

このチュートリアルでは、Amazon Fraud Detector モデルを構築してトレーニングし、このモデルを使用して を使用してリアルタイムの不正予測を生成する方法について説明します AWS SDK for Python (Boto3)。モデルは、Amazon S3 バケットにアップロードしたアカウント登録サンプルデータファイルを使用してトレーニングされます。

このチュートリアルを完了すると、以下のアクションが完了します。

- Amazon Fraud Detector モデルの構築とトレーニング
- リアルタイムの不正予測を生成する

## 前提条件

このチュートリアルの前提条件となる手順を次に示します。

- 完了しましたAmazon Fraud Detector のセットアップ。

  すでに を使用している場合はAWS SDK のセットアップ、Boto3 SDK バージョン 1.14.29 以降を使用していることを確認してください。
- このチュートリアルに必要な サンプルデータセットを取得してアップロードする ファイルの指示に従った。

## はじめに

ステップ 1: Python 環境をセットアップして検証する

Boto は、Amazon Web Services (AWS) SDK for Python です。これを使用して、作成、設定、管理を行うことができます AWS のサービス。Boto3 をインストールする方法については、「AWS SDK for Python (Boto3)」を参照してください。

インストール後 AWS SDK for Python (Boto3)、次の Python サンプルコマンドを実行して、環境が正しく設定されていることを確認します。環境が正しく設定されている場合、レスポンスにはディテクターのリストが含まれます。ディテクターが作成されていない場合、リストは空です。

```
import boto3
fraudDetector = boto3.client('frauddetector')

response = fraudDetector.get_detectors()
```

```
print(response)
```

### ステップ 2: 変数、エンティティタイプ、およびラベルを作成する

このステップでは、モデル、イベント、ルールの定義に使用されるリソースを作成します。

#### 変数の作成

変数は、イベントタイプ、モデル、ルールの作成に使用するデータセットのデータ要素です。

次の例では、CreateVariable API を使用して 2 つの変数を作成します。変数は email_addressとですip_address。対応する変数タイプに割り当てます: EMAIL_ADDRESSおよび IP_ADDRESS。これらの変数は、アップロードしたサンプルデータセットの一部です。変数タイプを指定すると、Amazon Fraud Detector はモデルトレーニング中および予測の取得時に変数を解釈します。モデルトレーニングに使用できるのは、関連する変数タイプを持つ変数のみです。

```
import boto3
fraudDetector = boto3.client('frauddetector')

 #Create variable email_address
fraudDetector.create_variable(
    name = 'email_address',
    variableType = 'EMAIL_ADDRESS',
    dataSource = 'EVENT',
    dataType = 'STRING',
    defaultValue = '<unknown>'
)

#Create variable ip_address
fraudDetector.create_variable(
    name = 'ip_address',
    variableType = 'IP_ADDRESS',
    dataSource = 'EVENT',
    dataType = 'STRING',
    defaultValue = '<unknown>'
)
```

#### エンティティタイプを作成する

エンティティはイベントを実行しているユーザーを表し、エンティティタイプはエンティティを分類します。分類の例には、顧客、マーチャント、またはアカウントが含まれます。

次の例では、PutEntityType API を使用してsample_customerエンティティタイプを作成します。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.put_entity_type(
   name = 'sample_customer',
   description = 'sample customer entity type'
)
```

ラベルの作成

ラベルは、イベントを不正または正当なものとして分類し、不正検出モデルのトレーニングに使用されます。モデルは、これらのラベル値を使用してイベントを分類する方法を学習します。

次の例では、Putlabel API を使用して fraudと の 2 つのラベルを作成しますlegit。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.put_label(
    name = 'fraud',
    description = 'label for fraud events'
)

fraudDetector.put_label(
    name = 'legit',
    description = 'label for legitimate events'
)
```

ステップ 3: イベントタイプを作成する

Amazon Fraud Detector を使用すると、リスクを評価し、個々のイベントの不正予測を生成するモデルを構築できます。イベントタイプは、個々のイベントの構造を定義します。

次の例では、PutEventType API を使用してイベントタイプ を作成しますsample_registration。イベントタイプを定義するには、前のステップで作成した変数 (email_address、ip_address ) 、エンティティタイプ (sample_customer ) 、ラベル (fraud、legit) を指定します。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.put_event_type (
     name = 'sample_registration',
     eventVariables = ['ip_address', 'email_address'],
     labels = ['legit', 'fraud'],
     entityTypes = ['sample_customer'])
```

ステップ 4: モデルを作成、トレーニング、デプロイする

Amazon Fraud Detector は、特定のイベントタイプの不正を検出する方法を学習するためにモデルをトレーニングします。前のステップでは、 イベントタイプを作成しました。このステップでは、イベントタイプのモデルを作成してトレーニングします。モデルは、モデルバージョンのコンテナとして機能します。モデルをトレーニングするたびに、新しいバージョンが作成されます。

次のサンプルコードを使用して、オンライン不正インサイトモデルを作成およびトレーニングします。このモデルは と呼ばれますsample_fraud_detection_model。Amazon S3 にアップロードしたアカウント登録サンプルデータセットsample_registrationを使用するイベントタイプ用です。

Amazon Fraud Detector がサポートするさまざまなモデルタイプの詳細については、「」を参照してくださいモデルタイプの選択。

モデルを作成する

次の例では、CreateModel API を使用してモデルを作成します。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.create_model (
       modelId = 'sample_fraud_detection_model',
       eventTypeName = 'sample_registration',
       modelType = 'ONLINE_FRAUD_INSIGHTS')
```

モデルをトレーニングする

次の例では、CreateModelVersion API を使用してモデルをトレーニングします。'EXTERNAL_EVENTS' に を指定trainingDataSourceし、サンプルデータセッ

トを保存した Amazon S3 の場所と、 の Amazon S3 バケットの RoleArn を指定しますexternalEventsDetail。trainingDataSchema パラメータには、Amazon Fraud Detector がサンプルデータを解釈する方法を指定します。具体的には、含める変数と、イベントラベルを分類する方法を指定します。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.create_model_version (
         modelId = 'sample_fraud_detection_model',
         modelType = 'ONLINE_FRAUD_INSIGHTS',
         trainingDataSource = 'EXTERNAL_EVENTS',
         trainingDataSchema = {
            'modelVariables' : ['ip_address', 'email_address'],
            'labelSchema' : {
               'labelMapper' : {
                   'FRAUD' : ['fraud'],
                   'LEGIT' : ['legit']
        }
    }
},
         externalEventsDetail = {
              'dataLocation' : 's3://amzn-s3-demo-bucket/your-example-data-
filename.csv',
              'dataAccessRoleArn' : 'role_arn'
}
)
```

モデルを複数回トレーニングできます。モデルをトレーニングするたびに、新しいバージョンが作成されます。モデルトレーニングが完了すると、モデルバージョンのステータスは に更新されますTRAINING_COMPLETE。モデルパフォーマンススコアやその他のモデルパフォーマンスメトリクスを確認できます。

モデルのパフォーマンスを確認する

Amazon Fraud Detector を使用する上で重要なステップは、モデルスコアとパフォーマンスメトリクスを使用してモデルの精度を評価することです。モデルトレーニングが完了すると、Amazon Fraud Detector は、モデルのトレーニングに使用されなかったデータの 15% を使用してモデルのパフォーマンスを検証します。モデルのパフォーマンススコアやその他のパフォーマンスメトリクスを生成します。

DescribeModelVersions API を使用してモデルのパフォーマンスを確認します。このモデルのモデルパフォーマンスの総合スコアと、Amazon Fraud Detector によって生成されたその他のすべてのメトリクスを確認します。

モデルのパフォーマンススコアとパフォーマンスメトリクスの詳細については、モデルスコア「」および「」を参照してくださいモデルパフォーマンスメトリクス。

トレーニング済みのすべての Amazon Fraud Detector モデルには、このチュートリアルのメトリクスと同様の実際の不正検出パフォーマンスメトリクスがあることが期待できます。

モデルのデプロイ

トレーニング済みモデルのパフォーマンスメトリクスを確認したら、モデルをデプロイし、Amazon Fraud Detector が不正予測を生成できるようにします。トレーニング済みモデルをデプロイするには、UpdateModelVersionStatus API を使用します。次の例では、モデルバージョンのステータスを ACTIVE に更新するために使用されます。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.update_model_version_status (
     modelId = 'sample_fraud_detection_model',
     modelType = 'ONLINE_FRAUD_INSIGHTS',
     modelVersionNumber = '1.00',
     status = 'ACTIVE'
)
```

ステップ 5: ディテクター、結果、ルール、ディテクターバージョンを作成する

ディテクターには、モデルやルールなどの検出ロジックが含まれています。このロジックは、不正について評価する特定のイベント用です。ルールは、予測中に変数値を解釈する方法を Amazon Fraud Detector に指示するために指定する条件です。また、結果は不正予測の結果です。ディテクターは複数のバージョンを持つことができ、各バージョンのステータスは DRAFT、ACTIVE、または INACTIVE です。ディテクターバージョンには、少なくとも 1 つのルールが関連付けられている必要があります。

次のサンプルコードを使用して、ディテクター、ルール、結果を作成し、ディテクターを発行します。

ディテクターを作成する

次の例では、PutDetector API を使用してsample_registrationイベントタイプのsample_detectorディテクターを作成します。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.put_detector (
     detectorId = 'sample_detector',
     eventTypeName = 'sample_registration'
)
```

結果を作成する

結果は、考えられる不正予測結果ごとに作成されます。次の例では、PutOutcome API を使用して、verify_customer、、 reviewの 3 つの結果を作成しますapprove。これらの結果は後でルールに割り当てられます。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.put_outcome(
     name = 'verify_customer',
     description = 'this outcome initiates a verification workflow'
    )

fraudDetector.put_outcome(
     name = 'review',
     description = 'this outcome sidelines event for review'
    )

fraudDetector.put_outcome(
     name = 'approve',
     description = 'this outcome approves the event'
)
```

ルールの作成

ルールは、データセットの 1 つ以上の変数、ロジック式、および 1 つ以上の結果で構成されます。

次の例では、CreateRule API を使用して、high_risk、、 medium_riskの 3 つの異なるルールを作成しますlow_risk。ルール式を作成して、モデルのパフォーマンススコアsample_fraud_detection_model_insightscore値をさまざまなしきい値と比較します。これは、イベントのリスクレベルを決定し、前のステップで定義した結果を割り当てます。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.create_rule(
     ruleId = 'high_fraud_risk',
     detectorId = 'sample_detector',
     expression = '$sample_fraud_detection_model_insightscore > 900',
     language = 'DETECTORPL',
     outcomes = ['verify_customer']
     )

fraudDetector.create_rule(
     ruleId = 'medium_fraud_risk',
     detectorId = 'sample_detector',
     expression = '$sample_fraud_detection_model_insightscore <= 900 and
 $sample_fraud_detection_model_insightscore > 700',
     language = 'DETECTORPL',
     outcomes = ['review']
     )

fraudDetector.create_rule(
     ruleId = 'low_fraud_risk',
     detectorId = 'sample_detector',
     expression = '$sample_fraud_detection_model_insightscore <= 700',
     language = 'DETECTORPL',
     outcomes = ['approve']
     )
```

ディテクターバージョンを作成する

ディテクターバージョンは、不正予測の取得に使用されるモデルとルールを定義します。

次の例では、CreateDetectorVersion API を使用してディテクターバージョンを作成します。これを行うには、モデルバージョンの詳細、ルール、ルール実行モード FIRST_MATCHED を指定します。ルール実行モードは、ルールを評価するシーケンスを指定します。ルール実行モード

FIRST_MATCHED は、ルールが最初から最後まで順番に評価され、最初に一致したルールで停止することを指定します。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.create_detector_version(
      detectorId = 'sample_detector',
      rules = [{
          'detectorId' : 'sample_detector',
          'ruleId' : 'high_fraud_risk',
          'ruleVersion' : '1'
},
{
          'detectorId' : 'sample_detector',
          'ruleId' : 'medium_fraud_risk',
          'ruleVersion' : '1'
},
{
          'detectorId' : 'sample_detector',
          'ruleId' : 'low_fraud_risk',
          'ruleVersion' : '1'
}
],
      modelVersions = [{
          'modelId' : 'sample_fraud_detection_model',
          'modelType': 'ONLINE_FRAUD_INSIGHTS',
          'modelVersionNumber' : '1.00'
}      ],
      ruleExecutionMode = 'FIRST_MATCHED'
)
```

ステップ 6: 不正予測を生成する

このチュートリアルの最後のステップでは、前のステップでsample_detector作成したディテクターを使用して、sample_registrationイベントタイプの不正予測をリアルタイムで生成します。ディテクターは、Amazon S3 にアップロードされたサンプルデータを評価します。レスポンスには、モデルのパフォーマンススコアと、一致したルールに関連付けられている結果が含まれます。

次の例では、GetEventPrediction API を使用して、リクエストごとに 1 つのアカウント登録からデータを提供します。このチュートリアルでは、アカウント登録サンプルデータファイルからデータ (email_address と ip_address) を取得します。上部のヘッダー行の後の各行 (行) は、単一のアカウント登録イベントからのデータを表します。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.get_event_prediction(
      detectorId = 'sample_detector',
      eventId = '802454d3-f7d8-482d-97e8-c4b6db9a0428',
      eventTypeName = 'sample_registration',
      eventTimestamp = '2020-07-13T23:18:21Z',
      entities = [{'entityType':'sample_customer', 'entityId':'12345'}],
 eventVariables = {
      'email_address': 'johndoe@exampledomain.com',
      'ip_address': '1.2.3.4'
}
)
```

このチュートリアルを完了したら、次の操作を実行しました。

- イベントデータセットの例を Amazon S3 にアップロードしました。
- モデルの作成とトレーニングに使用される変数、エンティティ、ラベルを作成しました。
- サンプルデータセットを使用してモデルを作成し、トレーニングしました。
- Amazon Fraud Detector が生成したモデルパフォーマンススコアおよびその他のパフォーマンスメトリクスを表示しました。
- 不正検出モデルをデプロイしました。
- ディテクターを作成し、デプロイされたモデルを追加しました。
- ディテクターにルール、ルールの実行順序、結果を追加しました。
- ディテクターバージョンを作成しました。
- 異なる入力を提供し、ルールとルールの実行順序が期待どおりに機能するかどうかを確認することで、ディテクターをテストしました。

## (オプション) Jupyter (iPython) ノートブックを使用した Amazon Fraud Detector API について詳しく知る

Amazon Fraud Detector APIsaws-fraud-detector-samples GitHub」を参照してください。ノートブックで取り上げるトピックには、Amazon Fraud Detector APIs を使用したモデルの構築とディテクター、および `GetEventPrediction` API を使用したバッチ不正予測リクエストの作成が含まれます。

## 次のステップ

モデルとディテクターを作成したので、より深く掘り下げて、モデルとディテクターの作成と不正予測の生成を開始できます。

「Amazon Fraud Detector ユーザーガイド」の以下のセクションでは、ビジネスまたは組織が Amazon Fraud Detector を使用して不正を検出する方法について説明します。

- モデルをトレーニングするためのイベントデータセットを準備して作成します。
- イベントタイプを作成する
- モデルを作成する
- ディテクターを作成する
- 不正予測の取得
- Amazon Fraud Detector リソース (具体的には、変数、エンティティ、結果、ラベル) を管理する
- セキュリティおよびコンプライアンスの目的を達成するように Amazon Fraud Detector を設定する
- Amazon Fraud Detector のモニタリングと Amazon Fraud Detector API コールのログ記録
- Amazon Fraud Detector に関する問題のトラブルシューティング

# イベントデータセット

イベントデータセットは、会社の過去の不正データです。このデータを Amazon Fraud Detector に提供して、不正検出モデルを作成します。

Amazon Fraud Detector は、機械学習モデルを使用して不正予測を生成します。各モデルは、モデルタイプを使用してトレーニングします。モデルタイプは、モデルのトレーニングに使用されるアルゴリズムと変換を指定します。モデルトレーニングとは、ユーザーが提供するデータセットを使用して、不正イベントを予測できるモデルを作成するプロセスです。詳細については、「Amazon Fraud Detector の仕組み」を参照してください。

不正検出モデルの作成に使用されるデータセットは、イベントの詳細を提供します。イベントとは、不正リスクについて評価の対象となるビジネス活動です。例えば、アカウント登録がイベントの例として挙げられます。アカウント登録イベントに関連付けられているデータは、イベントデータセットにすることができます。Amazon Fraud Detector は、このデータセットを使用してアカウント登録の不正行為を評価します。

モデルを作成するためにデータセットを Amazon Fraud Detector に提供する前に、モデルを作成するための目標を必ず定義してください。また、モデルの使用方法を決定し、特定の要件に基づいてモデルが実行されているかどうかを評価するためのメトリクスを定義する必要があります。

例えば、アカウント登録の不正を評価する不正検出モデルを作成する目標は、次のようになります。

- 正当な登録を自動承認すること。
- 後で調査するために不正な登録をキャプチャすること。

目標を決めたら、次のステップはモデルの使用方法を決定することです。不正検出モデルを使用して登録不正を評価する例をいくつか次に示します。

- 各アカウント登録をリアルタイムで不正検出する場合。
- すべてのアカウント登録を 1 時間ごとにオフラインで評価する場合。

モデルのパフォーマンスを測定するために使用できるメトリクスの例を次に示します。

- 本番環境において現在のベースラインよりも一貫して優れたパフォーマンスを発揮します。
- Y% 偽陽性率で X% の不正登録をキャプチャします。
- 不正である自動承認登録の最大 5% を受け入れます。

# イベントデータセット構造

Amazon Fraud Detector では、UTF-8 形式のカンマ区切り値 (CSV) を使用してイベントデータセットをテキストファイルに提供する必要があります。CSV データセットファイルの最初の行には、ファイルヘッダーが含まれている必要があります。ファイルヘッダーは、イベントメタデータと、イベントに関連付けられている各データ要素を記述するイベント変数で構成されています。ヘッダーにはイベントデータが続きます。各行は、1 つのイベントのデータ要素で構成されます。

- イベントメタデータ - イベントに関する情報を提供します。たとえば、EVENT_TIMESTAMP は、イベントが発生した時刻を指定するイベントメタデータです。ビジネスユースケースと不正検出モデルの作成とトレーニングに使用されるモデルタイプに応じて、Amazon Fraud Detector では特定のイベントメタデータを提供する必要があります。CSV ファイルヘッダーでイベントメタデータを指定する場合は、Amazon Fraud Detector で指定されたものと同じイベントメタデータ名を使用し、大文字のみを使用します。
- イベント変数 - 不正検出モデルの作成とトレーニングに使用するイベントに固有のデータ要素を表します。ビジネスユースケースと不正検出モデルの作成とトレーニングに使用されるモデルタイプによっては、Amazon Fraud Detector が特定のイベント変数の提供を要求するか、推奨する場合があります。オプションで、モデルのトレーニングに含めるイベントの他のイベント変数を指定することもできます。オンライン登録イベントのイベント変数の例としては、E メールアドレス、IP アドレス、電話番号などがあります。CSV ファイルヘッダーでイベント変数名を指定する場合は、任意の変数名を使用し、小文字のみを使用します。
- イベントデータ - 実際のイベントから収集されたデータを表します。CSV ファイルでは、ファイルヘッダーの後に続く各行は、1 つのイベントのデータ要素で構成されます。例えば、オンライン登録イベントデータファイルでは、各行に 1 つの登録のデータが含まれます。行内の各データ要素は、対応するイベントメタデータまたはイベント変数と一致する必要があります。

アカウント登録イベントのデータを含む CSV ファイルの例を次に示します。ヘッダー行には、大文字のイベントメタデータと、小文字のイベント変数、それに続くイベントデータの両方が含まれます。データセット内の各行には、1 つのアカウント登録に関連付けられたデータ要素が含まれ、各データ要素はヘッダーに対応しています。

# データモデルエクスプローラーを使用してイベントデータセットの要件を取得する

モデルの作成に選択したモデルタイプによって、データセットの要件が定義されます。Amazon Fraud Detector は、提供されたデータセットを使用して不正検出モデルを作成およびトレーニングします。Amazon Fraud Detector がモデルの作成を開始する前に、データセットがサイズ、形式、その他の要件を満たしているかどうかを確認します。データセットが要件を満たしていない場合、モデルの作成とトレーニングは失敗します。データモデルエクスプローラーを使用して、ビジネスユースケースに使用するモデルタイプを特定し、特定されたモデルタイプのデータセット要件に関するインサイトを得ることができます。

## データモデルエクスプローラー

データモデルエクスプローラーは、Amazon Fraud Detector コンソールのツールで、ビジネスユースケースを Amazon Fraud Detector がサポートするモデルタイプに合わせます。データモデルエクスプローラーは、Amazon Fraud Detector が不正検出モデルを作成するために必要なデータ要素に関するインサイトも提供します。イベントデータセットの準備を開始する前に、データモデルエクスプローラーを使用して、Amazon Fraud Detector がビジネスでの使用に推奨するモデルタイプを把握し、データセットの作成に必要な必須、推奨、およびオプションのデータ要素のリストを表示します。

データモデルエクスプローラーを使用するには、

1. AWS マネジメントコンソールを開き、アカウントにサインインします。Amazon Fraud Detector に移動します。
2. 左側のナビゲーションペインで、データモデルエクスプローラーを選択します。
3. データモデルエクスプローラーページの「ビジネスユースケース」で、不正リスクを評価するビジネスユースケースを選択します。
4. Amazon Fraud Detector には、ビジネスユースケースに一致する推奨モデルタイプが表示されます。モデルタイプは、Amazon Fraud Detector が不正検出モデルのトレーニングに使用するアルゴリズム、エンリッチメント、および変換を定義します。

   推奨されるモデルタイプを書き留めます。これは、後でモデルを作成するときに必要になります。

> **Note**
>
> ビジネスユースケースが見つからない場合は、説明の reach us リンクを使用して、ビジネスユースケースの詳細を入力します。ビジネスユースケースの不正検出モデルを作成するために使用するモデルタイプをお勧めします。

5. データモデルインサイトペインには、ビジネスユースケースの不正検出モデルの作成とトレーニングに必要な必須、推奨、およびオプションのデータ要素に関するインサイトが表示されます。インサイトペインの情報を使用して、イベントデータを収集し、データセットを作成します。

# イベントデータの収集

イベントデータを収集することは、モデルを作成する上で重要なステップです。これは、不正予測におけるモデルのパフォーマンスが、データセットの品質に依存するためです。イベントデータの収集を開始するときは、データモデルエクスプローラーがデータセットを作成するために提供したデータ要素のリストに注意してください。必須データ (イベントメタデータ) をすべて収集し、モデル作成の目標に基づいて、含める推奨データ要素とオプションのデータ要素 (イベント変数) を決定する必要があります。また、含めるイベント変数の形式とデータセットの合計サイズを決定することも重要です。

イベントデータセットの品質

モデルの高品質データセットを収集するには、以下をお勧めします。

- 成熟したデータを収集する - 最新のデータを使用すると、最新の不正パターンを特定するのに役立ちます。ただし、不正ユースケースを検出するには、データを成熟させます。成熟期間はビジネスによって異なり、2 週間から 3 か月かかる場合もあります。例えば、イベントにクレジットカード取引が含まれる場合、データの満期は、クレジットカードのチャージバック期間、または調査者が決定するのに要した時間によって決まる場合があります。

  モデルのトレーニングに使用されるデータセットが、ビジネスに合わせて成熟するのに十分な時間があることを確認します。
- データ分布が著しくドリフトしないようにする - Amazon Fraud Detector モデルトレーニングプロセスは、EVENT_TIMESTAMP に基づいてデータセットのサンプル作成とパーティショニングに行います。例えば、データセットが過去 6 か月から引き出された不正イベントで構成され、最後の月の正当なイベントのみが含まれる場合、データ分布はドリフトして不安定になると考えられます。不安定なデータセットは、モデルのパフォーマンス評価でバイアスを引き起こす可能性があ

ります。データ分布が大幅にドリフトしていることがわかった場合は、現在のデータ分布と同様のデータを収集してデータセットのバランスをとることを検討してください。

- データセットがモデルを実装/テストするユースケースを代表するものであることを確認します - そうしないと、推定されるパフォーマンスに偏りが生じる可能性があります。モデルを使用してすべての社内申請者を自動的に拒否しているが、モデルは以前に承認された履歴データ/ラベルを含むデータセットを使用してトレーニングされているとします。その場合、評価は却下された申請者の表現を持たないデータセットに基づいているため、モデルの評価が不正確になる可能性があります。

## イベントデータ形式

Amazon Fraud Detector は、モデルトレーニングプロセスの一環として、ほとんどのデータを必要な形式に変換します。ただし、Amazon Fraud Detector がデータセットを検証する際に問題を回避するのに役立つデータを提供するために簡単に使用できる標準形式がいくつかあります。次の表は、推奨されるイベントメタデータを提供するための形式に関するガイダンスです。

> **Note**
>
> CSV ファイルを作成するときは、以下に示すイベントメタデータ名を大文字で入力してください。

| メタデータ名 | 形式 | 必須 |
| --- | --- | --- |
| EVENT_ID | 指定する場合は、次の要件を満たしている必要があります。<br>• そのイベントはユニークである。<br>• ビジネスにとって有意義な情報を表している。<br>• 正規表現パターンに従っている (例えば、`^[0-9a-z_-]+$.`) | モデルのタイプによる |

| メタデータ名 | 形式 | 必須 |
| --- | --- | --- |
| | • 上記の要件に加えて、EVENT_ID にタイムスタンプを追加しないことをお勧めします。追加すると、イベントを更新するときに問題が発生する可能性があります。これは、追加する場合、まったく同じ EVENT_ID を指定する必要があるためです。 | |

| メタデータ名 | 形式 | 必須 |
| --- | --- | --- |
| EVENT_TIMESTAMP | • 次のいずれかの形式で有効な値を指定する必要があります。<br>• %yyyy-%mm-%ddT%hh:%mm:%ssZ (ミリ秒なし、UTC のみの ISO 8601標準)<br>例: 2019-11-30T13:01:01Z<br>• %yyyy/%mm/%dd %hh:%mm:%ss (AM/PM)<br>例: 2019/11/30 1:01:01 PM、または 2019/11/30 13:01:01<br>• %mm/%dd/%yyyy %hh:%mm:%ss<br>例: 11/30/2019 1:01:01 PM、または 11/30/2019 13:01:01<br>• %mm/%dd/%yy %hh:%mm:%ss<br>例: 11/30/19 1:01:01 PM、または 11/30/19 13:01:01<br>• Amazon Fraud Detector は、イベントタイムスタンプの日付/タイムスタンプ形式を解析するときに、次の仮定を行います。<br>• ISO 8601 標準を使用する場合は、前述の仕様と完 | はい |

| メタデータ名 | 形式 | 必須 |
| --- | --- | --- |
| | 全に一致する必要があります。<br>• 他の形式のいずれかを使用している場合は、さらに柔軟性があります。<br>  • 月および日には、1 桁または 2 桁の数字を指定できます。例えば、2019 年 1 月 12 日は有効な日付です。<br>  • hh:mm:ss を持っていない場合は、含める必要はありません (つまり、日付を指定するだけです)。時と分だけのサブセット (例えば、hh:mm) を指定することもできます。時のみの指定はサポートされていません。ミリ秒もサポートされていません。<br>  • AM/PM ラベルを指定した場合は、12 時間時計と見なされます。AM/PM 情報がない場合は、24 時間時計と見なされます。<br>  • 日付要素の区切り文字として「/」または「-」を使用できます。タイムスタンプ要素には「:」が想定されます。 | |

| メタデータ名 | 形式 | 必須 |
| --- | --- | --- |
| ENTITY_ID | • 正規表現のパターンを満たす必要があります: `^[0-9A-Za-z_.@+-]+$`<br>• 評価時にエンティティ ID が使用できない場合は、エンティティ ID を unknown として指定します。 | モデルのタイプによる |
| ENTITY_TYPE | 任意の文字列を使用できます。 | モデルのタイプによる |
| EVENT_LABEL | 「fraud」、「legit」、「1」、「0」など、任意のラベルを使用できます。 | LABEL_TIMESTAMP が含まれている場合は必須です |
| LABEL_TIMESTAMP | タイムスタンプ形式に従う必要があります。 | EVENT_LABEL が含まれている場合は必須です |

イベント変数の詳細については、「変数」を参照してください。

> **Important**
>
> Account Takeover Insights (ATI) モデルを作成する場合は、「」でデータの準備と選択の詳細データの準備を参照してください。

NULL または欠損値

EVENT_TIMESTAMP および EVENT_LABEL 変数には、NULL または欠損値を含めることはできません。他の変数には NULL または欠損値を指定できます。ただし、これらの変数には少数の NULL のみを使用することをお勧めします。Amazon Fraud Detector は、イベント変数の NULL または欠損値が多すぎると判断した場合、モデルから変数を自動的に省略します。

最小変数

モデルを作成する場合、データセットには、必要なイベントメタデータに加えて、少なくとも 2 つのイベント変数を含める必要があります。2 つのイベント変数は、検証チェックに合格する必要があります。

イベントデータセットのサイズ

必須

モデルトレーニングを成功させるには、データセットが次の基本要件を満たしている必要があります。

- 少なくとも 100 個のイベントからのデータ。
- データセットには、不正として分類されたイベント (行) が少なくとも 50 個含まれている必要があります。

推奨

モデルトレーニングを成功させ、モデルのパフォーマンスを向上させるには、データセットに以下を含めることをお勧めします。

- 最低 3 週間の履歴データを含めますが、最大 6 か月分のデータを含めます。
- 合計 10K 件以上のイベントデータを含めます。
- 不正と分類されるイベント (行) を少なくとも 400 件、正当と分類されるイベント (行) を少なくとも 400 件含めます。
- モデルタイプに ENTITY_ID が必要な場合は、100 を超える一意のエンティティを含めます。

# データセットの検証

Amazon Fraud Detector は、モデルの作成を開始する前に、モデルのトレーニングのためにデータセットに含まれる変数がサイズ、形式、およびその他の要件を満たしているかどうかをチェックします。データセットが検証に合格しない場合、モデルは作成されません。モデルを作成する前に、まず検証に合格しなかった変数を修正する必要があります。Amazon Fraud Detector は、モデルのトレーニングを開始する前に、データセットの問題を特定して修正するために使用できるデータプロファイラーを提供します。

データプロファイラー

Amazon Fraud Detector は、モデルトレーニングのためにデータをプロファイリングおよび準備するためのオープンソースツールを提供します。この自動データプロファイラーは、一般的なデータ準備エラーを回避し、モデルのパフォーマンスに悪影響を与える可能性のある変数タイプがマップされていないかなど、潜在的な問題を特定するのに役立ちます。プロファイラーは、変数統計、ラベル分布、カテゴリ分析、数値分析、変数とラベルの相関など、データセットの直感的で包括的なレポートを生成します。変数タイプに関するガイダンスと、データセットを Amazon Fraud Detector が必要とする形式に変換するオプションを提供します。

データプロファイラーの使用

自動データプロファイラーは AWS CloudFormation スタックで構築されており、数回クリックするだけで簡単に起動できます。すべてのコードは GitHub で利用できます。データプロファイラーの使用方法については、「Amazon Fraud Detector の自動データプロファイラーでモデルを迅速にトレーニングする」のブログ記事の指示に従ってください。

イベントデータセットの一般的なエラー

イベントデータセットの検証時に Amazon Fraud Detector で発生する一般的な問題のいくつかを次に示します。データプロファイラーを実行した後、モデルを作成する前に、このリストを使用してデータセットのエラーをチェックします。

- CSV ファイルは UTF-8 形式ではない。
- データセット内のイベント数が 100 未満です。
- 不正または正当として識別されるイベントの数は 50 未満です。
- 不正とされる一意のエンティティの数が 100 未満である。
- EVENT_TIMESTAMP の値の 0.1% 以上には、NULL、またはサポートされている日付/タイムスタンプ形式以外の値が含まれている。
- EVENT_LABEL の値の 1% 以上に、NULL、イベントタイプで定義されている値以外の値が含まれている。
- モデルトレーニングに使用できる変数が 2 つ未満である。

# データセットストレージ

データセットを収集したら、データセットを Amazon Fraud Detector を使用して内部に保存するか、Amazon Simple Storage Service (Amazon S3) を使用して外部に保存します。不正予測の生成に使用するモデルに基づいて、データセットの保存場所を選択することをお勧めします。モデルタイプ

の詳細については、「モデルタイプを選択する」を参照してください。データセットの保存の詳細については、「」を参照してくださいイベントデータストレージ。

# イベントタイプ

Amazon Fraud Detector を使用すると、イベントの不正予測を生成できます。イベントタイプは、Amazon Fraud Detector に送信される個々のイベントの構造を定義します。定義したら、特定のイベントタイプのリスクを評価するモデルとディテクターを構築できます。

イベントの構造には、次のものが含まれます。

- エンティティタイプ: イベントを実行しているユーザーを指定します。予測時に、エンティティタイプとエンティティ ID を指定して、イベントを実行したユーザーを定義します。
- 変数: イベントの一部として送信できる変数を定義します。変数は、不正リスクを評価するためにモデルとルールで使用されます。追加すると、変数をイベントタイプから削除することはできません。
- ラベル: イベントを不正または正当として分類します。モデルトレーニング中に使用されます。追加すると、ラベルをイベントタイプから削除することはできません。

## イベントタイプを作成する

不正検出モデルを作成する前に、まずイベントタイプを作成する必要があります。イベントタイプを作成するには、ビジネスアクティビティ (イベント) を定義して不正について評価する必要があります。イベントを定義するには、不正評価に含めるデータセット内のイベント変数を識別し、イベントを開始するエンティティと、イベントを分類するラベルを指定します。

イベントタイプを作成するための前提条件

イベントタイプの作成を開始する前に、以下が完了していることを確認してください。

- データモデルエクスプローラー ツールを使用して、Amazon Fraud Detector が不正検出モデルを作成するために必要なデータ要素に関するインサイトを取得しました。
- Data Models Explorer から取得したインサイトを使用してイベントデータセットを作成し、データセットを Amazon S3 バケットにアップロードしました。
- 変数、、および を作成しエンティティ、このイベントの不正検出モデルを作成するために Amazon Fraud Detector が使用ラベルできるようにします。作成した変数、エンティティタイプ、ラベルがイベントデータセットに含まれていることを確認します。

イベントタイプは、Amazon Fraud Detector コンソール、 API、、 AWS CLIまたは AWS SDK を使用して作成できます。

## Amazon Fraud Detector コンソールでイベントタイプを作成する

イベントタイプを作成するには、

1. AWS マネジメントコンソールを開き、アカウントにサインインします。Amazon Fraud Detector に移動します。
2. 左側のナビゲーションペインで [イベント] を選択します。
3. イベントタイプページで、作成を選択します。
4. イベントタイプの詳細で、
   a. 名前に、イベントの名前を入力します。
   b. 説明 で、オプションで説明を入力します。
   c. エンティティで、イベント用に作成したエンティティタイプを選択します。
5. イベント変数で、
   - 「このイベントの変数を定義する方法の選択」で、
     - このイベントのイベント変数を既に作成している場合は、変数リストから変数の選択を選択し、変数でこのイベント用に作成した変数を選択します。
     - このイベントの変数を作成していない場合は、トレーニングデータセットから変数を選択を選択し、
       - IAM ロールで、データセットを含む Amazon S3 バケットにアクセスするために Amazon Fraud Detector が使用する IAM ロールを選択します。
       - データロケーションに、データセットロケーションへのパスを入力します。次のようなS3 URIパスを使用します: `S3://your-bucket-name/example dataset filename.csv`。
       - [アップロード] を選択します。
       - 変数 の下に、Amazon Fraud Detector がデータセットファイルから抽出したすべてのイベント変数名が表示されます。

         不正を検出するために変数を含める場合は、変数タイプで変数タイプを選択します。削除を選択して、不正検出の対象から変数を削除します。リスト内の変数ごとにこのステップを繰り返します。

6. ラベル (オプション ) のラベルで、このイベント用に作成したラベルを選択します。不正および正当なイベントには、必ず各ラベルを 1 つ選択します。
7. このイベントの自動ダウンストリーム処理を設定する場合は、「Amazon EventBridge によるイベントオーケストレーション - オプション」で、「Amazon EventBridge によるイベントオーケストレーションを有効にする」をオンにします。イベントオーケストレーションの詳細については、「」を参照してくださいイベントのオーケストレーション。

> **Note**
>
> イベントタイプを作成した後で、イベントオーケストレーションを有効にすることもできます。

8. [イベントタイプの作成] を選択します。

## を使用してイベントタイプを作成する AWS SDK for Python (Boto3)

次の例は、PutEventType API のサンプルリクエストを示しています。この例では、変数 ip_address と email_address、ラベル legit と fraud、およびエンティティタイプ sample_customer を作成したと想定しています。これらのリソースの作成方法については、「リソース」を参照してください。

> **Note**
>
> 変数、エンティティタイプ、およびラベルを作成してから、イベントタイプに追加する必要があります。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.put_event_type (
name = 'sample_registration',
eventVariables = ['ip_address', 'email_address'],
labels = ['legit', 'fraud'],
entityTypes = ['sample_customer'])
```

# イベントまたはイベントタイプの削除

イベントを削除すると、Amazon Fraud Detector はそのイベントを完全に削除し、そのイベントに関連付けられたデータは Amazon Fraud Detector に保存されなくなります。

Amazon Fraud Detector が **`GetEventPrediction`** API を通じて評価したイベントを削除するには

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/frauddetector で Amazon Fraud Detector コンソールを開きます。
2. コンソールの左側のナビゲーションペインで、[過去の予測を検索] を選択します。
3. 削除するイベントを選択します。
4. [アクション] を選択してから、[イベントの削除] をクリックします。
5. 「**`delete`**」と入力してから、[イベントの削除] を選択します。

> **Note**
>
> これにより、そのイベント ID に関連付けられているすべてのレコードが削除されます。これには、`SendEvent` オペレーションに送信されたイベントデータや、`GetEventPrediction` オペレーションを通じて生成された予測データが含まれます。

Amazon Fraud Detector に保存されているが評価されていない (つまり、`SendEvent` オペレーションによって保存された) イベントを削除するには、`DeleteEvent` リクエストを行い、イベント ID とイベントタイプ ID を指定する必要があります。イベントとそのイベントに関連付けられた予測履歴の両方を削除する場合は、`deleteAuditHistory` パラメータの値を「true」に設定します。`deleteAuditHistory` パラメータを「true」に設定すると、削除オペレーションが完了してから最大 30 秒間、検索を通じてイベントデータを使用できます。

イベントタイプに関連付けられているすべてのイベントを削除するには

1. コンソールの左側のナビゲーションペインで、[イベントタイプ] を選択します。
2. すべてのイベントを削除するイベントタイプを選択します。
3. [ストアドイベント] タブに移動し、[ストアドイベントの削除] を選択します

イベントタイプの保存されたイベントの数によっては、保存されたすべてのイベントの削除に時間がかかる場合があります。例えば、1 GB のデータセット (平均的なお客様においては約 1～2 万件

のイベント) の削除には約 2 時間かかります。この期間中、このイベントタイプの Amazon Fraud Detector に送信した新しいイベントは保存されませんが、`GetEventPrediction` オペレーションを通じて引き続き不正予測を生成できます。

イベントタイプを削除するには

ディテクターで使用されているか、関連付けられたストアドイベントを含むイベントタイプは削除できません。イベントタイプを削除する前に、そのイベントタイプに関連付けられているすべてのイベントを削除する必要があります。

イベントタイプを削除すると、Amazon Fraud Detector はそのイベントタイプを完全に削除し、データは Amazon Fraud Detector に保存されなくなります。

1. Amazon Fraud Detector コンソールの左側のナビゲーションペインで、[リソース] を選択してから、[イベント] をクリックします。
2. 削除するイベントタイプを選択します。
3. [アクション] を選択してから、[イベントタイプの削除] をクリックします。
4. イベントタイプ名を入力してから、[イベントタイプの削除] を選択します。

# イベントデータストレージ

データセットを収集したら、Amazon Fraud Detector を使用して内部に、または Amazon Simple Storage Service (Amazon S3) を使用して外部にデータセットを保存します。不正予測の生成に使用するモデルに基づいて、データセットの保存場所を選択することをお勧めします。次に、これら 2 つのストレージオプションの詳細な内訳を示します。

- 内部ストレージ - データセットは Amazon Fraud Detector を使って保存されます。イベントに関連付けられているすべてのイベントデータは、一緒に保存されます。Amazon Fraud Detector に保存されているイベントデータセットは、いつでもアップロードできます。Amazon Fraud Detector API にイベントを 1 つずつストリーミングするか、バッチインポート機能を使用して大きなデータセット (最大 1 GB) をインポートできます。Amazon Fraud Detector に保存されたデータセットを使用してモデルをトレーニングする場合、時間範囲を指定してデータセットのサイズを制限できます。
- 外部ストレージ - データセットは、Amazon Fraud Detector 以外の外部データソースに保存されます。現在、Amazon Fraud Detector では、この目的のために Amazon Simple Storage Service (Amazon S3) の使用をサポートしています。モデルが Amazon S3 にアップロードされたファイルにある場合、そのファイルの非圧縮データは 5 GB を超えることはできません。それ以上の場合は、データセットの時間範囲を短くしてください。

次の表は、サポートされているモデルタイプとデータソースの詳細を示しています。

| モデルタイプ | 互換性のあるトレーニングデータソース |
| --- | --- |
| オンライン不正インサイト | 外部ストレージと内部ストレージです。 |
| トランザクション不正インサイト | 内部ストレージ |
| アカウント乗っ取りインサイト | 内部ストレージ |

Amazon Simple Storage Service でデータセットを外部に保存する方法の詳細については、「」を参照してくださいAmazon S3 を使用してイベントデータを外部に保存する 。Amazon Fraud Detector でデータセットを内部的に保存する方法については、「」を参照してくださいAmazon Fraud Detector を使用してイベントデータを内部に保存する。

# Amazon S3 を使用してイベントデータを外部に保存する

オンライン不正インサイトモデルをトレーニングしている場合は、Amazon S3 でイベントデータを外部に保存することができます。Amazon S3 にイベントデータを保存するには、まず CSV 形式のテキストファイルを作成し、イベントデータを追加してから、CSV ファイルを Amazon S3 バケットにアップロードする必要があります。

> **Note**
>
> Transaction Fraud Insights モデルタイプと Account Takeover Insights モデルタイプは、Amazon S3 で外部に保存されているデータセットをサポートしていません

## CSV ファイルを作成する

Amazon Fraud Detector では、CSV ファイルの最初の行に列ヘッダーが含まれている必要があります。CSV ファイルの列ヘッダーは、イベントタイプで定義されている変数に対応している必要があります。データセットの例については、「サンプルデータセットを取得してアップロードする」を参照してください。

オンライン不正インサイトモデルには、少なくとも 2 つの変数と最大 100 個の変数を持つトレーニングデータセットが必要です。イベント変数に加えて、トレーニングデータセットには、次のヘッダーが含まれている必要があります。

- EVENT_TIMESTAMP — いつイベントが発生したかを定義します
- EVENT_LABEL - イベントを不正または正当として分類します 列の値は、イベントタイプで定義されている値に対応している必要があります。

次のサンプル CSV データは、オンラインマーチャントからの登録イベントの履歴を表します。

```
EVENT_TIMESTAMP,EVENT_LABEL,ip_address,email_address
4/10/2019 11:05,fraud,209.146.137.48,fake_burtonlinda@example.net
12/20/2018 20:04,legit,203.0.112.189,fake_davidbutler@example.org
3/14/2019 10:56,legit,169.255.33.54,fake_shelby76@example.net
1/3/2019 8:38,legit,192.119.44.26,fake_curtis40@example.com
9/25/2019 3:12,legit,192.169.85.29,fake_rmiranda@example.org
```

> **Note**
>
> CSV データファイルには、データの一部として二重引用符とカンマを含めることができます。

対応するイベントタイプの簡略版を以下に示します。イベント変数は CSV ファイルのヘッダーに対応し、EVENT_LABEL の値はラベルリストの値に対応します。

```
(
name = 'sample_registration',
eventVariables = ['ip_address', 'email_address'],
labels = ['legit', 'fraud'],
entityTypes = ['sample_customer']
)
```

## イベントのタイムスタンプ形式

イベントのタイムスタンプが必須の形式であることを確認します。モデル構築プロセスの一環として、Online Fraud Insights モデルタイプは、イベントのタイムスタンプに基づいてデータを順序付けし、トレーニングとテストの目的でデータを分割します。パフォーマンスを公平に見積もるために、モデルはまずトレーニングデータセットでトレーニングを行い、次にテストデータセットでこのモデルをテストします。

Amazon Fraud Detector は、モデルトレーニング中、EVENT_TIMESTAMP の値に対して次の日付/タイムスタンプ形式をサポートしています。

- %yyyy-%mm-%ddT%hh:%mm:%ssZ (ミリ秒なし、UTC のみの ISO 8601標準)

  例: 2019-11-30T13:01:01Z
- %yyyy/%mm/%dd %hh:%mm:%ss (AM/PM)

  例: 2019/11/30 1:01:01 PM、または 2019/11/30 13:01:01
- %mm/%dd/%yyyy %hh:%mm:%ss

  例: 11/30/2019 1:01:01 PM、または 11/30/2019 13:01:01
- %mm/%dd/%yy %hh:%mm:%ss

  例: 11/30/19 1:01:01 PM、または 11/30/19 13:01:01

Amazon Fraud Detector は、イベントタイムスタンプの日付/タイムスタンプ形式を解析するときに、次の仮定を行います。

- ISO 8601 標準を使用する場合は、前述の仕様と完全に一致する必要があります。
- 他の形式のいずれかを使用している場合は、さらに柔軟性があります。
  - 月および日には、1 桁または 2 桁の数字を指定できます。例えば、2019 年 1 月 12 日は有効な日付です。
  - hh:mm:ss がない (つまり、単に日付を指定できる) 場合は、含める必要はありません。時と分だけのサブセット (例えば、hh:mm) を指定することもできます。時のみの指定はサポートされていません。ミリ秒もサポートされていません。
  - AM/PM ラベルを指定した場合は、12 時間時計と見なされます。AM/PM 情報がない場合は、24 時間時計と見なされます。
  - 日付要素の区切り文字として「/」または「-」を使用できます。タイムスタンプ要素には「:」が想定されます。

## 経時的なデータセットのサンプリング

不正のサンプルと正当なサンプルを同じ時間範囲で提供することをお勧めします。例えば、過去 6 か月間の不正イベントを提供する場合は、同じ期間に均等にまたがる正当なイベントも提供する必要があります。データセットに不正および正当なイベントの分布が非常に不均一に含まれている場合は、次のエラーが表示される場合があります。「時間の経過に伴う不正分布は容認できないほど変動しています。データセットを正しく分割できません」 通常、このエラーに対する最も簡単な修正は、不正イベントと正当なイベントが同じ期間にわたって均等にサンプリングされるようにすることです。また、短期間で不正の急増が発生した場合は、データの削除が必要になる場合があります。

均等に分散されたデータセットを作成するのに十分なデータを生成できない場合は、イベントの EVENT_TIMESTAMP をランダム化して、均等に分散されるようにする方法があります。ただし、Amazon Fraud Detector は EVENT_TIMESTAMP を使用して、データセット内の適切なイベントのサブセットでモデルを評価するため、多くの場合、パフォーマンスメトリクスが非現実的になります。

## NULL 値および欠損値

Amazon Fraud Detector は NULL 値および欠損値を処理します。ただし、変数の NULL の割合は制限する必要があります。EVENT_TIMESTAMP および EVENT_LABEL 列に欠損値を含めることはできません。

### ファイルの検証

Amazon Fraud Detector は、次のいずれかの条件が発生すると、モデルをトレーニングできません。

- CSV を解析できない場合
- 列のデータ型が間違っている場合

## イベントデータを Amazon S3 バケットにアップロードする

イベントデータで CSV ファイルを作成したら、このファイルを Amazon S3 バケットにアップロードします。

Amazon S3 バケットにアップロードするには

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/s3/ で Amazon S3 コンソールを開きます。
2. バケットの作成 を選択します。

   [バケットの作成] ウィザードが開きます。
3. [バケット名] に、バケットの DNS に準拠する名前を入力します。

   バケット名には次の条件があります。

   - すべての Amazon S3 で一意にする。
   - 3～63 文字で指定する。
   - 大文字を含めないでください。
   - 先頭の文字には小文字の英文字または数字を使用する。

   バケットを作成したら、その名前を変更することはできません。バケットの名前付けの詳細については、Amazon Simple Storage Service ユーザーガイドの「バケットの名前付けルール」を参照してください。

   > **Important**
   >
   > バケット名にアカウント番号などの機密情報を含めないでください。バケット名は、バケット内のオブジェクトを参照する URL に表示されます。

4. リージョンで、バケットを配置する AWS リージョンを選択します。Amazon Fraud Detector を使用しているリージョンと同じリージョンを選択する必要があります。これは、米国東部 (バージニア北部)、米国東部 (オハイオ)、米国西部 (オレゴン)、欧州 (アイルランド)、アジアパシフィック (シンガポール)、またはアジアパシフィック (シドニー) のいずれかです。
5. [バケットのパブリックアクセスブロック設定] で、バケットに適用するパブリックアクセスブロック設定を選択します。

   設定はすべて有効のままにしておくことをお勧めします。パブリックアクセスのブロックの詳細については、Amazon Simple Storage Service ユーザーガイドの「Amazon S3 ストレージへのパブリックアクセスのブロック」を参照してください。
6. [バケットを作成] を選択します。
7. トレーニングデータファイルを Amazon S3 バケットにアップロードします。トレーニングファイルの Amazon S3 の場所 (例: s3://bucketname/object.csv) を書き留めます。

# Amazon Fraud Detector を使用してイベントデータを内部に保存する

イベントデータを Amazon Fraud Detector に保存し、保存されたデータを後でモデルのトレーニングに使用できます。Amazon Fraud Detector にイベントデータを保存することで、自動計算変数を使用してパフォーマンスを改善し、モデルの再トレーニングを簡素化し、不正ラベルを更新して機械学習のフィードバックループを閉じるモデルをトレーニングできます。イベントはイベントタイプリソースレベルで保存されるため、同じイベントタイプのすべてのイベントが 1 つのイベントタイプデータセットにまとめて保存されます。イベントタイプの定義の一環として、Amazon Fraud Detector コンソールの [Event Ingestion] (イベントの取り込み) 設定を切り替えることで、そのイベントタイプのイベントを保存するかどうかを必要に応じて指定できます

Amazon Fraud Detector では、1 つのイベントを保存することも、多数のイベントデータセットをインポートすることもできます。単一のイベントは、GetEventPrediction API または SendEvent API を使用してストリーミングできます。大規模なデータセットは、Amazon Fraud Detector コンソールのバッチインポート機能または CreateBatchImportJob API を使用して、Amazon Fraud Detector にすばやく簡単にインポートできます。

Amazon Fraud Detector コンソールを使用して、イベントタイプごとに既に保存されているイベントの数をいつでも確認できます。

# ストレージ用のイベントデータを準備する

Amazon Fraud Detector の内部に保存されるイベントデータは、Event Typeリソースレベルで保存されます。したがって、同じイベントからのすべてのイベントデータは 1 つの に保存されますEvent Type。保存されたイベントは、後で新しいモデルをトレーニングしたり、既存のモデルを再トレーニングしたりするために使用できます。保存されたイベントデータを使用してモデルをトレーニングする場合、必要に応じてイベントの時間範囲を指定して、トレーニングデータセットのサイズを制限できます。

Amazon Fraud Detector コンソール、SendEventAPI、または CreateBatchImportJob API を使用して Amazon Fraud Detector にデータを保存するたびに、Amazon Fraud Detector は保存前にデータを検証します。データが検証に失敗した場合、イベントデータは保存されません。

Amazon Fraud Detector でデータを内部的に保存するための前提条件

- イベントデータが検証に合格し、データセットが正常に保存されるようにするには、データモデルエクスプローラーが提供するインサイトを使用してデータセットを準備していることを確認してください。
- Amazon Fraud Detector に保存するイベントデータのイベントタイプを作成しました。まだ作成していない場合は、 の手順に従ってイベントタイプを作成します。

## スマートデータ検証

バッチインポートのために Amazon Fraud Detector コンソールにデータセットをアップロードすると、Amazon Fraud Detector はスマートデータ検証 (SDV) を使用してデータセットを検証してから、データをインポートします。SDV は、アップロードされたデータファイルをスキャンして、欠落データ、誤った形式やデータ型などの問題を特定します。SDV は、データセットの検証に加えて、特定されたすべての問題を一覧表示し、最も影響の大きい問題を修正するためのアクションを提案する検証レポートも提供します。SDV で特定された問題の一部は重要であり、Amazon Fraud Detector がデータセットを正常にインポートする前に対処する必要があります。詳細については、「スマートデータ検証レポート」を参照してください。

SDV は、ファイルレベルとデータ (行) レベルでデータセットを検証します。ファイルレベルでは、SDV はデータファイルをスキャンし、ファイルにアクセスするための不適切なアクセス許可、誤ったファイルサイズ、ファイル形式、ヘッダー (イベントメタデータとイベント変数) などの問題を特定します。データレベルでは、SDV は各イベントデータ (行) をスキャンし、誤ったデータ形式、データの長さ、タイムスタンプ形式、null 値などの問題を特定します。

スマートデータ検証は現在 Amazon Fraud Detector コンソールでのみ使用でき、検証はデフォルトでオンになっています。データセットをインポートする前に Amazon Fraud Detector でスマートデータ検証を使用しない場合は、データセットのアップロード時に Amazon Fraud Detector コンソールで検証をオフにします。

### APIs または AWS SDK の使用時に保存されたデータを検証する

`SendEvent`、`GetEventPrediction`、または `CreateBatchImportJob` API オペレーションを介してイベントをアップロードする場合、Amazon Fraud Detector は以下を検証します。

- そのイベントタイプの EventIngestion 設定が ENABLED であること。
- イベントのタイムスタンプは更新できません。繰り返しイベント ID および異なる EVENT_TIMESTAMP を持つイベントは、エラーとして扱われます。
- 変数の名前と値は、期待される形式と一致します。詳細については、「変数の作成」を参照してください。
- 必須変数には値が入力されます。
- すべてのイベントタイムスタンプは 18 か月を超えず、今後もそうなることはありません。

## バッチインポートを使用してイベントデータを保存する

バッチインポート機能を使用すると、コンソール、API、または AWS SDK を使用して、大規模な履歴イベントデータセットを Amazon Fraud Detector にすばやく簡単にアップロードできます。バッチインポートを使用するには、すべてのイベントデータを含む CSV 形式の入力ファイルを作成し、CSV ファイルを Amazon S3 バケットにアップロードして、インポートジョブを開始します。Amazon Fraud Detector は、まずイベントタイプに基づいてデータを検証し、データセット全体を自動的にインポートします。データがインポートされると、新しいモデルのトレーニングや既存のモデルの再トレーニングに使用する準備が整います。

### 入力ファイルと出力ファイル

入力 CSV ファイルには、関連するイベントタイプで定義されている変数と 4 つの必須変数と一致するヘッダーが含まれている必要があります。詳細については「ストレージ用のイベントデータを準備する」を参照してください。入力データファイルの最大サイズは 20 GB (20 GB)、つまり約 5,000 万イベントです。イベントの数は、イベントのサイズによって異なります。インポートジョブが成功した場合、出力ファイルは空になります。インポートが失敗した場合、出力ファイルにエラーログが含まれます。

## CSV ファイルの作成

Amazon Fraud Detector は、カンマ区切り値 (CSV) 形式のファイルからのみデータをインポートします。CSV ファイルの最初の行には、関連するイベントタイプで定義された変数と、EVENT_ID、EVENT_TIMESTAMP、ENTITY_ID、および ENTITY_TYPE という 4 つの必須変数と完全に一致する列ヘッダーが含まれている必要があります。必要に応じて EVENT_LABEL と LABEL_TIMESTAMP を含めることもできます (EVENT_LABEL が含まれている場合は LABEL_TIMESTAMP が必要です)。

必須変数の定義

必須変数はイベントのメタデータと見なされ、大文字で指定する必要があります。イベントメタデータは、モデルトレーニングに自動的に含まれます。次の表に、必須変数、各変数の説明、および変数に必要な形式を示します。

| 名前 | 説明 | 要件 |
| --- | --- | --- |
| EVENT_ID | イベントの識別子。例えば、イベントがオンライントランザクションの場合、EVENT_ID は顧客に提供されたトランザクション参照番号になります。 | • EVENT_ID は、バッチインポートジョブに必要です。<br>• そのイベントで一意である必要があります<br>• ビジネスにとって有意義な情報を表す必要があります。<br>• 正規表現パターンに従う必要があります (例えば、`^[0-9a-z_-]+$.`)<br>• EVENT_ID にタイムスタンプを追加することはお勧めしません。追加すると、イベントを更新するときに問題が発生する可能性があります。これは、追加する場合、まったく同じ EVENT_ID を指定する必要があるためです。 |

| 名前 | 説明 | 要件 |
| --- | --- | --- |
| EVENT_TIMESTAMP | イベントが発生したときのタイムスタンプ。タイムスタンプは UTC の ISO 8601 標準である必要があります。 | • EVENT_ID は、バッチインポートジョブに必須です。<br>• 次のいずれかの形式で有効な値を指定する必要があります。<br>  • %yyyy-%mm-%ddT%hh:%mm:%ssZ (ミリ秒なし、UTC のみの ISO 8601標準)<br><br>  例: 2019-11-30T13:01:01Z<br>  • %yyyy/%mm/%dd %hh:%mm:%ss (AM/PM)<br><br>  例: 2019/11/30 1:01:01 PM、または 2019/11/30 13:01:01<br>  • %mm/%dd/%yyyy %hh:%mm:%ss<br><br>  例: 11/30/2019 1:01:01 PM、または 11/30/2019 13:01:01<br>  • %mm/%dd/%yy %hh:%mm:%ss<br><br>  例: 11/30/19 1:01:01 PM、または 11/30/19 13:01:01<br>• Amazon Fraud Detector は、イベントタイムスタンプの日付/タイムスタンプ形式を解析するときに、次の仮定を行います。 |

| 名前 | 説明 | 要件 |
| --- | --- | --- |
| | | • ISO 8601 標準を使用する場合は、前述の仕様と完全に一致する必要があります。<br>• 他の形式のいずれかを使用している場合は、さらに柔軟性があります。<br>  • 月および日には、1 桁または 2 桁の数字を指定できます。例えば、2019 年 1 月 12 日は有効な日付です。<br>  • hh:mm:ss を持っていない場合は、含める必要はありません (つまり、日付を指定するだけです)。時と分だけのサブセット (例えば、hh:mm) を指定することもできます。時のみの指定はサポートされていません。ミリ秒もサポートされていません。<br>  • AM/PM ラベルを指定した場合は、12 時間時計と見なされます。AM/PM 情報がない場合は、24 時間時計と見なされます。<br>  • 日付要素の区切り文字として「/」または「-」を使用できます。タ |

| 名前 | 説明 | 要件 |
| --- | --- | --- |
| | | イムスタンプ要素には「:」が想定されます。 |
| ENTITY_ID | イベントを実行するエンティティの識別子。 | • ENTITY_ID は、バッチインポートジョブに必要です。<br>• 正規表現のパターンを満たす必要があります: `^[0-9A-Za-z_.@+-]+$`<br>• 評価時にエンティティ ID が使用できない場合は、エンティティ ID を unknown として指定します。 |
| ENTITY_TYPE | マーチャントや顧客など、イベントを実行するエンティティ。 | ENTITY_ID は、バッチインポートジョブに必要です。 |
| EVENT_LABEL | イベントを `fraudulent` または `legitimate` として分類します。 | LABEL_TIMESTAMP が含まれている場合は EVENT_LABEL が必要です |
| LABEL_TIMESTAMP | イベントラベルが最後に入力または更新されたときのタイムスタンプ | • EVENT_LABEL が含まれている場合は LABEL_TIMESTAMP が必要です。<br>• タイムスタンプ形式に従う必要があります。 |

## バッチインポートのために CSV ファイルを Amazon S3 にアップロードする

データで CSV ファイルを作成したら、このファイルを Amazon Simple Storage Service (Amazon S3) バケットにアップロードします。

Amazon S3 バケットにイベントデータをアップロードするには

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/s3/ で Amazon S3 コンソールを開きます。

2. バケットの作成 を選択します。

   [バケットの作成] ウィザードが開きます。

3. [バケット名] に、バケットの DNS に準拠する名前を入力します。

   バケット名には次の条件があります。

   - すべての Amazon S3 で一意にする。
   - 3～63 文字で指定する。
   - 大文字を含めないでください。
   - 先頭の文字には小文字の英文字または数字を使用する。

   バケットを作成したら、その名前を変更することはできません。バケットの名前付けの詳細については、Amazon Simple Storage Service ユーザーガイドの「バケットの名前付けルール」を参照してください。

   > **Important**
   >
   > バケット名にアカウント番号などの機密情報を含めないでください。バケット名は、バケット内のオブジェクトを参照する URL に表示されます。

4. リージョンで、バケットを配置する AWS リージョンを選択します。Amazon Fraud Detector を使用しているリージョンと同じリージョンを選択する必要があります。これは、米国東部 (バージニア北部)、米国東部 (オハイオ)、米国西部 (オレゴン)、欧州 (アイルランド)、アジアパシフィック (シンガポール)、またはアジアパシフィック (シドニー) のいずれかです。

5. [バケットのパブリックアクセスブロック設定] で、バケットに適用するパブリックアクセスブロック設定を選択します。

   設定はすべて有効のままにしておくことをお勧めします。パブリックアクセスのブロックの詳細については、Amazon Simple Storage Service ユーザーガイドの「Amazon S3 ストレージへのパブリックアクセスのブロック」を参照してください。

6. [バケットを作成] を選択します。

7. トレーニングデータファイルを Amazon S3 バケットにアップロードします。トレーニングファイルの Amazon S3 の場所 (例: s3://bucketname/object.csv) を書き留めます。

# Amazon Fraud Detector コンソールでのイベントデータのバッチインポート

`CreateBatchImportJob` API または AWS SDK を使用して、Amazon Fraud Detector コンソールで多数のイベントデータセットを簡単にインポートできます。先に進む前に、データセットを CSV ファイルとして準備する手順に従ってください。CSV ファイルも Amazon S3 バケットにアップロードしていることを確認します。

Amazon Fraud Detector コンソールの使用

コンソールでイベントデータをバッチインポートするには

1. AWS コンソールを開いてアカウントにサインインし、Amazon Fraud Detector に移動します。
2. 左側のナビゲーションペインで [イベント] を選択します。
3. イベントタイプを選択します。
4. [保存されたイベント] タブを選択します。
5. [保存されたイベントの詳細] ペインで、[イベントの取り込み] が ON になっていることを確認します。
6. [イベントデータのインポート] ペインで [新規インポート] を選択します。
7. [新しいイベントのインポート] ページで、次の情報を指定します。
   - 〔推奨] このデータセットのスマートデータ検証を有効にする - 新しいセットをデフォルト設定のままにします。
   - [データの IAM ロール] で、インポートする予定の CSV ファイルを保持する Amazon S3 バケット用に作成した IAM ロールを選択します。
   - [入力データの場所] では、CSV ファイルがある S3 の場所を入力します。
   - インポート結果を保存するのに別の場所を指定する場合は、[入力と結果の個別のデータ位置] ボタンをクリックし、有効な Amazon S3 バケットの場所を指定します。

   > **Important**
   >
   > 選択した IAM ロールに、入力 Amazon S3 バケットへの読み取りアクセス許可と、出力 Amazon S3 バケットへの書き込みアクセス許可があることを確認します。

8. [開始] を選択します。

9. イベントデータをインポートするペインのステータス列には、検証ジョブとインポートジョブのステータスが表示されます。上部のバナーには、データセットが最初に検証を経てからインポートされるときのステータスの概要が表示されます。
10. 「」に記載されているガイダンスに従ってくださいデータセットの検証とインポートジョブの進行状況をモニタリングする。

データセットの検証とインポートジョブの進行状況をモニタリングする

Amazon Fraud Detector コンソールを使用してバッチインポートジョブを実行している場合、デフォルトでは、Amazon Fraud Detector はインポート前にデータセットを検証します。Amazon Fraud Detector コンソールの新しいイベントインポートページで、検証ジョブとインポートジョブの進行状況とステータスをモニタリングできます。ページ上部のバナーには、検証結果とインポートジョブのステータスの簡単な説明が表示されます。検証結果とインポートジョブのステータスによっては、データセットの検証とインポートを成功させるためのアクションを実行する必要がある場合があります。

次の表は、検証およびインポートオペレーションの結果に応じて実行する必要があるアクションの詳細を示しています。

| バナーメッセージ | ステータス | 意味 | 実行すべきこと |
| --- | --- | --- | --- |
| データ検証が開始されました | 検証中 | SDV がデータセットの検証を開始しました | ステータスが変更されるまで待機する |
| データセットのエラーのため、データ検証を続行できません。データファイルのエラーを修正し、新しいインポートジョブを開始します。詳細については、検証レポートを参照してください。 | 検証に失敗しました | SDV は、データファイルの問題を特定しました。これらの問題は、データセットのインポートを成功させるために対処 | イベントデータをインポートペインで、ジョブ ID を選択し、検証レポートを表示します。レポートの推奨事項に従って、リストされているすべてのエラーに対処します。詳細については、「検証レポートの使用」を参照してください。 |

| バナーメッセージ | ステータス | 意味 | 実行すべきこと |
| --- | --- | --- | --- |
| | | する必要があります。 | |
| データインポートが開始されました。検証が正常に完了しました | インポート中 | データセットが検証に合格しました。AFD がデータセットのインポートを開始しました | ステータスが変更されるまで待機する |
| 検証が完了し、警告が表示されました。データインポートが開始されました | インポート中 | データセット内の一部のデータの検証に失敗しました。ただし、検証に合格したデータは、インポートの最小データサイズ要件を満たしています。 | バナーのメッセージをモニタリングし、ステータスが変更されるまで待機する |

| バナーメッセージ | ステータス | 意味 | 実行すべきこと |
| --- | --- | --- | --- |
| データの一部がインポートされました。データの一部が検証に失敗し、インポートされませんでした。詳細については、「検証レポート」を参照してください。 | インポートされました。ステータスには警告アイコンが表示されます。 | 検証に失敗したデータファイル内のデータの一部はインポートされませんでした。検証に合格した残りのデータがインポートされました。 | イベントデータをインポートページで、ジョブ ID を選択し、検証レポートを表示します。データレベルの警告表の推奨事項に従って、リストされている警告に対処します。すべての警告に対処する必要はありません。ただし、データセットにインポートを成功させるための検証に合格するデータが 50% 以上あることを確認してください。警告に対処したら、新しいインポートジョブを開始します。詳細については、「検証レポートの使用」を参照してください。 |
| 処理エラーのため、データのインポートに失敗しました。新しいデータインポートジョブを開始する | インポート失敗 | 一時的なランタイムエラーによりインポートが失敗しました | 新しいインポートジョブを開始する |
| データは正常にインポートされました | インポート済み | 検証とインポートの両方が正常に完了しました | インポートジョブのジョブ ID を選択して詳細を表示し、モデルトレーニングに進みます。 |

> **Note**
>
> データセットが Amazon Fraud Detector に正常にインポートされてから 10 分待って、システムによって完全に取り込まれていることを確認することをお勧めします。

### スマートデータ検証レポート

スマートデータ検証は、検証が完了した後に検証レポートを作成します。検証レポートには、SDV がデータセット内で特定したすべての問題の詳細と、最も影響の大きい問題を修正するための推奨アクションが表示されます。検証レポートを使用して、問題の内容、データセット内の問題の場所、問題の重要度、および問題の修正方法を確認できます。検証レポートは、検証が正常に完了した場合でも作成されます。この場合、レポートを表示して、問題がリストされているかどうかを確認し、問題がある場合は、それらを修正するかどうかを決定できます。

> **Note**
>
> 現在のバージョンの SDV では、データセットをスキャンして、バッチのインポートが失敗する原因となる問題がないか確認します。検証とバッチインポートが成功しても、データセットにモデルトレーニングが失敗する原因となる問題が発生する可能性があります。検証とインポートが成功した場合でも検証レポートを表示し、モデルトレーニングを成功させるためにレポートに記載されている問題に対処することをお勧めします。問題に対処したら、新しいバッチインポートジョブを作成します。

### 検証レポートへのアクセス

次のいずれかのオプションを使用して、検証完了後いつでも検証レポートにアクセスできます。

1. 検証が完了し、インポートジョブの進行中に、上部のバナーで検証レポートの表示を選択します。
2. インポートジョブが完了したら、イベントデータをインポートペインで、完了したインポートジョブのジョブ ID を選択します。

### 検証レポートの使用

インポートジョブの検証レポートページには、このインポートジョブの詳細、見つかった場合は重大なエラーのリスト、見つかった場合はデータセット内の特定のイベント (行) に関する警告のリスト、無効な値や各変数の欠落値などの情報を含むデータセットの簡単な概要が表示されます。

- インポートジョブの詳細

  インポートジョブの詳細を提供します。インポートジョブが失敗したか、データセットが部分的にインポートされた場合は、結果ファイルに移動を選択して、インポートに失敗したイベントのエラーログを表示します。

- 重大なエラー

  SDV によって識別されるデータセット内の最も影響の大きい問題の詳細を提供します。このペインにリストされているすべての問題は重要であり、インポートを続行する前に対処する必要があります。重大な問題に対処せずにデータセットをインポートしようとすると、インポートジョブが失敗する可能性があります。

  重大な問題に対処するには、警告ごとに提供される推奨事項に従ってください。重大なエラーペインにリストされているすべての問題に対処したら、新しいバッチインポートジョブを作成します。

- データレベルの警告

  データセット内の特定のイベント (行) の警告の概要を提供します。データレベルの警告ペインが入力されている場合、データセットの一部のイベントが検証に失敗し、インポートされませんでした。

  警告ごとに、説明列には問題のあるイベントの数が表示されます。また、サンプルイベント IDs は、問題のある残りのイベントを見つけるための出発点として使用できるサンプルイベント IDs の一部のリストを提供します。警告に記載されている推奨事項を使用して問題を解決します。また、問題に関する追加情報については、出力ファイルのエラーログを使用します。エラーログは、バッチインポートに失敗したすべてのイベントに対して生成されます。エラーログにアクセスするには、ジョブの詳細のインポートペインで、結果ファイルに移動を選択します。

> **Note**
>
> データセット内のイベント (行) の 50% 以上が検証に失敗した場合、インポートジョブも失敗します。この場合、新しいインポートジョブを開始する前にデータを修正する必要があります。

- データセットの概要

  データセットの検証レポートの概要を提供します。警告数列に 0 個を超える警告が表示されている場合は、それらの警告を修正する必要があるかどうかを判断します。警告数列に 0 と表示されている場合は、モデルのトレーニングを続行します。

## AWS SDK for Python (Boto3) を使用したバッチインポートイベントデータ

次の例は、CreateBatchImportJob API のサンプルリクエストを示しています。バッチインポートジョブには、jobId、inputPath、outputPath、eventTypeName と iamRoleArn が含まれている必要があります。ジョブが CREATE_FAILED 状態でない限り、JobID に過去のジョブの同じ ID を含めることはできません。inputPath と outputPath は有効な S3 パスでなければなりません。OutputPath でファイル名の指定をオプトアウトできますが、有効な S3 バケットの場所を指定する必要があります。eventTypeName と iamRoLeArn が存在する必要があります。IAM ロールは、Amazon S3 バケットを入力するための読み取りアクセス許可と、Amazon S3 バケットを出力するための書き込みアクセス許可を付与する必要があります。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.create_batch_import_job (
jobId = 'sample_batch_import',
inputPath = 's3://bucket_name/input_file_name.csv',
outputPath = 's3://bucket_name/',
eventTypeName = 'sample_registration',
iamRoleArn: 'arn:aws:iam::************:role/service-role/AmazonFraudDetector-
DataAccessRole-*************'
)
```

## バッチインポートジョブをキャンセルする

進行中のバッチインポートジョブは、CancelBatchImportJob API または AWS SDK を使用して、Amazon Fraud Detector コンソールで、いつでもキャンセルできます。

コンソールでバッチインポートジョブをキャンセルするには、

1. AWS コンソールを開いてアカウントにサインインし、Amazon Fraud Detector に移動します。
2. 左側のナビゲーションペインで [イベント] を選択します。

3. イベントタイプを選択します。
4. [保存されたイベント] タブを選択します。
5. [イベントデータのインポート] ペインで、キャンセルする進行中のインポートジョブのジョブ ID を選択します。
6. [イベントジョブ] ページで、[アクション] を選択し、[イベントのインポートのキャンセル] をクリックします。
7. [イベントのインポートの停止] をクリックして、バッチインポートジョブをキャンセルします。

AWS SDK for Python (Boto3) を使用したバッチインポートジョブのキャンセル

次の例は、`CancelBatchImportJob` API のサンプルリクエストを示しています。インポートのキャンセルジョブには、進行中のバッチインポートジョブのジョブ ID を含める必要があります。

```
import boto3
fraudDetector = boto3.client('frauddetector')
fraudDetector.cancel_batch_import_job (
    jobId = 'sample_batch'
)
```

## GetEventPredictions API オペレーションを使用してイベントデータを保存する

デフォルトでは、評価のために `GetEventPrediction` API に送信されたすべてのイベントは Amazon Fraud Detector に保存されます。つまり、Amazon Fraud Detector は、予測を生成し、そのデータを使用して計算された変数をほぼリアルタイムで更新するときに、イベントデータを自動的に保存します。Amazon Fraud Detector コンソールでイベントタイプに移動し、[イベント取り込み] をオフに設定するか、`PutEventType` API オペレーションを使用して EventIngestion 値を DISABLED に更新することにより、データストレージを無効にできます。`GetEventPrediction` API オペレーションの詳細については、「不正予測」を参照してください。

> **Important**
>
> イベントタイプの [イベント取り込み] を有効にしたら、それを有効のままにしておくことを強くお勧めします。同じイベントタイプに対してイベント取り込みを無効にしてから予測を生成すると、動作が矛盾する可能性があります。

# SendEvent API オペレーションを使用してイベントデータを保存する

SendEvent API オペレーションを使用して、イベントの不正予測を生成せずに Amazon Fraud Detector にイベントを保存できます。例えば、SendEvent オペレーションを使用して履歴データセットをアップロードできます。このデータセットは、後でモデルのトレーニングに使用できます。

SendEvent API のイベントタイムスタンプ形式

SendEvent API を使用してイベントデータを保存する場合は、イベントのタイムスタンプが必要な形式であることを確認する必要があります。Amazon Fraud Detector は、次の日付/タイムスタンプ形式をサポートしています。

- %yyyy-%mm-%ddT%hh:%mm:%ssZ (ミリ秒なし、UTC のみの ISO 8601 標準)

  例: 2019-11-30T13:01:01Z

- %yyyy/%mm/%dd %hh:%mm:%ss (AM/PM)

  例: 2019/11/30 1:01:01 PM、または 2019/11/30 13:01:01

- %mm/%dd/%yyyy %hh:%mm:%ss

  例: 11/30/2019 1:01:01 PM、または 11/30/2019 13:01:01

- %mm/%dd/%yy %hh:%mm:%ss

  例: 11/30/19 1:01:01 PM、または 11/30/19 13:01:01

Amazon Fraud Detector は、イベントタイムスタンプの日付/タイムスタンプ形式を解析するときに、次の仮定を行います。

- ISO 8601 標準を使用する場合は、前述の仕様と完全に一致する必要があります。
- 他の形式のいずれかを使用している場合は、さらに柔軟性があります。
  - 月および日には、1 桁または 2 桁の数字を指定できます。例えば、2019 年 1 月 12 日は有効な日付です。
  - hh:mm:ss を持っていない場合は、含める必要はありません (つまり、日付を指定するだけです)。時と分だけのサブセット (例えば、hh:mm) を指定することもできます。時のみの指定はサポートされていません。ミリ秒もサポートされていません。
  - AM/PM ラベルを指定した場合は、12 時間時計と見なされます。AM/PM 情報がない場合は、24 時間時計と見なされます。

- 日付要素の区切り文字として「/」または「-」を使用できます。タイムスタンプ要素には「:」が想定されます。

SendEvent API コールの例を以下に示します。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.send_event(
            eventId        = '802454d3-f7d8-482d-97e8-c4b6db9a0428',
            eventTypeName  = 'sample_registration',
            eventTimestamp = '2020-07-13T23:18:21Z',
            eventVariables =  {
       'email_address' : 'johndoe@exampledomain.com',
       'ip_address' : '1.2.3.4'},
            assignedLabel  = 'legit',
            labelTimestamp = '2020-07-13T23:18:21Z',
            entities       = [{'entityType':'sample_customer', 'entityId':'12345'}],

)
```

## 保存されたイベントデータの詳細を取得する

Amazon Fraud Detector にイベントデータを保存した後、GetEvent API を使用して、イベント用に保存された最新のデータを確認できます。次のコード例では、sample_registrationイベントに保存されている最新のデータをチェックします。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.get_event(
            eventId        = '802454d3-f7d8-482d-97e8-c4b6db9a0428',
            eventTypeName  = 'sample_registration'
)
```

# 保存されたイベントデータセットのメトリクスを表示する

イベントタイプごとに、Amazon Fraud Detector コンソールで、保存されたイベントの数、保存されたイベントの合計サイズ、最も早い保存されたイベントと最新の保存されたイベントのタイムスタンプなどのメトリクスを表示できます。

イベントタイプの保存されたイベントメトリクスを表示するには、次の手順を実行します。

1. AWS コンソールを開き、アカウントにサインインします。Amazon Fraud Detector に移動します。
2. 左側のナビゲーションペインで [イベント] を選択します。
3. イベントタイプを選択します。
4. [保存されたイベント] タブを選択します。
5. [保存されたイベントの詳細] ペインにメトリクスが表示されます。これらのメトリクスは 1 日 1 回自動的に更新されます。
6. 必要に応じて、[イベントのメトリクスを更新]をクリックして、メトリクスを手動で更新します。

> **Note**
>
> データをインポートしたばかりの場合は、データのインポートが完了してから 5 ~ 10 分待って、メトリクスを更新して表示することをお勧めします。

# イベントのオーケストレーション

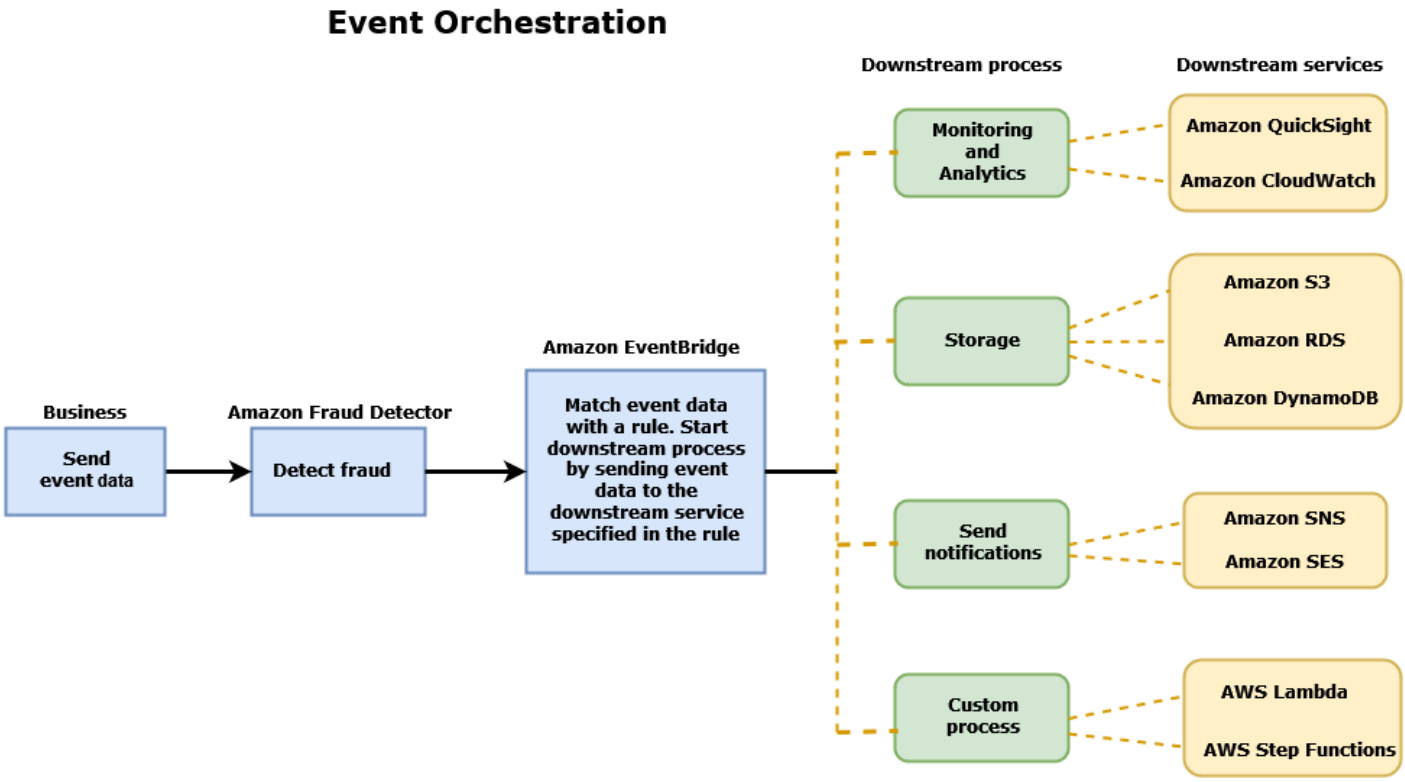
イベントオーケストレーションを使用すると、Amazon EventBridge を使用して、ダウンストリーム処理 AWS のサービス のために にイベントを簡単に送信できます。Amazon Fraud Detector には、不正検出後のイベント処理を自動化するために使用できる簡単なルールが用意されています。イベントオーケストレーションを使用すると、イベントデータからインサイトを取得するためのダッシュボードへのイベントの送信、不正検出の結果に基づく通知の生成、不正検出から学んだことに基づくラベルによるイベントの更新などのダウンストリームイベントプロセスを自動化できます。

イベントオーケストレーションは、Amazon EventBridge を介して AWS 環境内のサービスに簡単にアクセスできます。Amazon EventBridge は、イベントを API 送信先に直接送信するか AWS のサービス 、API 送信先を使用して間接的に送信するように設定できます。ダウンストリームプロセスのオーケストレーション AWS のサービス に使用する は、ターゲットとも呼ばれます。ダウンストリーム処理の調整に使用できるターゲットには、次のようなものがあります。

- モニタリングと分析の場合 — Amazon QuickSight、Amazon CloudWatch
- ストレージ用 — Amazon S3、Amazon RDS、Amazon DynamoDB
- 通知の送信 — Amazon SNS、Amazon SES
- カスタム処理の場合 — AWS Lambda、AWS Step Functions

Amazon EventBridge でサポートされているオーケストレーションターゲットの詳細については、「Amazon EventBridge ターゲット」を参照してください。

次の図は、イベントオーケストレーションの仕組みの概要を示しています。




# イベントオーケストレーションの設定

イベントのイベントオーケストレーションを設定するには、ターゲットサービスでプロセスを設定し、イベントデータを受信して送信するように Amazon EventBridge を設定し、ダウンストリームプロセスを開始するための条件を指定するルールを Amazon EventBridge で作成する必要があります。イベントオーケストレーションを設定するには、次のステップを実行します。

イベントオーケストレーションを設定するには

1. Amazon EventBridge ユーザーガイドに移動し、Amazon EventBridge を使用する方法について説明します。ユースケースに合わせて Amazon EventBridge でルールを作成する方法を確認してください。
2. 「」の手順に従いますAmazon Fraud Detector でイベントオーケストレーションを有効にする。

> **Note**
>
> イベントのイベントオーケストレーションはデフォルトで無効になっています。

3. イベントデータを受信して処理するようにターゲットサービスを設定します。例えば、ダウンストリームプロセスで通知を送信し、Amazon SNS を使用する場合は、Amazon SNS コンソールに移動して SNS トピックを作成し、エンドポイントをトピックにサブスクライブします。
4. 手順に従って Amazon EventBridge ルールを作成します。

> **Important**
>
> Amazon EventBridge でイベントパターンを構築するとき は、`aws.frauddetector`ソースフィールドに を、詳細タイプフィールドに を必ず指定`Event Prediction Result Returned`してください。

# Amazon Fraud Detector でイベントオーケストレーションを有効にする

イベントのイベントオーケストレーションは、イベントタイプを作成するとき、またはイベントタイプを作成した後に有効にできます。イベントオーケストレーションは、Amazon Fraud Detector コンソール、 `put-event-type` コマンド、 `PutEventType` API、または を使用して有効にできます AWS SDK for Python (Boto3)。

## Amazon Fraud Detector コンソールでイベントオーケストレーションを有効にする

この例では、既に作成されているイベントタイプのイベントオーケストレーションを有効にします。新しいイベントタイプを作成し、オーケストレーションを有効にする場合は、「」の手順に従いますイベントタイプを作成する。

イベントオーケストレーションを有効にするには

1. AWS マネジメントコンソールを開き、アカウントにサインインします。Amazon Fraud Detector に移動します。
2. 左側のナビゲーションペインで [イベント] を選択します。
3. イベントタイプ ページで、イベントタイプを選択します。
4. Amazon EventBridge でイベントオーケストレーションを有効にするを有効にします。
5. のステップ 3 の手順に進みますイベントオーケストレーションの設定。

## を使用してイベントオーケストレーションを有効にする AWS SDK for Python (Boto3)

次の例は、イベントオーケストレーションを有効にするsample_registrationためにイベントタイプを更新するリクエストの例を示しています。この例では PutEventType API を使用しており、変数 ip_addressと email_address、ラベル legitと fraud、エンティティタイプ を作成したことを前提としていますsample_customer。これらのリソースの作成方法については、「リソース」を参照してください。

```
import boto3
fraudDetector = boto3.client('frauddetector')
fraud_detector.put_event_type(
  name = 'sample_registration',
  eventVariables = ['ip_address', 'email_address'],
  eventOrchestration = {'eventBridgeEnabled': True},
  labels = ['legit', 'fraud'],
  entityTypes = ['sample_customer'])
```

# Amazon Fraud Detector でイベントオーケストレーションを無効にする

Amazon Fraud Detector コンソール、 put-event-type コマンド、 PutEventType API、または を使用して、イベントのイベントオーケストレーションをいつでも無効にできます AWS SDK for Python (Boto3)。

## Amazon Fraud Detector コンソールでイベントオーケストレーションを無効にする

イベントオーケストレーションを無効にするには

1. AWS マネジメントコンソールを開き、アカウントにサインインします。Amazon Fraud Detector に移動します。
2. 左側のナビゲーションペインで [イベント] を選択します。
3. イベントタイプ ページで、イベントタイプを選択します。
4. Amazon  EventBridge でイベントオーケストレーションを有効にするをオフにします。

# を使用してイベントオーケストレーションを無効にする AWS SDK for Python (Boto3)

次の例は、 PutEventType API を使用してイベントタイプを更新sample_registrationしてイベントオーケストレーションを無効にするリクエストの例を示しています。

```
import boto3
fraudDetector = boto3.client('frauddetector')
fraud_detector.put_event_type(
  name = 'sample_registration',
  eventVariables = ['ip_address', 'email_address'],
  eventOrchestration = {'eventBridgeEnabled': False},
  entityTypes = ['sample_customer'])
```

# モデル

Amazon Fraud Detector は、機械学習モデルを使用して不正予測を生成します。各モデルは、モデルタイプを使用してトレーニングします。モデルタイプは、モデルのトレーニングに使用されるアルゴリズムと変換を指定します。モデルトレーニングとは、ユーザーが提供するデータセットを使用して、不正イベントを予測できるモデルを作成するプロセスです。

モデルを作成するには、まずモデルタイプを選択し、モデルのトレーニングに使用するデータを準備して提供する必要があります。

## モデルタイプの選択

Amazon Fraud Detector では、次のモデルタイプを使用できます。ユースケースに適したモデルタイプを選択します。

- オンライン不正インサイト

  オンライン不正インサイトモデルタイプは、評価対象のエンティティに関する履歴データがほとんどない場合に、不正を検出するように最適化されています。例えば、新規顧客がオンラインで新しいアカウントの登録を行う場合などです。
- トランザクション不正インサイト

  トランザクション不正インサイトモデルタイプは、評価されるエンティティが予測精度を改善するために分析できるインタラクションの履歴を持っている可能性のある不正ユースケース (例えば、過去の購入履歴を持つ既存の顧客) を検出するのに最適です。
- アカウント乗っ取りインサイト

  Account Takeover Insights モデルタイプは、アカウントがフィッシングや他のタイプの攻撃によって侵害されたかどうかを検出します。ログイン時に使用するブラウザやデバイスなど、侵害されたアカウントのログインデータは、アカウントに関連付けられているログイン履歴データとは異なります。

### オンライン不正インサイト

オンライン不正インサイトは、教師付き機械学習モデルです。つまり、不正および正当なトランザクションのサンプル履歴を使用してモデルをトレーニングします。オンライン不正インサイトモデルは、わずかな履歴データに基づいて不正を検出できます。モデルの入力は柔軟性があるため、フェイ

クレビュー、プロモーションの不正使用、ゲストのチェックアウト不正など、さまざまな不正リスクを検出するように適応できます。

オンライン不正インサイトモデルは、データのエンリッチメント、変換、不正分類に機械学習アルゴリズムのアンサンブルを使用します。モデルトレーニングプロセスの一環として、オンライン不正インサイトは、IP アドレスや銀行識別番号などの raw データ要素を、IP アドレスのジオロケーションやクレジットカードの発行銀行などのサードパーティーデータで強化します。オンライン不正インサイトでは、サードパーティーデータに加えて、Amazon と AWSで見られた不正パターンを考慮した深層学習アルゴリズムを使用します。これらの不正パターンは、勾配ツリーブースティングアルゴリズムを使用して、モデルへの入力特徴になります。

パフォーマンスを向上させるために、オンライン不正インサイトは、ベイズ最適化プロセスを介して、勾配ツリーブースティングアルゴリズムのハイパーパラメータを最適化します。さまざまなモデルパラメータ (ツリーの数、ツリーの深さ、枝葉あたりのサンプル数など) を使用して、数十種類のモデルを順番にトレーニングします。また、マイノリティ不正集団の重み付けなど、さまざまな最適化戦略を使用して、非常に低い不正率を処理します。

## データソースの選択

オンライン不正インサイトモデルをトレーニングする場合、外部 (Amazon Fraud Detector の外) に格納されているイベントデータまたは Amazon Fraud Detector 内に格納されているイベントデータに基づいてモデルをトレーニングできます。Amazon Fraud Detector が現在サポートしている外部ストレージは、Amazon Simple Storage Service (Amazon S3) です。外部ストレージを使用している場合は、イベントデータセットをカンマ区切り値 (CSV) 形式として Amazon S3 バケットにアップロードする必要があります。これらのデータストレージオプションは、モデルトレーニング設定内で EXTERNAL_EVENTS (外部ストレージの場合) および INGESTED_EVENTS (内部ストレージの場合) と呼ばれます。使用可能なデータソースとそのデータソースにデータを保存する方法の詳細については、「」を参照してくださいイベントデータストレージ。

## データの準備

イベントデータの保存場所 (Amazon S3 または Amazon Fraud Detector) に関係なく、オンライン不正インサイトモデルタイプの要件は同じです。

データセットには、列ヘッダー EVENT_LABEL が含まれている必要があります。この変数は、イベントを不正または正当として分類します。CSV ファイル (外部ストレージ) を使用する場合は、ファイル内のイベントごとに EVENT_LABEL を含める必要があります。内部ストレージの場合、EVENT_LABEL フィールドは任意ですが、トレーニングデータセットに含めるには、すべてのイベントにラベルを付ける必要があります。モデルトレーニングを設定するときに、ラベルなしイベ

ントを無視するか、ラベルなしイベントは正当なラベルであると仮定するか、すべてのラベルなしイベントは不正なラベルであると仮定するかを選択できます。

### データの選択

オンライン不正インサイトモデルをトレーニングするためのデータの選択については、イベントデータの収集を参照してください。

オンライン不正インサイトトレーニングプロセスは、EVENT_TIMESTAMP に基づいて履歴データをサンプリングして分割します。データを手動でサンプリングする必要はありません。そうすると、モデルの結果に悪影響を与える可能性があります。

### イベント変数

オンライン不正インサイトモデルには、必要なイベントメタデータとは別に、モデルトレーニングのデータ検証に合格した少なくとも 2 つの変数が必要で、モデルごとに最大 100 個の変数を持つことができます。一般に、指定する変数が多いほど、モデルは不正イベントと正当なイベントを区別しやすくなります。オンライン不正インサイトモデルは、カスタム変数を含む多数の変数をサポートできますが、IP アドレスと E メールアドレスを含めることをお勧めします。これらの変数は通常、評価対象のエンティティを識別するのに最も効果的だからです。

### データの検証

トレーニングプロセスの一環として、オンライン不正インサイトは、モデルトレーニングに影響を与える可能性のあるデータ品質の問題についてデータセットを検証します。データを検証した後、Amazon Fraud Detector は最適なモデルを構築するために適切なアクションを実行します。これには、潜在的なデータ品質の問題に対する警告の発行、データ品質の問題がある変数の自動削除、エラーの発行、モデルトレーニングプロセスの停止などがあります。詳細については、「データセットの検証」を参照してください。

## トランザクション不正インサイト

トランザクション不正インサイトのモデルタイプは、オンラインまたはカードを提示しないトランザクションの不正を検出するように設計されています。トランザクション不正インサイトは、教師付き機械学習モデルです。つまり、不正および正当なトランザクションのサンプル履歴を使用してモデルをトレーニングします。

トランザクション不正インサイトモデルは、データのエンリッチメント、変換、不正分類に機械学習アルゴリズムのアンサンブルを使用します。特徴エンジニアリングエンジンを活用して、エンティ

ティレベルおよびイベントレベルの集計を作成します。モデルトレーニングプロセスの一環として、トランザクション不正インサイトは、IP アドレスや BIN 番号などの生データ要素を、IP アドレスのジオロケーションやクレジットカードの発行銀行などのサードパーティーデータで強化します。サードパーティーのデータに加えて、トランザクション不正インサイトは、Amazon および AWS で見られた不正パターンを考慮に入れた深層学習アルゴリズムを使用しています。このような不正パターンは、勾配ツリーブースティングアルゴリズムを使用してモデルへの入力特徴になります。

パフォーマンスを向上させるために、トランザクション不正インサイトは、ベイズ最適化プロセスを介して勾配ツリーブースティングアルゴリズムのハイパーパラメータを最適化し、さまざまなモデルパラメータ (ツリーの数、ツリーの深さ、枝葉あたりのサンプル数など) で数十の異なるモデルを順次トレーニングします。また、マイノリティ不正集団を重み付けして、非常に低い不正率に対処するなど、さまざまな最適化戦略もあります。

モデルトレーニングプロセスの一環として、トランザクション不正モデルの特徴エンジニアリングエンジンは、トレーニングデータセット内の各一意のエンティティの値を計算し、不正予測を改善します。例えば、トレーニングプロセス中に、Amazon Fraud Detector は、エンティティが最後に購入を行った時間を計算して保存し、`GetEventPrediction` または `SendEvent` API を呼び出すたびにこの値を動的に更新します。不正予測では、イベント変数が他のエンティティおよびイベントメタデータと組み合わされ、トランザクションが不正であるかどうかを予測します。

## データソースの選択

トランザクション不正インサイトモデルは、Amazon Fraud Detector (INGESTED_EVENTS) を使用して内部に格納されたデータセットでのみトレーニングされます。これにより、Amazon Fraud Detector は、評価しているエンティティに関する計算値を継続的に更新できます。使用可能なデータソースの詳細については、「イベントデータストレージ」を参照してください。

## データの準備

トランザクション不正インサイトモデルをトレーニングする前に、イベントデータセットの準備で説明したように、データファイルにすべてのヘッダーが含まれていることを確認してください。トランザクション不正インサイトモデルは、受け取った新しいエンティティと、データセット内の不正エンティティと正当なエンティティの例を比較するため、エンティティごとに多くの例を提供することが有用です。

Amazon Fraud Detector は、保存されたイベントデータセットをトレーニング用の正しい形式に自動的に変換します。モデルのトレーニングが完了したら、パフォーマンスメトリクスを確認して、トレーニングデータセットにエンティティを追加する必要があるかどうかを判断できます。

## データの選択

デフォルトでは、トランザクション不正インサイトは、選択したイベントタイプについて、保存されたデータセット全体をトレーニングします。オプションで、時間範囲を設定して、モデルのトレーニングに使用されるイベントを減らすことができます。時間範囲を設定するときは、モデルのトレーニングに使用されるレコードが成熟するのに十分な時間をかけるようにします。つまり、正当なレコードと不正なレコードを正しく特定するのに十分な時間が経過していることです。例えば、チャージバック不正の場合、不正イベントを正しく特定するのに 60 日以上かかることがよくあります。最適なモデルのパフォーマンスを得るには、トレーニングデータセット内のすべてのレコードが成熟していることを確認します。

理想的な不正率を表す時間範囲を選択する必要はありません。Amazon Fraud Detector は、不正率、時間範囲、エンティティ数のバランスをとるためにデータを自動的にサンプリングします。

モデルのトレーニングに十分なイベントがない時間範囲を選択すると、Amazon Fraud Detector はモデルトレーニング中に検証エラーを返します。保存されたデータセットの場合、EVENT_LABEL フィールドは任意ですが、トレーニングデータセットに含めるには、イベントにラベルを付ける必要があります。モデルトレーニングを設定するときに、ラベルなしイベントを無視するか、ラベルなしイベントは正当なラベルであると仮定するか、ラベルなしイベントは不正なラベルであると仮定するかを選択できます。

## イベント変数

モデルのトレーニングに使用されるイベントタイプには、必要なイベントメタデータの他に、データ検証に合格した変数が少なくとも 2 つ含まれている必要があり、また最大 100 個の変数を含めることができます。一般に、指定する変数が多いほど、モデルは不正イベントと正当なイベントを区別しやすくなります。トランザクション不正インサイトモデルは、カスタム変数を含む多数の変数をサポートできますが、IP アドレス、E メールアドレス、支払い手段の種類、注文価格、クレジットカードの銀行識別番号を含めることをお勧めします。

## データの検証

トレーニングプロセスの一環として、トランザクション不正インサイトは、モデルトレーニングに影響を与える可能性のあるデータ品質の問題についてトレーニングデータセットを検証します。データを検証した後、Amazon Fraud Detector は最適なモデルを構築するために適切なアクションを実行します。これには、潜在的なデータ品質の問題に対する警告の発行、データ品質の問題がある変数の自動削除、エラーの発行、モデルトレーニングプロセスの停止などがあります。詳細については、「データセットの検証」を参照してください。

Amazon Fraud Detector は警告を発行しますが、一意のエンティティの数が 1,500 未満の場合は、トレーニングデータの品質に影響を与える可能性があるため、モデルのトレーニングを続行します。警告が表示された場合は、パフォーマンスメトリクスを確認してください。

## アカウント乗っ取りに関するインサイト

Account Takeover Insights (ATI) モデルタイプは、悪意のある乗っ取り、フィッシング、または盗まれた認証情報によってアカウントが侵害されたかどうかを検出することで、不正なオンラインアクティビティを識別します。Account Takeover Insights は、オンラインビジネスからのログインイベントを使用してモデルをトレーニングする機械学習モデルです。

トレーニング済みの Account Takeover Insights モデルをリアルタイムログインフローに埋め込んで、アカウントが侵害されているかどうかを検出できます。このモデルは、さまざまな認証タイプとログインタイプを評価します。これには、ウェブアプリケーションログイン、API ベースの認証、single-sign-on (SSO) が含まれます。Account Takeover Insights モデルを使用するには、有効なログイン認証情報が表示されたら、GetEventPrediction API を呼び出します。API は、アカウントが侵害されるリスクを定量化するスコアを生成します。Amazon Fraud Detector は、スコアと定義したルールを使用して、ログインイベントの結果を 1 つ以上返します。結果は、設定した結果です。受け取った結果に基づいて、ログインごとに適切なアクションを実行できます。つまり、ログインに提示された認証情報を承認またはチャレンジできます。例えば、追加の検証としてアカウント PIN を要求することで、認証情報にチャレンジできます。

Account Takeover Insights モデルを使用して、アカウントログインを非同期的に評価し、高リスクアカウントに対してアクションを実行することもできます。例えば、リスクの高いアカウントを人間によるレビューワーの調査キューに追加して、アカウントの停止など、さらにアクションを実行する必要があるかどうかを判断できます。

Account Takeover Insights モデルは、ビジネスの過去のログインイベントを含むデータセットを使用してトレーニングされます。このデータを提供します。オプションで、アカウントを正当または不正としてラベル付けできます。ただし、モデルのトレーニングには必要ありません。Account Takeover Insights モデルは、アカウントの正常なログイン履歴に基づいて異常を検出します。また、悪意のあるアカウント乗っ取りのイベントのリスクが高いことを示すユーザーの行動の異常を検出する方法についても説明します。例えば、通常、同じデバイスと IP アドレスのセットからログインするユーザーなどです。不正行為者は通常、別のデバイスや位置情報からログインします。この手法では、アクティビティが異常であるというリスクスコアが生成されます。これは通常、悪意のあるアカウント乗っ取りの主な特徴です。

Account Takeover Insights モデルをトレーニングする前に、Amazon Fraud Detector は機械学習手法を組み合わせてデータエンリッチメント、データ集約、データ変換を実行します。次に、トレーニン

グプロセス中に、Amazon Fraud Detector は指定した raw データ要素を強化します。raw データ要素の例には、IP アドレスとユーザーエージェントが含まれます。Amazon Fraud Detector は、これらの要素を使用して、ログインデータを記述する追加の入力を作成します。これらの入力には、デバイス、ブラウザ、および位置情報入力が含まれます。また、Amazon Fraud Detector は、指定したログインデータを使用して、過去のユーザー動作を記述する集計変数を継続的に計算します。ユーザー動作の例には、ユーザーが特定の IP アドレスからサインインした回数が含まれます。これらの追加のエンリッチメントと集計を使用すると、Amazon Fraud Detector はログインイベントからの少数の入力から強力なモデルパフォーマンスを生成できます。

Account Takeover Insights モデルは、悪意のあるアクターが人間かロボットかに関係なく、正当なアカウントが悪意のあるアクターによってアクセスされたインスタンスを検出します。このモデルは、アカウント侵害の相対リスクを示す単一のスコアを生成します。侵害された可能性のあるアカウントには、高リスクアカウントとしてフラグが付けられます。高リスクアカウントは、2 つの方法のいずれかで処理できます。どちらの場合も、追加の ID 検証を適用できます。または、手動調査のためにアカウントをキューに送信することもできます。

## データソースの選択

Account Takeover Insights モデルは、Amazon Fraud Detector に内部的に保存されたデータセットでトレーニングされます。Amazon Fraud Detector でログインイベントデータを保存するには、ユーザーのログインイベントを含む CSV ファイルを作成します。イベントごとに、イベントのタイムスタンプ、ユーザー ID、IP アドレス、ユーザーエージェント、ログインデータが有効かどうかなどのログインデータを含めます。CSV ファイルを作成したら、まずファイルを Amazon Fraud Detector にアップロードし、次にインポート機能を使用してデータを保存します。その後、保存されたデータを使用してモデルをトレーニングできます。Amazon Fraud Detector でイベントデータセットを保存する方法の詳細については、「」を参照してください。 Amazon Fraud Detector を使用してイベントデータを内部に保存する

## データの準備

Amazon Fraud Detector では、UTF-8 形式でエンコードされたカンマ区切り値 (CSV) ファイルにユーザーアカウントのログインデータを指定する必要があります。CSV ファイルの最初の行には、ファイルヘッダーが含まれている必要があります。ファイルヘッダーは、各データ要素を記述するイベントメタデータとイベント変数で構成されます。イベントデータは ヘッダーに従います。イベントデータの各行は、単一のログインイベントからのデータで構成されます。

Accounts Takeover Insights モデルでは、CSV ファイルのヘッダー行に次のイベントメタデータとイベント変数を指定する必要があります。

### イベントメタデータ

CSV ファイルヘッダーには、次のメタデータを指定することをお勧めします。イベントメタデータは大文字にする必要があります。

- EVENT_ID - ログインイベントの一意の識別子。
- ENTITY_TYPE - マーチャントや顧客など、ログインイベントを実行するエンティティ。
- ENTITY_ID - ログインイベントを実行するエンティティの識別子。
- EVENT_TIMESTAMP - ログインイベントが発生したときのタイムスタンプ。タイムスタンプは UTC の ISO 8601 標準である必要があります。
- EVENT_LABEL (推奨) - イベントを不正または正当として分類するラベル。「fraud」、「legit」、「1」、「0」など、任意のラベルを使用できます。

> **Note**
>
> - イベントメタデータは大文字である必要があります。大文字と小文字が区別されます。
> - ログインイベントにはラベルは必要ありません。ただし、EVENT_LABEL メタデータを含め、ログインイベントのラベルを指定することをお勧めします。ラベルが不完全または散発的であれば問題ありません。ラベルを指定すると、Amazon Fraud Detector はそれらを使用してアカウント乗っ取り検出率を自動的に計算し、モデルのパフォーマンスチャートとテーブルに表示します。

### イベント変数

Accounts Takeover Insights モデルには、必須 (必須) 変数とオプションの変数の両方があります。変数を作成するときは、必ず変数を適切な変数タイプに割り当ててください。モデルトレーニングプロセスの一環として、Amazon Fraud Detector は変数に関連付けられた変数タイプを使用して、変数エンリッチメントと特徴量エンジニアリングを実行します。

> **Note**
>
> イベント変数名は小文字にする必要があります。大文字と小文字が区別されます。

### 必須変数

Accounts Takeover Insights モデルのトレーニングには、次の変数が必要です。

| カテゴリ | 変数タイプ | 説明 |
| --- | --- | --- |
| IP アドレス | IP_ADDRESS | ログインイベントで使用される IP アドレス |
| ブラウザとデバイス | USERAGENT | ログインイベントで使用されるブラウザ、デバイス、OS |
| 有効な認証情報 | 検証済み | ログインに使用された認証情報が有効かどうかを示します |

オプションの変数

以下の変数は、Accounts Takeover Insights モデルのトレーニングではオプションです。

| カテゴリ | タイプ | 説明 |
| --- | --- | --- |
| ブラウザとデバイス | FINGERPRINT | ブラウザまたはデバイスのフィンガープリントの一意の識別子 |
| セッション ID | SESSION_ID | 認証セッションの識別子 |
| ラベル | EVENT_LABEL | イベントを不正または正当として分類するラベル。「fraud」、「legit」、「1」、「0」など、任意のラベルを使用できます。 |
| Timestamp | LABEL_TIMESTAMP | ラベルが最後に更新されたときのタイムスタンプ。これは、EVENT_LABEL が指定されている場合に必要です。 |

> **Note**
>
> - 両方の必須変数のオプション変数に任意の変数名を指定できます。必須変数とオプション変数をそれぞれ適切な変数タイプに割り当てることが重要です。
> - 追加の変数を指定できます。ただし、Amazon Fraud Detector には、Accounts Takeover Insights モデルのトレーニングにこれらの変数は含まれません。

## データの選択

データの収集は、Account Takeover Insights モデルを作成するための重要なステップです。ログインデータの収集を開始するときは、次の要件と推奨事項を考慮してください。

必須

- 少なくとも 1,500 個のユーザーアカウントの例を示し、それぞれに少なくとも 2 つのログインイベントが関連付けられていること。
- データセットには、少なくとも 30 日間のログインイベントが含まれている必要があります。後で、モデルのトレーニングに使用するイベントの特定の時間範囲を指定できます。

推奨

- データセットには、失敗したログインイベントの例が含まれています。オプションで、失敗したログインに「詐欺的」または「正当」というラベルを付けることができます。
- 6 か月を超えるログインイベントを含む履歴データを準備し、100Kのエンティティを含めます。

最小要件を満たしているデータセットがない場合は、SendEvent API オペレーションを呼び出して Amazon Fraud Detector にイベントデータをストリーミングすることを検討してください。

## データの検証

Account Takeover Insights モデルを作成する前に、Amazon Fraud Detector は、モデルのトレーニングのためにデータセットに含めたメタデータと変数がサイズと形式の要件を満たしているかどうかを確認します。詳細については、「データセットの検証」を参照してください。また、他の要件もチェックします。データセットが検証に合格しない場合、モデルは作成されません。モデルを正常に作成するには、再度トレーニングする前に、検証に合格しなかったデータを修正してください。

一般的なデータセットエラー

Account Takeover Insights モデルをトレーニングするためのデータセットを検証する場合、Amazon Fraud Detector はこれらの問題やその他の問題をスキャンし、1 つ以上の問題が発生した場合に エラーをスローします。

- CSV ファイルは UTF-8 形式ではない。
- CSV ファイルヘッダーには、、、EVENT_ID、ENTITY_IDまたは のメタデータが少なくとも 1 つ含まれていませんEVENT_TIMESTAMP。
- CSV ファイルヘッダーには、、、IP_ADDRESS、USERAGENTまたは の変数タイプの変数が少なくとも 1 つ含まれていませんVALIDCRED。
- 同じ変数タイプに関連付けられている変数が複数あります。
- の 0.1% を超える値には、サポートされている日付とタイムスタンプ形式以外の null または 値EVENT_TIMESTAMPが含まれています。
- 最初のイベントから最後のイベントまでの日数は 30 日未満です。
- 変数タイプのIP_ADDRESS変数の 10% 以上が無効なか null です。
- 変数タイプのUSERAGENT変数の 50% 以上が null を含んでいます。
- 変数タイプのすべてのVALIDCRED変数は に設定されますfalse。

# モデルの構築

Amazon Fraud Detector モデルは、特定のイベントタイプの不正を検出する方法を学びます。Amazon Fraud Detector では、まずモデルを作成します。これは、モデルバージョンのコンテナとして機能します。モデルをトレーニングするたびに、新しいバージョンが作成されます。 AWS コンソールを使用してモデルを作成およびトレーニングする方法の詳細については、「」を参照してくださいステップ 3: モデルを作成する。

各モデルには、対応するモデルスコア変数があります。Amazon Fraud Detector は、モデルを作成するときにユーザーに代わってこの変数を作成します。この変数をルール式で使用して、不正評価中にモデルスコアを解釈できます。

## を使用してモデルをトレーニングおよびデプロイする AWS SDK for Python (Boto3)

CreateModel および CreateModelVersion オペレーションを呼び出すことにより、CreateModel はモデルを開始します。これは、モデルバージョンのコンテナとして機能します。CreateModelVersion がトレーニングプロセスを開始すると、特定のバージョンのモデルが

作成されます。ソリューションの新しいバージョンは、CreateModelVersion を呼び出すたびに作成されます。

次の例は、CreateModel API のサンプルリクエストを示しています。この例では、オンライン不正インサイトモデルタイプで、イベントタイプ sample_registration を作成したと仮定します。イベントタイプの作成の詳細については、「イベントタイプを作成する」を参照してください。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.create_model (
modelId = 'sample_fraud_detection_model',
eventTypeName = 'sample_registration',
modelType = 'ONLINE_FRAUD_INSIGHTS')
```

CreateModelVersion API を使用して最初のバージョンをトレーニングします。TrainingDataSource と には、トレーニングデータセットのソースと Amazon S3 の場所ExternalEventsDetailを指定します。には、Amazon Fraud Detector がトレーニングデータを解釈する方法、具体的には含めるイベント変数とイベントラベルの分類方法TrainingDataSchemaを指定します。デフォルトでは、Amazon Fraud Detector はラベル付けされていないイベントを無視します。このサンプルコードではunlabeledEventsTreatment、AUTOの を使用して、Amazon Fraud Detector がラベル付けされていないイベントの使用方法を決定するように指定します。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.create_model_version (
modelId = 'sample_fraud_detection_model',
modelType = 'ONLINE_FRAUD_INSIGHTS',
trainingDataSource = 'EXTERNAL_EVENTS',
trainingDataSchema = {
    'modelVariables' : ['ip_address', 'email_address'],
    'labelSchema' : {
        'labelMapper' : {
            'FRAUD' : ['fraud'],
            'LEGIT' : ['legit']
        }
       unlabeledEventsTreatment = 'AUTO'
    }
},
```

```
externalEventsDetail = {
    'dataLocation' : 's3://bucket/file.csv',
    'dataAccessRoleArn' : 'role_arn'
}
)
```

リクエストが成功すると、ステータス TRAINING_IN_PROGRESS の新しいモデルバージョンになります。トレーニング中の任意の時点で、UpdateModelVersionStatus を呼び出し、ステータスを [TRAINING_CANCELLED] に更新することでトレーニングをキャンセルできます。トレーニングが完了すると、モデルバージョンのステータスは [TRAINING_COMPLETE] に更新されます。Amazon Fraud Detector コンソールを使用するか、DescribeModelVersions を呼び出すことにより、モデルのパフォーマンスを確認できます。モデルのスコアとパフォーマンスの解釈の詳細については、「モデルスコア」および「モデルパフォーマンスメトリクス」を参照してください。

モデルのパフォーマンスを確認したら、モデルをアクティブ化し、Detectors でリアルタイムの不正予測を使用できるようにします。Amazon Fraud Detector は、Auto Scaling をオンにした状態で冗長性を確保するために、モデルを複数のアベイラビリティーゾーンにデプロイし、不正予測の数に合わせてモデルをスケーリングするようにします。モデルをアクティブ化するには、UpdateModelVersionStatus API を呼び出し、ステータスを [ACTIVE] に更新します。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.update_model_version_status (
modelId = 'sample_fraud_detection_model',
modelType = 'ONLINE_FRAUD_INSIGHTS',
modelVersionNumber = '1.00',
status = 'ACTIVE'
)
```

# モデルスコア

Amazon Fraud Detector は、モデルタイプごとにモデルスコアを生成します。

Account Takeover Insights (ATI) モデルの場合、Amazon Fraud Detector は集計値 (一連の未加工変数を組み合わせて計算された値) のみを使用してモデルスコアを生成します。新しいエンティティの最初のイベントに対してスコア -1 が生成され、不明なリスクが示されます。これは、新しいエンティティの場合、集計の計算に使用される値がゼロまたは null になるためです。Account Takeeover Insights (ATI) モデルは、同じエンティティと既存のエンティティの後続のすべてのイベントについ

て 0 ~ 1000 のモデルスコアを生成します。0 は低い不正リスクを示し、1000 は高い不正リスクを示します。ATI モデルの場合、モデルスコアはチャレンジレート (CR) に直接関係します。例えば、スコア 500 は推定 5% のチャレンジレートに対応し、スコア 900 は推定 0.1% のチャレンジレートに対応します。

オンライン不正インサイト (OFI) モデルとトランザクション不正インサイト (TFI) モデルの場合、Amazon Fraud Detector は、集計値 (raw 変数のセットを組み合わせて計算された値) と raw 値 (変数に指定された値) の両方を使用してモデルスコアを生成します。モデルスコアは 0 ~ 1000 の間で、0 は不正リスクが低いことを示し、1000 は不正リスクが高いことを示します。OFI モデルと TFI モデルの場合、モデルスコアは誤検出率 (FPR) に直接関係します。例えば、600 のスコアは推定 10% の偽陽性率に相当し、900 のスコアは推定 2% の偽陽性率に相当します。次の表に、特定のモデルスコアが推定偽陽性率とどのように関連しているかを詳しく説明します。

| モデルスコア | 推定 FPR |
|---|---|
| 975 | 0.50% |
| 950 | 1% |
| 900 | 2% |
| 860 | 3% |
| 775 | 5% |
| 700 | 7% |
| 600 | 10% |

# モデルパフォーマンスメトリクス

モデルトレーニングが完了すると、Amazon Fraud Detector は、モデルのトレーニングに使用されなかったデータの 15% を使用してモデルのパフォーマンスを検証します。トレーニング済みの Amazon Fraud Detector モデルには、検証パフォーマンスメトリクスに似た実世界の不正検出パフォーマンスが期待できます。

ビジネスとして、より多くの不正行為を検出することと、正当な顧客に対してより多くの軋轢を生むことのバランスをとる必要があります。適切な残高の選択を支援するために、Amazon Fraud Detector はモデルのパフォーマンスを評価するための次のツールを提供しています。

- スコア分布チャート — モデルスコア分布のヒストグラムでは、100,000 イベントの母集団の例を想定しています。左側のY 軸は正当なイベントを表し、右側の Y 軸は不正イベントを表しています。チャート領域をクリックすると、特定のモデルのしきい値を選択できます。これにより、混同行列と ROC チャートの対応するビューが更新されます。
- 混同行列 — モデルの予測と実際の結果を比較して、特定のスコアのしきい値のモデル精度を要約します。Amazon Fraud Detector では、100,000 イベントの人口例を想定しています。不正イベントや正当なイベントの分布は、ビジネスにおける不正率をシミュレートしています。
  - 真陽性 — モデルは不正を予測し、イベントも実際に不正です。
  - 偽陰性 — モデルは不正を予測しますが、イベントは実際には正当です。
  - 真陰性 — モデルは正当を予測し、イベントも実際に正当です。
  - 誤陰性 — モデルは正当を予測しますが、イベントは実際には不正です。
  - 真陽性率 (TPR) — モデルが検出した不正行為の割合。キャプチャレートとも呼ばれます。
  - 偽陽性率 (FPR) — 不正として誤って予測された正当なイベントの総数の割合。
- 受信者操作特性曲線 (ROC) — 可能なすべてのモデルスコアのしきい値に対する偽陽性率の関数として真陽性率をプロットします。[アドバンストメトリクス] を選択して、このチャートを表示します。
- 曲線下面積 (AUC) - 考えられるすべてのモデルスコアのしきい値にわたって TPR と FPR を要約します。予測検出力のないモデルの AUC は 0.5 ですが、完全モデルのスコアは 1.0 です。
- 不確実性の範囲 – モデルから予想される AUC の範囲を示します。範囲が大きいほど (AUC の上限と下限の差 > 0.1)、モデルの不確実性が高くなります。不確実性の範囲が大きい (>0.1) 場合は、よりラベル付けされたイベントを提供し、モデルを再トレーニングすることを検討してください。

モデルパフォーマンスメトリクスを使用するには

1. スコアの分布チャートから初めて、不正イベントと正当なイベントのモデルスコアの分布をグラフで確認します。理想的には、不正イベントと正当なイベントは明確に区別されます。これは、どのイベントが不正イベントでどれが正当なイベントであるかをモデルが正確に特定できていることを示しています。チャート領域をクリックして、モデルのしきい値を選択します。モデルスコアのしきい値の調整が真陽性率と偽陽性率にどのように影響するかを確認できます。

> **Note**
>
> スコア分布チャートは、2 つの異なる Y 軸に不正イベントと正当ないベントをプロットします。左側のY 軸は正当なイベントを表し、右側の Y 軸は不正イベントを表しています。

2. 混行列を確認します。選択したモデルのスコアのしきい値に応じて、100,000 イベントのサンプルに基づいてシミュレートされた影響を確認できます。不正イベントや正当なイベントの分布は、ビジネスにおける不正率をシミュレートしています。この情報を使用して、真陽性率と偽陽性率の適切なバランスを特定します。
3. 詳細を知るには、[アドバンストメトリクス] を選択します。ROC チャートを使用して、モデルスコアのしきい値に対する真陽性率と偽陽性率の関係を理解します。ROC カーブは、真陽性率と偽陽性率の間のトレードオフを微調整するのに役立ちます。

> **Note**
>
> [表] を選択して、テーブル形式でメトリクスを確認することもできます。
> テーブルビューには精度のメトリクスも表示されます。精度は、不正であると予測されたすべてのイベントと比較して、不正と正しく予測できた不正イベントの割合です。

4. パフォーマンスメトリクスを使用して、目標と不正検出のユースケースに基づいて、ビジネスに最適なモデルのしきい値を決定します。例えば、モデルを使用して新しいアカウント登録を高、中、または低リスクに分類する場合は、次のように 3 つのルール条件をドラフトできるように、2 つのしきい値スコアを特定する必要があります。

   - スコア > Xは高リスク
   - スコア < X but > Y は中リスク
   - スコア < Y は低リスク

# モデル変数の重要度

モデル変数の重要度は、モデルバージョン内のモデル変数をランク付けする Amazon Fraud Detector の機能です。各モデル変数には、モデルの全体的なパフォーマンスに対する相対的な重要度に基づいて値が与えられます。最も高い値を持つモデル変数は、そのモデルバージョンのデータセット内の他のモデル変数よりもモデルにとって重要であり、デフォルトで最上位にリストされます。同様に、最小値を持つモデル変数はデフォルトで最下位にリストされ、他のモデル変数と比較して最も重要度が

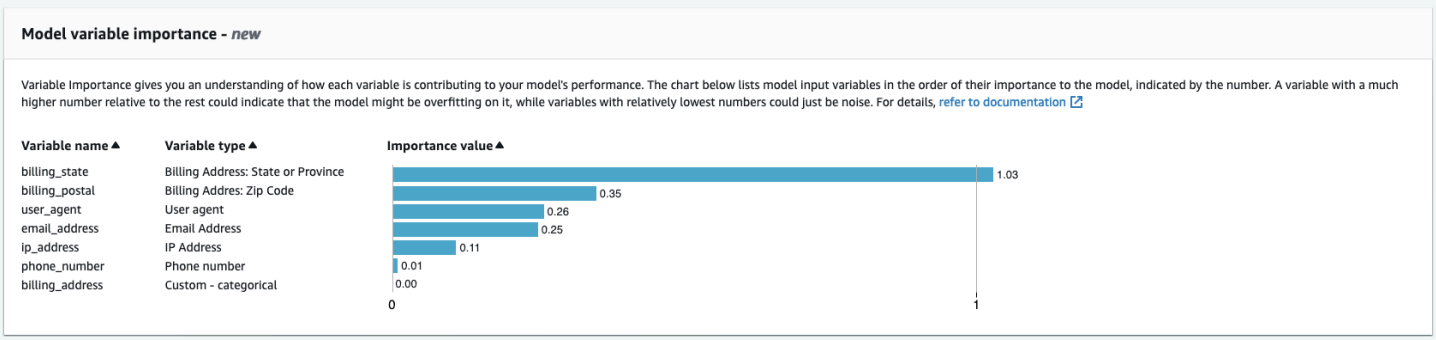
低くなります。モデル変数重要度値を使用すると、モデルのパフォーマンスを左右するのはどのような入力かを把握することができます。

トレーニング済みモデルバージョンのモデル変数重要度値は、Amazon Fraud Detector コンソールで、または DescribeModelVersion API を使用して確認できます。

モデル変数の重要度は、モデルバージョンのトレーニングに使用される変数ごとに次の値のセットを示します。

- 変数タイプ: 変数のタイプ (IP アドレスや E メールなど ) 。詳細については、「変数タイプ」を参照してください。Account Takeover Insights (ATI) モデルの場合、Amazon Fraud Detector は raw 変数タイプと集計変数タイプの両方に変数重要度値を提供します。Raw 変数タイプは、指定した変数に割り当てられます。集計変数タイプは、Amazon Fraud Detector が集計された重要度値を計算するために組み合わせた一連の raw 変数に割り当てられます。
- 変数名: モデルバージョンのトレーニングに使用されたイベント変数の名前 (、ip_addressemail_address、 などare_creadentials_valid ) 。集計変数タイプでは、集計変数重要度値の計算に使用されたすべての変数の名前が一覧表示されます。
- 変数重要度値: モデルのパフォーマンスに対する raw 変数または集計変数の相対的な重要度を表す数値。標準範囲:0 ~ 10

Amazon Fraud Detector コンソールでは、オンライン不正インサイト (OFI) モデルまたはトランザクション不正インサイト (TFI) モデルについて、モデル変数の重要度値が次のように表示されます。Account Takeover Insight (ATI) モデルは、raw 変数の重要度値に加えて、集計された変数重要度値を提供します。ビジュアルチャートでは、最高位の変数の重要度を参照する垂直点線を使用して、変数間の相対的な重要度を簡単に確認できます。




Amazon Fraud Detector は、追加費用なしで、すべてのFraud Detector モデルのバージョンに対して変数重要度値を生成します。

> **Important**
>
> 2021 年 7 月 9 日以前に作成されたモデルバージョンには、変数重要度値はありません。モデル変数重要度値を生成するには、モデルの新しいバージョンをトレーニングする必要があります。

## モデル変数重要度値の使用

モデル変数重要度値を使用して、モデルのパフォーマンスを上げるか下げる要因と、最も寄与する変数を把握できます。次に、モデルを微調整して、全体的なパフォーマンスを向上させます。

具体的には、モデルのパフォーマンスを向上させるには、ドメインの知識に対する変数重要度値を調べ、トレーニングデータ内の問題をデバッグします。例えば、Account ID がモデルへの入力として使用され、それが上部にリストされている場合は、その変数重要度値を確認します。変数重要度値が他の値よりも大幅に高い場合、モデルが特定の不正パターンで過剰適合している可能性があります (例えば、すべての不正イベントが同じアカウント ID からのものである)。ただし、変数が不正ラベルに依存している場合、ラベル漏れが発生する場合もあります。ドメイン知識に基づく分析の結果によっては、変数を削除してより多様なデータセットを使用してトレーニングさせたり、モデルをそのまま維持したりできます。

同様に、最下位にランク付けされた変数を見てみましょう。変数重要度値が他の値よりも大幅に低い場合、このモデル変数はモデルのトレーニングにおいて重要性を持たない可能性があります。その変数を削除して、より単純なモデルバージョンをトレーニングすることを検討できます。モデルに変数が 2 つしかないなど、変数が少ない場合でも、Amazon Fraud Detector は変数重要度値を示し、変数をランク付けします。ただし、この場合のインサイトは限られます。

> **Important**
>
> 1. モデル変数の重要度チャート中の変数が欠落していることに気付いた場合、次のいずれかの原因による場合があります。データセット内の変数を変更し、モデルを再トレーニングすることを検討してください。
>    - トレーニングデータセット内の変数の一意の値の数が 100 未満である。
>    - 0.9 より大きい変数の値がトレーニングデータセットから欠落している。
> 2. モデルの入力変数を調整するたびに、新しいモデルバージョンをトレーニングする必要があります。

## モデル変数重要度値の評価

モデル変数重要度値を評価する場合は、以下を考慮することをお勧めします。

- 変数重要度値は、常にドメイン知識と組み合わせて評価する必要があります。
- モデルバージョン内の他の変数の変数重要度値と比較した場合の変数重要度値を調べます。1 つの変数の変数重要度値を個別に考慮しないでください。
- 同じモデルバージョン内の変数の変数重要度値を比較します。モデルバージョン内の変数の変数重要度値が、異なるモデルバージョンの同じ変数の値と異なる可能性があるため、モデルバージョン間で同じ変数の変数重要度値を比較しないでください。同じ変数とデータセットを使用して異なるモデルのバージョンをトレーニングする場合、これは必ずしも同じ変数重要度値を生成するとは限りません。

## モデル変数の重要度ランキングの表示

モデルトレーニングが完了したら、トレーニング済みモデルバージョンのモデル変数の重要度ランキングは、Amazon Fraud Detector コンソールで、またはDescribeModelVersion API を使用して確認できます。

コンソールを使用してモデル変数の重要度ランキングを表示するには、

1. AWS コンソールを開き、アカウントにサインインします。Amazon Fraud Detector に移動します。
2. 左側のナビゲーションペインで [モデル] を選択します。
3. モデルを選択してから、モデルバージョンを選択します。
4. 概要タブが選択されていることを確認します。
5. 下にスクロールしてモデル変数の重要度ペインを表示します。

## モデル変数重要度値の計算方法の理解

各モデルバージョントレーニングが完了すると、Amazon Fraud Detector はモデル変数重要度値とモデルのパフォーマンスメトリクスを自動的に生成します。このために Amazon Fraud Detector は SHapley Additive exPlanations (SHAP) を用いています。SHAP は、基本的に、すべてのモデル変数の可能なすべての組み合わせを考慮した後のモデル変数の平均期待寄与率です。

SHAP は、まず、各モデル変数の寄与度をイベントの予測に割り当てます。次に、これらの予測を集約して、モデルレベルで変数のランキングを作成します。予測に各モデル変数の寄与度を割り当てるために、SHAP は、可能なすべての変数の組み合わせにおけるモデル出力の差を考慮します。モデルの出力を生成するために特定の変数セットを含めるか削除する可能性をすべて含めると、SHAP は各モデル変数の重要度に正確にアクセスできます。これは、モデル変数が互いに高い相関関係にある場合に特に重要です。

機械学習モデルでは、ほとんどの場合、変数を削除することはできません。代わりに、モデル内で削除または欠落している変数を、1 つ以上のベースラインの対応する変数値 (例えば、不正でないイベント) に置き換えることができます。適切なベースラインインスタンスを選択するのは難しい場合がありますが、Amazon Fraud Detector では、このベースラインを人口平均として設定することで、これを簡単にしています。

# SageMaker AI モデルのインポート

必要に応じて、SageMaker AI がホストするモデルを Amazon Fraud Detector にインポートできます。モデルと同様に、SageMaker AI モデルをディテクターに追加し、 `GetEventPrediction` API を使用して不正予測を生成できます。`GetEventPrediction` リクエストの一環として、Amazon Fraud Detector は SageMaker AI エンドポイントを呼び出し、結果をルールに渡します。

`GetEventPrediction` リクエストの一部として送信されたイベント変数を使用するように Amazon Fraud Detector を設定できます。イベント変数を使用する場合は、入力テンプレートを指定する必要があります。Amazon Fraud Detector は、このテンプレートを使用してイベント変数を必要な入力ペイロードに変換し、SageMaker AI エンドポイントを呼び出します。または、`GetEventPrediction`リクエストの一部として送信される byteBuffer を使用するように SageMaker AI モデルを設定することもできます。

Amazon Fraud Detector は、JSON または CSV 入力形式と JSON または CSV 出力形式を使用する SageMaker AI アルゴリズムのインポートをサポートしています。サポートされている SageMaker AI アルゴリズムの例には、XGBoost、線形学習、ランダムカットフォレストなどがあります。

## を使用して SageMaker AI モデルをインポートする AWS SDK for Python (Boto3)

SageMaker AI モデルをインポートするには、 `PutExternalModel` API を使用します。次の例では、SageMaker AI エンドポイント`sagemaker-transaction-model`がデプロイされ、`InService`ステータスが であり、XGBoost アルゴリズムを使用していることを前提としています。

入力設定は、イベント変数を使用してモデル入力を構築することを指定します (useEventVariables は TRUE に設定)。XGBoost では CSV 入力が必要であることから、入力形式は TEXT_CSV です。csvIntputTemplate は、GetEventPrediction リクエストの一部として送信される変数から CSV 入力を作成する方法を指定します。この例では、order_amt、prev_amt、hist_amt および payment_type 変数を作成したと仮定します。

出力設定は SageMaker AI モデルのレスポンス形式を指定し、適切な CSV インデックスを Amazon Fraud Detector 変数 にマッピングしますsagemaker_output_score。設定したら、ルールで出力変数を使用できます。

> **Note**
>
> SageMaker AI モデルからの出力は、ソース を持つ変数にマッピングする必要がありますEXTERNAL_MODEL_SCORE。変数を使用してコンソールでこれらの変数を作成することはできません。代わりに、モデルのインポートを設定するときに、それらを作成する必要があります。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.put_external_model (
modelSource = 'SAGEMAKER',
modelEndpoint = 'sagemaker-transaction-model',
invokeModelEndpointRoleArn = 'your_SagemakerExecutionRole_arn',
inputConfiguration = {
    'useEventVariables' : True,
    'eventTypeName' : 'sample_transaction',
    'format' : 'TEXT_CSV',
    'csvInputTemplate' : '{{order_amt}}, {{prev_amt}}, {{hist_amt}}, {{payment_type}}'
},

outputConfiguration = {
    'format' : 'TEXT_CSV',
    'csvIndexToVariableMap' : {
        '0' : 'sagemaker_output_score'
    }
},

modelEndpointStatus = 'ASSOCIATED'
```

```
)
```

# モデルまたはモデルバージョンの削除

ディテクターのバージョンに関連付けられていない場合は、Amazon Fraud Detector でモデルとモデルバージョンを削除できます。モデルを削除すると、Amazon Fraud Detector はそのモデルを完全に削除し、データは Amazon Fraud Detector に保存されなくなります。

ディテクターバージョンに関連付けられていない場合は、Amazon SageMaker AI モデルを削除することもできます。SageMaker AI モデルを削除すると Amazon Fraud Detector から切断されますが、モデルは SageMaker AI で引き続き使用できます。

モデルバージョンを削除するには

ステータスが [Ready to deploy] のモデルバージョンのみ削除できます。モデルバージョンのステータスを [ACTIVE] から [Ready to deploy] に変更するには、モデルバージョンのデプロイを解除します。

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/frauddetector で Amazon Fraud Detector コンソールを開きます。
2. Amazon Fraud Detector コンソールの左のナビゲーションペインで、[モデル] を選択します。
3. 削除するモデルバージョンを含むモデルを選択します。
4. 削除するモデルバージョンを選択します。
5. [アクション] を選択してから、[削除] をクリックします。
6. モデルバージョン名を入力してから、[モデルバージョンの削除] を選択します。

モデルバージョンのデプロイを解除するには

ディテクタバージョン (ACTIVE、INACTIVE、DRAFT) で使用中のモデルバージョンのデプロイを解除することはできません。したがって、ディテクターバージョンによって使用されているモデルバージョンのデプロイを解除するには、まずディテクターバージョンからモデルバージョンを削除します。

1. Amazon Fraud Detector コンソールの左のナビゲーションペインで、[モデル] を選択します。
2. デプロイ解除するモデルバージョンを含むモデルを選択します。
3. 削除するモデルバージョンを選択します。

4. [アクション] を選択し、[モデルバージョンのデプロイの解除] を選択します。

モデルを削除するには

モデルを削除する前に、モデルに関連付けられているすべてのモデルバージョンを削除する必要があります。

1. Amazon Fraud Detector コンソールの左のナビゲーションペインで、[モデル] を選択します。
2. 削除するモデルを選択します。
3. [アクション] を選択し、[削除] を選択します。
4. モデル名を入力してから、[モデルの削除] を選択します。

Amazon SageMaker AI モデルを削除するには

1. Amazon Fraud Detector コンソールの左のナビゲーションペインで、[モデル] を選択します。
2. 削除する SageMaker AI モデルを選択します。
3. [アクション] を選択してから、[モデルの削除] をクリックします。
4. モデル名を入力し、SageMaker AI モデルの削除を選択します。

# ディテクター

ディテクターは、不正について評価する特定のビジネスイベントのモデルやルールなどの不正検出ロジックを含むコンテナです。最初に、既に定義したイベントを指定してディテクターを作成し、オプションで、Amazon Fraud Detector によってイベント用に作成およびトレーニング済みのモデルバージョンを追加します。

次に、ディテクターにルールとルール実行順序を追加して、ディテクターのバージョンを作成します。ディテクターバージョンは、不正予測の生成リクエストの一部として実行されるルールとオプションでモデルを定義します。ディテクター内で定義された任意のルールをディテクターバージョンに追加できます。評価されたイベントタイプでトレーニングされたモデルをディテクターバージョンに追加することもできます。ディテクターは複数のバージョンを持つことができ、各バージョンは複数のユースケースに合わせて異なるルールとルール実行順序を持ちます。

各ディテクターバージョンのステータスは、DRAFT、ACTIVE、または である必要がありますINACTIVE。一度に 1 つのディテクターバージョンのみが ACTIVE ステータスになることができます。Amazon Fraud Detector は、 ACTIVEステータスのディテクターバージョンを使用して不正予測を生成します。

# ディテクターの作成

ディテクターを作成するには、既に定義したイベントタイプを指定します。オプションで、Amazon Fraud Detector によってトレーニングおよびデプロイ済みのモデルを追加できます。モデルを追加する場合は、ルールの作成時に Amazon Fraud Detector によって生成されたモデルスコアをルール式で使用できます (例: `$model score < 90` ) 。

Amazon Fraud Detector コンソール、PutDetector API、put-detector コマンド、または AWS SDK を使用してディテクターを作成できます。API、コマンド、または SDK を使用してディテクターを作成している場合は、ディテクターを作成した後、「」の手順に従いますディテクターバージョンの作成。

## Amazon Fraud Detector コンソールでディテクターを作成する

この例では、イベントタイプを作成し、不正予測に使用するモデルバージョンを作成してデプロイしていることを前提としています。

## ステップ 1: ディテクターを構築する

1. Amazon Fraud Detector コンソールの左のナビゲーションペインで、[ディテクター] をクリックします。
2. [ディテクターの作成] を選択します。
3. ディテクターの詳細の定義 ページで、ディテクター名sample_detectorに を入力します。必要に応じて、 などのディテクターの説明を入力しますmy sample fraud detector。
4. イベントタイプ で、不正予測用に作成したイベントタイプを選択します。
5. [Next (次へ)] を選択します。

## ステップ 2: デプロイされたモデルバージョンを追加する

1. これはオプションのステップであることに注意してください。ディテクターにモデルを追加する必要はありません。このステップをスキップするには、[Next] (次へ) を選択します。
2. モデルの追加 - オプションで、モデルの追加を選択します。
3. モデルの追加ページのモデルの選択で、前にデプロイした Amazon Fraud Detector モデル名を選択します。バージョンの選択 で、デプロイされたモデルのモデルバージョンを選択します。
4. [モデルの追加] を選択します。
5. [Next (次へ)] を選択します。

## ステップ 3: ルールを追加する

ルールは、不正予測を評価するときに変数値を解釈する方法を Amazon Fraud Detector に指示する条件です。この例では、モデルスコアを変数値として使用して high_fraud_risk、、medium_fraud_riskの 3 つのルールを作成しますlow_fraud_risk。独自のルール、ルール式、ルール実行順序、結果を作成するには、モデルとユースケースに適した値を使用します。

1. 「ルールの追加」ページの「ルールの定義high_fraud_risk」で、ルール名に「説明 - オプション」で、ルールの説明**This rule captures events with a high ML model score**として「」と入力します。
2. [式] で、Amazon Fraud Detector 簡易ルール式言語を使用して、次のルール式を入力します。

   ```
   $sample_fraud_detection_model_insightscore > 900
   ```

3. [結果] では、[新しい結果の作成] を選択します。結果は、不正予測の結果であり、評価中にルールが一致した場合に返されます。

4. [新しい結果の作成] には、結果名として「verify_customer」と入力します。必要に応じて説明に説明を入力します。
5. [結果の保存] を選択します。
6. [ルールの追加] を選択して、ルール検証チェッカーを実行し、ルールを保存します。作成後、Amazon Fraud Detector はルールをディテクターで使用できるようにします。
7. [別のルールの追加] を選択してから、[ルールの作成] タブをクリックします。
8. このプロセスをさらに 2 回繰り返して、次のルールの詳細を使用して medium_fraud_risk と low_fraud_risk ルールを作成します。

   - medium_fraud_risk

     ルール名: `medium_fraud_risk`

     結果: `review`

     式:

     ```
     $sample_fraud_detection_model_insightscore <= 900 and
     $sample_fraud_detection_model_insightscore > 700
     ```

   - low_fraud_risk

     ルール名: `low_fraud_risk`

     結果: `approve`

     式:

     ```
     $sample_fraud_detection_model_insightscore <= 700
     ```

9. ユースケースのすべてのルールを作成したら、次へを選択します。

   ルールの作成と記述の詳細については、「ルール」および「ルール言語リファレンス」を参照してください。

## ステップ 4: ルールの実行とルールの順序を設定する

ディテクターに含まれるルールのルール実行モードは、定義したすべてのルールが評価されるか、最初に一致したルールでルール評価が停止されるかを決定します。また、ルールの順序によって、ルールを実行する順序が決まります。

デフォルトのルール実行モードは FIRST_MATCHED です。

最初の一致

最初の一致のルール実行モードは、定義されたルールの順序に基づいて、最初の一致ルールの結果を返します。FIRST_MATCHED を指定した場合、Amazon Fraud Detector はルールを順番に評価し、最初に一致したルールで停止します。Amazon Fraud Detector は、その 1 つのルールの結果を示します。

ルールを実行する順序は、結果として生じる不正予測の結果に影響を与える可能性があります。ルールを作成したら、以下の手順に従ってルールの順序を変更し、必要な順序で実行します。

high_fraud_risk ルールがまだルールリストの上部にない場合は、順序を選択し、1 を選択します。これにより、high_fraud_risk が一番上に移動します。

このプロセスを繰り返して、medium_fraud_risk ルールが 2 番目の位置に来て、low_fraud_risk ルールが 3 番目の位置に来るようにします。

すべての一致

すべての一致ルール実行モードは、ルールの順序に関係なく、一致したすべてのルールの結果を返します。ALL_MATCHED を指定した場合、Amazon Fraud Detector はすべてのルールを評価し、一致したすべてのルールの結果を返します。

このチュートリアルFIRST_MATCHEDで を選択し、次へ を選択します。

## ステップ 5: ディテクターバージョンを確認して作成する

ディテクターバージョンは、不正予測の生成に使用される特定のモデルとルールを定義します。

1. 確認と作成ページで、設定したディテクターの詳細、モデル、ルールを確認します。何らかの変更を加える必要がある場合は、対応するセクションの隣にある [編集] をクリックします。
2. [ディテクターの作成] を選択します。ディテクターの作成後、ディテクターの最初のバージョンが Draft ステータスで Detector バージョンテーブルに表示されます。

   ドラフトバージョンを使用して、ディテクターをテストします。

## を使用してディテクターを作成する AWS SDK for Python (Boto3)

次の例は、PutDetector API のサンプルリクエストです。ディテクターは、ディテクターバージョンのコンテナとして機能します。PutDetector API は、ディテクターが評価するイベントタイプを指定します。次の例では、イベントタイプ sample_registration を作成したと想定しています。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.put_detector (
detectorId = 'sample_detector',
eventTypeName = 'sample_registration'
)
```

# ディテクターバージョンの作成

ディテクターバージョンは、不正予測の生成リクエストの一部として使用されるルール、ルール実行順序、およびオプションでモデルバージョンを定義します。ディテクター内で定義された任意のルールをディテクターバージョンに追加できます。また、評価されたイベントタイプでトレーニングされたモデルを追加することもできます。

ディテクターの各バージョンのステータスは、DRAFT、ACTIVE、または INACTIVE です。一度に 1 つのディテクターバージョンのみが ACTIVE ステータスになることができます。GetEventPrediction リクエスト中、DetectorVersion が指定されていない場合、Amazon Fraud Detector は ACTIVE ディテクターを使用します。

## ルール実行モード

Amazon Fraud Detector は、FIRST_MATCHED と ALL_MATCHED の 2 つの異なるルール実行モードをサポートしています。

- ルール実行モードが FIRST_MATCHED の場合、Amazon Fraud Detector はルールを最初から最後の順で順番に評価し、最初に一致したルールで停止します。Amazon Fraud Detector は、その 1 つのルールの結果を示します。ルールが false ( 一致しない ) と評価されると、リスト内の次のルールが評価されます。
- ルール実行モードが ALL_MATCHED である場合、順序に関係なく、評価内のすべてのルールが並行して実行されます。Amazon Fraud Detector はすべてのルールを実行し、一致するルールごとに定義された結果を返します。

# を使用してディテクターバージョンを作成する AWS SDK for Python (Boto3)

次の例は、CreateDetectorVersion API のサンプルリクエストを示しています。ルール実行モードが FIRST_MATCHED に設定されていると、Amazon Fraud Detector はルールを最初から最後の順で順番に評価し、最初に一致したルールで停止します。Amazon Fraud Detector は、GetEventPrediction response 中、その 1 つのルールの結果を示します。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.create_detector_version(
detectorId = 'sample_detector',
rules = [{
    'detectorId' : 'sample_detector',
    'ruleId' : 'high_fraud_risk',
    'ruleVersion' : '1'
},
{
    'detectorId' : 'sample_detector',
    'ruleId' : 'medium_fraud_risk',
    'ruleVersion' : '1'
},
{
    'detectorId' : 'sample_detector',
    'ruleId' : 'low_fraud_risk',
    'ruleVersion' : '1'
}
],
modelVersions = [{
    'modelId' : 'sample_fraud_detection_model',
    'modelType': 'ONLINE_FRAUD_INSIGHTS',
    'modelVersionNumber' : '1.00'
}],
ruleExecutionMode = 'FIRST_MATCHED'
)
```

ディテクターバージョンのステータスを更新するには、UpdateDetectorVersionStatus API を使用します。次の例では、ディテクターバージョンのステータスを DRAFT から ACTIVE に更新します。GetEventPrediction リクエスト中、ディテクター ID が指定されていない場合、Amazon Fraud Detector は ACTIVE バージョンのディテクターを使用します。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.update_detector_version_status(
detectorId = 'sample_detector',
detectorVersionId = '1',
status = 'ACTIVE'
)
```

# ディテクター、ディテクターバージョン、ルールバージョンの削除

Amazon Fraud Detector のディテクターを削除する前に、ディテクターに関連付けられたすべてのディテクターバージョンとルールバージョンを削除する必要があります。

ディテクター、ディテクターバージョン、またはルールバージョンを削除すると、Amazon Fraud Detector はそのリソースを完全に削除し、データは Amazon Fraud Detector に保存されなくなります。

ディテクターバージョンを削除するには

ステータスが DRAFT または INACTIVE のディテクターバージョンのみ削除できます。

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/frauddetector で Amazon Fraud Detector コンソールを開きます。
2. Amazon Fraud Detector コンソールの左のナビゲーションペインで、[ディテクター] をクリックします。
3. 削除するディテクターバージョンを含むディテクターを選択します。
4. 削除するディテクターバージョンを選択します。
5. [アクション] を選択してから、[削除] をクリックします。
6. 「**delete**」と入力し、[ディテクターの削除] を選択します。

実行バージョンを削除するには

ルールバージョンの削除は、ACTIVE または INACTIVE ディテクターバージョンでは使用されていない場合にのみ行えます。必要に応じて、ルールバージョンを削除する前に、最初に ACTIVE ディテクターバージョンを INACTIVE に移動させてから、INACTIVE ディテクターバージョンを削除します。

1. Amazon Fraud Detector コンソールの左のナビゲーションペインで、[ディテクター] をクリックします。
2. 削除するルールバージョンを含むディテクターを選択します。
3. [関連付けられたルール] タブをクリックし、削除するルールを選択します。
4. 削除するルールバージョンを選択します。
5. [アクション] を選択し、[ルールバージョンの削除] を選択します。
6. 「**delete**」と入力してから、[バージョンの削除] を選択します。

### ディテクターを削除するには

ディテクターを削除する前に、ディテクターに関連付けられたすべてのディテクターバージョンとルールバージョンを削除する必要があります。

1. Amazon Fraud Detector コンソールの左のナビゲーションペインで、[ディテクター] をクリックします。
2. 削除するディテクターを選択します。
3. [アクション] を選択して、[ディテクターの削除] をクリックします。
4. 「**delete**」と入力し、[ディテクターの削除] を選択します。

# リソース

モデル、ルール、ディテクターは、変数、結果、ラベル、リスト、エンティティなどのリソースを使用して、不正リスクのイベントを評価します。このセクションでは、 リソースの作成と管理について説明します。

トピック


- 変数
- ラベル
- ルール
- Lists
- 結果
- エンティティ
- を使用して Amazon Fraud Detector リソースを管理する AWS CloudFormation


## 変数

変数は、不正予測で使用するデータ要素を表します。これらの変数は、モデルのトレーニング用に準備したイベントデータセット、Amazon Fraud Detector モデルのリスクスコア出力、または Amazon SageMaker AI モデルから取得できます。イベントデータセットから取得される変数の詳細については、「」を参照してくださいデータモデルエクスプローラーを使用してイベントデータセットの要件を取得する。

不正予測で使用する変数は、まず作成してから、イベントタイプを作成するときにイベントに追加する必要があります。作成する各変数には、データ型、デフォルト値、およびオプションで変数型を割り当てる必要があります。Amazon Fraud Detector は、IP アドレス、銀行識別番号 (BINs)、電話番号など、指定した変数の一部を強化して、追加の入力を作成し、これらの変数を使用するモデルのパフォーマンスを向上させます。

### データ型

変数には、変数が表すデータ要素のデータ型が必要です。オプションで、事前定義された のいずれかを割り当てることができます変数タイプ。変数タイプに割り当てられた変数の場合、データ型が事前に選択されています。可能なデータ型には、次の型が含まれます。

| データ型 | 説明 | デフォルト値 | 値の例 |
| --- | --- | --- | --- |
| String | 文字、整数、またはその両方の任意の組み合わせ | <空> | abc、123、1D3B |
| 整数 | 正または負の整数 | 0 | 1、-1 |
| ブール値 | True または False | False | True、False |
| DateTime | ISO 8601 標準 UTC 形式で指定された日付と時刻のみ | <空> | 2019-11-30T13:01:01Z |
| 浮動小数点数 | 小数点のある数字 | 0.0 | 4.01、0.10 |

## デフォルト値

変数にはデフォルト値が必要です。Amazon Fraud Detector が不正予測を生成する場合、Amazon Fraud Detector が変数の値を受信しない場合、このデフォルト値を使用してルールまたはモデルを実行します。指定するデフォルト値は、選択したデータ型と一致する必要があります。AWS コンソールでは、Amazon Fraud Detector は整数`0`に のデフォルト値を、ブール値`false`に を、浮動小数点に を、文字列に (空) `0.0` を割り当てます。これらのデータ型のいずれかにカスタムデフォルト値を設定できます。

## 変数タイプ

変数を作成するときに、オプションで変数を変数タイプに割り当てることができます。変数タイプは、モデルのトレーニングや不正予測の生成に使用される一般的なデータ要素を表します。モデルトレーニングに使用できるのは、関連する変数タイプを持つ変数のみです。モデルトレーニングプロセスの一環として、Amazon Fraud Detector は変数に関連付けられた変数タイプを使用して、変数エンリッチメント、特徴量エンジニアリング、リスクスコアリングを実行します。

Amazon Fraud Detector には、変数に割り当てるために使用できる以下の変数タイプが事前に定義されています。

| カテゴリ | 変数タイプ | 説明 | データ型 | 例 |
| --- | --- | --- | --- | --- |
| セッション | IP_ADDRESS | イベント中に収集される IP アドレス | String | 192.0.2.0<br><br>注：Amazon Fraud Detector はこのデータを充実させます。詳細については、「位置情報エン |

| カテゴリ | 変数タイプ | 説明 | データ型 | 例 |
| --- | --- | --- | --- | --- |
| | | | | リッチメントト」を参照してください |
| | USERAGEN | イベント中に収集されるユーザーエージェント | Strin | Mozilla 5.0 (Windows NT 10.0、Win64、x64、rv:68.0)™ 20100101 |
| | FINGERPRINT | イベントに使用されるデバイスの一意の識別子 | Strin | sadfow987u234 |
| | SESSION_ID | イベントのアクティブなセッションのセッション ID | Strin | sid123456789 |

| カテゴリ | 変数タイプ | 説明 | データ型 | 例 |
| --- | --- | --- | --- | --- |
| | are_CREDE NTIALS_VA LID | イベントログインに使用される認証情報が有効かどうかを示します | ブール値 | 真 |
| ユザ | EMAIL_ADI RESS | イベント中に収集された E メールアドレス | Strin | abc@domai n.com |

| カテゴリ | 変数タイプ | 説明 | データ型 | 例 |
| --- | --- | --- | --- | --- |
| | PHONE_NUMBER | イベント中に収集された電話番号 | String | +1 555-0100<br><br>注：Amazon Fraud Detector はこのデータを充実させます。詳細については、「電話番号エン |

| カテゴリ | 変数タイプ | 説明 | データ型 | 例 |
| --- | --- | --- | --- | --- |
| | | | | リッチメント」を参照してください |
| 「請求 | BILLING_NAME | 請求先住所に関連付けられている名前 | Strin | John Doe |

| カテゴリ | 変数タイプ | 説明 | データ型 | 例 |
| --- | --- | --- | --- | --- |
| | BILLING_PHONE | 請求先住所に関連付けられている電話番号 | Strin | +1 555-0100<br><br>注：Amazon Fraud Detector はこのデータを充実させます。詳細については、「電話番号エン |

| カテゴリ | 変数タイプ | 説明 | データ型 | 例 |
| --- | --- | --- | --- | --- |
| | | | | リッチメント」を参照してください |
| | BILLING_ADDRESS_L | 請求先住所の最初の行 | Strin | 任意の通り |
| | BILLING_ADDRESS_L | 請求先住所の 2 行目 | Strin | 任意のユニット 123 |

| カテゴリ | 変数タイプ | 説明 | データ型 | 例 |
| --- | --- | --- | --- | --- |
| | BILLING_CITY | 請求先住所の市区町村 | Strin | 任意の市区町村 |
| | BILLING_STATE | 請求先住所にある州または都道府県 | Strin | 任意の州または県 |

| カテゴリ | 変数タイプ | 説明 | データ型 | 例 |
| --- | --- | --- | --- | --- |
| | BILLING_COUNTRY | 請求先住所の国 | Strin | 任意の国<br><br>注：Amazon Fraud Detector はこのデータを充実させます。詳細については、「位置情報 |

| カテゴリ | 変数タイプ | 説明 | データ型 | 例 |
| --- | --- | --- | --- | --- |
| | | | | エンリッチメント」を参照してください |

| カテゴリ | 変数タイプ | 説明 | データ型 | 例 |
| --- | --- | --- | --- | --- |
| | BILLING_ZIP | 請求先住所にある郵便番号 | Strin | 01234<br><br>注：Amazon Fraud Detector はこのデータを充実させます。詳細については、「位置情報エンリッ |

| カテゴリ | 変数タイプ | 説明 | データ型 | 例 |
| --- | --- | --- | --- | --- |
| | | | | チメント」を参照してください |
| 配送 | SHIPPING_NAME | 配送先住所に関連付けられている名前 | Strin | John Doe |

| カテゴリ | 変数タイプ | 説明 | データ型 | 例 |
| --- | --- | --- | --- | --- |
| | SHIPPING_PHONE | 配送先住所に関連付けられている電話番号 | Strin | +1 555-0100<br><br>注：Amazon Fraud Detector はこのデータを充実させます。詳細については、「電話番号エン |

| カテゴリ | 変数タイプ | 説明 | データ型 | 例 |
| --- | --- | --- | --- | --- |
| | | | | リッチメント」を参照してください |
| | SHIPPING_ADDRESS_1 | 配送先住所の最初の行 | Strin | 123 Any Street |
| | SHIPPING_ADDRESS_2 | 配送先住所の 2 行目 | Strin | Unit 123 |
| | SHIPPING_CITY | 配送先住所の市区町村 | Strin | 任意の市区町村 |

| カテゴリ | 変数タイプ | 説明 | データ型 | 例 |
| --- | --- | --- | --- | --- |
| | SHIPPING_STATE | 配送先住所の州または都道府県 | Strin | 任意の状態 |

| カテゴリ | 変数タイプ | 説明 | データ型 | 例 |
| --- | --- | --- | --- | --- |
| | SHIPPING_COUNTRY | 配送先住所がある の国 | Strin | 任意の国<br><br>注：Amazon Fraud Detector はこのデータを充実させます。詳細については、「位置情報 |

| カテゴリ | 変数タイプ | 説明 | データ型 | 例 |
| --- | --- | --- | --- | --- |
| | | | | エンリッチメント」を参照してください |

| カテゴリ | 変数タイプ | 説明 | データ型 | 例 |
| --- | --- | --- | --- | --- |
| | SHIPPING_ZIP | 配送先住所にある郵便番号 | Strin | 01234<br><br>注：Amazon Fraud Detector はこのデータを充実させます。詳細については、「位置情報エンリッ |

| カテゴリ | 変数タイプ | 説明 | データ型 | 例 |
| --- | --- | --- | --- | --- |
| | | | | チメント」を参照してください |
| Payment | ORDER_ID | トランザクションの一意の識別子 | Strin | LUX60 |
| | 料金 | 合計注文価格 | Strin | 560.00 |
| | CURRENCY | ISO 4217 通貨コード | Strin | USD |
| | PAYMENT_TYPE | イベント中の支払いに使用される支払い方法 | Strin | クレジットカード |

| カテゴリ | 変数タイプ | 説明 | データ型 | 例 |
|---|---|---|---|---|
| | AUTH_COD | クレジットカード発行者または発行銀行によって送信される英数字コード | Strin | 0000 |
| | AVS | カードプロセッサからのアドレス検証システム (AVS) レスポンスコード | Strin | Y |
| 製品 | PRODUCT_ATEGORY | 注文項目の製品カテゴリ | Strin | キッチン |
| カスタム | NUMERIC | 実数として表現できる任意の変数 | 浮動小数点数 | 1.224 |
| | CATEGORIAL | カテゴリ、セグメント、またはグループを記述する変数 | Strin | Large |

| カテゴリ | 変数タイプ | 説明 | データ型 | 例 |
| --- | --- | --- | --- | --- |
| | FREE_FORM_TEXT | イベントの一部としてキャプチャされた自由形式のテキスト (顧客レビューやコメントなど ) | String | フリーフォームテキスト入力の例 |

## 変数タイプへの変数の割り当て

モデルのトレーニングに変数を使用する予定の場合は、変数に割り当てる適切な変数タイプを選択することが重要です。変数タイプの割り当てが正しくないと、モデルのパフォーマンスに悪影響を及ぼす可能性があります。また、特に複数のモデルやイベントで 変数が使用されている場合、後で割り当てを変更するのが非常に難しくなる可能性があります。

変数は、事前定義された変数タイプのいずれか、または FREE_FORM_TEXT、、 CATEGORICALのいずれかのカスタム変数タイプに割り当てることができますNUMERIC。

適切な変数タイプに変数を割り当てるための重要な注意事項

1. 変数が事前定義された変数タイプのいずれかと一致する場合は、それを使用します。変数タイプが 変数に対応していることを確認します。例えば、ip_address 変数をEMAIL_ADDRESS変数型に割り当てると、ip_address 変数は ASN、ISP、geo-location、リスクスコアなどのエンリッチメントで強化されません。詳細については、「変数エンリッチメント」を参照してください。
2. 変数が事前定義された変数タイプのいずれとも一致しない場合は、以下に示す推奨事項に従って、カスタム変数タイプのいずれかを割り当てます。

3. 通常、自然な順序がなく、カテゴリ、セグメント、またはグループに配置できる変数に変数CATEGORICALタイプを割り当てます。モデルのトレーニングに使用しているデータセットには、 merchant_id、 campaign_id、policy_id などの ID 変数が含まれている場合があります。これらの変数はグループを表します (たとえば、同じ policy_id を持つすべての顧客がグループを表します ) 。次のデータを持つ変数には、カテゴリ変数タイプ - を割り当てる必要があります。

   - customer_ID、segment_ID、color_ID、Department_code、product_ID などのデータを含む変数。
   - true、false、または null 値を持つブールデータを含む変数。
   - 会社名、製品カテゴリ、カードタイプ、紹介メディアなど、グループまたはカテゴリに含めることができる変数。

   > **Note**
   >
   > ENTITY_ID は、Amazon Fraud Detector が ENTITY_ID 変数に割り当てるために使用される予約変数タイプです。ENTITY_ID 変数は、評価するアクションを開始するエンティティの ID です。トランザクション不正インサイト (TFI) モデルタイプを作成する場合は、ENTITY_ID 変数を指定する必要があります。データ内のどの変数がアクションを開始するエンティティを一意に識別し、それを ENTITY_ID 変数として渡すかを決定する必要があります。カテゴリ変数タイプは、データセット内の他のすべての IDs が存在し、モデルトレーニングに使用している場合に割り当てます。データセット内のエンティティではない他の IDs の例は、 merchant_ID、policy_ID、および campaign_ID です。

4. テキストのブロックを含むFREE_FORM_TEXT変数に変数タイプを割り当てます。="_FORM_TEXT 変数タイプの例には、ユーザーレビュー、コメント、日付、紹介コードなどがあります。":"_FORM_TEXT データには、区切り文字で区切られた複数のトークンが含まれています。区切り文字は、英数字とアンダースコア記号以外の任意の文字にすることができます。たとえば、ユーザーレビューとコメントは「スペース」区切り文字で区切ることができます。日付と紹介コードでは、区切り文字としてハイフンを使用して、プレフィックス、サフィックス、中間部分を区切ることができます。Amazon Fraud Detector は、区切り文字を使用して、™_FORM_TEXT 変数からデータを抽出します。

5. NUMERIC 変数タイプは、実数で固有の順序を持つ変数に割り当てます。NUMERIC 変数の例には、day_of_the_week、incident_severity、customer_rating などがあります。これらの変数にはカテゴリ変数タイプを割り当てることができますが、固有の順序を持つすべての実数変数を NUMERIC 変数タイプに割り当てることを強くお勧めします。

# 変数エンリッチメント

Amazon Fraud Detector は、IP アドレス、銀行識別番号 (BINs)、電話番号など、指定した未加工のデータ要素の一部を強化して、追加の入力を作成し、これらのデータ要素を使用するモデルのパフォーマンスを向上させます。エンリッチメントは、潜在的に疑わしい状況を特定し、モデルがより多くの不正をキャプチャするのに役立ちます。

## 電話番号エンリッチメント

Amazon Fraud Detector は、位置情報、元のキャリア、電話番号の有効性に関連する追加情報で電話番号データを強化します。電話番号エンリッチメントは、2021 年 12 月 13 日以降にトレーニングされ、国コード (+xxx) を含む電話番号を持つすべてのモデルで自動的に有効になります。モデルに電話番号変数を含め、2021 年 12 月 13 日より前にトレーニングした場合は、このエンリッチメントを活用できるようにモデルを再トレーニングしてください。

電話番号変数に次の形式を使用して、データが正常に強化されるようにすることを強くお勧めします。

| 変数 | 形式 | 説明 |
| --- | --- | --- |
| PHONE_NUMBER | E.164 標準 | 電話番号に国コード (+xxx) を必ず含めてください。 |
| BILLING_PHONE と ShipPING_PHONE | E.164 標準 | 電話番号に国コード (+xxx) を必ず含めてください。 |

## 位置情報エンリッチメント

2022 年 2 月 8 日以降、Amazon Fraud Detector は、イベントに指定した IP_ADDRESS、BILLING_ZIP、および ShipPING_ZIP の値の間の物理的な距離を計算します。計算された距離は、不正検出モデルへの入力として使用されます。

位置情報エンリッチメントを有効にするには、イベントデータに IP_ADDRESS、BILLING_ZIP、または ShipPING_ZIP の 3 つの変数のうち少なくとも 2 つが含まれている必要があります。さらに、各 BILLING_ZIP および ShipPING_ZIP 値には、それぞれ有効な BILLING_COUNTRY コードと

ShipPING_COUNTRY コードが必要です。2022 年 2 月 8 日より前にトレーニングされたモデルがあり、これらの変数が含まれている場合は、モデルを再トレーニングして位置情報エンリッチメントを有効にする必要があります。

データが無効であるために、Amazon Fraud Detector がイベントの IP_ADDRESS、BILLING_ZIP、または ShipPING_ZIP 値に関連付けられている場所を特定できない場合、代わりに特別なプレースホルダー値が使用されます。例えば、イベントに有効な IP_ADDRESS 値と BILLING_ZIP 値があるが、ShipPING_ZIP 値が無効であるとします。この場合、エンリッチメントは IP_ADDRESS–> BILLING_ZIP に対してのみ行われます。エンリッチメントは、IP_ADDRESS–>ShipPING_ZIP および BILLING_ZIP–>ShipPING_ZIP では行われません。代わりに、プレースホルダー値が代わりに使用されます。モデルに対して位置情報エンリッチメントが有効になっているかどうかにかかわらず、モデルのパフォーマンスは変わりません。

BILLING_ZIP 変数と ShipPING_ZIP 変数を CUSTOM_CATEGORICAL 変数タイプにマッピングすることで、位置情報エンリッチメントをオプトアウトできます。変数タイプを変更しても、モデルのパフォーマンスには影響しません。

位置情報変数の形式

位置情報データが正常に強化されるように、位置情報変数には次の形式を使用することを強くお勧めします。

| 変数 | 形式 | 説明 |
| --- | --- | --- |
| IP_ADDRESS | IPv4 アドレス | 例 - 1.1.1.1 |
| BILLING_ZIP および ShipPING_ZIP | 指定された国の ISO 3166-1 alpha-2 郵便番号 | 詳細については、このトピックの「国と地域のコード」セクションを参照してください。 |
| BILLING_COUNTRY と ShipPING_COUNTRY | ISO 3166-1 alpha-2 2 文字の標準国コード | 詳細については、このトピックの「国と地域のコード」セクションを参照してください。Amazon Fraud Detector は、国名の一般的なバリエーションをすべて ISO 3166-1 |

| 変数 | 形式 | 説明 |
| --- | --- | --- |
| | | 2 文字の標準国コードと一致させようとします。ただし、それらが正しく一致することを保証することはできません。 |

国と地域のコード

次の表は、Amazon Fraud Detector で位置情報エンリッチメントがサポートされている国と地域の詳細なリストです。各国と地域には、国コード (特に ISO 3166-1 alpha-2 2 文字の国コード) と郵便番号が割り当てられます。

郵便番号形式

- 9 - 数値
- a - 文字
- [X] - X はオプションです。例えば、ガースニーGY9[9] 9aa」は、GY9 9aa」とGY99 9aa」の両方が有効であることを意味します。1 つの形式を使用します。
- [X/XX] - X または XX を使用できます。例えば、「aa[aa/99]」とは、「aa aa」と「aa 99」の両方が有効であることを意味します。これらの形式のいずれかを使用しますが、両方を使用しないでください。
- 一部の国では、プレフィックスが固定されています。例えば、Andorra の郵便番号は AD999 です。つまり、国コードは AD 文字で始まり、その後に 3 つの数字が続く必要があります。

| コード | 名前 | 郵便番号 |
| --- | --- | --- |
| AD | アンドラ | AD999 |
| AR | オランダ領アンティル | 9999 |
| AT | オーストリア | 9999 |
| AU | オーストラリア | 9999 |

| コード | 名前 | 郵便番号 |
| --- | --- | --- |
| AZ | アゼルバイジャン | AZ 9999 |
| BD | バングラデシュ | 9999 |
| BE | ベルギー | 9999 |
| BG | ブルガリア | 9999 |
| BM | バミューダ | aa[aa/99] |
| BY | ベラルーシ | 999999 |
| CA | カナダ | a9a 9a9 |
| CH | スイス | 9999 |
| CL | チリ | 9999999 |
| CO | コロンビア | 999999 |
| CR | コスタリカ | 99999 |
| CY | キプロス | 9999 |
| CZ | チェコ共和国 | 999 99 |
| DE | ドイツ | 99999 |
| DK | デンマーク | 9999 |
| DO | ドミニカ共和国 | 99999 |
| DZ | アルジェリア | 99999 |
| EE | エストニア | 99999 |
| ES | スペイン | 99999 |
| FI | フィンランド | 99999 |

| コード | 名前 | 郵便番号 |
| --- | --- | --- |
| FM | ミクロネシア連邦 | 99999 |
| FO | フェロー諸島 | 999 |
| FR | フランス | 99999 |
| GB | 英国 | a[a]9[a/9] 9aa |
| GG | ガーンジー代官管轄区 | GY9[9] 9aa |
| GL | グリーンランド | 9999 |
| GP | グアドループ | 99999 |
| GT | グアテマラ | 99999 |
| GU | グアム | 99999 |
| HR | クロアチア | 99999 |
| hu | ハンガリー | 9999 |
| IE | アイルランド | a99[a/9][a/9][a/9][a/9] |
| IM | マン島 | IM9[9]9aa |
| IN | インド | 999999 |
| IS | アイスランド | 999 |
| IT | イタリア | 99999 |
| JE | ジャージー | JE9[9]9aa |
| JP | 日本 | 999-9999 |
| KR | 韓国 | 99999 |
| LI | リヒテンシュタイン | 9999 |

| コード | 名前 | 郵便番号 |
| --- | --- | --- |
| LK | スリランカ | 99999 |
| LT | リトアニア | 99999 |
| LU | ルクセンブルグ | L-9999 |
| LV | ラトビア | LV-9999 |
| MC | モナコ | 99999 |
| MD | ピドロバ共和国 | 9999 |
| MH | マーシャル諸島共和国 | 99999 |
| MK | 北マケドニア | 9999 |
| MP | 北マリアナ諸島 | 99999 |
| MQ | マティニーク | 99999 |
| MT | マルタ | aaa 9999 |
| MX | メキシコ | 99999 |
| MY | マレーシア | 99999 |
| NL | オランダ | 9999 aa |
| いいえ | ノルウェー | 9999 |
| NZ | ニュージーランド | 9999 |
| PH | フィリピン | 9999 |
| PK | パキスタン | 99999 |
| PL | ポーランド | 99-999 |
| PR | プエルトリコ | 99999 |

| コード | 名前 | 郵便番号 |
| --- | --- | --- |
| PT | ポルトガル | 9999-999 |
| PW | パラオ | 99999 |
| RE | レユニオン | 99999 |
| RO | ルーマニア | 999999 |
| RU | ロシア連邦 | 999999 |
| SE | スウェーデン | 999 99 |
| SG | シンガポール | 999999 |
| SI | スロベニア | 9999 |
| SK | スロバキア | 999 99 |
| SM | サンマリノ | 99999 |
| TH | タイ | 99999 |
| TR | トルコ | 99999 |
| UA | ウクライナ | 99999 |
| 米国 | アメリカ | 99999 |
| UY | ウルグアイ | 99999 |
| VI | 米領バージン諸島 | 99999 |
| WF | ウォリス・フツナ | 99999 |
| YT | マヨット | 99999 |
| ZA | 南アフリカ | 9999 |

### Useragent エンリッチメント

Account Takeover Insights (ATI) モデルを作成する場合は、データセットに変数タイプのuseragent変数を指定する必要があります。この変数には、ログインイベントのブラウザ、デバイス、OS データが含まれます。Amazon Fraud Detector は、、、OS_family、user_agent_family などの追加情報でユーザーエージェントデータを強化しますdevice_family。

## 変数の作成

Amazon Fraud Detector コンソール、create-variable コマンド、CreateVariable、または を使用して変数を作成できます。 AWS SDK for Python (Boto3)

### Amazon Fraud Detector コンソールを使用して変数を作成する

この例では、 email_addressと の 2 つの変数を作成しip_address、対応する変数タイプ (EMAIL_ADDRESS と ) に割り当てますIP_ADDRESS。これらの変数は例として使用されます。モデルトレーニングに使用する変数を作成する場合は、ユースケースに適したデータセットの変数を使用します。変数を作成する変数エンリッチメント前に、 変数タイプと について必ずお読みください。

変数を作成するには、

1. AWS マネジメントコンソールを開き、アカウントにサインインします。
2. Amazon Fraud Detector に移動し、左側のナビゲーションで変数を選択し、作成を選択します。
3. 新しい変数ページで、変数名email_addressとして を入力します。必要に応じて、変数の説明を入力します。
4. 変数タイプで、E メールアドレスを選択します。
5. この変数タイプは事前定義されているため、Amazon Fraud Detector は、この変数タイプのデータ型を自動的に選択します。変数に変数タイプが自動的に割り当てられていない場合は、リストから変数タイプを選択します。詳細については、「変数タイプ」を参照してください。
6. 変数のデフォルト値を指定する場合は、カスタムデフォルト値を定義 を選択し、変数のデフォルト値を入力します。この例に従っている場合は、このステップをスキップします。
7. [Create] (作成) を選択します。
8. email_address 概要ページで、先ほど作成した変数の詳細を確認します。

   更新する必要がある場合は、編集を選択して更新を指定します。[Save changes] (変更の保存) をクリックします。

9. このプロセスを繰り返して別の変数を作成しip_address、変数タイプの IP アドレスを選択します。
10. 変数ページには、新しく作成された変数が表示されます。

> **Important**
>
> データセットから必要な数の変数を作成することをお勧めします。後でイベントタイプを作成するときに、モデルをトレーニングして不正を検出し、不正検出を生成するために含める変数を決定できます。

## を使用して変数を作成する AWS SDK for Python (Boto3)

次の例は、CreateVariable API のリクエストを示しています。この例では、email_address と ip_address の 2 つの変数を作成し、それらを対応する変数型 (EMAIL_ADDRESS および IP_ADDRESS) に割り当てます。

これらの変数は例として使用されます。モデルトレーニングに使用する変数を作成する場合は、ユースケースに適したデータセットの変数を使用します。変数を作成する変数エンリッチメント前に、変数タイプと について必ずお読みください。

必ず変数ソースを指定してください。変数値がどこから派生するかを特定するのに役立ちます。変数ソースが EVENT の場合、変数値は GetEventPrediction リクエストの一部として送信されます。変数値が の場合MODEL_SCORE、Amazon Fraud Detector によって入力されます。の場合EXTERNAL_MODEL_SCORE、変数値はインポートされた SageMaker AI モデルによって入力されます。

```
import boto3
fraudDetector = boto3.client('frauddetector')

 #Create variable email_address
   fraudDetector.create_variable(
     name = 'email_address',
     variableType = 'EMAIL_ADDRESS',
     dataSource = 'EVENT',
     dataType = 'STRING',
     defaultValue = '<unknown>'
     )
```

```
#Create variable ip_address
   fraudDetector.create_variable(
     name = 'ip_address',
     variableType = 'IP_ADDRESS',
     dataSource = 'EVENT',
     dataType = 'STRING',
     defaultValue = '<unknown>'
     )
```

## 変数の削除

変数を削除すると、Amazon Fraud Detector はその変数を完全に削除し、データは Amazon Fraud Detector に保存されなくなります。

Amazon Fraud Detector のイベントタイプに含まれる変数は削除できません。まず、変数が関連付けられているイベントタイプを削除してから、変数を削除する必要があります。

Amazon Fraud Detector モデル出力変数と SageMaker AI モデル出力変数を手動で削除することはできません。Amazon Fraud Detector は、モデルを削除するとモデル出力変数を自動的に削除します。

Amazon Fraud Detector コンソールで、delete-variable CLI コマンド、DeleteVariable API、または を使用して変数を削除できます。 AWS SDK for Python (Boto3)

### コンソールを使用して変数を削除する

変数を削除するには、

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/frauddetector で Amazon Fraud Detector コンソールを開きます。
2. Amazon Fraud Detector コンソールの左側のナビゲーションペインで、[リソース] を選択してから、[変数] をクリックします。
3. 削除する変数を選択します。
4. [アクション] を選択し、[削除] を選択します。
5. 変数名を入力してから、[変数の削除] を選択します。

### を使用して変数を削除する AWS SDK for Python (Boto3)

次のコードサンプルは、DeleteVariable API を使用して変数 customer_name を削除します。DeleteVariable

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.delete_variable (

name = 'customer_name'

)
```

# ラベル

ラベルは、イベントを不正または正当として分類します。ラベルはイベントタイプに関連付けられ、Amazon Fraud Detector で機械学習モデルをトレーニングするために使用されます。オンライン不正インサイト (OFI) モデルまたはトランザクション不正インサイト (TFI) モデルのいずれかをトレーニングする予定の場合、トレーニングデータセット内の少なくとも 400 個のイベントを、細分性または正当のいずれかに分類する必要があります。トレーニングデータセット内のイベントを分類するには、不正、正当性、1、0 などの任意のラベルを使用できます。トレーニングが完了すると、トレーニングされたモデルはイベントを不正と評価し、これらの値を使用してイベントを不正または正当と分類します。

まず、トレーニングデータセットで使用される値でラベルを作成し、不正検出モデルの構築とトレーニングに使用されるイベントタイプにラベルを関連付ける必要があります。

## ラベルの作成

Amazon Fraud Detector コンソール、put-label コマンド、PutLabel API、または を使用してラベルを作成できます AWS SDK for Python (Boto3)。

### Amazon Fraud Detector コンソールを使用してラベルを作成する

ラベルを作成するには、

1. AWS マネジメントコンソールを開き、アカウントにサインインします。
2. Amazon Fraud Detector に移動し、左側のナビゲーションでラベルを選択し、作成を選択します。
3. 「ラベルの作成」ページで、不正イベントのラベル名をラベル名として入力します。ラベル名は、トレーニングデータセット内の不正行為を表すラベルに対応している必要があります。必要に応じて、ラベルの説明を入力します。

4. [ラベルの作成] を選択します。
5. 2 番目のラベルを作成し、正当なイベントのラベル名を入力します。ラベル名がトレーニングデータセットの正当なアクティビティを表す値に対応していることを確認します。

### AWS SDK for Python (Boto3)を使用したラベルの作成

次のコード AWS SDK for Python (Boto3) 例では、PutLabel API を使用して 2 つのラベル (不正、正当) を作成します。ラベルを作成したら、イベントタイプに追加して特定のイベントを分類できます。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.put_label(
name = 'fraud',
description = 'label for fraud events'
)

fraudDetector.put_label(
name = 'legit',
description = 'label for legitimate events'
)
```

## ラベルの更新

イベントデータセットが Amazon Fraud Detector に保存されている場合は、イベントのオフライン不正調査を実行して機械学習フィードバックループを閉じる場合など、保存されたイベントのラベルを追加または更新する必要がある場合があります。

保存されたイベントのラベルを追加または更新するには、update-event-label コマンド、UpdateEventLabel API、または を使用します。 AWS SDK for Python (Boto3)

次のコード AWS SDK for Python (Boto3) 例では、 `UpdateEventLabel` API を使用したイベントタイプの登録に関連付けられたラベル不正を追加します。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.update_event_label(
            eventId        = '802454d3-f7d8-482d-97e8-c4b6db9a0428',
```

```
            eventTypeName  = 'registration',
            assignedLabel  = 'fraud',
            labelTimestamp = '2020-07-13T23:18:21Z'
)
```

## Amazon Fraud Detector に保存されているイベントデータのイベントラベルの更新

イベントのオフライン不正調査を実行し、機械学習フィードバックループを閉じる場合など、Amazon Fraud Detector に既に保存されているイベントの不正ラベルを追加または更新する必要がある場合があります。Amazon Fraud Detector に既に保存されているイベントのラベルを更新するには、`UpdateEventLabel` API オペレーションを使用します。以下は、UpdateEventLabel API コールの例を示しています。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.update_event_label(
            eventId        = '802454d3-f7d8-482d-97e8-c4b6db9a0428',
            eventTypeName  = 'sample_registration',
            assignedLabel  = 'fraud',
            labelTimestamp = '2020-07-13T23:18:21Z'
)
```

## ラベルの削除

ラベルを削除すると、Amazon Fraud Detector はそのラベルを完全に削除し、データは Amazon Fraud Detector に保存されなくなります。

Amazon Fraud Detector のイベントタイプに含まれるラベルは削除できません。また、イベント ID に割り当てられたラベルを削除することもできません。まず、関連するイベント ID を削除する必要があります。

Amazon Fraud Detector コンソール、delete-label コマンド、DeleteLabel API、または を使用してラベルを削除できます。 AWS SDK for Python (Boto3)

### コンソールを使用してラベルを削除する

ラベルを削除するには

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/frauddetector で Amazon Fraud Detector コンソールを開きます。
2. Amazon Fraud Detector コンソールの左側のナビゲーションペインで、[リソース] を選択してから、[ラベル] をクリックします。
3. 削除するラベルを選択します。
4. [アクション] を選択し、[削除] を選択します。
5. ラベル名を入力してから、[ラベルの削除] を選択します。

### を使用してラベルを削除する AWS SDK for Python (Boto3)

次の AWS SDK for Python (Boto3) サンプルコードは、DeleteLabel API を使用してラベル Legit を削除します。 DeleteLabel

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.delete_event_label (
    name = 'legit'
)
```

## ルール

ルールは、不正予測時に変数値の解釈方法を Amazon Fraud Detector に指示する条件です。ルールはディテクターロジックの一部であり、次の要素で構成されます。

- 変数またはリスト – 変数は、不正予測で使用するイベントデータセット内のデータ要素を表します。リストは、イベントデータセット内の変数の入力データ要素のセットです。ルールで使用される変数は、評価されたイベントタイプで事前定義され、ルールで使用されるリストは変数タイプに関連付けられている必要があります。詳細については、変数およびListsを参照してください。
- 式 – ルール内の式はビジネスロジックをキャプチャします。ルールで変数を使用している場合、単純なルール式は、変数、 > 、 < 、<=、>=、== などの比較演算子、および値を使用して構築されます。リストを使用している場合、ルール式はリストエントリ、、inおよびリスト名として作成されます。詳細については、「ルール言語リファレンス」を参照してください。and と or を使用

して、複数の式を組み合わせることができます。すべての式は、ブール値 (true または false) と評価され、長さが 4,000 文字未満である必要があります。if-else 型の条件はサポートされていません。

- 結果 – 結果は、ルールが一致したときに Amazon Fraud Detector によって返されるレスポンスです。結果は、不正予測の結果を示します。考えられる不正予測ごとに結果を作成し、ルールに追加できます。詳細については、「結果」を参照してください。

ディテクターには少なくとも 1 つのルールが必要です。ルールには最大 3 つのリストを含めることができ、ディテクターには最大 30 個のリストを含めることができます。ディテクター作成プロセスの一環としてルールを作成します。新しいルールを作成して既存のディテクターに関連付けることもできます。

# ルール言語リファレンス

次のセクションでは、Amazon Fraud Detector の式 (ルールの書き込み) 機能の概要を説明します。

## 変数の使用

評価されたイベントタイプで定義されている変数は、式の一部として使用できます。変数を示すには、ドル記号を使用します。

```
$example_variable < 100
```

## リストの使用

変数タイプに関連付けられ、ルール式の一部としてエントリが入力されている任意のリストを使用できます。ドル記号を使用して、リストエントリ値を示します。

```
$example_list_variable in @list_name
```

## 比較、メンバーシップ、およびアイデンティティ演算子

Amazon Fraud Detector には、>、>=、<、<=、!=、==、in、not in の比較演算子が含まれています。

次に例を示します。

例: <

```
$variable < 100
```

例: in、not in

```
$variable in [5, 10, 25, 100]
```

例: !=

```
$variable != "US"
```

例: ==

```
$variable == 1000
```

演算子テーブル

| 演算子 | Amazon Fraud Detector 演算子 |
| --- | --- |
| 等しい | == |
| 等しくない | != |
| 超 | > |
| 未満 | < |
| 以上 | >= |
| 以下 | <= |
| 中 | 中 |
| And | and |
| Or | or |
| Not | ! |

## 基本的な数学

式には基本的な数学演算子 (+、-、*、/など) を使用できます。代表的なユースケースには、評価中に変数を組み合わせる必要があるときです。

以下のルールでは、変数 $variable_1 に $variable_2 を追加し、合計が 10 未満かどうかをチェックしています。

```
$variable_1 + $variable_2 < 10
```

基本的な数学テーブルデータ

| 演算子 | Amazon Fraud Detector 演算子 |
| --- | --- |
| + (足し算) | + |
| - (引き算) | - |
| Multiply | * |
| Divide | / |
| モジュロ | % |

## 正規表現 (regex)

正規表現を使用して、式の一部として特定のパターンを検索できます。これは、変数の 1 つに特定の文字列または数値を一致させる場合に特に便利です。Amazon Fraud Detector は、正規表現を使用する場合にのみ一致をサポートします (例えば、指定された文字列が正規表現と一致するかどうかによって True/False を返します)。Amazon Fraud Detector の正規表現のサポートは、Java の.matches() に基づいています (RE2J 正規表現ライブラリを使用)。インターネットには、さまざまな正規表現パターンのテストに役立つウェブサイトがいくつかあります。

以下の最初の例では、まず変数 email を小文字に変換します。次に、パターン @gmail.com が email 変数内にあるかどうかをチェックします。文字列 .com を明示的にチェックできるように、2 番目のピリオドがエスケープされていることに注目してください。

```
regex_match(".*@gmail\.com", lowercase($email))
```

2 番目の例では、変数 phone_number に国コード +1 が含まれているかどうかをチェックして、電話番号が米国からのものかどうかを判断します。文字列 +1 を明示的にチェックできるように、プラス記号がエスケープされています。

```
regex_match(".*\+1", $phone_number)
```

正規表現テーブル

| 演算子 | Amazon Fraud Detector の例 |
| --- | --- |
| 以下で始まる任意の文字列に一致 | regex_match("^mystring", $variable) |
| 文字列全体を正確に一致 | regex_match("mystring", $variable) |
| 改行以外の任意の文字に一致 | regex_match(".", $variable) |
| 改行前の 'mystring' 以外の任意の数の文字に一致 | regex_match(".*mystring", $variable) |
| 特殊文字のエスケープ | \ |

## 欠落している値のチェック

場合によっては、値が欠落しているかどうかをチェックすることが有益です。Amazon Fraud Detector では、これは null で表されます。これを行うには、次の構文を使用します。

```
$variable != null
```

同様に、値が存在しないかどうかをチェックする場合は、以下を行えます。

```
$variable == null
```

## 複数の条件

and と or を使用して、複数の式を組み合わせることができます。Amazon Fraud Detector は、真の値が 1 つ見つかったときに OR 式で停止し、偽の値が 1 つ見つかったときに AND で停止します。

次の例では、and 条件を使用して 2 つの条件をチェックしています。最初のステートメントでは、変数 1 が 100 未満かどうかをチェックしています。2 つ目のステートメントでは、変数 2 が米国ではないかどうかをチェックしています。

ルールは and を使用していることを考えると、条件全体が TRUE と評価されるためには、両方とも TRUE でなければなりません。

```
$variable_1 < 100 and $variable_2 != "US"
```

括弧を使用して、次のようにブール演算をグループ化することができます。

```
$variable_1 < 100 and $variable_2 != "US" or ($variable_1 * 100.0 > $variable_3)
```

## その他の式タイプ

DateTime 関数

| 関数 | 説明 | 例 |
| --- | --- | --- |
| getcurrentdatetime() | ルール実行の現在の時刻を ISO8601 UTC 形式で指定します。getepochmilliseconds(getcurrentdatetime()) を使用して追加のオペレーションを実行できます。 | getcurrentdatetime() == "2023-03-28T18:34:02Z" |
| isbefore(DateTime1、DateTime2) | 発信者 DateTime1 が DateTime2 より前の場合、ブール値 (True/False) を返します。 | isbefore(getcurrentdatetime(), "2019-11-30T01:01:01Z") == "False"<br><br>isbefore(getcurrentdatetime(), "2050-11-30T01:05:01Z") == "True" |
| isafter(DateTime1, DateTime2) | 発信者 DateTime1 が DateTime2 より後の場合、ブール値 (True/False) を返します。 | isafter(getcurrentdatetime(), "2019-11-30T01:01:01Z") == "True"<br><br>isafter(getcurrentdatetime(), "2050-11-30T01:05:01Z") == "False" |
| getepochmilliseconds(DateTime) | DateTime を取得し、その DateTime をエポックミリ秒単位で返します。日付に対して数学演算を実行するのに役立ちます | getepochmilliseconds("2019-11-30T01:01:01Z") == 1575032461 |

文字列演算子

| 演算子 | 例 |
| --- | --- |
| 文字列を大文字に変換 | uppercase($variable) |
| 文字列を小文字に変換 | lowercase($variable) |

その他

| 演算子 | コメント |
| --- | --- |
| コメントを追加する | # 自分のコメント |

## ルールを作成する

Amazon Fraud Detector コンソール、create-rule コマンド、CreateRule API、または を使用してルールを作成できます AWS SDK for Python (Boto3)。

各ルールには、ビジネスロジックをキャプチャする 1 つの式が含まれている必要があります。すべての式は、ブール値 (true または false) と評価され、長さが 4,000 文字未満である必要があります。if-else 型の条件はサポートされていません。式で使用されるすべての変数は、評価されたイベントタイプであらかじめ定義されている必要があります。同様に、式で使用されるすべてのリストは事前定義され、可変型に関連付けられ、エントリが入力されている必要があります。

次の の例では、既存のディテクター `high_risk`のルールを作成します`payments_detector`。ルールは、式と結果をルール`verify_customer`に関連付けます。

前提条件

以下のステップに従うには、ルールの作成に進む前に、必ず以下を完了してください。

- ディテクターの作成
- 結果の作成

ユースケースのディテクター、ルール、結果を作成する場合は、ディテクター名、ルール名、ルール式、結果名の例を、ユースケースに関連する名前と式に置き換えます。

## Amazon Fraud Detector コンソールで新しいルールを作成する

1. AWS マネジメントコンソールを開き、アカウントにサインインします。Amazon Fraud Detector に移動します。
2. 左側のナビゲーションペインで、ディテクター を選択し、ユースケース用に作成したディテクター、たとえば payment_detector を選択します。
3. payment_detector ページで、関連ルールタブを選択し、ルールの作成を選択します。
4. 新しいルールページで、次のように入力します。

    a. 名前にルールの名前を入力します。例: **high_risk**

    b. 説明 - オプションで、オプションでルールの説明を入力します。例: **This rule captures events with a high ML model score**

    c. 式で、式クイックリファレンスガイドを使用してユースケースのルール式を入力します。`$sample_fraud_detection_model_insightscore >900` の例

    d. 結果で、ユースケース用に作成した結果を選択します。例: verify_customer。結果は、不正予測の結果であり、評価中にルールが一致した場合に返されます。
5. 保存ルールを選択する

ディテクターの新しいルールを作成しました。これは、Amazon Fraud Detector が自動的にディテクターで使用できるようにするルールのバージョン 1 です。

## を使用してルールを作成する AWS SDK for Python (Boto3)

次のコード例では、CreateRule API を使用して既存のディテクター `high_risk`のルールを作成します`payments_detector`。サンプルコードでは、ルール式と結果を`verify_customer`ルールに追加します。

前提条件

サンプルコードを使用するには、ルールの作成に進む前に、次の作業を完了していることを確認してください。

- ディテクターの作成
- 結果の作成

ユースケースのディテクター、ルール、結果を作成する場合は、ディテクター名、ルール名、ルール式、結果名の例を、ユースケースに関連する名前と式に置き換えます。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.create_rule(
ruleId = 'high_risk',
detectorId = 'payments_detector',
expression = '$sample_fraud_detection_model_insightscore > 900',
language = 'DETECTORPL',
outcomes = ['verify_customer']
)
```

Amazon Fraud Detector がディテクターで使用できるように自動的にするルールのバージョン 1 を作成しました。

## 更新ルール

ルールの説明を追加または更新するか、ルール式を更新するか、ルールの結果を追加または削除することで、ルールをいつでも更新できます。ルールを更新すると、新しいルールバージョンが作成されます。

Amazon Fraud Detector コンソールでルールを更新するには、update-rule-version コマンドを使用するか、UpdateRuleVersion API を使用するか、 AWS SDK を使用して を使用します。

ルールを更新したら、新しいルールバージョンを使用するようにディテクターバージョンを更新してください。

### Amazon Fraud Detector コンソールでルールを更新する

ルールを更新するには、

1. AWS マネジメントコンソールを開き、アカウントにサインインします。Amazon Fraud Detector に移動します。
2. 左側のナビゲーションペインで、ディテクターを選択します。
3. ディテクターペインで、更新するルールに関連付けられているディテクターを選択します。
4. ディテクターページで、関連ルールタブを選択し、更新するルールを選択します。
5. ルールページで、アクションを選択し、バージョンの作成を選択します。
6. バージョンが変更されたことに注意してください。更新された説明、式、または結果を入力します。
7. 新しいバージョンの保存を選択する

を使用してルールを更新する AWS SDK for Python (Boto3)

次のコード例では、UpdateRuleVersion API を使用してルールのしきい値を 900 high_riskから 950 に更新します。このルールはディテクター に関連付けられていますpayments_detector。

```
fraudDetector.update_rule_version(
rule = {
    'detectorId' : 'payments_detector',
    'ruleId' : 'high_risk',
    'ruleVersion' : '1'
},
expression = '$sample_fraud_detection_model_insightscore > 950',
language = 'DETECTORPL',
outcomes = ['verify_customer']
)
```

# Lists

リストは、イベントデータセット内の変数の入力データのセットです。ディテクターに関連付けられているルールで入力データを使用します。ルールは、不正予測中に入力データを解釈する方法を Amazon Fraud Detector に指示する条件です。例えば、IP アドレスのリストを作成し、特定の IP アドレスがリストにある場合にアクセスを拒否するルールを作成できます。リストを使用するルールは、 形式で表現$ip_address_valueされます@list_name。

Amazon Fraud Detector を使用すると、関連するルールを更新することなく、データを追加または削除することでリストを管理できます。リストに関連付けられたルールには、新しく追加または削除されたデータが自動的に組み込まれます。

リストには最大 100,000 個の一意のエントリを含めることができ、各エントリの長さは最大 320 文字です。ルールで使用するすべてのリストは、デフォルトで Amazon Fraud Detector の変数タイプ":"_FORM_TEXT に関連付けられています。変数タイプはいつでもリストに割り当てることができます。ルールでは最大 3 つのリストを使用できます。

リストの作成、リストへのエントリの追加、リストの削除、リスト内の 1 つ以上のエントリの削除、Amazon Fraud Detector コンソールでのリストへの変数タイプの割り当て、 API の使用、 AWS CLIの使用、または AWS SDK の使用を行うことができます。

# リストを作成する

イベントデータセット内の変数の入力データ (エントリ) を含むリストを作成し、そのリストをルール式で使用できます。リスト内のエントリは、リストを使用しているルールを更新せずに動的に管理できます。

リストを作成するには、まず名前を指定し、オプションでリストを Amazon Fraud Detector で変数タイプサポートされている に関連付ける必要があります。デフォルトでは、Amazon Fraud Detector はリストが™_FORM_TEXT 変数タイプであると見なします。

Amazon Fraud Detector コンソール、 API、、 AWS CLIまたは AWS SDK を使用してリストを作成できます。

## Amazon Fraud Detector コンソールを使用してリストを作成する

リストを作成するには

1. AWS マネジメントコンソールを開き、アカウントにサインインします。Amazon Fraud Detector に移動します。
2. 左側のナビゲーションペインで、リストを選択します。
3. リストの詳細の下

   a. リスト名に、リストの名前を入力します。

   b. 説明 で、オプションで説明を入力します。

   c. ( オプション) 変数タイプで、リストの変数タイプを選択します。

   > **Important**
   >
   > リストに IP アドレスが含まれている場合は、必ず IP_ADDRESS を変数タイプとして選択してください。変数タイプを選択しない場合、Amazon Fraud Detector はリストが™_FORM_TEXT 変数タイプであると見なします。

4. リストデータを追加で、リストエントリを各行に 1 つずつ追加します。スプレッドシートからエントリをコピーして貼り付けることもできます。

> **Note**
>
> エントリがカンマで区切られておらず、リスト内で一意であることを確認してください。2 つの同じエントリを入力すると、1 つのエントリのみが追加されます。

5. [Create] (作成) を選択します。

## を使用してリストを作成する AWS SDK for Python (Boto3)

リストを作成するには、リスト名を指定します。オプションで、リストの作成時に説明を指定したり、変数タイプを関連付けたり、リストにエントリを追加したりできます。または、エントリまたは説明を追加して、後でリストを更新することもできます。リストの作成時に変数タイプを割り当てていない場合は、後で変数タイプをリストに割り当てることができます。リストの変数タイプは、割り当て後に変更することはできません。

> **Important**
>
> リストに IP アドレスが含まれている場合は、必ず IP_ADDRESS を変数タイプとして割り当ててください。変数タイプを割り当てない場合、Amazon Fraud Detector はリストが™_FORM_TEXT 変数タイプであると見なします。

次の例では、CreateList API オペレーションを使用して、`allow_email_ids`説明、変数タイプ、および 4 つのリストエントリを追加して リストを作成します。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.create_list (
     name = 'allow_email_ids',
     description = 'legitimate email_ids'
     variableType = 'EMAIL_ADDRESS',
     elements = ['emailId _1', 'emailId_2', 'emailId_3','emailId_4']
     )
```

# リストにエントリを追加する

リストを作成したら、いつでもリストにエントリを追加または追加できます。リストにエントリを追加または追加する場合、リストが関連付けられているルールを更新する必要はありません。ルールには、新しく追加されたエントリが自動的に組み込まれます。

リストには最大 100,000 個の一意のエントリを含めることができ、各エントリは最大 320 文字です。

Amazon Fraud Detector コンソール、 API、、 AWS CLIまたは AWS SDK を使用してエントリを追加できます。

## Amazon Fraud Detector コンソールを使用してリストにエントリを追加する

リストに 1 つ以上のエントリを追加するには

1. AWS マネジメントコンソールを開き、アカウントにサインインします。Amazon Fraud Detector に移動します。
2. 左側のナビゲーションペインで、リストを選択します。
3. リストページで、エントリを追加するリストを選択します。
4. リストの詳細ページで、データのリストタブを選択し、データの追加を選択します。
5. リストデータの追加ボックスで、各行に 1 つのエントリを追加するか、スプレッドシートからエントリをコピーして貼り付けます。エントリを区切るためにカンマを使用しないでください。
6. [追加] を選択します。

## を使用してリストにエントリを追加する AWS SDK for Python (Boto3)

次の例では、UpdateList API オペレーションを使用して、`allow_email_ids`リストに 2 つの新しいエントリを追加します。追加するエントリがリスト内で一意であることを確認します。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.update_list (
     name = 'allow_email_ids',
     updateMode = 'APPEND'
     elements = ['emailId_11','emailId_12']
```

# 変数タイプをリストに割り当てる

ルールで使用するすべてのリストは、Amazon Fraud Detector の変数タイプ変数タイプに関連付ける必要があります。デフォルトでは、Amazon Fraud Detector はリストが™_FORM_TEXT 変数タイプであると見なします。IP アドレスで構成されるリストは、IP_ADDRESS 変数タイプに関連付ける必要があることに注意してください。

リストの作成時または作成後にいつでも、リストを変数タイプに関連付けることができます。リストを変数タイプに既に関連付けていて、後で変更する場合は、新しいリストを作成する必要があります。リストの変数タイプを変更することはできません。

Amazon Fraud Detector コンソール、 API、、 AWS CLIまたは AWS SDK を使用して変数タイプを割り当てることができます。

## Amazon Fraud Detector コンソールを使用して変数タイプをリストに割り当てる

変数タイプをリストに割り当てるには

1. AWS マネジメントコンソールを開き、アカウントにサインインします。Amazon Fraud Detector に移動します。
2. 左側のナビゲーションペインで、リストを選択します。
3. リストページで、変数タイプを割り当てるリストを選択します。
4. リストの詳細ページで、アクションを選択し、リストの編集を選択します。
5. リストの編集ボックスで、リストの変数タイプを選択します。
6. [Save] を選択します。

## を使用して変数タイプをリストに割り当てる AWS SDK for Python (Boto3)

次の例では、UpdateList API オペレーションを使用して、`allow_ip_address`リストする変数タイプを割り当てます。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.update_list (
     name = 'allow_ip_address',
     variableType = 'IP_ADDRESS'
)
```

## リストを削除する

どのルールでも使用されていないリストも削除できます。リストを削除すると、Amazon Fraud Detector はそのリストとリスト内のすべてのエントリを完全に削除します。

Amazon Fraud Detector コンソールで、 API、 AWS CLI または AWS SDK を使用してリストを削除できます。

### Amazon Fraud Detector コンソールを使用してリストを削除する

リストを削除するには

1. AWS マネジメントコンソールを開き、アカウントにサインインします。Amazon Fraud Detector に移動します。
2. 左側のナビゲーションペインで、リストを選択します。
3. リストページで、削除するリストを選択します。
4. リストの詳細ページで、アクションを選択し、リストの削除を選択します。
5. 削除リストを選択します。

### を使用してリストを削除する AWS SDK for Python (Boto3)

次の例では、DeleteList API オペレーションを使用して を削除しますallow_email_ids。

```
import boto3
                        fraudDetector = boto3.client('frauddetector')
fraudDetector.delete_list(
   name = 'allow_email_ids'
)
```

## リストからエントリを削除する

リストから 1 つ以上のエントリをいつでも削除できます。リスト内のエントリを削除する場合、リストが関連付けられているルールを更新する必要はありません。ルールには、更新されたリストが自動的に組み込まれます。

Amazon Fraud Detector コンソールのリストからエントリを削除するには、 API、 AWS CLI または AWS SDK を使用します。

### Amazon Fraud Detector コンソールを使用してリストからエントリを削除する

リストから 1 つ以上のエントリを削除するには

1. AWS マネジメントコンソールを開き、アカウントにサインインします。Amazon Fraud Detector に移動します。
2. 左側のナビゲーションペインで、リストを選択します。
3. リストページで、削除するエントリを含むリストを選択します。
4. リストの詳細ページで、リストデータタブを選択し、削除するエントリを選択します。
5. 削除を選択し、もう一度削除を選択して確認します。

### を使用してリストからエントリを削除する AWS SDK for Python (Boto3)

次の例では、UpdateList API オペレーションは`allow_email_ids`リストからエントリを削除します。

```
import boto3
                        fraudDetector = boto3.client('frauddetector')
fraudDetector.update_list(
   name = 'allow_email_ids',
   updateMode = 'REMOVE',
   elements = ['emailId_4', 'emailId_12']
)
```

## リストからすべてのエントリを削除する

リストがルールで使用されていない場合は、リスト内のすべてのエントリを削除できます。リスト内のすべてのエントリを削除し、後で同じリストにエントリを追加できます。

Amazon Fraud Detector コンソールのリストからエントリを削除するには、 API、 AWS CLI または AWS SDK を使用します。

### Amazon Fraud Detector コンソールを使用してリストからすべてのエントリを削除する

リストからすべてのエントリを削除するには

1. AWS マネジメントコンソールを開き、アカウントにサインインします。Amazon Fraud Detector に移動します。
2. 左側のナビゲーションペインで、リストを選択します。
3. リストページで、削除するエントリを含むリストを選択します。
4. リストの詳細ページで、データのリストタブを選択し、すべて削除を選択します。
5. 「すべて削除」ボックスに「」と入力delete allして確認し、「すべてのリストデータを削除」を選択します。

### を使用してリストからすべてのエントリを削除する AWS SDK for Python (Boto3)

次の例では、UpdateList API オペレーションはallow_email_idsリストからすべてのエントリを削除します。

```
import boto3
                        fraudDetector = boto3.client('frauddetector')
fraudDetector.update_list(
   name = 'allow_email_ids',
   updateMode = 'REPLACE',
   elements = []
)
```

## 結果

結果は、不正予測の結果です。考えられる不正予測結果ごとに結果を作成できます。例えば、結果でリスクレベル (high_risk、medium_risk、low_risk) またはアクション (承認、レビュー) を表すことができます。結果を作成したら、ルールに 1 つ以上の結果を追加できます。GetEventPrediction レスポンスの一部として、Amazon Fraud Detector は一致したルールに対して定義された結果を返します。

## 結果の作成

Amazon Fraud Detector コンソール、put-outcome コマンド、PutOutcome API、または を使用して結果を作成できます AWS SDK for Python (Boto3)。

## Amazon Fraud Detector コンソールを使用して結果を作成する

1 つ以上の結果を作成するには、

1. AWS マネジメントコンソールを開き、アカウントにサインインします。Amazon Fraud Detector に移動します。
2. 左側のナビゲーションペインで、結果を選択します。
3. 結果ページで、作成を選択します。
4. 新しい結果ページで、次のように入力します。
    a. 結果名に、結果の名前を入力します。
    b. 結果の説明に、オプションで説明を入力します。
5. [結果の保存] を選択します。
6. 追加の結果を作成するには、ステップ 2 ~ 5 を繰り返します。

## を使用して結果を作成する AWS SDK for Python (Boto3)

次の例では、 `PutOutcome` API を使用して 3 つの結果を作成します。これらは `verify_customer`、`review`、および ですapprove。結果を作成したら、ルールに割り当てることができます。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.put_outcome(
name = 'verify_customer',
description = 'this outcome initiates a verification workflow'
)

fraudDetector.put_outcome(
name = 'review',
description = 'this outcome sidelines event for review'
)

fraudDetector.put_outcome(
name = 'approve',
description = 'this outcome approves the event'
)
```

## 結果の削除

ルールバージョンで使用されている結果を削除することはできません。

結果を削除すると、Amazon Fraud Detector はその結果を完全に削除し、データは Amazon Fraud Detector に保存されなくなります。

Amazon Fraud Detector コンソール、delete-outcome コマンド、DeleteOutcome API、または を使用して結果を削除できます。 AWS SDK for Python (Boto3)

### Amazon Fraud Detector コンソールで結果を削除する

結果を削除するには

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/frauddetector で Amazon Fraud Detector コンソールを開きます。
2. Amazon Fraud Detector コンソールの左側のナビゲーションペインで、[リソース] を選択してから、[結果] をクリックします。
3. 削除する結果を選択します。
4. [アクション] を選択し、[削除] を選択します。
5. 結果名を入力してから、[結果の削除] を選択します。

### を使用して結果を削除する AWS SDK for Python (Boto3)

次の例では、DeleteOutcome API を使用して`verify_customer`結果を削除します。結果を削除すると、ルールに割り当てることができなくなります。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.delete_outcome(
name = 'verify_customer'
)
```

# エンティティ

エンティティは、イベントを実行している人またはモノを表します。エンティティタイプは、エンティティを分類します。分類の例には、顧客、マーチャント、ユーザー、またはアカウントが含まれ

ます。イベントデータセットの一部としてエンティティタイプ (ENTITY_TYPE) とエンティティ識別子 (ENTITY_ID) を指定し、イベントを実行した特定のエンティティを示します。

Amazon Fraud Detector は、イベントの不正予測を生成するときにエンティティタイプを使用して、イベントを実行したユーザーを示します。不正予測で使用するエンティティタイプは、まず Amazon Fraud Detector で作成してから、イベントタイプの作成時にイベントに追加する必要があります。

## エンティティタイプの作成

Amazon Fraud Detector コンソールで、put-entity-type コマンド、PutEntityType API、または を使用してエンティティタイプを作成できます AWS SDK for Python (Boto3)。以下の例では、Amazon Fraud Detector コンソール`customer`で SDK for Python (Boto3) を使用してエンティティタイプを作成します。不正検出モデルをトレーニングするためのイベントタイプに関連付けるエンティティタイプを作成する場合は、ユースケースに適したイベントデータセットのエンティティタイプを使用します。

### Amazon Fraud Detector コンソールを使用してエンティティタイプを作成する

エンティティタイプを作成するには、

1. AWS マネジメントコンソールを開き、アカウントにサインインします。
2. Amazon Fraud Detector に移動し、左側のナビゲーションでエンティティを選択し、作成を選択します。
3. エンティティの作成ページで、エンティティタイプ名として customer と入力します。必要に応じて、エンティティの説明を入力します。
4. [エンティティの作成] を選択します。

### を使用してエンティティタイプを作成する AWS SDK for Python (Boto3)

次の AWS SDK for Python (Boto3) コード例では、 `PutEntityType` API を使用してエンティティタイプ を作成します`customer`。不正検出モデルをトレーニングするためのイベントタイプに関連付けるエンティティタイプを作成する場合は、ユースケースに適したイベントデータセットのエンティティを使用します。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.put_entity_type(
name = 'customer',
```

```
description = 'customer'
)
```

# エンティティタイプの削除

Amazon Fraud Detector では、イベントタイプに含まれているエンティティタイプは削除できません。エンティティが関連付けられているイベントタイプを削除してから、エンティティタイプを削除する必要があります。

エンティティタイプを削除すると、Amazon Fraud Detector はそのエンティティタイプを完全に削除し、データは Amazon Fraud Detector に保存されなくなります。

エンティティタイプは、Amazon Fraud Detector コンソール、delete-entity-type コマンド、DeleteEntityType API、または を使用して削除できます。 AWS SDK for Python (Boto3)

## Amazon Fraud Detector コンソールでエンティティタイプを削除する

エンティティタイプを削除するには、

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/frauddetector で Amazon Fraud Detector コンソールを開きます。
2. Amazon Fraud Detector コンソールの左側のナビゲーションペインで、[リソース] を選択してから、[エンティティ] をクリックします。
3. 削除するエンティティタイプを選択します。
4. [アクション] を選択してから、[削除] をクリックします。
5. エンティティタイプ名を入力してから、[エンティティタイプの削除] を選択します。

## を使用してエンティティタイプを削除する AWS SDK for Python (Boto3)

次のコード AWS SDK for Python (Boto3) 例では、DeleteEntityType API を使用してエンティティタイプの顧客を削除します。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.delete_entity_type (

name = 'customer'
```

```
)
```

# を使用して Amazon Fraud Detector リソースを管理する AWS CloudFormation

Amazon Fraud Detector は AWS CloudFormation、Amazon Fraud Detector リソースのモデル化とセットアップに役立つサービスである と統合されているため、リソースとインフラストラクチャの作成と管理に費やす時間を短縮できます。必要なすべての Amazon Fraud Detector リソース (ディテクター、変数、EntityType、EventType、結果、ラベルなど) を記述するテンプレートを作成し、それらのリソースを AWS CloudFormation プロビジョニングして設定します。テンプレートを再利用すると、リソースを複数の AWS アカウントとリージョンで一貫して繰り返しプロビジョニングして設定することができます。

AWS CloudFormation の使用に関して別途料金が発生することはありません。

## Amazon Fraud Detector テンプレートの作成

Amazon Fraud Detector および関連サービスのリソースをプロビジョニングして設定するには、AWS CloudFormation テンプレートについて理解しておく必要があります。テンプレートは、JSON や YAML でフォーマットされたテキストファイルです。これらのテンプレートは、 AWS CloudFormation スタックでプロビジョニングするリソースを記述します。JSON または YAML に慣れていない場合は、 AWS CloudFormation Designer を使用して AWS CloudFormation テンプレートの使用を開始できます。詳細については、「 AWS CloudFormation ユーザーガイド」のAWS CloudFormation 「デザイナーとは」を参照してください。

AWS CloudFormation テンプレートを使用して Amazon Fraud Detector リソースを作成、更新、削除することもできます。リソースの JSON テンプレートと YAML テンプレートの例を含む詳細については、AWS CloudFormation ユーザーガイドの「Amazon Fraud Detector リソースタイプのリファレンス」を参照してください。

CloudFormation を既に使用している場合は、追加の IAM ポリシーや CloudTrail ログ記録を管理する必要はありません。

## Amazon Fraud Detector スタックの管理

CloudFormation コンソールまたは AWS CLI を使用して、Amazon Fraud Detector スタックを作成、更新、削除できます。

スタックを作成するには、AWS CloudFormation がスタックに含めるリソースを示すテンプレートが必要になります。作成済みの Amazon Fraud Detector リソースを新規または既存のスタックにインポートすることにより、CloudFormation 管理に取り込むこともできます。

スタックを管理するための詳細な手順については、AWS CloudFormation ユーザーガイドを参照して、スタックを作成、更新、および削除する方法を確認してください。

### Amazon Fraud Detector スタックの整理

AWS CloudFormation スタックを整理する方法は、完全にユーザー次第です。一般的にベストプラクティスは、ライフサイクルと所有権別にスタックを整理することです。つまり、リソースの変更頻度、またはリソースの更新を担当するチームごとにリソースをグループ化します。

各ディテクターとその検出ロジック ( ルール、変数など ) のスタックを作成して、スタックを整理するように選択できます。他のサービスを使用している場合は、Amazon Fraud Detector のリソースを他のサービスのリソースと一緒にスタックするかどうかを検討する必要があります。例えば、データの収集に役立つ Kinesis リソースと、データを処理する Amazon Fraud Detector リソースを含むスタックを作成できます。これは、詐欺対策チームのすべての製品が連携していることを確認する効果的な方法です。

## Amazon Fraud Detector CloudFormation パラメータの理解

Amazon Fraud Detector には、すべての CloudFormation テンプレートで使用できる標準パラメータに加えて、デプロイ動作の管理に役立つ 2 つの追加パラメータが導入されています。これらのパラメータの 1 つまたは両方を含めない場合、CloudFormation では以下に示すデフォルト値が使用されます。

| パラメータ | 値 | デフォルト値 |
| --- | --- | --- |
| DetectorVersionStatus | ACTIVE: 新規/更新されたディテクターバージョンを [アクティブ] ステータスに設定<br><br>DDRAFT: 新規/更新されたディテクターバージョンを [ドラフト] ステータスに設定 | DRAFT |
| インライン | TRUE: CloudFormation はスタックを作成/更新/削除するときに、リソースを作成/更新/削除できます。 | 正 |

| パラメータ | 値 | デフォルト値 |
| --- | --- | --- |
| | FALSE: CloudFormation がオブジェクトが存在することを検証できるようにしますが、オブジェクトに変更を加えることはりません。 | |

# Amazon Fraud Detector リソースのサンプル AWS CloudFormation テンプレート

以下は、ディテクターと関連するディテクターバージョンを管理するための AWS CloudFormation YAML テンプレートのサンプルです。

```
# Simple Detector resource containing inline Rule, EventType, Variable, EntityType and
 Label resource definitions
Resources:
  TestDetectorLogicalId:
    Type: AWS::FraudDetector::Detector
    Properties:
      DetectorId: "sample_cfn_created_detector"
      DetectorVersionStatus: "DRAFT"
      Description: "A detector defined and created in a CloudFormation stack!"

      Rules:
        - RuleId: "over_threshold_investigate"
          Description: "Automatically sends transactions of $10000 or more to an
 investigation queue"
          DetectorId: "sample_cfn_created_detector"
          Expression: "$amount >= 10000"
          Language: "DETECTORPL"
          Outcomes:
            - Name: "investigate"
              Inline: true
        - RuleId: "under_threshold_approve"
          Description: "Automatically approves transactions of less than $10000"
          DetectorId: "sample_cfn_created_detector"
          Expression: "$amount <10000"
          Language: "DETECTORPL"
          Outcomes:
            - Name: "approve"
              Inline: true
      EventType:
```

```
        Inline: "true"
        Name: "online_transaction"
        EventVariables:
          - Name: "amount"
            DataSource: 'EVENT'
            DataType: 'FLOAT'
            DefaultValue: '0'
            VariableType: "PRICE"
            Inline: 'true'
        EntityTypes:
          - Name: "customer"
            Inline: 'true'
        Labels:
          - Name: "legitimate"
            Inline: 'true'
          - Name: "fraudulent"
            Inline: 'true'
```

## の詳細 AWS CloudFormation

詳細については AWS CloudFormation、以下のリソースを参照してください。

- AWS CloudFormation
- AWS CloudFormation ユーザーガイド
- AWS CloudFormation API リファレンス
- AWS CloudFormation コマンドラインインターフェイスユーザーガイド

# 不正予測

Amazon Fraud Detector を使用して、1 つのイベントの不正予測をリアルタイムで取得したり、一連のイベントの不正予測をオフラインで取得したりできます。1 つのイベントまたは一連のイベントに対して不正予測を生成するには、Amazon Fraud Detector に次の情報を提供する必要があります。

- 不正予測ロジック
- イベントのメタデータ

## 不正検出ロジック

不正予測ロジックは、1 つ以上のルールを使用してイベントに関連付けられたデータを評価し、結果と不正予測スコアを提供します。不正予測ロジックは、次のコンポーネントを使用して作成します。

- イベントタイプ - イベントの構造を定義します
- モデル - 不正を予測するためのアルゴリズムとデータ要件を定義します
- 変数 - イベントに関連付けられたデータ要素を表します
- ルール - 不正検出時に変数値をどのように解釈するかを Amazon Fraud Detector に対して指示します
- 結果 - 不正予測から生成された結果
- ディテクターバージョン - 特定のイベントの不正予測ロジックが含まれています

不正検出ロジックの作成に使用されるコンポーネントの詳細については、「Amazon Fraud Detector コンセプト」を参照してください。不正予測の生成を開始する前に、不正予測ロジックを含むディテクターバージョンを作成して発行していることを確認してください。Fraud Detector コンソールまたは API を使用して、ディテクターバージョンを作成して発行できます。コンソールを使用する手順については、「開始方法 (コンソール)」を参照してください。API の使用手順については、「ディテクターバージョンの作成」を参照してください。

## イベントメタデータ

イベントメタデータは、評価されるイベントの詳細を提供します。評価する各イベントには、ディテクターバージョンに関連付けられたイベントタイプ内の各変数の値を含める必要があります。さらに、イベントメタデータには次のものを含める必要があります。

- EVENT_ID — イベントの識別子。例えば、イベントがオンライントランザクションの場合、EVENT_ID は顧客に提供されるトランザクション参照番号になります。

  EVENT_ID に関する重要な注意事項

  - そのイベントで一意である必要があります
  - ビジネスにとって有意義な情報を表す必要があります
  - 正規表現のパターンを満たす必要があります: `^[0-9a-z_-]+$.`
  - 保存する必要があります。EVENT_ID はイベントのリファレンスで、イベントの削除などのイベントに対する操作を実行するために使用されます。
  - EVENT_ID にタイムスタンプを追加することは、後でイベントを更新するときに問題が発生する可能性があるため、推奨されません。これは、まったく同じ EVENT_ID を指定する必要があるためです。
- ENTITY_TYPE — マーチャントや顧客など、イベントを実行するエンティティ。
- ENTITY_ID - イベントを実行するエンティティの識別子。ENTITY_ID は次の正規表現のパターンを満たす必要があります: `^[0-9a-z_-]+$`。ENTITY_ID が評価時に使用できない場合は、unknown という文字列を渡します。
- EVENT_TIMESTAMP - イベントが発生したときのタイムスタンプ。タイムスタンプは UTC の ISO 8601 標準である必要があります。

# リアルタイム予測

`GetEventPrediction` API を呼び出して、オンライン上の不正行為をリアルタイムで評価できます。各リクエストで 1 つのイベントに関する情報を提供し、指定したディテクターに関連付けられた不正予測ロジックに基づいて、モデルスコアと結果を同期的に受け取ります。

## リアルタイム不正予測の仕組み

-`GetEventPrediction` API は、指定されたディテクターバージョンを使用して、イベントに提供されたイベントメタデータを評価します。評価中、Amazon Fraud Detector は、まずディテクターバージョンに追加されたモデルのモデルスコアを生成してから、その結果をルールに渡して評価します。ルールは、ルール実行モードで指定されたとおりに実行されます (ディテクターバージョンの作成を参照)。Amazon Fraud Detector は、レスポンスの一部として、一致したルールに関連する結果だけでなく、モデルスコアも提供します。

# リアルタイムの不正予測の取得

リアルタイムの不正予測を取得するには、不正予測モデルとルール (つまりルールセット) を含むディテクターを作成して発行していることを確認してください。

コマンドラインインターフェイス (AWS CLI) またはいずれかの Amazon Fraud Detector SDKs を使用して GetEventPrediction API AWS オペレーションを呼び出すことで、イベントの不正予測をリアルタイムで取得できます。

API を使用するには、リクエストごとに 1 つのイベントの情報を指定します。リクエストの一部として、Amazon Fraud Detector がイベントの評価に使用する `detectorId` を指定する必要があります。必要に応じて、`detectorVersionId` を指定できます。`detectorVersionId` が指定されていない場合、Amazon Fraud Detector は `ACTIVE` バージョンのディテクターを使用します。

オプションで、フィールド にデータを渡して SageMaker AI モデルを呼び出すためにデータを送信できます`externalModelEndpointBlobs`。

## を使用して不正予測を取得する AWS SDK for Python (Boto3)

不正予測を生成するには、`GetEventPrediction` API を呼び出します。以下の例では、パート B: 不正予測を生成する を完了していることを前提としています。レスポンスの一環として、モデルスコア、一致したルールとそれに対応する結果を受け取ります。`GetEventPrediction` リクエストのその他の例については、aws-fraud-detector-samples GitHub リポジトリをご覧ください。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.get_event_prediction(
detectorId = 'sample_detector',
eventId = '802454d3-f7d8-482d-97e8-c4b6db9a0428',
eventTypeName = 'sample_registration',
eventTimestamp = '2020-07-13T23:18:21Z',
entities = [{'entityType':'sample_customer', 'entityId':'12345'}],
eventVariables = {
    'email_address' : 'johndoe@exampledomain.com',
    'ip_address' : '1.2.3.4'
}
)
```

# バッチ予測

Amazon Fraud Detector のバッチ予測ジョブを使用して、リアルタイムのスコアリングを必要としない一連のイベントの予測を取得できます。例えば、バッチ予測ジョブを作成して、オフラインの概念実証を実行したり、イベントのリスクを時間単位、日単位、週単位で遡及的に評価したりできます。

Amazon Fraud Detector コンソールを使用するか、 コマンドラインインターフェイス (AWS CLI) または Amazon Fraud Detector SDKs のいずれかを使用して CreateBatchPredictionJob API AWS オペレーションを呼び出すことで、バッチ予測ジョブを作成できます。

トピック

- バッチ予測の仕組み
- 入力および出力ファイル
- バッチ予測の取得
- IAM ロールに関するガイダンス
- を使用してバッチ不正予測を取得する AWS SDK for Python (Boto3)

## バッチ予測の仕組み

-`CreateBatchPredictionJob`API オペレーションでは、指定されたディテクタバージョンを使用して、Amazon S3 バケットにある入力 CSV ファイルで指定されたデータに基づいて予測を行います。次に API は、結果の CSV ファイルを S3 バケットに返します。

バッチ予測ジョブは、`GetEventPrediction` オペレーションと同じようにモデルのスコアと予測結果を計算します。`GetEventPrediction` と同様に、バッチ予測ジョブを作成するには、まずイベントタイプを作成し、オプションでモデルをトレーニングしてから、バッチジョブのイベントを評価するディテクタバージョンを作成します。

バッチ予測ジョブによって評価されるイベントリスクスコアの料金は、`GetEventPrediction` API で作成されたスコアの料金と同じです。詳細については、「Amazon Fraud Detector の料金」を参照してください。

バッチ予測ジョブは、一度に 1 回のみ実行することができます。

## 入力および出力ファイル

入力 CSV ファイルには、選択したディテクターバージョンに関連付けられているイベントタイプに一致するヘッダーが含まれている必要があります。入力データファイルの最大サイズは 1GB です。イベントの数は、イベントのサイズによって異なります。

Amazon Fraud Detector は、出力データ用に別の場所を指定しない限り、入力ファイルと同じバケットに出力ファイルを作成します。出力ファイルには、入力ファイルの元のデータと、次の追加列が含まれます。

- MODEL_SCORES — 選択したディテクターバージョンに関連付けられている各モデルのイベントのモデルスコアの詳細を示します。
- OUTCOMES — 選択したディテクターバージョンとそのルールによって評価されたイベントの結果の詳細を示します。
- STATUS — イベントが正常に評価されたかどうかを示します。イベントが正常に評価されなかった場合、この列には失敗の理由コードが表示されます。
- RULE_RESULTS — ルール実行モードに基づく、一致したすべてのルールのリスト。

## バッチ予測の取得

次の手順では、既にイベントタイプを作成し、そのイベントタイプを使用してモデルをトレーニングし (オプション)、そのイベントタイプのディテクターバージョンを作成していることを前提としています。

バッチ予測を取得するには

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/frauddetector で Amazon Fraud Detector コンソールを開きます。
2. Amazon Fraud Detector コンソールの左側のナビゲーションペインで、[バッチ予測] を選択してから、[新しいバッチ予測] をクリックします。
3. [ジョブ名] で、バッチ予測ジョブの名前を指定します。名前を指定しない場合、Amazon Fraud Detector はジョブ名をランダムに生成します。
4. [ディテクター] で、このバッチ予測のディテクターを選択します。
5. [ディテクターバージョン] で、このバッチ予測のディテクターバージョンを選択します。どのステータスのディテクターバージョンも選択できます。お使いのディテクターに Active ステー

タスのディテクターバージョンがある場合、そのバージョンが自動的に選択されますが、必要に応じてこの選択を変更することもできます。

6. [IAM ロール] で、入力および出力の Amazon S3 バケットに対する読み取りおよび書き込みアクセス権限を持つロールを選択または作成します。詳細については「IAM ロールに関するガイダンス」を参照してください。

   バッチ予測を取得するには、 `CreateBatchPredictionJob`オペレーションを呼び出す IAM ロールに、入力 S3 バケットへの読み取りアクセス許可と出力 S3 バケットへの書き込みアクセス許可が必要です。バケットのアクセス権限の詳細については、Amazon S3 ユーザーガイドの「ユーザーポリシーの例」を参照してください。

7. [入力データの場所] で、入力データの Amazon S3 の場所を指定します。出力ファイルを別の S3 バケットに含める場合は、[出力用の個別のデータの場所] を選択し、出力データの Amazon S3 の場所を指定します。

8. (オプション) バッチ予測ジョブのタグを作成します。

9. [開始] を選択します。

   Amazon Fraud Detector はバッチ予測ジョブを作成し、ジョブのステータスは [`In progress`] になります。Batch 予測ジョブの処理時間は、イベントの数とディテクターのバージョンの設定によって異なります。

進行中のバッチ予測ジョブを停止するには、バッチ予測ジョブの詳細ページに移動し、[アクション] を選択してから、[バッチ予測の停止] をクリックします。バッチ予測ジョブを停止すると、ジョブの結果が表示されません。

バッチ予測ジョブのステータスが [`Complete`] に変わると、指定した出力 Amazon S3 バケットからジョブの出力を取得できます。出力ファイルの名前は `batch prediction job name_file creation timestamp_output.csv` という形式になります。例えば、`mybatchjob` という名前のジョブの出力ファイルは、`mybatchjob_ 1611170650_output.csv` です。

バッチ予測ジョブによって評価された特定のイベントを検索するには、Amazon Fraud Detector コンソールの左側のナビゲーションペインで、[過去の予測の検索] を選択します。

完了したバッチ予測ジョブを削除するには、バッチ予測ジョブの詳細ページに移動し、[アクション] を選択してから、[バッチ予測の削除] をクリックします。

## IAM ロールに関するガイダンス

バッチ予測を取得するには、CreateBatchPredictionJob オペレーションを呼び出す IAM ロールに、入力 S3 バケットへの読み取りアクセス許可と、出力 S3 バケットへの書き込みアクセス許可が必要です。バケットのアクセス許可の詳細については、Amazon S3 ユーザーガイドの「ユーザーポリシーの例」を参照してください。Amazon Fraud Detector コンソールでは、以下のように、Batch 予測の IAM ロールを選択するためのオプションが 3 つあります。

1. 新しいBatch 予測ジョブを作成するときに、ロールを作成します。
2. Amazon Fraud Detector コンソールで以前に作成した既存の IAM ロールを選択します。この手順を実行する前に、必ずロールに `S3:PutObject` アクセス許可を追加してください。
3. 以前に作成した IAM ロールのカスタム ARN を入力します。

IAM ロールに関連するエラーが表示された場合は、以下を確認します。

1. Amazon S3 の入出力バケットは、ディテクターと同じリージョンにあります。
2. 使用している IAM ロールには、入力 S3 バケット用の `s3:GetObject` アクセス許可と出力 S3 バケット用の `s3:PutObject` アクセス許可があること。
3. 使用している IAM ロールには、サービスプリンシパル `frauddetector.amazonaws.com` の信頼ポリシーがあること。

## を使用してバッチ不正予測を取得する AWS SDK for Python (Boto3)

次の例は、CreateBatchPredictionJob API のリクエストの例を示しています。バッチ予測ジョブには、ディテクター、ディテクターバージョン、およびイベントタイプ名の既存のリソースを含める必要があります。次の例では、イベントタイプ `sample_registration`、ディテクター `sample_detector`、およびディテクターバージョン `1` を作成したとします。

```
import boto3
fraudDetector = boto3.client('frauddetector')

fraudDetector.create_batch_prediction_job (
   jobId = 'sample_batch',
   inputPath = 's3://bucket_name/input_file_name.csv',
   outputPath = 's3://bucket_name/',
   eventTypeName = 'sample_registration',
   detectorName = 'sample_detector',
```

```
   detectorVersion = '1',
   iamRoleArn = 'arn:aws:iam::**:role/service-role/AmazonFraudDetector-DataAccessRole-
**'
)
```

# 予測説明

予測説明は、各イベント変数がモデルの不正予測スコアにどのような影響を与えたかについてのインサイトを提供し、不正予測の一部として自動的に生成されます。各不正予測には、1～1000 のリスクスコアが付けられます。予測説明では、マグニチュード (0～5、5が最高) と方向 (スコアを高くするか低くする) の観点から、各イベント変数がリスクスコアに与える影響の詳細を示します。予測説明を次のタスクで使用することもできます。

- イベントにレビューのフラグが設定されたときに、手動による調査中にトップリスク指標を特定するタスク。
- 偽陽性の予測につながる根本原因を絞り込むタスク (例えば、正当なイベントのリスクスコアが高い ) 。
- イベントデータ全体にわたる不正パターンを分析し、データセット内のバイアス (存在する場合) を検出するタスク。

> **Important**
>
> 予測説明は自動的に生成され、2021 年 6 月 30 日以降にトレーニングされたモデルでのみ使用できます。2021 年 6 月 30 日以前にトレーニングしたモデルの予測説明を受け取るには、そのモデルを再トレーニングします。

予測説明では、モデルのトレーニングに使用されたイベント変数ごとに次の値のセットが提供されます。

相対的な影響

不正予測スコアに対する大きさの観点から、変数の影響を視覚的に確認できるようにします。相対的な影響値は、不正リスクの星評価 (0～5、5が最高) と方向 (増加/減少) の影響で構成されます。

- 不正リスクが増加した変数は、赤色の星で示されます。赤い星の数が多いほど、変数が不正スコアを上げて、不正の可能性が高まります。

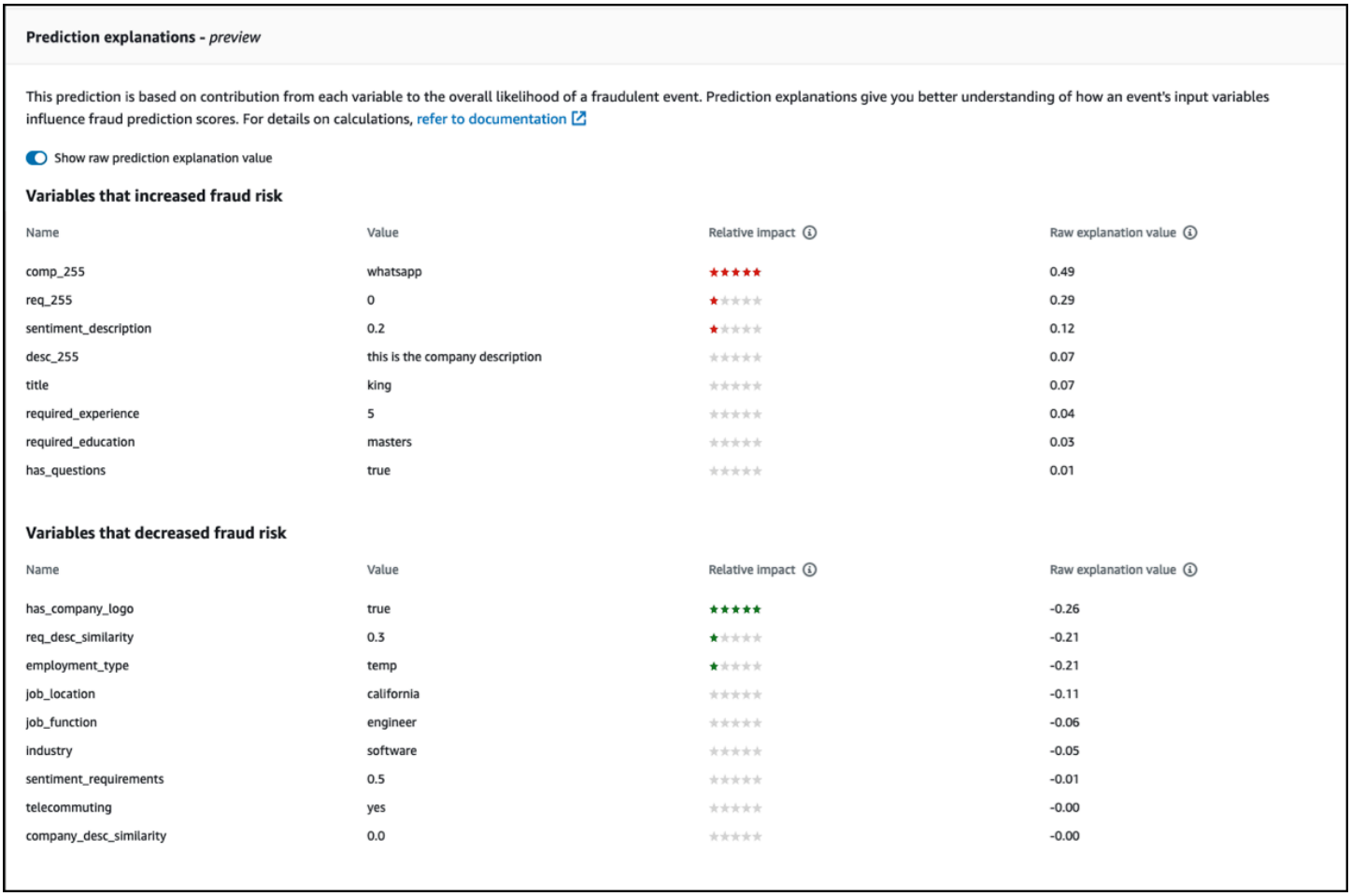
- 不正リスクが減少した変数は、緑色の星で示されます。緑の星の数が多いほど、変数が不正リスクスコアを下げて、不正の可能性が低くなります。
- すべての変数の星がゼロの場合、どの変数もそれ自体では不正リスクを大きく変えなかったことを示しています。

### 未加工の説明値

不正行為の対数オッズとして表される、未解釈の未加工の値を示します。これらの値は、通常 -10 から +10 の間ですが、- 無限大から + 無限大までの範囲を取れます。

- 正の値は、変数がリスクスコアを上げたことを示しています。
- 負の値は、変数がリスクスコアを下げたことを示しています。

Amazon Fraud Detector コンソールでは、予測説明の値が次のように表示されます。色付きの星の評価とそれに対応する未加工の数値により、変数間の相対的な影響を簡単に確認できます。



**Prediction explanations** - *preview*

This prediction is based on contribution from each variable to the overall likelihood of a fraudulent event. Prediction explanations give you better understanding of how an event's input variables influence fraud prediction scores. For details on calculations, refer to documentation

Show raw prediction explanation value

**Variables that increased fraud risk**

| Name | Value | Relative impact | Raw explanation value |
|---|---|---|---|
| comp_255 | whatsapp | ★★★★★ | 0.49 |
| req_255 | 0 | ★☆☆☆☆ | 0.29 |
| sentiment_description | 0.2 | ★☆☆☆☆ | 0.12 |
| desc_255 | this is the company description | ☆☆☆☆☆ | 0.07 |
| title | king | ☆☆☆☆☆ | 0.07 |
| required_experience | 5 | ☆☆☆☆☆ | 0.04 |
| required_education | masters | ☆☆☆☆☆ | 0.03 |
| has_questions | true | ☆☆☆☆☆ | 0.01 |

**Variables that decreased fraud risk**

| Name | Value | Relative impact | Raw explanation value |
|---|---|---|---|
| has_company_logo | true | ★★★★★ | -0.26 |
| req_desc_similarity | 0.3 | ★☆☆☆☆ | -0.21 |
| employment_type | temp | ★☆☆☆☆ | -0.21 |
| job_location | california | ☆☆☆☆☆ | -0.11 |
| job_function | engineer | ☆☆☆☆☆ | -0.06 |
| industry | software | ☆☆☆☆☆ | -0.05 |
| sentiment_requirements | 0.5 | ☆☆☆☆☆ | -0.01 |
| telecommuting | yes | ☆☆☆☆☆ | -0.00 |
| company_desc_similarity | 0.0 | ☆☆☆☆☆ | -0.00 |

# 予測説明の表示

不正予測を生成したら、Amazon Fraud Detector コンソールで予測の説明を表示できます。 AWS SDK の APIs を使用して予測の説明を表示するには、まず `ListEventPrediction` API を呼び出してイベントの予測タイムスタンプを取得し、次に `GetEventPredictionMetadata` API を呼び出して予測の説明を取得する必要があります。

## Amazon Fraud Detector コンソールを使用して予測の説明を表示する

コンソールを使用して予測説明を表示するには

1. AWS コンソールを開き、アカウントにサインインします。Amazon Fraud Detector に移動します。
2. 左側のナビゲーションペインで、[過去の予測を検索] を選択します。
3. プロパティ、演算子、および値のフィルターを使用して、レビューする予測を選択します。
4. 上位のフィルターペインで、確認する予測が生成された期間を必ず選択してください。
5. [結果] ペインには、指定した期間中に生成されたすべての予測のリストが表示されます。予測説明を表示するには、予測の [イベント ID] をクリックします。
6. [予測説明] ペインまで下にスクロールします。
7. すべての変数の未加工の予測説明値を表示するには、[未加工の予測説明値を表示] ボタンをオンに設定します。

## AWS SDK for Python (Boto3) を使用して予測の説明を表示する

次の例は、 AWS SDK の `ListEventPredictions`および `GetEventPredictionMetadata` APIs を使用して予測の説明を表示するためのサンプルリクエストを示しています。

例 1: **`ListEventPredictions`** API を使用して最新の予測のリストを取得する

```
import boto3
fraudDetector = boto3.client('frauddetector')
fraudDetector.list_event_predictions(
  maxResults = 10,
  predictionTimeRange = {
     end_time: '2022-01-13T23:18:21Z',
     start_time: '2022-01-13T20:18:21Z'
    }
 )
```

例 2: **ListEventPredictions** API を使用してイベントタイプ「登録」の過去の予測のリストを取得する

```
import boto3
fraudDetector = boto3.client('frauddetector')
fraudDetector.list_event_predictions(
   eventType = {
      value = 'registration'
    }
   maxResults = 70,
   nextToken = "10",
   predictionTimeRange = {
     end_time: '2021-07-13T23:18:21Z',
     start_time: '2021-07-13T20:18:21Z'
    }
 )
```

例 3: **GetEventPredictionMetadata** API を使用して、指定された期間に生成された指定されたイベント ID、イベントタイプ、ディテクター ID、ディテクターバージョン ID の過去の予測の詳細を取得します。

このリクエストにpredictionTimestamp指定された は、最初に ListEventPredictions API を呼び出すことで取得されます。

```
import boto3
fraudDetector = boto3.client('frauddetector')
fraudDetector.get_event_prediction_metadata (
   detectorId = 'sample_detector',
   detectorVersionId = '1',
   eventId = '802454d3-f7d8-482d-97e8-c4b6db9a0428',
   eventTypeName = 'sample_registration',
   predictionTimestamp = '2021-07-13T21:18:21Z'
 )
```

## 予測説明の計算方法の理解

Amazon Fraud Detector は、SHAP (SHapeley Additive exPlanations) を使用して、モデルトレーニングに使用される各イベント変数の未加工の説明値を計算することにより、個々のイベント予測を説明

します。未加工の説明値は、予測を生成する際に、分類アルゴリズムの一部としてモデルによって計算されます。このような未加工の説明値は、不正の確率の対数に対して各入力がどれだけ寄与しているかを表しています。未加工の説明値 (-無限大から +無限大まで) は、マッピングを使用して相対的な影響値 (-5 から+5) に変換されます。未加工の説明値から導かれる相対的影響値は、不正 (正) または正当 (負) の確立が増加する回数を表し、予測説明を理解しやすくしています。

# Amazon Fraud Detector のセキュリティ

でのクラウドセキュリティが最優先事項 AWS です。お客様は AWS 、セキュリティを最も重視する組織の要件を満たすように構築されたデータセンターとネットワークアーキテクチャからメリットを得られます。

セキュリティは、 AWS お客様とお客様の間の責任共有です。責任共有モデルでは、これをクラウドのセキュリティおよびクラウド内のセキュリティと説明しています。

- クラウドのセキュリティ – AWS クラウドで AWS サービスを実行するインフラストラクチャを保護する AWS 責任があります。 AWS また、 では、安全に使用できるサービスも提供しています。サードパーティーの監査者は、AWS コンプライアンスプログラムコンプライアンスプログラムの一環として、当社のセキュリティの有効性を定期的にテストおよび検証。Amazon Fraud Detector に適用するコンプライアンスプログラムの詳細については、コンプライアンスプログラムによる対象範囲内の AWS のサービスを参照してください。
- クラウド内のセキュリティ – お客様の責任は、使用する AWS サービスによって決まります。また、ユーザーは、データの機密性、会社の要件、適用される法律や規制など、その他の要因についても責任を負います。

このドキュメントは、Amazon Fraud Detector の使用時に責任共有モデルがどのように適用されるかを理解するために役立ちます。以下のトピックでは、セキュリティおよびコンプライアンス上の目的を達成するように Amazon Fraud Detector を設定する方法について説明します。また、Amazon Fraud Detector リソースのモニタリングと保護に役立つ他の AWS サービスの使用方法についても説明します。

トピック


- Amazon Fraud Detector でのデータ保護
- Amazon Fraud Detector の Identity and Access Management
- Amazon Fraud Detector でのログ記録とモニタリング
- Amazon Fraud Detector のコンプライアンス検証
- Amazon Fraud Detector の復元力
- Amazon Fraud Detector のインフラストラクチャセキュリティ

# Amazon Fraud Detector でのデータ保護

責任 AWS 共有モデル、Amazon Fraud Detector でのデータ保護に適用されます。このモデルで説明されているように、 AWS はすべての を実行するグローバルインフラストラクチャを保護する責任があります AWS クラウド。ユーザーは、このインフラストラクチャでホストされるコンテンツに対する管理を維持する責任があります。また、使用する「 AWS のサービス 」のセキュリティ設定と管理タスクもユーザーの責任となります。データプライバシーの詳細については、データプライバシーに関するよくある質問を参照してください。欧州でのデータ保護の詳細については、AWS セキュリティブログに投稿された AWS 責任共有モデルおよび GDPR のブログ記事を参照してください。

データ保護の目的で、認証情報を保護し AWS アカウント 、 AWS IAM Identity Center または AWS Identity and Access Management (IAM) を使用して個々のユーザーを設定することをお勧めします。この方法により、それぞれのジョブを遂行するために必要な権限のみが各ユーザーに付与されます。また、次の方法でデータを保護することもお勧めします:

- 各アカウントで多要素認証 (MFA) を使用します。
- SSL/TLS を使用して AWS リソースと通信します。TLS 1.2 が必須で、TLS 1.3 をお勧めします。
- で API とユーザーアクティビティのログ記録を設定します AWS CloudTrail。CloudTrail 証跡を使用して AWS アクティビティをキャプチャする方法については、「 AWS CloudTrail ユーザーガイド」のCloudTrail 証跡の使用」を参照してください。
- AWS 暗号化ソリューションと、 内のすべてのデフォルトのセキュリティコントロールを使用します AWS のサービス。
- Amazon Macie などの高度な管理されたセキュリティサービスを使用します。これらは、Amazon S3 に保存されている機密データの検出と保護を支援します。
- コマンドラインインターフェイスまたは API AWS を介して にアクセスするときに FIPS 140-3 検証済みの暗号化モジュールが必要な場合は、FIPS エンドポイントを使用します。利用可能な FIPS エンドポイントの詳細については、「連邦情報処理規格 (FIPS) 140-3」を参照してください。

お客様の E メールアドレスなどの極秘または機密情報を、タグ、または [名前] フィールドなどの自由形式のテキストフィールドに含めないことを強くお勧めします。これは、コンソール、API、または SDK を使用して Amazon Fraud Detector AWS CLIまたは他の AWS のサービス を使用する場合も同様です。 AWS SDKs タグ、または名前に使用される自由記述のテキストフィールドに入力したデータは、請求または診断ログに使用される場合があります。外部サーバーに URL を提供する場合、そのサーバーへのリクエストを検証できるように、認証情報を URL に含めないことを強くお勧めします。

# 保管中のデータの暗号化

Amazon Fraud Detector は、選択した暗号化キーを使用して、保管中のデータを暗号化します。次のいずれかを選択できます。

- AWS 所有の KMS キー。暗号化キーを指定しない場合、デフォルトでは、データはこのキーを使用して暗号化されます。
- カスタマーマネージド KMS キー。キーポリシーを使用して、カスタマーマネージド KMS キーへのアクセスを制御できます。カスタマーマネージド KMS キーの作成と管理については、「キーの管理」を参照してください。

## Encrypting data in transit

Amazon Fraud Detector は、アカウントからデータをコピーし、内部 AWS システムで処理します。デフォルトでは、Amazon Fraud Detector は TLS 1.2 と AWS 証明書を使用して転送中のデータを暗号化します。

## キーの管理

Amazon Fraud Detector は、次の 2 種類のキーのいずれかを使用してデータを暗号化します。

- AWS 所有の KMS キー。これがデフォルトのトランスコードプリセットです。
- カスタマーマネージド KMS キー。

### カスタマーマネージド KMS キーの作成

カスタマーマネージド KMS キーは、KMS AWS コンソールまたは CreateKey API を使用して作成できます。キーを作成するときは、以下を確認してください。

- 対称暗号化カスタマーマネージド KMS キーを選択します。Amazon Fraud Detector は非対称 KMS キーをサポートしていません。詳細については、「 Key Management Service デベロッパーガイド」の「 の非対称 AWS KMS AWS キー」を参照してください。
- 単一リージョン KMS キーを作成します。Amazon Fraud Detector は、マルチリージョン KMS キーをサポートしていません。詳細については、「 Key Management Service デベロッパーガイド」の「 のマルチリージョン AWS KMS AWS キー」を参照してください。
- 次のキーポリシーを指定して、キーを使用するためのアクセス許可を Amazon Fraud Detector に付与します。

```
        {
    "Effect": "Allow",
    "Principal": {
        "Service": "frauddetector.amazonaws.com"
    },
    "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey",
        "kms:CreateGrant",
        "kms:RetireGrant"
    ],
    "Resource": "*"
}
```

キーポリシーの詳細については、「 Key Management Service デベロッパーガイド」の「KMS AWS でのキーポリシーの使用」を参照してください。 AWS

## カスタマーマネージド KMS キーを使用したデータの暗号化

カスタマーマネージド KMS キーを使って保管中の Amazon Fraud Detector データを暗号化するには、Amazon Fraud Detector PutKMSEncryptionKey API を使用します。`PutKMSEncryptionKey` API を使用して暗号化設定はいつでも変更できます。

暗号化されたデータに関する重要な注意事項

- カスタマーマネージド KMS キーの設定後に生成されたデータは暗号化されます。カスタマーマネージド KMS キーを設定する前に生成されたデータは、暗号化されないままになります。
- カスタマーマネージド KMS キーが変更された場合、以前の暗号化設定を使用して暗号化されたデータは再暗号化されません。

## データの表示

カスタマーマネージド KMS キーを使用して Amazon Fraud Detector のデータを暗号化する場合、この方法を使用して暗号化されたデータは、Amazon Fraud Detector コンソールの [過去の予測の検索]

領域のフィルターを使用して検索できません。検索結果に漏れがないようにするには、次のプロパティの 1 つ以上のプロパティを使用して結果をフィルタリングします。

- Event ID
- 評価タイムスタンプ
- ディテクターステータス
- ディテクターバージョン
- モデルバージョン
- モデルタイプ
- ルール評価ステータス
- ルール実行モード
- ルール一致ステータス
- ルールバージョン
- 可変データソース

カスタマーマネージド KMS キーが削除されたか、削除がスケジュールされている場合、データは利用できない可能性があります。詳細については、「KMS キーの削除」を参照してください。

## Amazon Fraud Detector とインターフェイス VPC エンドポイント (AWS PrivateLink)

VPC と Amazon Fraud Detector とのプライベート接続を確立するには、インターフェイス VPC エンドポイントを作成します。インターフェイスエンドポイントは、インターネットゲートウェイ、NAT デバイス、VPN 接続、AWS Direct Connect 接続のいずれも必要とせずに Amazon Fraud Detector API にプライベートにアクセスできるテクノロジーである AWS PrivateLink を利用しています。VPC のインスタンスは、パブリック IP アドレスがなくても Amazon Fraud Detector API と通信できます。VPC と Amazon Fraud Detector との間のトラフィックは、Amazon ネットワークを離れません。

各インターフェースエンドポイントは、サブネット内の 1 つ以上の Elastic Network Interface によって表されます。

詳細については、「Amazon VPC ユーザーガイド」の「インターフェイス VPC エンドポイント (AWS PrivateLink)」を参照してください。

## Amazon Fraud Detector VPC エンドポイントに関する考慮事項

Amazon Fraud Detector 用の VPC エンドポイントを設定する前に、Amazon VPC ユーザーガイドの「インターフェイスエンドポイントのプロパティと制限」を確認してください。

Amazon Fraud Detector は、VPC からのすべての API アクションの呼び出しをサポートしています。

VPC エンドポイントポリシーは Amazon Fraud Detector でサポートされています。デフォルトでは、エンドポイントを通じた Amazon Fraud Detector へのフルアクセスが許可されています。詳細については、「Amazon VPC ユーザーガイド」の「VPC エンドポイントによるサービスのアクセスコントロール」を参照してください。

## Amazon Fraud Detector 用のインターフェイス VPC エンドポイントの作成

Amazon Fraud Detector サービスの VPC エンドポイントは、Amazon VPC コンソールまたは AWS Command Line Interface () を使用して作成できますAWS CLI。詳細については、Amazon VPC ユーザーガイドの「インターフェイスエンドポイントの作成」を参照してください。

Amazon Fraud Detector 用の VPC エンドポイントを作成するには、次のサービス名を使用します。

- com.amazonaws.*region*.frauddetector

エンドポイントのプライベート DNS を有効にすると、リージョンのデフォルト DNS 名 (`frauddetector.us-east-1.amazonaws.com` など) を使用して、Amazon Fraud Detector への API リクエストを実行できます。

詳細については、Amazon VPC ユーザーガイドの「インターフェイスエンドポイントを介したサービスへのアクセス」を参照してください。

## Amazon Fraud Detector 用の VPC エンドポイントポリシーの作成

Amazon Fraud Detector のインターフェイス VPC エンドポイントに対するポリシーを作成して、以下を指定することができます。

- アクションを実行できるプリンシパル
- 実行可能なアクション
- アクションを実行できるリソース

詳細については、Amazon VPC ユーザーガイドの「VPC エンドポイントによるサービスのアクセスコントロール」を参照してください。

次の例の VPC エンドポイントポリシーでは、VPC インターフェイスエンドポイントにアクセスできるすべてのユーザーが、Amazon WorkSpaces でホストされた、my_detector という名前の Amazon Fraud Detector ディテクターにアクセスすることが許可されます。

```
{
  "Statement": [
      {
          "Action": "frauddetector:*Detector",
          "Effect": "Allow",
          "Resource": "arn:aws:frauddetector:us-east-1:123456789012:detector/
my_detector",
          "Principal": "*"
      }
  ]
}
```

この例では、以下が拒否されます。

- 他の Amazon Fraud Detector API アクション
- Amazon Fraud Detector GetEventPrediction API の呼び出し

> **Note**
>
> この例では、ユーザーは VPC の外部からその他の Amazon Fraud Detector API アクションをまだ実行できます。VPC 内からこれらの API コールを制限する方法については、「Amazon Fraud Detector のアイデンティティベースポリシー」を参照してください。

## サービス改善のためのデータの使用をオプトアウトする

モデルのトレーニングと予測の生成のために提供した履歴イベントデータは、サービスの提供と保守のみに使用されます。このデータは、Amazon Fraud Detector の品質を向上させるために使用される場合もあります。お客様の信頼、プライバシー、およびお客様のコンテンツのセキュリティは当社の最優先事項であり、当社の使用がお客様に対する当社のコミットメントに準拠していることを確認します。詳細については、「データプライバシーに関するよくある質問」を参照してください。

AWS Organizations ユーザーガイドの AI サービスのオプトアウトポリシーページにアクセスし、そこで説明されているプロセスに従うことで、イベントデータを使用して Amazon Fraud Detector の品質を開発または改善することをオプトアウトできます。 AWS Organizations

> **Note**
>
> オプトアウトポリシーを使用するには、AWS Organizations が AWS アカウントを一元管理する必要があります。AWS アカウントの組織をまだ作成していない場合は、「組織の作成と管理」ページにアクセスし、そこで説明されているプロセスに従います。

# Amazon Fraud Detector の Identity and Access Management

AWS Identity and Access Management (IAM) は、管理者が AWS リソースへのアクセスを安全に制御 AWS のサービス するのに役立つ です。IAM 管理者は、誰を認証 (サインイン) し、誰に Amazon Fraud Detector リソースの使用を承認する (アクセス許可を付与する) かを制御します。IAM は、追加料金なしで使用できる AWS のサービス です。

トピック


- 対象者
- アイデンティティを使用した認証
- ポリシーを使用したアクセスの管理
- Amazon Fraud Detector で IAM が機能する仕組み
- Amazon Fraud Detector のアイデンティティベースポリシーの例
- 混乱した代理の防止
- Amazon Fraud Detector アイデンティティとアクセスのトラブルシューティング


## 対象者

AWS Identity and Access Management (IAM) の使用方法は、Amazon Fraud Detector で行う作業によって異なります。

サービスユーザー – ジョブを実行するために Amazon Fraud Detector サービスを使用する場合は、管理者から必要なアクセス許可と認証情報が与えられます。業務のために使用する Amazon Fraud Detector 機能が増えるにつれて、追加の許可が必要になる可能性があります。アクセスの管理方法を理解すると、管理者に適切なアクセス許可をリクエストするのに役に立ちます。Amazon Fraud

Detector の機能にアクセスできない場合は、「Amazon Fraud Detector アイデンティティとアクセスのトラブルシューティング」を参照してください。

サービス管理者 – 社内の Amazon Fraud Detector リソースを担当している場合は、通常、Amazon Fraud Detector へのフルアクセスがあります。サービスユーザーがどの Amazon Fraud Detector 機能とリソースにアクセスするかを決めるのは管理者の仕事です。その後、IAM 管理者にリクエストを送信して、サービスユーザーの権限を変更する必要があります。このページの情報を点検して、IAM の基本概念を理解してください。会社で Amazon Fraud Detector と IAM を併用する方法の詳細については、「Amazon Fraud Detector で IAM が機能する仕組み」を参照してください。

IAM 管理者 – IAM 管理者には、Amazon Fraud Detector へのアクセスを管理するポリシーの作成方法の詳細を理解することが推奨されます。IAM で使用できる Amazon Fraud Detector アイデンティティベースのポリシーの例を表示するには、「Amazon Fraud Detector のアイデンティティベースポリシーの例」を参照してください。

## アイデンティティを使用した認証

認証とは、ID 認証情報 AWS を使用して にサインインする方法です。として、IAM ユーザーとして AWS アカウントのルートユーザー、または IAM ロールを引き受けて認証 (サインイン AWS) される必要があります。

ID ソースを介して提供された認証情報を使用して、フェデレーティッド ID AWS として にサインインできます。 AWS IAM Identity Center （ IAM Identity Center) ユーザー、会社のシングルサインオン認証、Google または Facebook 認証情報は、フェデレーション ID の例です。フェデレーティッド ID としてサインインする場合、IAM ロールを使用して、前もって管理者により ID フェデレーションが設定されています。フェデレーション AWS を使用して にアクセスすると、間接的にロールを引き受けることになります。

ユーザーのタイプに応じて、 AWS Management Console または AWS アクセスポータルにサインインできます。へのサインインの詳細については AWS、「 AWS サインイン ユーザーガイド」の「 へのサインイン AWS アカウント方法」を参照してください。

AWS プログラムで にアクセスする場合、 はソフトウェア開発キット (SDK) とコマンドラインインターフェイス (CLI) AWS を提供し、認証情報を使用してリクエストを暗号化して署名します。AWS ツールを使用しない場合は、自分でリクエストに署名する必要があります。リクエストに自分で署名する推奨方法の使用については、「IAM ユーザーガイド」の「API リクエストに対するAWS Signature Version 4」を参照してください。

使用する認証方法を問わず、追加セキュリティ情報の提供をリクエストされる場合もあります。たとえば、 では、アカウントのセキュリティを高めるために多要素認証 (MFA) を使用する AWS ことを

お勧めします。詳細については、「AWS IAM Identity Center ユーザーガイド」の「多要素認証」および「IAM ユーザーガイド」の「IAM のAWS 多要素認証」を参照してください。

### AWS アカウント ルートユーザー

を作成するときは AWS アカウント、アカウント内のすべての およびリソースへの AWS のサービス完全なアクセス権を持つ 1 つのサインインアイデンティティから始めます。この ID は AWS アカウント ルートユーザーと呼ばれ、アカウントの作成に使用した E メールアドレスとパスワードでサインインすることでアクセスできます。日常的なタスクには、ルートユーザーを使用しないことを強くお勧めします。ルートユーザーの認証情報は保護し、ルートユーザーでしか実行できないタスクを実行するときに使用します。ルートユーザーとしてサインインする必要があるタスクの完全なリストについては、「IAM ユーザーガイド」の「ルートユーザー認証情報が必要なタスク」を参照してください。

### ユーザーとグループ

IAM ユーザーは、単一のユーザーまたはアプリケーションに対して特定のアクセス許可 AWS アカウント を持つ 内の ID です。可能であれば、パスワードやアクセスキーなどの長期的な認証情報を保有する IAM ユーザーを作成する代わりに、一時的な認証情報を使用することをお勧めします。ただし、IAM ユーザーでの長期的な認証情報が必要な特定のユースケースがある場合は、アクセスキーをローテーションすることをお勧めします。詳細については、「IAM ユーザーガイド」の「長期的な認証情報を必要とするユースケースのためにアクセスキーを定期的にローテーションする」を参照してください。

IAM グループは、IAM ユーザーの集団を指定するアイデンティティです。グループとしてサインインすることはできません。グループを使用して、複数のユーザーに対して一度に権限を指定できます。多数のユーザーグループがある場合、グループを使用することで権限の管理が容易になります。例えば、IAMAdmins という名前のグループを設定して、そのグループに IAM リソースを管理する許可を与えることができます。

ユーザーは、ロールとは異なります。ユーザーは 1 人の人または 1 つのアプリケーションに一意に関連付けられますが、ロールはそれを必要とする任意の人が引き受けるようになっています。ユーザーには永続的な長期の認証情報がありますが、ロールでは一時認証情報が提供されます。詳細については、「IAM ユーザーガイド」の「IAM ユーザーに関するユースケース」を参照してください。

### IAM ロール

IAM ロールは、特定のアクセス許可 AWS アカウント を持つ 内の ID です。これは IAM ユーザーに似ていますが、特定のユーザーには関連付けられていません。で IAM ロールを一時的に引き受ける

には AWS Management Console、ユーザーから IAM ロール (コンソール) に切り替えることができます。ロールを引き受けるには、 または AWS API オペレーションを AWS CLI 呼び出すか、カスタム URL を使用します。ロールを使用する方法の詳細については、「IAM ユーザーガイド」の「ロールを引き受けるための各種方法」を参照してください。

IAM ロールと一時的な認証情報は、次の状況で役立ちます:

- フェデレーションユーザーアクセス – フェデレーティッド ID に許可を割り当てるには、ロールを作成してそのロールの許可を定義します。フェデレーティッド ID が認証されると、その ID はロールに関連付けられ、ロールで定義されている許可が付与されます。フェデレーションのロールについては、「IAM ユーザーガイド」の「サードパーティー ID プロバイダー (フェデレーション) 用のロールを作成する」を参照してください。IAM Identity Center を使用する場合は、許可セットを設定します。アイデンティティが認証後にアクセスできるものを制御するため、IAM Identity Center は、権限セットを IAM のロールに関連付けます。アクセス許可セットの詳細については、「AWS IAM Identity Center User Guide」の「Permission sets」を参照してください。
- 一時的な IAM ユーザー権限 - IAM ユーザーまたはロールは、特定のタスクに対して複数の異なる権限を一時的に IAM ロールで引き受けることができます。
- クロスアカウントアクセス - IAM ロールを使用して、自分のアカウントのリソースにアクセスすることを、別のアカウントの人物 (信頼済みプリンシパル) に許可できます。クロスアカウントアクセス権を付与する主な方法は、ロールを使用することです。ただし、一部の では AWS のサービス、 (プロキシとしてロールを使用する代わりに) リソースに直接ポリシーをアタッチできます。クロスアカウントアクセスにおけるロールとリソースベースのポリシーの違いについては、「IAM ユーザーガイド」の「IAM でのクロスアカウントのリソースへのアクセス」を参照してください。
- クロスサービスアクセス — 一部の は他の の機能 AWS のサービス を使用します AWS のサービス。例えば、あるサービスで呼び出しを行うと、通常そのサービスによって Amazon EC2 でアプリケーションが実行されたり、Amazon S3 にオブジェクトが保存されたりします。サービスでは、呼び出し元プリンシパルの許可、サービスロール、またはサービスリンクロールを使用してこれを行う場合があります。
  - 転送アクセスセッション (FAS) – IAM ユーザーまたはロールを使用してアクションを実行すると AWS、プリンシパルと見なされます。一部のサービスを使用する際に、アクションを実行することで、別のサービスの別のアクションがトリガーされることがあります。FAS は、 を呼び出すプリンシパルのアクセス許可を AWS のサービス、ダウンストリームサービス AWS のサービス へのリクエストをリクエストする と組み合わせて使用します。FAS リクエストは、サービスが他の AWS のサービス またはリソースとのやり取りを完了する必要があるリクエストを受け取った場合にのみ行われます。この場合、両方のアクションを実行するためのアクセス許可が必

要です。FAS リクエストを行う際のポリシーの詳細については、「転送アクセスセッション」を参照してください。

- サービスロール - サービスがユーザーに代わってアクションを実行するために引き受ける IAM ロールです。IAM 管理者は、IAM 内からサービスロールを作成、変更、削除することができます。詳細については、「IAM ユーザーガイド」の「AWS のサービスに許可を委任するロールを作成する」を参照してください。
- サービスにリンクされたロール – サービスにリンクされたロールは、 にリンクされたサービスロールの一種です AWS のサービス。サービスは、ユーザーに代わってアクションを実行するロールを引き受けることができます。サービスにリンクされたロールは に表示され AWS アカウント 、サービスによって所有されます。IAM 管理者は、サービスリンクロールのアクセス許可を表示できますが、編集することはできません。

- Amazon EC2 で実行されているアプリケーション – IAM ロールを使用して、EC2 インスタンスで実行され、 AWS CLI または AWS API リクエストを行うアプリケーションの一時的な認証情報を管理できます。これは、EC2 インスタンス内でのアクセスキーの保存に推奨されます。EC2 インスタンスに AWS ロールを割り当て、そのすべてのアプリケーションで使用できるようにするには、インスタンスにアタッチされたインスタンスプロファイルを作成します。インスタンスプロファイルにはロールが含まれ、EC2 インスタンスで実行されるプログラムは一時的な認証情報を取得できます。詳細については、「IAM ユーザーガイド」の「Amazon EC2 インスタンスで実行されるアプリケーションに IAM ロールを使用して許可を付与する」を参照してください。

## ポリシーを使用したアクセスの管理

でアクセスを制御する AWS には、ポリシーを作成し、ID AWS またはリソースにアタッチします。ポリシーは AWS 、アイデンティティまたはリソースに関連付けられているときにアクセス許可を定義する のオブジェクトです。 は、プリンシパル (ユーザー、ルートユーザー、またはロールセッション) がリクエストを行うときに、これらのポリシー AWS を評価します。ポリシーでの権限により、リクエストが許可されるか拒否されるかが決まります。ほとんどのポリシーは JSON ドキュメント AWS として に保存されます。JSON ポリシードキュメントの構造と内容の詳細については、「IAM ユーザーガイド」の「JSON ポリシー概要」を参照してください。

管理者は JSON AWS ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどのリソースに対してどのような条件下でアクションを実行できるかということです。

デフォルトでは、ユーザーやロールに権限はありません。IAM 管理者は、リソースで必要なアクションを実行するための権限をユーザーに付与する IAM ポリシーを作成できます。その後、管理者はロールに IAM ポリシーを追加し、ユーザーはロールを引き受けることができます。

IAM ポリシーは、オペレーションの実行方法を問わず、アクションの許可を定義します。例えば、`iam:GetRole` アクションを許可するポリシーがあるとします。そのポリシーを持つユーザーは、 AWS Management Console、、 AWS CLIまたは AWS API からロール情報を取得できます。

### アイデンティティベースのポリシー

アイデンティティベースポリシーは、IAM ユーザーグループ、ユーザーのグループ、ロールなど、アイデンティティにアタッチできる JSON 許可ポリシードキュメントです。これらのポリシーは、ユーザーとロールが実行できるアクション、リソース、および条件をコントロールします。アイデンティティベースポリシーの作成方法については、「IAM ユーザーガイド」の「カスタマー管理ポリシーでカスタム IAM アクセス許可を定義する」を参照してください。

アイデンティティベースのポリシーは、さらにインラインポリシーまたはマネージドポリシーに分類できます。インラインポリシーは、単一のユーザー、グループ、またはロールに直接埋め込まれています。管理ポリシーは、 内の複数のユーザー、グループ、ロールにアタッチできるスタンドアロンポリシーです AWS アカウント。管理ポリシーには、 AWS 管理ポリシーとカスタマー管理ポリシーが含まれます。マネージドポリシーまたはインラインポリシーのいずれかを選択する方法については、「IAM ユーザーガイド」の「管理ポリシーとインラインポリシーのいずれかを選択する」を参照してください。

### リソースベースのポリシー

リソースベースのポリシーは、リソースに添付する JSON ポリシードキュメントです。リソースベースのポリシーには例として、IAM ロールの信頼ポリシーや Amazon S3 バケットポリシーがあげられます。リソースベースのポリシーをサポートするサービスでは、サービス管理者はポリシーを使用して特定のリソースへのアクセスを制御できます。ポリシーがアタッチされているリソースの場合、指定されたプリンシパルがそのリソースに対して実行できるアクションと条件は、ポリシーによって定義されます。リソースベースのポリシーでは、プリンシパルを指定する必要があります。プリンシパルには、アカウント、ユーザー、ロール、フェデレーティッドユーザー、または を含めることができます AWS のサービス。

リソースベースのポリシーは、そのサービス内にあるインラインポリシーです。リソースベースのポリシーでは、IAM の AWS マネージドポリシーを使用できません。

### アクセスコントロールリスト (ACL)

アクセスコントロールリスト (ACL) は、どのプリンシパル (アカウントメンバー、ユーザー、またはロール) がリソースにアクセスするための許可を持つかを制御します。ACL はリソースベースのポリシーに似ていますが、JSON ポリシードキュメント形式は使用しません。

Amazon S3、および Amazon VPC は AWS WAF、ACLs。ACL の詳細については、「Amazon Simple Storage Service デベロッパーガイド」の「アクセスコントロールリスト (ACL) の概要」を参照してください。

## その他のポリシータイプ

AWS は、一般的でない追加のポリシータイプをサポートしています。これらのポリシータイプでは、より一般的なポリシータイプで付与された最大の権限を設定できます。

- アクセス許可の境界 - アクセス許可の境界は、アイデンティティベースポリシーによって IAM エンティティ (IAM ユーザーまたはロール) に付与できる権限の上限を設定する高度な機能です。エンティティにアクセス許可の境界を設定できます。結果として得られる権限は、エンティティのアイデンティティベースポリシーとそのアクセス許可の境界の共通部分になります。`Principal` フィールドでユーザーまたはロールを指定するリソースベースのポリシーでは、アクセス許可の境界は制限されません。これらのポリシーのいずれかを明示的に拒否した場合、権限は無効になります。アクセス許可の境界の詳細については、「IAM ユーザーガイド」の「IAM エンティティのアクセス許可の境界」を参照してください。
- サービスコントロールポリシー (SCPs) – SCPs は、 の組織または組織単位 (OU) の最大アクセス許可を指定する JSON ポリシーです AWS Organizations。 AWS Organizations は、ビジネスが所有する複数の AWS アカウント をグループ化して一元管理するためのサービスです。組織内のすべての機能を有効にすると、サービスコントロールポリシー (SCP) を一部またはすべてのアカウントに適用できます。SCP は、各 を含むメンバーアカウントのエンティティのアクセス許可を制限します AWS アカウントのルートユーザー。Organizations と SCP の詳細については、「AWS Organizations ユーザーガイド」の「サービスコントロールポリシー (SCP)」を参照してください。
- リソースコントロールポリシー (RCP) – RCP は、所有する各リソースにアタッチされた IAM ポリシーを更新することなく、アカウント内のリソースに利用可能な最大数のアクセス許可を設定するために使用できる JSON ポリシーです。RCP は、メンバーアカウントのリソースのアクセス許可を制限し、組織に属しているかどうかにかかわらず AWS アカウントのルートユーザー、 を含む ID の有効なアクセス許可に影響を与える可能性があります。RCP をサポートする のリストを含む Organizations と RCP の詳細については、AWS Organizations RCPs「リソースコントロールポリシー (RCPs」を参照してください。 AWS のサービス
- セッションポリシー - セッションポリシーは、ロールまたはフェデレーションユーザーの一時的なセッションをプログラムで作成する際にパラメータとして渡す高度なポリシーです。結果としてセッションの権限は、ユーザーまたはロールのアイデンティティベースポリシーとセッションポリシーの共通部分になります。また、リソースベースのポリシーから権限が派生する場合もありま

す。これらのポリシーのいずれかを明示的に拒否した場合、権限は無効になります。詳細については、「IAM ユーザーガイド」の「セッションポリシー」を参照してください。

### 複数のポリシータイプ

1 つのリクエストに複数のタイプのポリシーが適用されると、結果として作成される権限を理解するのがさらに難しくなります。が複数のポリシータイプが関与する場合にリクエストを許可するかどうか AWS を決定する方法については、「IAM ユーザーガイド」の「ポリシー評価ロジック」を参照してください。

## Amazon Fraud Detector で IAM が機能する仕組み

IAM を使用して Amazon Fraud Detector へのアクセスを管理する前に、Amazon Fraud Detector で使用できる IAM 機能について理解しておく必要があります。Amazon Fraud Detector およびその他の AWS のサービスが IAM と連携する方法の概要を把握するには、IAM ユーザーガイドのAWS「IAM と連携する のサービス」を参照してください。

トピック

- Amazon Fraud Detector のアイデンティティベースポリシー
- Amazon Fraud Detector のリソースベースのポリシー
- Amazon Fraud Detector タグに基づく認可
- Amazon Fraud Detector IAM ロール

### Amazon Fraud Detector のアイデンティティベースポリシー

IAM アイデンティティベースのポリシーでは許可または拒否するアクションとリソース、またアクションを許可または拒否する条件を指定できます。Amazon Fraud Detector は、特定のアクション、リソース、および条件キーをサポートしています。JSON ポリシーで使用するすべての要素については、「IAM ユーザーガイド」の「IAM JSON ポリシー要素のリファレンス」を参照してください。

Amazon Fraud Detector の使用を開始するには、Amazon Fraud Detector オペレーションと必要なアクセス許可に制限されたアクセス権限を持つ ユーザーを作成することをお勧めします。必要に応じて他のアクセス許可を追加できます。`AmazonFraudDetectorFullAccessPolicy` および `AmazonS3FullAccess` のポリシーは、Amazon Fraud Detector を使用するために必要なアクセス許可を示します。これらのポリシーを使用した Amazon Fraud Detector の設定の詳細については、「Amazon Fraud Detector のセットアップ」を参照してください。

アクション

管理者は JSON AWS ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどのリソースに対してどのような条件下でアクションを実行できるかということです。

JSON ポリシーの `Action` 要素にはポリシー内のアクセスを許可または拒否するために使用できるアクションが記述されます。ポリシーアクションの名前は通常、関連付けられた AWS API オペレーションと同じです。一致する API オペレーションのない許可のみのアクションなど、いくつかの例外があります。また、ポリシーに複数のアクションが必要なオペレーションもあります。これらの追加アクションは依存アクションと呼ばれます。

このアクションは関連付けられたオペレーションを実行するためのアクセス許可を付与するポリシーで使用されます。

Amazon Fraud Detector のポリシーアクションは、アクションの前にプレフィックス `frauddetector:` を使用します。例えば、Amazon Fraud Detector `CreateRule` API オペレーションを使用してデータセットを作成するには、ポリシーに `frauddetector:CreateRule` アクションを含めます。ポリシーステートメントには`Action` または `NotAction` 要素を含める必要があります。Amazon Fraud Detector は、このサービスで実行できるタスクを記述する独自のアクションのセットを定義します。

単一のステートメントに複数のアクションを指定するには次のようにコンマで区切ります。

```
"Action": [
      "frauddetector:action1",
      "frauddetector:action2"
```

ワイルドカード (*) を使用して複数アクションを指定できます。例えば、`Describe` という単語で始まるすべてのアクションを指定するには次のアクションを含めます。

```
"Action": "frauddetector:Describe*"
```

Amazon Fraud Detector アクションのリストを確認するには、IAM ユーザーガイドの「Amazon Fraud Detector で定義されるアクション」を参照してください。

リソース

管理者は JSON AWS ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどのリソースに対してどのような条件下でアクションを実行できるかということです。

Resource JSON ポリシー要素はアクションが適用されるオブジェクトを指定します。ステートメントにはResource または NotResource 要素を含める必要があります。ベストプラクティスとして、Amazon リソースネーム (ARN) を使用してリソースを指定します。これは、リソースレベルの許可と呼ばれる特定のリソースタイプをサポートするアクションに対して実行できます。

オペレーションのリスト化など、リソースレベルの権限をサポートしないアクションの場合は、ステートメントがすべてのリソースに適用されることを示すために、ワイルドカード (*) を使用します。

```
"Resource": "*"
```

Amazon Fraud Detector で定義されるリソースタイプは、すべての Amazon Fraud Detector リソース ARN をリストしています。

例えば、ステートメントで my_detector ディテクターを指定するには、次の ARN を使用します。

```
"Resource": "arn:aws:frauddetector:us-east-1:123456789012:detector/my_detector"
```

ARN の形式の詳細については、「Amazon リソースネーム (ARNs)」と AWS 「サービス名前空間」を参照してください。

特定のアカウントに属するすべてのディテクターを指定するには、ワイルドカード (*) を使用します。

```
"Resource": "arn:aws:frauddetector:us-east-1:123456789012:detector/*"
```

リソースを作成するためのアクションなど、Amazon Fraud Detector アクションには特定のリソースで実行できないものがあります。このような場合はワイルドカード *を使用する必要があります。

```
"Resource": "*"
```

Amazon Fraud Detector のリソースタイプとそれらの ARN のリストを確認するには、IAM ユーザーガイドの「Amazon Fraud Detector で定義されるリソースタイプ」を参照してください。どのアク

ションで各リソースの ARN を指定できるかについては、「Amazon Fraud Detector で定義されるアクション」を参照してください。

条件キー

管理者は JSON AWS ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどのリソースに対してどのような条件下でアクションを実行できるかということです。

Condition 要素 (または Condition ブロック) を使用すると、ステートメントが有効な条件を指定できます。Condition 要素はオプションです。イコールや未満などの 条件演算子 を使用して条件式を作成して、ポリシーの条件とリクエスト内の値を一致させることができます。

1 つのステートメントに複数の Condition 要素を指定する場合、または 1 つの Condition 要素に複数のキーを指定する場合、 AWS では AND 論理演算子を使用してそれらを評価します。1 つの条件キーに複数の値を指定すると、 は論理ORオペレーションを使用して条件 AWS を評価します。ステートメントの権限が付与される前にすべての条件が満たされる必要があります。

条件を指定する際にプレースホルダー変数も使用できます。例えば IAM ユーザーに、IAM ユーザー名がタグ付けされている場合のみリソースにアクセスできる権限を付与することができます。詳細については、「IAM ユーザーガイド」の「IAM ポリシーの要素: 変数およびタグ」を参照してください。

AWS は、グローバル条件キーとサービス固有の条件キーをサポートしています。すべての AWS グローバル条件キーを確認するには、「IAM ユーザーガイド」のAWS 「グローバル条件コンテキストキー」を参照してください。

Amazon Fraud Detector では独自の条件キーが定義されており、また一部のグローバル条件キーの使用がサポートされています。すべての AWS グローバル条件キーを確認するには、IAM ユーザーガイドのAWS 「グローバル条件コンテキストキー」を参照してください。

Amazon Fraud Detector 条件キーのリストについては、IAM ユーザーガイドの「Amazon Fraud Detector の条件キー」を参照してください。どのアクションおよびリソースと条件キーを使用できるかについては、「Amazon Fraud Detector で定義されるアクション」を参照してください。

例

Amazon Fraud Detector のアイデンティティベースポリシーの例を確認するには、「Amazon Fraud Detector のアイデンティティベースポリシーの例」を参照してください。

## Amazon Fraud Detector のリソースベースのポリシー

Amazon Fraud Detector では、 リソースベースのポリシーはサポートされていません。

## Amazon Fraud Detector タグに基づく認可

タグは、Amazon Fraud Detector リソースにアタッチする、または Amazon Fraud Detector へのリクエストで渡すことができます。タグに基づいてアクセスを制御するには`aws:ResourceTag/key-name`、`aws:RequestTag/key-name`、または `aws:TagKeys` の条件キーを使用して、ポリシーの条件要素でタグ情報を提供します。

## Amazon Fraud Detector IAM ロール

IAM ロールは、特定のアクセス許可を持つ AWS アカウント内のエンティティです。

### Amazon Fraud Detector での一時的な認証情報の使用

一時的な認証情報を使用して、フェデレーションでサインインする、IAM 役割を引き受ける、またはクロスアカウント役割を引き受けることができます。一時的なセキュリティ認証情報を取得するには、AssumeRole や GetFederationToken などの AWS STS API オペレーションを呼び出します。

### Amazon Fraud Detector は一時的な認証情報の使用をサポート

### サービスにリンクされた役割

サービスにリンクされたロールを使用すると、 AWS サービスは他の サービスのリソースにアクセスして、ユーザーに代わってアクションを実行できます。サービスリンクロールは IAM アカウント内に表示され、サービスによって所有されます。IAM 管理者は、サービスリンクロールの許可を表示できますが、編集することはできません。

### Amazon Fraud Detector は、サービスにリンクされたロールをサポートしていません。

### サービス役割

この機能により、ユーザーに代わってサービスがサービス役割を引き受けることが許可されます。この役割により、サービスがお客様に代わって他のサービスのリソースにアクセスし、アクションを完了することが許可されます。サービスロールは、 アカウントに表示され、サービスによって所有されます。つまり、 管理者は、このロールのアクセス許可を変更できます。ただし、それにより、サービスの機能が損なわれる場合があります。

### Amazon Fraud Detector は、サービスロールをサポートしています。

# Amazon Fraud Detector のアイデンティティベースポリシーの例

デフォルトでは、ユーザーと IAM ロールには Amazon Fraud Detector リソースを作成または変更するアクセス許可はありません。また、 AWS Management Console、 AWS CLI、または AWS API を使用してタスクを実行することはできません。IAM 管理者は、指定されたリソースで特定の API 操作を実行するための許可をユーザーとロールに付与する IAM ポリシーを作成する必要があります。続いて、管理者はそれらのアクセス許可が必要なユーザーまたはグループにそのポリシーをアタッチします。

JSON ポリシードキュメントのこれらの例を使用して、IAM アイデンティティベースのポリシーを作成する方法については、「IAM ユーザーガイド」の「JSON タブでのポリシーの作成」を参照してください。

トピック


- ポリシーに関するベストプラクティス
- Amazon Fraud Detector の AWS 管理 (事前定義) ポリシー
- 自分の権限の表示をユーザーに許可する
- Amazon Fraud Detector リソースへのフルアクセスを許可する
- Amazon Fraud Detector リソースへの読み取り専用アクセスを許可する
- 特定のリソースへのアクセスを許可する
- デュアルモード API の使用時に特定のリソースへのアクセスを許可する
- タグに基づくアクセスの制限


## ポリシーに関するベストプラクティス

ID ベースのポリシーは、アカウント内で誰かが Amazon Fraud Detector リソースを作成、アクセス、または削除できるかどうかを決定します。これらのアクションを実行すると、 AWS アカウントに料金が発生する可能性があります。アイデンティティベースポリシーを作成したり編集したりする際には、以下のガイドラインと推奨事項に従ってください:

- AWS 管理ポリシーを開始し、最小特権のアクセス許可に移行する – ユーザーとワークロードにアクセス許可の付与を開始するには、多くの一般的なユースケースにアクセス許可を付与するAWS 管理ポリシーを使用します。これらは で使用できます AWS アカウント。ユースケースに固有の AWS カスタマー管理ポリシーを定義することで、アクセス許可をさらに減らすことをお勧めします。詳細については、「IAM ユーザーガイド」の「AWS マネージドポリシー」または「ジョブ機能のAWS マネージドポリシー」を参照してください。

- 最小特権を適用する – IAM ポリシーで許可を設定する場合は、タスクの実行に必要な許可のみを付与します。これを行うには、特定の条件下で特定のリソースに対して実行できるアクションを定義します。これは、最小特権アクセス許可とも呼ばれています。IAM を使用して許可を適用する方法の詳細については、「IAM ユーザーガイド」の「IAM でのポリシーとアクセス許可」を参照してください。
- IAM ポリシーで条件を使用してアクセスをさらに制限する - ポリシーに条件を追加して、アクションやリソースへのアクセスを制限できます。例えば、ポリシー条件を記述して、すべてのリクエストを SSL を使用して送信するように指定できます。条件を使用して、サービスアクションが などの特定の を通じて使用されている場合に AWS のサービス、サービスアクションへのアクセスを許可することもできます AWS CloudFormation。詳細については、「IAM ユーザーガイド」の「IAM JSON ポリシー要素:条件」を参照してください。
- IAM Access Analyzer を使用して IAM ポリシーを検証し、安全で機能的な権限を確保する - IAM Access Analyzer は、新規および既存のポリシーを検証して、ポリシーが IAM ポリシー言語 (JSON) および IAM のベストプラクティスに準拠するようにします。IAM アクセスアナライザーは 100 を超えるポリシーチェックと実用的な推奨事項を提供し、安全で機能的なポリシーの作成をサポートします。詳細については、「IAM ユーザーガイド」の「IAM Access Analyzer でポリシーを検証する」を参照してください。
- 多要素認証 (MFA) を要求する – で IAM ユーザーまたはルートユーザーを必要とするシナリオがある場合は AWS アカウント、MFA をオンにしてセキュリティを強化します。API オペレーションが呼び出されるときに MFA を必須にするには、ポリシーに MFA 条件を追加します。詳細については、「IAM ユーザーガイド」の「MFA を使用した安全な API アクセス」を参照してください。

IAM でのベストプラクティスの詳細については、「IAM ユーザーガイド」の「IAM でのセキュリティのベストプラクティス」を参照してください。

## Amazon Fraud Detector の AWS 管理 (事前定義) ポリシー

AWS は、 によって作成および管理されるスタンドアロン IAM ポリシーを提供することで、多くの一般的なユースケースに対処します AWS。これらの AWS 管理ポリシーは、一般的なユースケースに必要なアクセス許可を付与するため、必要なアクセス許可を調査する必要がなくなります。詳細については、AWS Identity and Access Management 「 管理ユーザーガイド」の「AWS 管理ポリシー」を参照してください。

アカウントのユーザーにアタッチできる次の AWS 管理ポリシーは、Amazon Fraud Detector に固有のものです。

`AmazonFraudDetectorFullAccess`: 以下を含む、Amazon Fraud Detector のリソース、アクションおよびサポートされているオペレーションにフルアクセスできます

- Amazon SageMaker AI のすべてのモデルエンドポイントを一覧表示して記述する
- アカウント内のすべての IAM ロールを一覧表示する
- Amazon S3 バケットをすべて一覧表示する
- IAM パスロールが Amazon Fraud Detector にロールを渡すことを許可する

このポリシーでは、無制限の S3 アクセスは提供されません。モデルトレーニングデータセットを S3 にアップロードする必要がある場合は、`AmazonS3FullAccess` 管理ポリシー (またはスコープダウンカスタム Amazon S3 アクセスポリシー) も必要です。

ポリシーのアクセス許可は、IAM コンソールにサインインしてポリシー名で検索することで確認できます。独自のカスタム IAM ポリシーを作成し、必要に応じて Amazon Fraud Detector のアクションとリソースのためのアクセスを許可することもできます。これらのカスタムポリシーは、それらを必要とする ユーザーにアタッチできます。

## 自分の権限の表示をユーザーに許可する

この例では、ユーザーアイデンティティにアタッチされたインラインおよびマネージドポリシーの表示を IAM ユーザーに許可するポリシーの作成方法を示します。このポリシーには、コンソールで、または AWS CLI または AWS API を使用してプログラムでこのアクションを実行するアクセス許可が含まれています。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
```

```
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

## Amazon Fraud Detector リソースへのフルアクセスを許可する

次の例では、すべての Amazon Fraud Detector リソースとアクションへの AWS アカウント フルアクセスを のユーザーに付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "frauddetector:*"
            ],
            "Resource": "*"
        }
    ]
}
```

## Amazon Fraud Detector リソースへの読み取り専用アクセスを許可する

この例では、Amazon Fraud Detector リソースへの AWS アカウント 読み取り専用アクセスを ユーザーに付与します。

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "frauddetector:GetEventTypes",
                "frauddetector:BatchGetVariable",
                "frauddetector:DescribeDetector",
                "frauddetector:GetModelVersion",
                "frauddetector:GetEventPrediction",
                "frauddetector:GetExternalModels",
                "frauddetector:GetLabels",
                "frauddetector:GetVariables",
                "frauddetector:GetDetectors",
                "frauddetector:GetRules",
                "frauddetector:ListTagsForResource",
                "frauddetector:GetKMSEncryptionKey",
                "frauddetector:DescribeModelVersions",
                "frauddetector:GetDetectorVersion",
                "frauddetector:GetPrediction",
                "frauddetector:GetOutcomes",
                "frauddetector:GetEntityTypes",
                "frauddetector:GetModels"
            ],
            "Resource": "*"
        }
    ]
}
```

## 特定のリソースへのアクセスを許可する

このリソースレベルのポリシーの例では、1 つの特定の Detector リソースを除くすべてのアクションとリソース AWS アカウント へのアクセスを のユーザーに許可します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "frauddetector:*"
            ],
            "Resource": "*"
        },
```

```
        {
            "Effect": "Deny",
            "Action": [
                "frauddetector:*Detector"
            ],
            "Resource": "arn:${Partition}:frauddetector:${Region}:${Account}:detector/
${detector-name}"
        }
    ]
}
```

## デュアルモード API の使用時に特定のリソースへのアクセスを許可する

Amazon Fraud Detector は、List オペレーションと Describe オペレーションの両方として機能するデュアルモード取得 APIs を提供します。パラメータなしで呼び出されたデュアルモード API は、に関連付けられた指定されたリソースのリストを返します AWS アカウント。パラメータで呼び出されたデュアルモード API は、指定されたリソースの詳細を返します。リソースは、モデル、変数、イベントタイプ、またはエンティティタイプです。

デュアルモード APIsIAM ポリシーでリソースレベルのアクセス許可をサポートします。ただし、リソースレベルのアクセス許可は、リクエストの一部として 1 つ以上のパラメータが指定されている場合にのみ適用されます。たとえば、ユーザーが GetVariables API を呼び出して変数名を指定し、変数リソースまたは変数名に IAM 拒否ポリシーがアタッチされている場合、ユーザーはAccessDeniedExceptionエラーを受け取ります。ユーザーが GetVariables API を呼び出して変数名を指定しない場合、すべての変数が返されるため、情報漏洩が発生する可能性があります。

ユーザーが特定のリソースの詳細のみを表示できるようにするには、IAM 拒否NotResourceポリシーで IAM ポリシー要素を使用します。このポリシー要素を IAM 拒否ポリシーに追加すると、ユーザーは NotResourceブロックで指定されたリソースの詳細のみを表示できます。詳細については、IAM ユーザーガイドの「IAM JSON ポリシー要素: NotResource」を参照してください。

次のポリシー例では、ユーザーに Amazon Fraud Detector のすべてのリソースへのアクセスを許可します。ただし、NotResourceポリシー要素は、GetVariables API コールをプレフィックス user*、、job_*および を持つ変数名のみに制限するために使用されますvar*。

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
    "Effect": "Allow",
    "Action": "frauddetector:*",
```

```
    "Resource": "*"
  },
 {
    "Effect": "Deny",
    "Action": "frauddetector:GetVariables",
    "NotResource": [
       "arn:aws:frauddetector:*:*:variable/user*",
       "arn:aws:frauddetector:*:*:variable/job_*",
       "arn:aws:frauddetector:*:*:variable/var*"
    ]
  }
 ]
}
```

レスポンス

このポリシー例では、レスポンスは次の動作を示します。

- 変数名を含まない GetVariables 呼び出しでは、リクエストが Deny ステートメントにマッピングされるため、AccessDeniedException エラーが発生します。
- 許可されていない変数名を含む GetVariables 呼び出しでは、変数名が NotResource ブロック内の変数名にマッピングされないため、AccessDeniedException エラーが発生します。たとえば、変数名を持つ GetVariables 呼び出しは AccessDeniedException エラー email_address になります。
- NotResource ブロック内の変数名に一致する変数名を含む GetVariables 呼び出しが想定どおりに返されます。たとえば、変数名を含む GetVariables 呼び出しは、job_cpa 変数の詳細 job_cpa を返します。

## タグに基づくアクセスの制限

このポリシーの例では、リソースタグに基づいて Amazon Fraud Detector へのアクセスを制限する方法を説明します。この例では、次のことを前提としています。

- で、Team1 と Team2 という 2 つの異なるグループを定義 AWS アカウント しました。
- 4 つのディテクターが作成されました
- Team1 のメンバーが 2 つのディテクターで API 呼び出しを行うことを許可したいと考えています
- Team2 のメンバーが他の 2 つのディテクターで API 呼び出しを行うことを許可したいと考えています

API コールへのアクセスをコントロールするには (例)

1. Team1 が使用するディテクターに、キー Project と値 A を含むタグを追加します。
2. Team2 が使用するディテクターに、キー Project と値 B を含むタグを追加します。
3. キー Project と値 B のタグを含むディテクターへのアクセスを拒否する ResourceTag 条件で IAM ポリシーを作成し、そのポリシーを Team1 にアタッチします。
4. キー Project と値 A のタグを含むディテクターへのアクセスを拒否する ResourceTag 条件で IAM ポリシーを作成し、そのポリシーを Team2 にアタッチします。

以下は、Project のキーと B の値を含むタグを持つ Amazon Fraud Detector リソースに対する特定のアクションを拒否するポリシーの例です。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "frauddetector:*",
      "Resource": "*"
    },
    {
      "Effect": "Deny",

      "Action": [

        "frauddetector:CreateModel",
        "frauddetector:CancelBatchPredictionJob",
        "frauddetector:CreateBatchPredictionJob",
        "frauddetector:DeleteBatchPredictionJob",
        "frauddetector:DeleteDetector"
      ],

      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Project": "B"
        }
      }
    }
  ]
}
```

## 混乱した代理の防止

混乱した代理問題は、アクションを実行するアクセス許可を持たないエンティティが、より特権のあるエンティティにアクションを実行するよう強制する可能性がある場合に発生します。 は、サードパーティー (クロスアカウントと呼ばれる) または他の AWS サービス (クロスサービスと呼ばれる) にアカウント内のリソースへのアクセスを提供する場合、アカウントを保護するのに役立つツール AWS を提供します。

サービス間の混乱した代理問題は、あるサービス (呼び出し元のサービス) が別のサービス (呼び出し元のサービス) を呼び出すと発生する可能性があります。呼び出し元サービスは、本来ならアクセスすることが許可されるべきではない方法でその許可を使用して、別のお客様のリソースに対する処理を実行するように操作される場合があります。これを防ぐには、サービスのリソースへのアクセス権が付与されたサービスプリンシパルを持つすべてのサービスのデータを保護するのに役立つポリシーを作成できます。

Amazon Fraud Detector では、アクセス許可ポリシーでサービスロールを使用して、サービスがユーザーに代わって別のサービスのリソースにアクセスすることを許可できます。ロールには 2 つのポリシーが必要です。ロールを引き受けることができるプリンシパルを指定する、ロールの信頼ポリシーと、ロールで実行できる操作を指定する、アクセス許可ポリシーです。サービスがユーザーに代わってロールを引き受ける場合、サービスプリンシパルが `sts:AssumeRole` アクションを実行できるように、ロールの信頼ポリシーで許可されている必要があります。サービスが を呼び出すと`sts:AssumeRole`、 は、サービスプリンシパルがロールのアクセス許可ポリシーで許可されているリソースにアクセスするために使用する一時的なセキュリティ認証情報のセット AWS STS を返します。

サービス間の混乱した代理問題を防ぐために、Amazon Fraud Detector では、ロール信頼ポリシーで `aws:SourceArn`および `aws:SourceAccount` グローバル条件コンテキストキーを使用して、ロールへのアクセスを、予想されるリソースによって生成されたリクエストのみに制限することをお勧めします。

`aws:SourceAccount` はアカウント ID を指定し、 はクロスサービスアクセスに関連付けられたリソースの ARN `aws:SourceArn`を指定します。は、ARN 形式を使用して指定`aws:SourceArn`する必要があります。同じポリシーステートメントで使用する場合は`aws:SourceArn`、`aws:SourceAccount`と の両方が同じアカウント ID を使用していることを確認します。

混乱した代理問題から保護するための最も効果的な方法は、リソースの完全な ARN を指定して `aws:SourceArn` グローバル条件コンテキストキーを使用することです。リソースの完全な

ARN がわからない場合、または複数のリソースを指定する場合は、ARN の不明な部分にワイルドカード (*) を含むaws:SourceArnグローバルコンテキスト条件キーを使用します。例えば、arn:aws:*servicename*:*:*123456789012*:*。アクセス許可ポリシーで使用できる Amazon Fraud Detector リソースとアクションの詳細については、「Amazon Fraud Detector のアクション、リソース、および条件キー」を参照してください。

次のロール信頼ポリシーの例では、 aws:SourceArn 条件キーでワイルドカード(*) を使用して、Amazon Fraud Detector がアカウント ID に関連付けられた複数のリソースにアクセスできるようにします。

```
{
        "Version": "2012-10-17",
        "Statement": [
             {
               "Effect": "Allow",
               "Principal": {
               "Service": [
                   "frauddetector.amazonaws.com"
                   ]
               },
               "Action": "sts:AssumeRole",
               "Condition": {
               "StringEquals": {
                   "aws:SourceAccount": "123456789012"
               },
                 "StringLike": {
                    "aws:SourceArn": "arn:aws:frauddetector:us-west-2:123456789012:*"
        }
      }
    }
  ]
}
```

次のロール信頼ポリシーは、Amazon Fraud Detector にexternal-modelリソースのみへのアクセスを許可します。条件ブロックの aws:SourceArnパラメータに注目してください。リソース修飾子は、PutExternalModelAPI コールを行うために提供されるモデルエンドポイントを使用して構築されます。

```
    {
  "Version": "2012-10-17",
```

```
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "frauddetector.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        },
        "StringLike": {
          "aws:SourceArn": "arn:aws:frauddetector:us-west-2:123456789012:external-
model/MyExternalModeldoNotDelete-ReadOnly"
        }
      }
    }
  ]
}
```

# Amazon Fraud Detector アイデンティティとアクセスのトラブルシューティング

以下の情報を使用して、Amazon Fraud Detector と IAM の使用時に発生する可能性がある一般的な問題の診断と修正に役立てます。

トピック

- Amazon Fraud Detector でアクションを実行する権限がない
- iam:PassRole を実行する権限がない
- AWS アカウント以外のユーザーに Amazon Fraud Detector リソースへのアクセスを許可したい
- Amazon Fraud Detector は、指定されたロールを引き受けることができませんでした

## Amazon Fraud Detector でアクションを実行する権限がない

にアクションを実行する権限がないと AWS Management Console 通知された場合は、管理者に連絡してサポートを依頼する必要があります。管理者とは、サインイン認証情報を提供した担当者です。

次の例のエラーは、mateojacksonユーザーがコンソールを使用して*######*の詳細を表示しようとしているが、 アクセスfrauddetector:*GetDetectors*許可がない場合に発生します。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 frauddetector:GetDetectors on resource: my-example-detector
```

この場合、Mateo は、frauddetector:*GetDetectors* アクションを使用して *my-example-detector* リソースにアクセスできるように、管理者にポリシーの更新を依頼します。

### iam:PassRole を実行する権限がない

iam:PassRole アクションを実行する権限がないというエラーが表示された場合は、ポリシーを更新して Amazon Fraud Detector にロールを渡すことができるようにする必要があります。

一部の AWS のサービス では、新しいサービスロールまたはサービスにリンクされたロールを作成する代わりに、そのサービスに既存のロールを渡すことができます。そのためには、サービスにロールを渡す権限が必要です。

以下の例のエラーは、marymajor という IAM ユーザーがコンソールを使用して Amazon Fraud Detector でアクションを実行しようする場合に発生します。ただし、このアクションをサービスが実行するには、サービスロールから付与された権限が必要です。メアリーには、ロールをサービスに渡す許可がありません。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

この場合、Mary のポリシーを更新してメアリーに iam:PassRole アクションの実行を許可する必要があります。

サポートが必要な場合は、 AWS 管理者にお問い合わせください。サインイン資格情報を提供した担当者が管理者です。

### AWS アカウント以外のユーザーに Amazon Fraud Detector リソースへのアクセスを許可したい

他のアカウントのユーザーや組織外の人が、リソースにアクセスするために使用できるロールを作成できます。ロールの引き受けを委託するユーザーを指定できます。リソースベースのポリシーまたはアクセスコントロールリスト (ACL) をサポートするサービスの場合、それらのポリシーを使用して、リソースへのアクセスを付与できます。

詳細については、以下を参照してください:

- Amazon Fraud Detector がこれらの機能をサポートしているかどうかを確認するには、「Amazon Fraud Detector で IAM が機能する仕組み」を参照してください。
- 所有 AWS アカウント している のリソースへのアクセスを提供する方法については、IAM ユーザーガイドの「所有 AWS アカウント している別の の IAM ユーザーへのアクセスを提供する」を参照してください。
- リソースへのアクセスをサードパーティーに提供する方法については AWS アカウント、IAM ユーザーガイドの「サードパーティー AWS アカウント が所有する へのアクセスを提供する」を参照してください。
- ID フェデレーションを介してアクセスを提供する方法については、「IAM ユーザーガイド」の「外部で認証されたユーザー (ID フェデレーション) へのアクセスの許可」を参照してください。
- クロスアカウントアクセスにおけるロールとリソースベースのポリシーの使用方法の違いについては、「IAM ユーザーガイド」の「IAM でのクロスアカウントのリソースへのアクセス」を参照してください。

## Amazon Fraud Detector は、指定されたロールを引き受けることができませんでした

Amazon Fraud Detector が特定のロールを引き受けることができなかったというエラーが表示された場合は、指定したロールの信頼関係を更新する必要があります。Amazon Fraud Detector を信頼できるエンティティとして指定することで、サービスはロールを引き受けられます。Amazon Fraud Detector を使用してロールを作成すると、この信頼関係が自動的に設定されます。Amazon Fraud Detector によって作成されない IAM ロールに対しては、この信頼関係を確立する必要があります。

Amazon Fraud Detector への既存のロールの信頼関係を確立するには

1. IAM コンソール (https://console.aws.amazon.com/iam/) を開きます。
2. ナビゲーションペインで [ロール] を選択します。
3. 変更するロールの名前を選択し、[信頼関係] タブを選択します。
4. [信頼関係の編集] を選択します。
5. [ポリシードキュメント] の下に以下の内容を貼り付け、[信頼ポリシーの更新] を選択します。

```
{
    "Version": "2012-10-17",
    "Statement": [ {
        "Effect": "Allow",
        "Principal": {
            "Service": "frauddetector.amazonaws.com"
```

```
            },
            "Action": "sts:AssumeRole"
        } ]
    }
```

# Amazon Fraud Detector でのログ記録とモニタリング

AWS には、Amazon Fraud Detector を監視して異常を検出した場合に報告し、必要に応じて自動的に対処するために、次のモニタリングツールが用意されています。

- Amazon CloudWatch は、 AWS リソースと で実行されるアプリケーションを AWS リアルタイムでモニタリングします。CloudWatch の詳細については、Amazon CloudWatch ユーザーガイドを参照してください。
- AWS CloudTrail は、 AWS アカウントによって、またはアカウントに代わって行われた API コールおよび関連イベントをキャプチャし、指定した Amazon S3 バケットにログファイルを配信します。CloudTrail の詳細については、AWS CloudTrail ユーザーガイドを参照してください。

Amazon Fraud Detector のモニタリングの詳細については、「Amazon Fraud Detector のモニタリング」を参照してください。

# Amazon Fraud Detector のコンプライアンス検証

サードパーティーの監査者は、SOC、PCI、FedRAMP、HIPAA などの複数のコンプライアンスプログラムの一環として、 AWS サービスのセキュリティと AWS コンプライアンスを評価します。

AWS のサービス が特定のコンプライアンスプログラムの範囲内にあるかどうかを確認するには、AWS のサービス「コンプライアンスプログラムによる対象範囲内」を参照して、関心のあるコンプライアンスプログラムを選択します。一般的な情報については、AWS 「 Compliance ProgramsAssurance」を参照してください。

を使用して、サードパーティーの監査レポートをダウンロードできます AWS Artifact。詳細については、「Downloading Reports in AWS Artifact」を参照してください。

を使用する際のお客様のコンプライアンス責任 AWS のサービス は、お客様のデータの機密性、貴社のコンプライアンス目的、適用される法律および規制によって決まります。 は、コンプライアンスに役立つ以下のリソース AWS を提供します。

- セキュリティのコンプライアンスとガバナンス – これらのソリューション実装ガイドでは、アーキテクチャ上の考慮事項について説明し、セキュリティとコンプライアンスの機能をデプロイする手順を示します。
- HIPAA 対応サービスのリファレンス – HIPAA 対応サービスの一覧が提供されています。すべての AWS のサービス が HIPAA の対象となるわけではありません。
- AWS コンプライアンスリソース – このワークブックとガイドのコレクションは、お客様の業界や地域に適用される場合があります。
- AWS カスタマーコンプライアンスガイド – コンプライアンスの観点から責任共有モデルを理解します。このガイドは、複数のフレームワーク (米国国立標準技術研究所 (NIST)、Payment Card Industry Security Standards Council (PCI)、国際標準化機構 (ISO) を含む) のセキュリティコントロールを保護し、そのガイダンスに AWS のサービス マッピングするためのベストプラクティスをまとめたものです。
- 「 デベロッパーガイド」の「ルールによるリソースの評価」 – この AWS Config サービスは、リソース設定が内部プラクティス、業界ガイドライン、および規制にどの程度準拠しているかを評価します。 AWS Config
- AWS Security Hub – これにより AWS のサービス 、 内のセキュリティ状態を包括的に把握できます AWS。Security Hub では、セキュリティコントロールを使用して AWS リソースを評価し、セキュリティ業界標準とベストプラクティスに対するコンプライアンスをチェックします。サポートされているサービスとコントロールの一覧については、Security Hub のコントロールリファレンスを参照してください。
- Amazon GuardDuty – 不審なアクティビティや悪意のあるアクティビティがないか環境をモニタリングすることで AWS アカウント、、ワークロード、コンテナ、データに対する潜在的な脅威 AWS のサービス を検出します。GuardDuty を使用すると、特定のコンプライアンスフレームワークで義務付けられている侵入検知要件を満たすことで、PCI DSS などのさまざまなコンプライアンス要件に対応できます。
- AWS Audit Manager – これにより AWS のサービス 、 AWS 使用状況を継続的に監査し、リスクの管理方法と規制や業界標準への準拠を簡素化できます。

## Amazon Fraud Detector の復元力

AWS の グローバルインフラストラクチャは、AWS リージョンとアベイラビリティーゾーンを中心として構築されます。AWS リージョンには、低レイテンシー、高いスループット、そして高度の冗長ネットワークで接続されている、複数の物理的に独立し隔離されたアベイラビリティーゾーンがあります。アベイラビリティーゾーンでは、ゾーン間で中断することなく自動的にフェイルオーバーするアプリケーションとデータベースを設計および運用することができます。アベイラビリティーゾー

ンは、従来の単一または複数のデータセンターインフラストラクチャよりも可用性が高く、フォールトトレラントで、スケーラブルです。

AWS リージョンとアベイラビリティーゾーンの詳細については、AWS グローバルインフラストラクチャを参照してください。

# Amazon Fraud Detector のインフラストラクチャセキュリティ

マネージドサービスである Amazon Fraud Detector は、 AWS グローバルネットワークセキュリティで保護されています。 AWS セキュリティサービスと がインフラストラクチャ AWS を保護する方法については、AWS 「 クラウドセキュリティ」を参照してください。インフラストラクチャセキュリティのベストプラクティスを使用して AWS 環境を設計するには、「Security Pillar AWS Well-Architected Framework」の「Infrastructure Protection」を参照してください。

AWS 公開された API コールを使用して、ネットワーク経由で Amazon Fraud Detector にアクセスします。クライアントは以下をサポートする必要があります。

- Transport Layer Security (TLS)。TLS 1.2 が必須で、TLS 1.3 をお勧めします。
- DHE (楕円ディフィー・ヘルマン鍵共有) や ECDHE (楕円曲線ディフィー・ヘルマン鍵共有) などの完全前方秘匿性 (PFS) による暗号スイート。これらのモードはJava 7 以降など、ほとんどの最新システムでサポートされています。

また、リクエストにはアクセスキー ID と、IAM プリンシパルに関連付けられているシークレットアクセスキーを使用して署名する必要があります。またはAWS Security Token Service (AWS STS) を使用して、一時的なセキュリティ認証情報を生成し、リクエストに署名することもできます。

# Amazon Fraud Detector のモニタリング

モニタリングは、Amazon Fraud Detector および AWS のその他のソリューションの信頼性、可用性、およびパフォーマンスを維持するための重要な部分です。AWS には、Amazon Fraud Detector を監視して異常を検出した場合に報告し、必要に応じて自動的に対処するために、次のモニタリングツールが用意されています。

- Amazon CloudWatch は、 AWS リソースと で実行されるアプリケーションを AWS リアルタイムでモニタリングします。メトリクスを収集および追跡し、カスタマイズされたダッシュボードを作成し、指定されたメトリックが指定したしきい値に達したときに通知またはアクションを実行するアラームを設定できます。詳細については、『Amazon CloudWatch ユーザーガイド』を参照してください。
- AWS CloudTrail は、 AWS アカウントによって、またはアカウントに代わって行われた API コールおよび関連イベントをキャプチャし、指定した Amazon S3 バケットにログファイルを配信します。 AWSを呼び出したユーザーとアカウント、呼び出し元の IP アドレス、および呼び出しの発生日時を特定できます。詳細については、「AWS CloudTrail ユーザーガイド」を参照してください。

トピック

- Amazon CloudWatch を使用した Amazon Fraud Detector のモニタリング
- を使用した Amazon Fraud Detector API コールのログ記録 AWS CloudTrail

# Amazon CloudWatch を使用した Amazon Fraud Detector のモニタリング

CloudWatch を使用して Amazon Fraud Detector をモニタリングすることで、raw データを収集し、リアルタイムに近い読み取り可能なメトリクスに加工することができます。これらの統計は 15 か月間保持されるため、履歴情報にアクセスし、ウェブアプリケーションまたはサービスの動作をより的確に把握できます。また、特定のしきい値を監視するアラームを設定し、これらのしきい値に達したときに通知を送信したりアクションを実行したりできます。詳細については、「Amazon CloudWatch ユーザーガイド」を参照してください。

トピック

- Amazon Fraud Detector の CloudWatch メトリクスの使用。

- Amazon Fraud Detector メトリクス

# Amazon Fraud Detector の CloudWatch メトリクスの使用。

メトリクスを使用するには、以下の情報を指定する必要があります。

- メトリクスの名前空間。名前空間は、Amazon Fraud Detector がメトリクスを公開するために使用する CloudWatch コンテナです。CloudWatch ListMetrics API または list-metrics コマンドを使用して Amazon Fraud Detector のメトリクスを表示している場合は、名前空間`AWS/FraudDetector`に を指定します。
- メトリクスディメンション。ディメンションは、メトリクスを一意に識別するための名前と値のペアです。例えば、`DetectorId` というディメンション名にすることができます。メトリクスディメンションの指定はオプションです。
- メトリクス名 (`GetEventPrediction` など)。

Amazon Fraud Detector のモニタリングデータは AWS CLI、 AWS Management Console、、または CloudWatch API を使用して取得できます。CloudWatch API は、いずれかの Amazon AWS Software Development Kit (SDK) または CloudWatch API ツールでも使用できます。コンソールには、CloudWatch API の raw データに基づいて一連のグラフが表示されます。必要に応じて、コンソールに表示されるグラフまたは API から取得したグラフを使用できます。

以下のリストは、メトリクスの一般的な利用方法をいくつか示しています。ここで紹介するのは開始するための提案事項です。すべてを網羅しているわけではありません。

| 目的 | 関連するメトリクス |
| --- | --- |
| 実行された予測数を追跡する方法を教えてください。 | `GetEventPrediction` メトリクスをモニタリングします。 |
| `GetEventPrediction` エラーをモニタリングするにはどうすればよいですか? | `GetEventPrediction5xxError` と `GetEventPrediction4xxError` メトリクスを使用します。 |
| `GetEventPrediction` 呼び出しのレイテンシーをモニタリングするにはどうすればよいですか。 | `GetEventPredictionLatency` メトリクスを使用します。 |

CloudWatch で Amazon Fraud Detector をモニタリングするには、適切な CloudWatch アクセス許可が必要です。詳細については、「Amazon CloudWatch に対する認証とアクセスコントロール」を参照してください。

## Amazon Fraud Detector メトリクスへのアクセス

以下の手順は、CloudWatch コンソールを使用して Amazon Fraud Detector メトリクスにアクセスする方法を示しています。

メトリクスを表示するには (コンソール)

1. CloudWatch コンソールの https://console.aws.amazon.com/cloudwatch/ を開いてください。
2. [メトリクス] を選択し、[すべてのメトリクス] タブをクリックして、[Fraud Detector] を選択します。
3. メトリクスディメンションを選択します。
4. リストから目的のメトリクスを選択して、グラフの期間を選択します。

## アラームの作成

アラームの状態が変わったときに Amazon Simple Notification Service (Amazon SNS) メッセージを送信する CloudWatch のアラームを作成することができます。1 つのアラームで、指定した期間中、1 つのメトリクスをモニタリングします。このアラームは、複数の期間にわたる一定のしきい値とメトリクスの値の関係性に基づき、1 つ以上のアクションを実行します。アクションは、Amazon SNS トピックまたは Auto Scaling ポリシーに送信される通知です。

アラームは持続している状態変化に対してのみアクションを呼び出します。CloudWatch アラームは、特定の状態にあるというだけの理由ではアクションを呼び出しません。状態が変わって、変わった状態が指定期間にわたって維持される必要があります。

アラームを設定するには (コンソール)

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/cloudwatch/ で CloudWatch コンソールを開きます。
2. ナビゲーションペインで、[アラーム] を選択し、[アラームの作成] を選択します。これにより、[アラームウィザードの作成] が起動します。
3. [メトリクスの選択] を選択します。
4. [すべてのメトリクス] タブで、[Fraud Detector を選択します。

5. [ディテクター ID 別] を選択し、[GetEventPrediction] メトリクス選択します。
6. [グラフ化したメトリクス] タブを選択します。
7. [統計] で、[合計] を選択します。
8. [メトリクスの選択] を選択します。
9. 条件 で、しきい値タイプに静的を選択し、常により大きい... を選択し、選択した最大値を入力します。 [次へ] を選択します。
10. 既存の Amazon SNS トピックにアラームを送信するには、[通知の送信先:] で既存の SNS トピックを選択します。新しいメールサブスクリプションリスト用の名前とメールアドレスを設定するには、[新しいリスト] を選択します。CloudWatch はリストを保存してフィールドに表示されるため、以降のアラーム設定に利用できます。

> **Note**
>
> [新しいリスト] を使用して新しい Amazon SNS トピックを作成する場合は、宛先に通知を送信する前にメールアドレスを検証する必要があります。Amazon SNS は、アラームがアラーム状態になったときにのみメールを送信します。アラーム状態になったときに E メールアドレスの検証がまだ完了していない場合、宛先には通知が届きません。

11. [次へ] を選択します。アラームの名前とオプションの説明を追加します。[次へ] を選択します。
12. [Create Alarm] (アラームの作成) を選択します。

## Amazon Fraud Detector メトリクス

Amazon Fraud Detector は、次のメトリクスを CloudWatch に送信します。すべてのメトリクスで `Average`、`Minimum`、`Maximum`、`Sum` の統計がサポートされます。

| メトリクス | 説明 |
| --- | --- |
| `GetEventPrediction` | GetEventPrediction API リクエストの数。<br><br>有効なディメンション: `DetectorID` |
| `GetEventPredictionLatency` | GetEventPrediction リクエストからのクライアントリクエストに応答するのにかかる時間の間隔。<br><br>有効なディメンション: `DetectorID` |

| メトリクス | 説明 |
| --- | --- |
| | 単位: ミリ秒 |
| GetEventPrediction4XXError | Amazon Fraud Detector で 4xx HTTP レスポンスコードが返された GetEventPrediction リクエストの数。各 4xx レスポンスについて、1 が送信されます。<br><br>有効なディメンション: DetectorID |
| GetEventPrediction5XXError | Amazon Fraud Detector で 5xx HTTP レスポンスコードが返された GetEventPrediction リクエストの数。各 5xx レスポンスについて、1 が送信されます。<br><br>有効なディメンション: DetectorID |
| Prediction | 予測の数。成功すると 1 が送信されます。<br><br>有効なディメンション: DetectorID 、DetectorVersionID |
| PredictionLatency | 予測オペレーションに要する時間間隔。<br><br>有効なディメンション: DetectorID 、DetectorVersionID<br><br>単位: ミリ秒 |
| PredictionError | Amazon Fraud Detector でエラーが発生した予測の数。エラーが発生した場合は 1 が送信されます。<br><br>有効なディメンション: DetectorID 、DetectorVersionID |

| メトリクス | 説明 |
| --- | --- |
| VariableUsed | 変数が評価の一部として使用された GetEventPrediction リクエストの数。<br><br>有効なディメンション: DetectorID 、DetectorVersionID 、VariableName |
| VariableDefaultReturned | 変数がイベント属性の一部として存在しなかったため、評価時に変数のデフォルト値が使用された GetEventPrediction リクエストの数。<br><br>有効なディメンション: DetectorID 、DetectorVersionID 、VariableName |
| RuleNotEvaluated | 前のルールが一致したためにルールが評価されなかった GetEventPrediction リクエストの数。<br><br>有効なディメンション: DetectorID 、DetectorVersionID 、RuleID |
| RuleEvaluateTrue | ルールが True としてトリガーされ、ルールの結果が返された GetEventPrediction リクエストの数。<br><br>有効なディメンション: DetectorID 、DetectorVersionID 、RuleID |
| RuleEvaluateFalse | ルールが False と評価された GetEventPrediction リクエストの数。<br><br>有効なディメンション: DetectorID 、DetectorVersionID 、RuleID |
| RuleEvaluateError | ルールがエラーで評価される GetEventPrediction リクエストの数<br><br>有効なディメンション: DetectorID 、DetectorVersionID 、RuleID |

| メトリクス | 説明 |
| --- | --- |
| OutcomeReturned | 指定された結果が返された GetEventPrediction 呼び出しの数。<br><br>有効なディメンション: DetectorID 、DetectorVersionID 、OutcomeName |
| ModelInvocation (Amazon SageMaker model endpoint) | SageMaker モデルエンドポイントが評価の一部として呼び出された GetEventPrediction リクエストの数。<br><br>有効なディメンション: DetectorID 、DetectorVersionID 、ModelEndpoint |
| ModelInvocationError (Amazon SageMaker model endpoint) | 呼び出された SageMaker モデルエンドポイントが評価中にエラーを返した GetEventPrediction リクエストの数。<br><br>有効なディメンション: DetectorID 、DetectorVersionID 、ModelEndpoint |
| ModelInvocationLatency (Amazon SageMaker model endpoint) | Amazon Fraud Detector から見た、インポートされたモデルの応答時間の間隔。この間隔には、モデルの呼び出しのみが含まれます。<br><br>有効なディメンション: DetectorID 、DetectorVersionID 、ModelEndpoint<br><br>単位: ミリ秒 |
| ModelInvocation | モデルが評価の一部として呼び出された GetEventPrediction リクエストの数。<br><br>有効なディメンション: DetectorID 、DetectorVersionID 、ModelType 、ModelID |

| メトリクス | 説明 |
| --- | --- |
| ModelInvocationError | Amazon Fraud Detector モデルが評価中にエラーを返した GetEventPrediction リクエストの数。<br><br>有効なディメンション: DetectorID 、DetectorVersionID 、ModelType 、ModelID |
| ModelInvocationLatency | Amazon Fraud Detector から見た、Amazon Fraud Detector モデルによる応答時間の間隔。この間隔には、モデルの呼び出しのみが含まれます。<br><br>有効なディメンション: DetectorID 、DetectorVersionID 、ModelType 、ModelID<br><br>単位: ミリ秒 |

# を使用した Amazon Fraud Detector API コールのログ記録 AWS CloudTrail

Amazon Fraud Detector は AWS CloudTrail、Amazon Fraud Detector のユーザー、ロール、または のサービスによって実行されたアクションを記録する AWS サービスである と統合されています。CloudTrail は、Amazon Fraud Detector コンソールからの呼び出しや Amazon Fraud Detector API への呼び出しを含む、Amazon Fraud Detector のすべての API コールをイベントとしてキャプチャします。

証跡を作成する場合は、Amazon Fraud Detector のイベントなど、Amazon S3 バケットへの CloudTrail イベントの継続的な配信を有効にすることができます。証跡を設定しない場合でも、CloudTrail コンソールの [イベント履歴] で最新のイベントを表示できます。CloudTrail で収集された情報を使用して、Amazon Fraud Detector に対するリクエスト、そのリクエストが発信された IP アドレス、リクエストの作成者、リクエスト作成日時、その他の詳細情報などを確認できます。

CloudTrail の詳細については、「AWS CloudTrail ユーザーガイド」を参照してください。

## CloudTrail での Amazon Fraud Detector 情報

CloudTrail は、 AWS アカウントの作成時にアカウントで有効になります。Amazon Fraud Detector でアクティビティが発生すると、そのアクティビティはイベント履歴の他の AWS サービスイベント

とともに CloudTrail イベントに記録されます。 AWS アカウントで最近のイベントを表示、検索、ダウンロードできます。詳細については、CloudTrail イベント履歴でのイベントの表示を参照してください。

Amazon Fraud Detector のイベントなど、 AWS アカウント内のイベントの継続的な記録については、証跡を作成します。追跡により、CloudTrail はログファイルを Simple Storage Service (Amazon S3) バケットに配信できます。デフォルトでは、コンソールで作成した証跡がすべての AWS リージョンに適用されます。証跡は、 AWS パーティション内のすべてのリージョンからのイベントをログに記録し、指定した Amazon S3 バケットにログファイルを配信します。さらに、CloudTrail ログで収集したイベントデータをより詳細に分析し、それに基づいて対応するようにその他の AWS サービスを設定できます。詳細については、次を参照してください:

- 証跡の作成のための概要
- CloudTrail がサポートするサービスと統合
- CloudTrail 用 Amazon SNS 通知の構成
- 複数のリージョンから CloudTrail ログファイルを受け取るおよび複数のアカウントから CloudTrail ログファイルを受け取る

Amazon Fraud Detector では、すべてのアクション (API オペレーション) をイベントとして CloudTrail ログファイルに記録できます。詳細については、「アクション」を参照してください。

各イベントまたはログエントリには、誰がリクエストを生成したかという情報が含まれます。ID 情報は次の判断に役立ちます。

- リクエストが、ルートとユーザー認証情報のどちらを使用して送信されたか。
- リクエストが、ロールとフェデレーションユーザーの一時的なセキュリティ認証情報のどちらを使用して送信されたか。
- リクエストが別の AWS サービスによって行われたかどうか。

詳細については、「CloudTrail userIdentity 要素」を参照してください。

## Amazon Fraud Detector ログファイルエントリの概要

「トレイル」は、指定した Amazon S3 バケットにイベントをログファイルとして配信するように設定できます。CloudTrail のログファイルには、単一または複数のログエントリがあります。各イベントは任意の送信元からの単一のリクエストを表し、リクエストされたオペレーション、オペレーションの日時、リクエストパラメータなどに関する情報を含みます。CloudTrail ログファイルは、パブ

リック API コールの順序付けられたスタックトレースではないため、特定の順序では表示されません。

GetDetectors オペレーションを示す CloudTrail ログエントリの例は、次のとおりです。

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "principal-id",
        "arn": "arn:aws:iam::user-arn",
        "accountId": "account-id",
        "accessKeyId": "access-key",
        "userName": "user-name"
    },
    "eventTime": "2019-11-22T02:18:03Z",
    "eventSource": "frauddetector.amazonaws.com",
    "eventName": "GetDetectors",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "source-ip-address",
    "userAgent": "aws-cli/1.11.16 Python/2.7.11 Darwin/15.6.0 botocore/1.4.73",
    "requestParameters": null,
    "responseElements": null,
    "requestID": "request-id",
    "eventID": "event-id",
    "eventType": "AwsApiCall",
    "recipientAccountId": "recipient-account-id"
}
```

# トラブルシューティング

以下のセクションは、Amazon Fraud Detector を使用する際に発生する可能性がある問題のトラブルシューティングに役立ちます。

## トレーニングデータに関する問題のトラブルシューティング

このセクションの情報を使用して、モデルのトレーニング時に Amazon Fraud Detector コンソールのモデルトレーニング診断ペインに表示される可能性のある問題の診断と解決に役立ててください。

モデルトレーニング診断ペインに表示される問題は、次のように分類されます。問題に対処するための要件は、問題のカテゴリによって異なります。

-
  エラー- これにより、モデルトレーニングが失敗します。モデルのトレーニングを正常に行うには、この問題に対処する必要があります。
-
  警告- モデルトレーニングは続行しますが、一部の変数がトレーニングプロセスで除外される可能性があります。このセクションで関連するガイダンスを確認して、データセットの品質を向上させましょう。
-
  情報 (info)- モデルトレーニングには影響を与えず、すべての変数がトレーニングに使用されます。このセクションの関連するガイダンスを確認して、データセットとモデルのパフォーマンスをさらに改善することをお勧めします。

トピック


- 指定されたデータセットの不安定な不正率
- 不十分なデータ
- 異なる EVENT_LABEL 値がない
- EVENT_TIMESTAMP 値が欠落しているか正しくない
- データが取り込まれない
- 変数が不十分
- 変数タイプが欠落しているか、正しくない

- 欠落している変数値
- 一意な変数値が不十分
- 変数式が誤っている
- 一意のエンティティが不十分

## 指定されたデータセットの不安定な不正率

問題タイプ: エラー

説明

特定のデータの不正率が、時間の経過とともに過度に不安定に。不正イベントと正当なイベントが、経時的に一様にサンプリングされていることを確認してください。

原因

このエラーは、データセット内の不正イベントと正当なイベントが不均等に分散され、異なるタイムスロットから取得された場合に発生します。Amazon Fraud Detector モデルトレーニングプロセスは、EVENT_TIMESTAMP に基づいてデータセットのサンプル抽出とパーティショニングを行います例えば、データセットが過去 6 か月から引き出された不正イベントで構成され、最後の月の正当なイベントのみが含まれる場合、データセットは不安定と見なされます。不安定なデータセットは、モデルのパフォーマンス評価でバイアスを引き起こす可能性があります。

解決策

不正イベントデータと正当なイベントデータが同じタイムスロットから提供されるようにし、不正率が時間の経過とともに劇的に変化しないようにします。

## 不十分なデータ

1. 問題タイプ: エラー

   説明

   不正イベントとしてラベル付けされる行は 50 行未満です。不正イベントと正当なイベントの両方が最小数である 50 を超え、モデルを再トレーニングします。

   原因

このエラーは、データセット中の不正とラベル付けされているイベント数がモデルトレーニングに必要なイベント数よりも少ない場合に発生します。Amazon Fraud Detector では、モデルのトレーニングに少なくとも 50 件の不正イベントが必要です。

解決策

データセットに少なくとも 50 個の不正イベントが含まれていることを確認してください。必要に応じて、より長い期間を力バーすることで、これを保証できます。

2. 問題タイプ: エラー

説明

正当なイベントとしてラベル付けされる行は 50 行未満です。不正イベントと正当なイベントの両方が最小数の $threshold を超えていることを確認し、モデルを再トレーニングします。

原因

このエラーは、データセット中の正当とラベル付けされているイベント数がモデルトレーニングに必要なイベント数よりも少ない場合に発生します。Amazon Fraud Detector では、モデルのトレーニングに少なくとも 50 の正当なイベントが必要です。

解決策

データセットに最低 50 個の正当なイベントが含まれていることを確認します。必要に応じて、より長い期間を力バーすることで、これを保証できます。

3. 問題タイプ: エラー

説明

不正とされる一意のエンティティの数が 100 未満です。パフォーマンスを向上させるために、不正なエンティティの例をさらに含めることを検討してください。

原因

このエラーは、データセット中にある不正なイベントを持つエンティティの数がモデルトレーニングに必要な数よりも少ない場合に発生します。トランザクション不正インサイト (TFI) モデルでは、不正スペースの最大のカバレッジを確保するために、不正イベントを持つエンティティが少なくとも 100 必要です。すべての不正イベントが少数のエンティティグループによって実行されている場合、モデルは一般化されないことがあります。

解決策

データセットに不正イベントを持つエンティティが少なくとも 100 含まれていることを確認してください。必要に応じて、より長い期間をカバーすることで、これを保証できます。

4. 問題タイプ: エラー

   説明

   正当とされる一意のエンティティの数が 100 未満です。パフォーマンスを向上させるために、正当なエンティティの例をさらに含めることを検討してください。

   原因

   このエラーは、データセット中にある正当なイベントを持つエンティティの数がモデルトレーニングに必要な数よりも少ない場合に発生します。トランザクション不正インサイト (TFI) モデルでは、不正スペースの最大のカバレッジを確保するために、正当なイベントを持つエンティティが少なくとも 100 必要です。すべての正当なイベントが少数のエンティティによって実行された場合、モデルはうまく一般化しない可能性があります。

   解決策

   データセットに正当なイベントを持つエンティティが少なくとも 100 含まれていることを確認してください。必要に応じて、より長い期間をカバーすることで、これを保証できます。

5. 問題タイプ: エラー

   説明

   データセットに含まれる行が 100 行未満です。データセットの合計に 100 行以上あり、少なくとも 50 行が不正とラベル付けされていることを確認します。

   原因

   このエラーは、データセットに含まれるレコードが 100 未満である場合に発生します。Amazon Fraud Detector では、モデルトレーニングのためにデータセット内の少なくとも 100 個のイベント (レコード) からのデータが必要です。

   解決策

   データセットに 100 を超えるイベントのデータがあることを確認します。

# 異なる EVENT_LABEL 値がない

1. 問題タイプ: エラー

   説明

   EVENT_LABEL 列の 1% 以上が NULL であるか、モデル設定 **$label_values** で定義されている値以外の値である。EVENT_LABEL 列の欠落している値が 1% 未満で、その値がモデル設定 **$label_values** で定義されている値であることを確認します。

   原因

   このエラーは、次のいずれかの原因で発生することがあります。

   - トレーニングデータを含む CSV ファイル内のレコードの 1% 以上の EVENT_LABEL 列に欠落している値があります。
   - トレーニングデータを含む CSV ファイル内のレコードの 1% 以上の EVENT_LABEL 列の値が、イベントタイプに関連付けられている値とは異なります。

   オンライン不正インサイト (OFI) モデルでは、各レコードの EVENT_LABEL 列に、イベントタイプに関連付けられている (または、`CreateModelVersion` にマップされている) ラベルのいずれかが入力されている必要があります。

   解決策

   このエラーの原因が EVENT_LABEL 値が欠落していることにある場合は、それらのレコードに適切なラベルを割り当てるか、データセットからそれらのレコードを削除することを検討してください。このエラーの原因が一部のレコードのラベルが **label_values** 含まれていないことにある場合は、EVENT_LABEL 列のすべての値をイベントタイプのラベルに追加し、モデル作成で不正または正当な (fraud、legit) にマップされていることを確認してください。

2. 問題タイプ: 情報

   説明

   EVENT_LABEL 列には、モデル設定 **$label_values** で定義された値以外の NULL 値またはラベル値が含まれています。これらの矛盾した値は、トレーニングの前に「不正ではない」に変換されました。

   原因

この情報は、次のいずれかが原因で受け取ります。

- トレーニングデータを含む CSV ファイル内のレコードの 1% 未満の EVENT_LABEL 列に欠落している値があります。
- トレーニングデータを含む CSV ファイル内のレコードの 1% 未満の EVENT_LABEL 列の値が、イベントタイプに関連付けられている値とは異なります。

どちらの場合も、モデルトレーニングは成功します。ただし、ラベル値が欠落またはマッピングされていないイベントのラベル値は、正当なラベル値に変換されます。これを問題と見なす場合は、以下のソリューションに従ってください。

解決策

データセット中に欠落している EVENT_LABEL 値がある場合は、データセットからそれらのレコードを削除することを検討してください。これらの EVENT_LABELS に指定された値がマッピングされていない場合は、イベントごとにすべての値が不正または正当 (fraud、legit) にマップされていることを確認してください。

## EVENT_TIMESTAMP 値が欠落しているか正しくない

1. 問題タイプ: エラー

   説明

   トレーニングデータセットに、許容される形式に準拠しないタイムスタンプを含む EVENT_TIMESTAMP が含まれています。形式が有効な日付/タイムスタンプ形式の 1 つであることを確認します。

   原因

   このエラーは、EVENT_TIMESTAMP 列に、Amazon Fraud Detector でサポートされているタイムスタンプ形式に準拠していない値が含まれている場合に発生します。

   解決策

   EVENT_TIMESTAMP 列に指定された値が、サポートされているタイムスタンプ形式に準拠していることを確認します。EVENT_TIMESTAMP 列に欠落している値がある場合は、サポートされているタイムスタンプ形式を使用した値で埋めるか、none、null、または missing の文字列を入力するのではなく、イベントを完全に削除することを検討してください。

2. 問題タイプ: エラー

   トレーニングデータセットには、欠落している値がある EVENT_TIMESTAMP が含まれています。欠落している値がないことを確認します。

   原因

   このエラーは、データセットの EVENT_TIMESTAMP 列に欠落している値がある場合に発生します。Amazon Fraud Detector では、データセットの EVENT_TIMESTAMP 列に値が必要です。

   解決策

   データセットの EVENT_TIMESTAMP 列に値があり、それらの値がサポートされているタイムスタンプ形式に準拠していることを確認します。EVENT_TIMESTAMP 列に欠落している値がある場合は、サポートされているタイムスタンプ形式を使用した値で埋めるか、`none`、`null`、または `missing` のような文字列を入力するのではなく、イベントを完全に削除することを検討してください。

## データが取り込まれない

問題タイプ: エラー

説明

トレーニングで取り込まれたイベントが見つかりません。トレーニング設定を確認してください。

原因

このエラーは、Amazon Fraud Detector で保存されたイベントデータを含むモデルを作成しているが、モデルのトレーニングを開始する前にデータセットを Amazon Fraud Detector にインポートしなかった場合に発生します。

解決策

`SendEvent` API オペレーション、`CreateBatchImportJob` API オペレーション、または Amazon Fraud Detector コンソールのバッチインポート機能を使用して、最初にイベントデータをインポートし、次にモデルをトレーニングします。詳細については、「ストアドイベントデータセット」を参照してください。

> **Note**
>
> データのインポートが完了してから 10 分待ってから、データを使用してモデルをトレーニングすることをお勧めします。

Amazon Fraud Detector コンソールを使用して、イベントタイプごとに既に保存されているイベントの数を確認できます。詳細については、「ストアドイベントのメトリクスの表示」を参照してください。

## 変数が不十分

問題タイプ: エラー

説明

データセットには、トレーニングに適した変数が少なくとも 2 つ含まれている必要があります。

原因

このエラーは、データセットに含まれる、モデルトレーニングに適している変数が 2 つ未満の場合に発生します。Amazon Fraud Detector は、すべての検証に合格した場合にのみ、モデルトレーニングに適した変数と見なします。変数が検証に失敗すると、モデルトレーニングでは除外され、モデルトレーニング診断にメッセージが表示されます。

解決策

データセットに少なくとも 2 つの変数が値で入力され、すべてのデータ検証に合格していることを確認します。列ヘッダーを指定したイベントメタデータ行 (EVENT_TIMESTAMP、EVENT_ID、ENTITY_ID、EVENT_LABEL など) は変数と見なされないことに注意してください。

## 変数タイプが欠落しているか、正しくない

問題の種類: 警告

説明

**`$variable_name`** に期待されるデータ型は、NUMERIC です。データセット中の **`$variable_name`** を確認して更新し、モデルを再トレーニングします。

原因

この警告は、変数が NUMERIC 変数として定義されているが、データセットに NUMERIC に変換できない値がある場合に表示されます。その結果、その変数はモデルトレーニングでは除外されます。

解決策

NUMERIC 変数として保持する場合は、指定する値が浮動小数点数に変換できることを確認します。変数に欠落している値が含まれている場合は、nonene、null、または missing などの文字列を入力しないでください。変数に数値以外の値が含まれている場合は、CATEGORICAL または FREE_FORM_TEXT 変数タイプとして再作成します。

## 欠落している変数値

問題の種類: 警告

説明

**$threshold** の値より大きい **$variable_name** がトレーニングデータセットから欠落しています。パフォーマンスを向上させるために、データセット中の **$variable_name** を変更して再トレーニングすることを検討してください。

原因

この警告は、欠落している値が多すぎるために指定された変数が削除されている場合に表示されます。Amazon Fraud Detector では、変数の値の欠落が許可されています。ただし、1 つの変数に欠落している値が多すぎると、その変数はモデルにほとんど寄与せず、その変数はモデルトレーニングで削除されます。

解決策

まず、これらの欠落している値がデータの収集と準備のミスによるものではないことを確認します。それらが間違いであれば、モデルトレーニングからそれらを削除することを検討できます。ただし、これらの欠落している値に価値があると考え、その変数を保持したい場合は、モデルトレーニングとリアルタイム推論の両方で、欠落している値を定数で手動で入力できます。

## 一意な変数値が不十分

問題の種類: 警告

説明

**$variable_name** の一意の値の数が 100 未満です。データセット中の **$variable_name** を確認して更新し、モデルを再トレーニングします。

原因

この警告は、指定された変数の一意の値の数が 100 より小さい場合に表示されます。しきい値は、変数のタイプによって異なります。一意の値が非常に少ないため、データセットがその変数の特徴空間をカバーするのに十分なほど一般的ではないというリスクがあります。その結果、モデルがリアルタイム予測ではうまく一般化されないことがあります。

解決策

まず、変数分布が実際のビジネストラフィックを代表していることを確認します。次に、個別に `first_name` や `last_name` を使用する代わりに `full_customer_name` を使用するなど、カーディナリティの高い、より詳細にトレーニングされた変数を採用するか、変数タイプを CATEGORICAL に変更して、カーディナリティを低くすることができます。

## 変数式が誤っている

1. 問題タイプ: 情報

   説明

   50% を超える**$email_variable_name** 値が、予想される正規表現 http://emailregex.com と一致しません。パフォーマンスを向上させるために、データセット中の **$email_variable_name** を変更して再トレーニングすることを検討してください。

   原因

   この情報は、データセット内の 50% を超えるレコードに通常の E メール式に準拠しない E メール値が含まれているため、検証に失敗した場合に表示されます。

   解決策

   E メール変数の値を正規表現に準拠するようにフォーマットします。E メールの値が欠落している場合は、`none`、`null`、または `missing` のような文字列で入力するのではなく、空のままにすることをお勧めします。
2. 問題タイプ: 情報

   説明

50% 超の **$IP_variable_name** 値が、IPv4 または IPv6 アドレス https://digitalfortress.tech/tricks/top-15-commonly-used-regex/ の正規表現と一致しません。パフォーマンスを向上させるために、データセット中の **$IP_variable_name** を変更して再トレーニングすることを検討してください。

原因

この情報は、データセット内の 50% を超えるレコードに通常の IP 式に準拠しない IP 値が含まれているため、検証に失敗した場合に表示されます。

解決策

IP 値を正規表現に準拠するようにフォーマットします。IP 値が欠落している場合は、none、null、または missing のような文字列で入力するのではなく、空のままにすることをお勧めします。

3. 問題タイプ: 情報

説明

50% 超の **$phone_variable_name** 値が、基本的な電話の正規表現 /$pattern/ と一致しません。パフォーマンスを向上させるために、データセット中の **$phone_variable_name** を変更して再トレーニングすることを検討してください。

原因

この情報は、データセット内の 50% を超えるレコードに通常の電話番号式に準拠しない電話番号が含まれているため、検証に失敗した場合に表示されます。

解決策

電話番号を正規表現に準拠するようにフォーマットします。電話番号が欠落している場合は、none、null、または missing のような文字列で入力するのではなく、空のままにすることをお勧めします。

## 一意のエンティティが不十分

問題タイプ: 情報

説明

一意のエンティティの数が 1500 未満です。パフォーマンスを向上させるために、より多くのデータを含めることを検討してください。

原因

この情報は、データセットの一意のエンティティ数が推奨数よりも少ない場合に表示されます。トランザクション不正インサイト (TFI) モデルは、時系列集計と汎用トランザクション機能の両方を使用して、最高のパフォーマンスを提供します。データセットの一意のエンティティが少なすぎる場合、IP_ADDRESS、EMAIL_ADDRESS などの汎用データのほとんどは一意の値を持たない可能性があります。すると、データセットがその変数の特徴空間をカバーするのに十分なほど一般的ではないというリスクもあります。その結果、新しいエンティティからのトランザクションでは、モデルがうまく一般化されない可能性があります。

解決策

より多くのエンティティを含めます。必要に応じて、トレーニングデータの時間範囲を拡張します。

# クォータ

AWS アカウント には、Amazon Web Service ごとに、以前は制限と呼ばれていたデフォルトのクォータがあります。特に明記していない限り、クォータはリージョン固有です。次の表に示すすべての調整可能なクォータに対してクォータの引き上げをリクエストできます。詳細については、「クォータの引き上げのリクエスト」を参照してください。

次の表は、コンポーネント別の Amazon Fraud Detector クォータの概要を示しています。

## Amazon Fraud Detector モデル

| クォータ名 | デフォルトのクォータ | 調整可能 |
|---|---|---|
| トレーニングデータサイズ | 5 GB | いいえ |
| アカウントあたりのモデル数 | 50 | いいえ |
| モデルごとのバージョンの数 | 200 | いいえ |
| アカウントあたりのデプロイされたモデルのバージョン | 5 | いいえ |
| アカウントあたりの同時トレーニングジョブ数 | 3 | いいえ |
| モデルごとの同時トレーニングジョブ数 | 1 | いいえ |

## Amazon Fraud Detector ディテクター/変数/結果/ルール

| クォータ名 | デフォルトのクォータ | 調整可能 |
|---|---|---|
| アカウントごとの変数の数 | 5000 | いいえ |
| アカウントごとのルールの数 | 5000 | いいえ |
| ルールあたりのリスト | 3 | いいえ |

| クォータ名 | デフォルトのクォータ | 調整可能 |
| --- | --- | --- |
| アカウントごとの結果の数 | 5000 | いいえ |
| アカウントごとのディテクター数 | 100 | いいえ |
| ディテクターあたりのリスト | 30 | いいえ |
| ディテクターごとのドラフトバージョンの数 | 100 | いいえ |
| ディテクターバージョンごとのモデル数 | 10 | いいえ |
| アカウントあたりのラベル数 | 100 | いいえ |
| アカウントごとのイベントタイプの数 | 100 | いいえ |
| アカウントごとのエンティティタイプの数 | 100 | いいえ |

## Amazon Fraud Detector API

| クォータ名 | デフォルトのクォータ | 調整可能 |
| --- | --- | --- |
| GetEventPrediction API コール/秒 | 200 TPS | はい |
| GetEventPrediction API コールあたりのペイロードのサイズ | 256 KB | いいえ |
| GetEventPrediction API コールあたりの入力数 | 5000 | いいえ |

# ドキュメント履歴

以下の表は、Amazon Fraud Detector ユーザーガイドの重要な変更点をまとめたものです。また、お客様からいただいたフィードバックに対応するために、Amazon Fraud Detector ユーザーガイドを頻繁に更新しています。

| 変更 | 説明 | 日付 |
| --- | --- | --- |
| 新しい変数とデータ型 | Amazon Fraud Detector では、新しい変数タイプと、有用な情報を抽出するために使用できるデータ型が導入されています。 | 2023 年 6 月 5 日 |
| イベントのオーケストレーション | イベントオーケストレーションを使用すると、Amazon EventBridge を使用して、ダウンストリーム処理 AWS のサービス のために にイベントを簡単に送信できます。 | 2023 年 5 月 30 日 |
| Lists | Lists リソースを使用すると、ルールの一部として IP アドレスや E メールアドレスなどの値のセットを参照できます。ルールのリストを使用して、アクセスまたはトランザクションを許可または拒否します。 | 2023 年 2 月 14 日 |
| データモデルエクスプローラー | Data Models Explorer は、Amazon Fraud Detector が不正検出モデルを作成するために必要なデータ要素に関するインサイトを提供します。イベントデータセットを準備する前に、データモデル | 2022 年 12 月 15 日 |

| | | |
|---|---|---|
| | エクスプローラーを使用します。 | |
| アカウント乗っ取りインサイトモデル | アカウント乗っ取りインサイト (ATI) モデルを使用して、悪意のある乗っ取り、フィッシング、または盗まれた認証情報によって侵害されたアカウントを検出します。 | 2022 年 7 月 21 日 |
| 章の更新 | Amazon Fraud Detector に関する追加情報を含む概要の章を更新しました。 | 2022 年 4 月 11 日 |
| 変数エンリッチメント | これらのデータ要素を使用し、2022 年 2 月 8 日より前にトレーニングされたモデルのパフォーマンスを向上させるために、指定した一部の raw データの強化を有効にします。 | 2022 年 2 月 8 日 |
| オプトアウトポリシー | オプトアウトポリシーを使用して、イベントデータを使用して Amazon Fraud Detector の品質を開発または改善することをオプトアウトします。 | 2022 年 1 月 6 日 |
| 混乱した代理の防止 | サードパーティーまたはクロスサービスエンティティが、アカウント内のリソースにアクセスするために、そのエンティティに代わって行動するアクセス許可を持つエンティティを操作しないようにするポリシーを作成します。 | 2021 年 12 月 6 日 |

| | | |
|---|---|---|
| イベントデータセットの作成 | [イベントデータセットの作成] で示されるガイダンスを使用して、モデルをトレーニングするためのデータを準備および収集します。 | 2021 年 11 月 22 日 |
| 予測の説明 | 予測の説明を使用して、各イベント変数がモデルの不正予測スコアにどのように影響したかを把握できます。 | 2021 年 11 月 10 日 |
| トラブルシューティング | トレーニングデータの問題のトラブルシューティングの情報と使用すると、モデルのトレーニング時に Amazon Fraud Detector コンソールに表示される可能性のある問題の診断と解決に役立ちます。 | 2021 年 10 月 11 日 |
| トランザクション不正インサイトモデル | トランザクション不正インサイト (TFI) モデルを使用して、オンラインまたはcard-not-presentトランザクション不正を検出します。 | 2021 年 10 月 11 日 |

| | | |
|---|---|---|
| ストアドイベント | イベントデータを Amazon Fraud Detector に保存し、保存されたデータを使用して後でモデルをトレーニングします。Amazon Fraud Detector にイベントデータを保存することで、自動計算変数を使用してパフォーマンスを改善し、モデルの再トレーニングを簡素化し、不正ラベルを更新して機械学習のフィードバックループを閉じるモデルをトレーニングできます。 | 2021 年 10 月 11 日 |
| モデル変数の重要度 | モデル変数の重要度を使用して、モデルのパフォーマンスを上昇または低下させている要因と、どのモデル変数が最も寄与しているかを把握します。次に、全体的なパフォーマンスを向上させるために、モデルを微調整します。 | 2021 年 7 月 9 日 |
| AWS CloudFormationとの統合 | AWS CloudFormation を使用して Amazon Fraud Detector リソースを管理します。 | 2021 年 5 月 10 日 |
| バッチ予測 | バッチ予測を使用して、リアルタイムスコアリングを必要としない一連のイベントの予測を取得します。 | 2021 年 3 月 31 日 |
| チャプター再編 | 開始方法とその他のセクションの再編 | 2020 年 7 月 17 日 |
| 初回リリース | 初回リリース | 2019 年 12 月 2 日 |

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。