aws

ユーザーガイド

Amazon Fraud Detector

Version latest

Amazon Fraud Detector: ユーザーガイド

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 をセットアップする ... 16
Amazon Fraud Detector の使用を開始する ... 17
サンプルデータセットの取得とアップロード ... 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
開始方法 ... 30
(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 ... 41
イベントデータセットの構造 ... 42
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 ... 43
データモデルエクスプローラー ... 43
イベントデータの収集 ... 44

データセットの検証 50
データセットストレージ 51
イベントタイプ 53
イベントタイプを作成 53
Amazon 詐欺検出器コンソールでイベントタイプを作成する 54
AWS SDK for Python (Boto3) を使用したイベントタイプの作成 55
イベントまたはイベントタイプの削除 56
イベントデータストレージ 58
Amazon S3 でのイベントデータの外部保存 59
CSV ファイルの作成 59
Amazon S3 バケットにイベントデータをアップロードする 62
Amazon Fraud Detector を使用してイベントデータを社内に保存する 63
ストレージ用のイベントデータの準備 64
バッチインポートを使用したイベントデータの保存 65
GetEventPredictions API オペレーションを使用したイベントデータの保存 78
SendEvent API オペレーションを使用したイベントデータの保存 79
保存されたイベントデータの詳細の取得 80
保存されたイベントデータセットのメトリクスの表示 81
イベントオーケストレーション 82
イベントオーケストレーションの設定 83
Amazon Fraud Detector でイベントオーケストレーションを有効にする 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有効にする 84
を使用してイベントオーケストレーションを有効にする AWS SDK for Python (Boto3) 85
Amazon Fraud Detector でイベントオーケストレーションを無効にする 85
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 85
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) 86
モデル 87
モデルタイプの選択 87
オンライン不正インサイト 87
トランザクション不正インサイト 89
アカウント乗っ取りインサイト 92
モデルの構築 97
AWS SDK for Python (Boto3) を使用したモデルのトレーニングとデプロイ 97
モデルスコア 99
モデルパフォーマンスメトリクス 100
モデル変数の重要度 102

モデル変数重要度値の使用 ... 104
モデル変数重要度値の評価 ... 105
モデル変数の重要度ランキングの表示 ... 105
モデル変数重要度値の計算方法の理解 ... 105
モデルのインポート SageMaker ... 106
を使用して SageMaker モデルをインポートする AWS SDK for Python (Boto3) ... 106
モデルまたはモデルバージョンの削除 ... 108
ディテクター ... 110
ディテクターの作成 ... 110
Amazon 詐欺検出器コンソールで検出器を作成する ... 110
AWS SDK for Python (Boto3) を使用したディテクターの作成 ... 114
ディテクターバージョンの作成 ... 114
ルール実行モード ... 114
AWS SDK for Python (Boto3) を使用したディテクターバージョンの作成 ... 115
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 ... 116
リソース ... 118
可変 ... 118
データ型 ... 118
デフォルト値 ... 119
変数タイプ ... 119
可変エンリッチメント ... 142
変数の作成 ... 149
変数の削除 ... 151
Labels ... 152
ラベルの作成 ... 152
ラベルを更新 ... 153
Amazon Fraud Detector に保存されているイベントデータのイベントデータラベルを更新する ... 154
ラベルの削除 ... 154
[Rules] (ルール) ... 155
ルール言語リファレンス ... 156
ルールを作成する ... 161
ルールを更新 ... 163
リスト ... 164
リストを作成する ... 165
リストにエントリを追加する ... 167

変数タイプをリストに割り当てる 168
リストを削除する 169
リストからエントリを削除する 169
リストからすべてのエントリを削除する 170
結果 171
結果の作成 172
結果の削除 173
エンティティ 174
エンティティタイプの作成 174
エンティティタイプの削除 175
AWS CloudFormation を使用したリソースの管理 176
Amazon Fraud Detector テンプレートの作成 176
Amazon Fraud Detector スタックの管理 177
Amazon Fraud Detector CloudFormation パラメータの理解 177
Amazon Fraud Detector サンプル AWS CloudFormation テンプレート 178
AWS CloudFormation の詳細はこちら 179
不正予測の取得 180
リアルタイム予測の取得 181
リアルタイム不正予測の仕組み 181
リアルタイムの不正予測の取得 182
バッチ予測 183
バッチ予測の仕組み 183
入力および出力ファイル 184
バッチ予測の取得 184
IAM ロールに関するガイダンス 186
AWS SDK for Python (Boto3) を使用したバッチ詐欺の予測の取得 186
予測説明 187
予測説明の表示 189
予測説明の計算方法の理解 190
セキュリティ 192
データ保護 193
保管中の暗号化 193
転送中の暗号化 194
キーの管理 194
VPC エンドポイントAWS PrivateLink 196
オプトアウト 198

Identity and Access Management 199
対象者 199
アイデンティティを使用した認証 200
ポリシーを使用したアクセスの管理 203
Amazon Fraud Detector で IAM が機能する仕組み 206
アイデンティティベースポリシーの例 210
混乱した代理の防止 218
トラブルシューティング 221
Amazon Fraud Detector のモニタリング 223
コンプライアンス検証 224
耐障害性 225
インフラストラクチャセキュリティ 226
Amazon Fraud Detector のモニタリング 227
によるモニタリング CloudWatch 227
Amazon Fraud Detector の CloudWatch メトリクスの使用。 228
Amazon Fraud Detector メトリクス 230
を使用した Amazon Fraud Detector API コールのログ記録 AWS CloudTrail 234
の Amazon Fraud Detector 情報 CloudTrail 234
Amazon Fraud Detector ログファイルエントリの概要 235
トラブルシューティング 237
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 237
指定されたデータセットの不安定な不正率 238
不十分なデータ 238
異なる EVENT_LABEL 値がない 241
EVENT_TIMESTAMP 値が欠落しているか正しくない 242
データが取り込まれない 243
変数が不十分 244
変数タイプが欠落しているか、正しくない 244
欠落している変数値 245
一意な変数値が不十分 245
変数式が誤っている 246
一意のエンティティが不十分 247
クォータ 249
Amazon Fraud Detector モデル 249
Amazon Fraud Detector ディテクター/変数/結果/ルール 249
Amazon Fraud Detector API 250

ドキュメント履歴 251
..... cclv

Amazon Fraud Detector とは

Amazon Fraud Detector は、オンライン上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、不正な取引や偽の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 年を超える不正検出の長年の専門知識に基づく方法で行われます。

Amazon Fraud Detector を使用して、カスタマイズされた不正検出モデルの構築、モデルの不正評価を解釈する決定ロジックの追加、可能性のある不正評価ごとにレビューのために合格または送信などの結果の割り当てを行うことができます。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、 AWSおよび Amazon 独自の不正専門知識から不正のパターンを学習した機械学習モデルを使用して、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に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、次の利点があります。これらの利点により、不正管理システムの構築と保守に従来必要な時間とリソースに投資することなく、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 モデルを使用して、新しいアカウント作成、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を破棄できます。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のみをアップロードし、モデルタイプを選択します。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、または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、変化する不正状況に追いつくように絶えず進化する必要があります。Amazon Fraud Detector は、アカウント年齢、最後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、モデルは、頻繁に取引を行う信頼された顧客と、不正行為者の一般的な継続的な試行の違いを学習します。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をトレーニングすると、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用意されています。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促進す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、組織のビジネスアクティビティ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説明を生成します。予測説明は、モデルのトレーニングに使用される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洞察を提供します。予測説明は、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与える影響を視覚的に識別できます。その後、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、バイアスがある場合は検出できます。最後に、予測説明を使用して、手動不正調査プロセス中に最上位リスク指標を特定することもできます。これにより、誤検出の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、評価データの承認、レビュー用データの送信、データ収集など、評価データに対してアクションを実行するルールを追加できます。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、レビュー対象の疑わしい顧客アカウント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の両方が既定のしきい値を超え、アカウント支払いの認証コード (AUTH_CODE) が有効でない場合は、このルールを開始できます。

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

イベントとは、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、会社のイベントはオンラインアカウント登録である可能性があります。単一のイベント (登録) 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、請求先住所、イベ

ントタイムスタンプが含まれる場合があります。不正検出モデルを作成およびトレーニングするために、Amazon Fraud Detector にイベントデータセットを提供します。

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、および特徴量変換を定義します。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に、Amazon Fraud Detector はデータセットの不正行為を評価し、0 ~ 1000 のスコアを生成します。このスコアでは、0 は不正リスクが低く、1000 は不正リスクが最も高いことを示します。スコア自体は偽陽性率 (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 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 SageMakerホストモデルエンドポイントを Amazon Fraud Detector 評価で使用できます。

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

ディテクター

ディテクターには、不正を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、派生することもできますAmazon SageMaker。

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の不正の評価です。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の を実行したり proof-of-concept、不正リスクを時間単位、日単位、週単位で遡及的に評価したりできます。

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洞察を提供します。各変数がリスクスコアにどのように影響するかに関する情報は、大きさ (0～5 の範囲、5 が最も高い) と方向 (スコアを高くまたは低くする) で提供されます。

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活動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。まずは、ビジネスユースケースを用意してください。ビジネスユースケースに応じて、Amazon Fraud Detector は不正検知モデルの作成に使用するモデルタイプを推奨しています。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のあるデータ要素に関する洞察も得られます。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では、特定のビジネスユースケースに合わせて不正行為を検出する機械学習アルゴリズムを選択し、提供したデータを検証し、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を実行します。モデルをトレーニングすると、Amazon Fraud Detector はモデルスコアと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、トレーニング用に提供したデータセットに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、有効化したら、ビジネスで生成されたデータをどのように解釈するかをモデルに指示し、各アクティビティの解釈にどう対処するかについての結果を割り当てる意思決定ロジック (ルールとも呼ばれる) を設定する必要があります。アウトカムは、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場合もあれば、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場合もあります。

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

本番環境に導入されたディテクタは、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。不正評価を行うために、この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、高度な機械学習アルゴリズムとユーザーが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、単一のビジネスアクティビティ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のデータをオフラインで評価することもできます。

例えば、オンラインでの送金をその活動の 1 つとして行っている企業があるとします。Amazon Fraud Detector を使用して、不正な送金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、不正な送金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、そのモデルがデータを解釈するためのルールを設定して、ディテクターを作成します。オンライン送金アクティビティのルールの

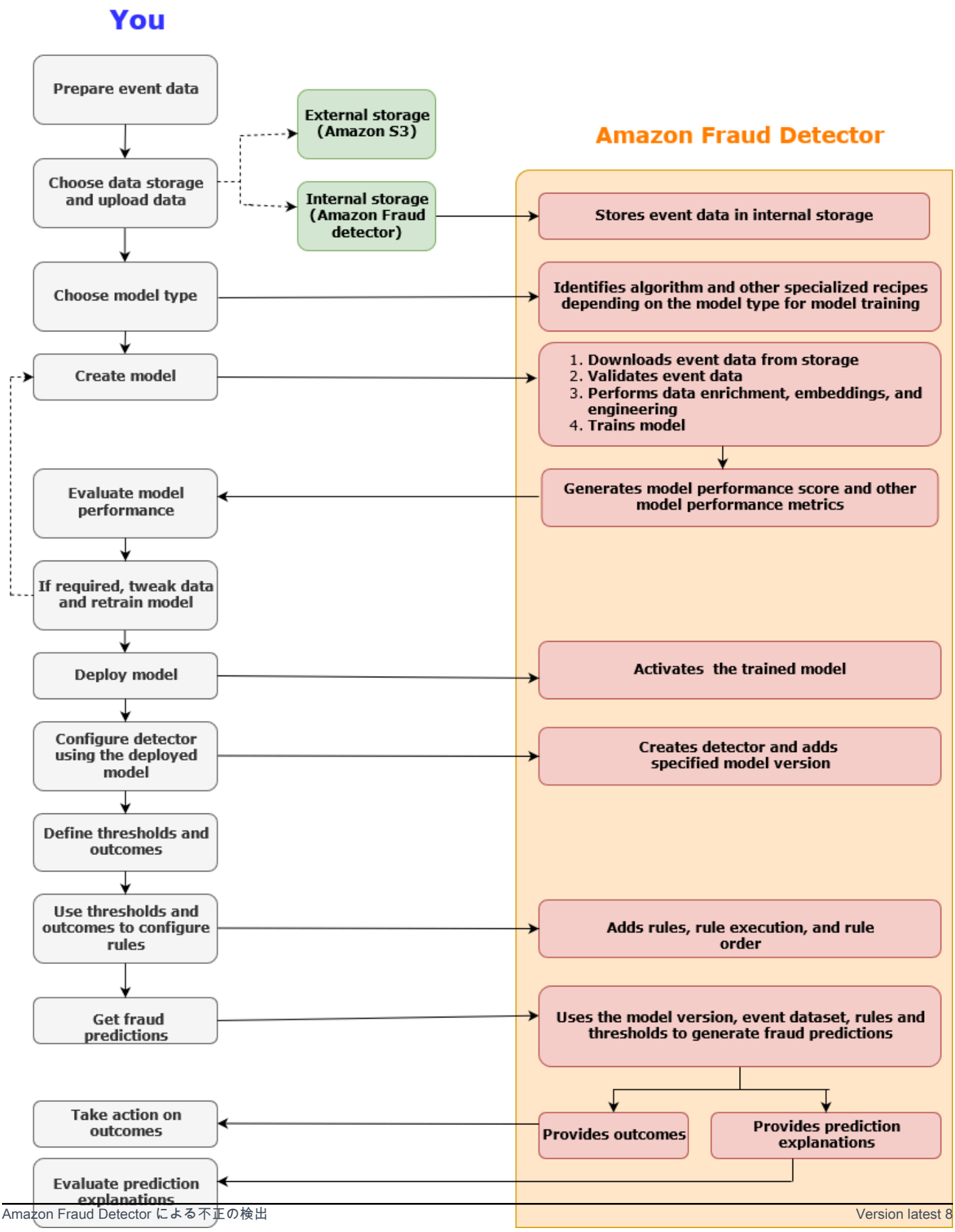
例としては、送金リクエストの送信元が次の場合です。xyz@example.comメールアドレス、審査依頼を送信 ビジネスの本番環境では、資金移動のリクエストが届くと、モデルはそのリクエストに付随するデータを分析し、ルールを使用して結果を割り当てます。その後、割り当てられた結果に応じて、リクエストに対してアクションを実行できます。

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、不正評価ロジックをビジネスに提供します。

Amazon Fraud Detector を使用して詐欺を検出するために使用するワークフローについては、以下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、これらのタスクをどのように達成できるか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、上位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には、データセットを微調整して新しいデータ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。利用可能なデータが増えるにつれて、モデルの再トレーニングを続行して精度を向上させます。

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) 、および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。にサインアップした場合は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 の幅広い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、接続のさまざまな詳細を管理します。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 リソースとプロパティを定義するため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、「 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AWS CloudFormation」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る計算時間、使用するストレージの量、不正予測の数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、所有するリソースにアクセスできるように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完了します。

- 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、Amazon Fraud Detector を含む 上のすべてのサービスに が自動的にサインアップ AWS アカウント 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環として、通話呼び出しを受け取り、電話キーパッドで検証コードを入力するように求められます。

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。セキュリティのベストプラクティスとして、ユーザーに管理アクセスを割り当て、ルートユーザーのみを使用してルートユーザーアクセスが必要なタスクを実行してください。

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] を選んで、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、 を保護し AWS アカウントのルートユーザー、 を有効にして AWS IAM Identity Center、日常的なタスクにルートユーザーを使用しないように管理ユーザーを作成します。

のセキュリティ保護 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 「ルートユーザーの仮想 MFA デバイスを有効にする (コンソール)」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、「 ユーザーガイド」の「デフォルト でユーザーアクセス IAM アイデンティティセンターディレクトリを設定するAWS IAM Identity Center 」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 「 アクセスポータルへのサインイン」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「グループの参加」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
  - のすべてのモデルエンドポイントを一覧表示して説明する SageMaker
  - 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。

ユーザーを作成して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、新しいユーザーが最初にサインインするときに新しいパスワードを作成することを AWS 要求します。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。[グループを作成] を選択します。

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 のサービスには、 からアクセスできます AWS Management Console。は AWS アカウント、 へのアクセスを許可します 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 コンソールで提供される機能と同様の機能のコマンド AWS CLI を実行できます。

を設定するには 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 をセットアップする

AWS SDKs、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 をサポートします。

を設定するには 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てください。

をインストールしたら 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なかった場合、リストは空になります。

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

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

AWS SDKs for Java 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手順については AWS SDK for JavaScript、「 SDK for のセットアップ JavaScript」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、Amazon Fraud Detector による不正の検出の手順を読み、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。

このセクションのハンズオン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、デプロイする方法を学んでください。このチュートリアルでは、機械学習モデルを使用して新規アカウント登録が不正かどうかを予測する詐欺アナリストの役割を担います。このモデルは、アカウント登録のデータを使用してトレーニングされている必要があります。Amazon Fraud Detector は、このチュートリアル用のサンプルアカウント登録データセットの例を提供しています。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、次のインターフェイスのいずれかを使用します。チュートリアルを開始する前に、以下の要件を満たしていることを確認してください。サンプルデータセットの取得とアップロード

- 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の取得とアップロード

このチュートリアルで使用するサンプルデータセットには、オンラインアカウント登録の詳細が記載されています。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。ヘッダー行の後には、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。便宜上、データにはラベルが付けられています。データセットの列は、アカウント登録が不正かどうかを識別します。

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を含むデータファイルには、registration_data_20K_minimum.csv と registration_data_20K_full.csv の 2 つがあります。`registration_data_20K_minimum`このファイルには、ip_address と email_address の 2 つの変数しか含まれていません。`registration_data_20K_full`ファイルには他の変数が

含まれています。これらの変数は各イベント用で、請求先住所、電話番号、ユーザーエージェントが含まれます。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、データセットを格納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. AWS Management Console にサインインし、Amazon S3 コンソール (https://console.aws.amazon.com/s3/) 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。必ずコンソールのバケット命名規則に従い、グローバルに一意な名前を付けてください。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、AWS リージョンバケットを作成したい場所を選択してください。選択する地域は Amazon Fraud Detector をサポートしている必要があります。待ち時間を短縮するには、地理的な場所に最も近い場所を選択してください。AWS リージョン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テーブルを参照してください。
   e. このチュートリアルでは、「オブジェクトの所有権」、「パブリックアクセスをブロック」のバケット設定、「バケットのバージョン管理」、および「タグ」はデフォルト設定のままにします。
   f. このチュートリアルでは、「デフォルトの暗号化」で「無効」を選択します。
   g. バケット設定を確認して、[Create bucket] を選択します。

3. データサンプルを Amazon S3 バケットにアップロードします。

   バケットが用意できたので、以前に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、バケット名が表示されます。バケットを選択します。

b. [Upload] (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. コンピューター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. [宛先]、[権限]、[プロパティ] はデフォルト設定のままにします。

f. 設定を確認して、[アップロード] を選択します。

g. サンプルトデータファイルは 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ておきます。オブジェクトで、アップロードしたばかりのサンプルデータファイルを選択します。

h. オブジェクト概要で、S3 URI の下のロケーションをコピーします。これは、サンプルデータファイルの Amazon S3 ロケーションです。この情報は後で必要になります。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 つのパートで構成されています。第 1 部では、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。第 2 部では、このモデルを使用して不正行為の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終了すると、次のアクションを完了します。

- 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**
>
> 先に進む前に、次の指示に従ってください。サンプルデータセットの取得とアップロード

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 がサポートする不正検出モデルタイプと照合します。データモデルエクスプローラーは Amazon Fraud Detector コンソールに統合されたツールで、ビジネスユースケースに合わせた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では、データセットに含める必要のある必須、推奨、およびオプションのデータ要素に関する洞察も得られます。このデータセットは、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として新規アカウント登録を行います。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が不正検知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ので、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、[新規アカウント詐欺] を選択します。

  d. Amazon Fraud Detector には、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でき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 が不正検出モデルをトレーニングするために使用するアルゴリズム、エンリッチメント、およびトランスフォーメーションを定義します。

     推奨モデルタイプを書き留めておきます。この情報は後でモデルを作成するときに必要になります。

  e. データモデルインサイトペインでは、不正検出モデルの作成とトレーニングに必要な必須データ要素と推奨データ要素に関する洞察が得られます。

     ダウンロードしたサンプルデータセットを見て、表に記載されている必須データ要素と推奨データ要素がすべて含まれていることを確認してください。

     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、不正行為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、イベントを分類するラベルの設定が必要です。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] に、`sample_registration`イベントタイプ名として入力し、必要に応じてイベントの説明を入力します。

  e. [エンティティ] で、[エンティティの作成] を選択します。

  f. 「エンティティの作成」ページで、`sample_customer`エンティティタイプ名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] の [このイベントの変数の定義方法の選択] 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 ロールには、「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、[Create role] を選択します。

  k. [データの場所] に、サンプルデータのパスを入力します。これは、`S3 URI`サンプルデータをアップロードした後に保存したパスです。パスはこれに似ています:`S3://your-bucket-name/example dataset filename.csv`.

  l. [Upload] (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、変数タイプでマッピングします。マッピングはコンソールに表示されます。

  m. [ラベル-オプション] の [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、`fraud`名前として入力します。このラベルは、サンプルデータセットの不正なアカウント登録を表す値に対応しています。

  o. [ラベルの作成] を選択します。

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

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、`sample_fraud_detection_model`モデル名として入力します。必要に応じて、モデルの説明を追加することもできます。
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] に [サンプル登録] を選択します。ステップ 1 で作成したイベントタイプです。
5. 過去のイベントデータでは、

   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 ロールでは、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に、サンプルデータファイルの 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. [不正ラベル] の [ラベルの分類] では、[不正] を選択します。このラベルは、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しているためです。[正当なラベル] では、[正当] を選択します。このラベルは、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しているためです。
3. ラベルなしイベントの処理では、このサンプルデータセットのデフォルト選択である [ラベルのないイベントを無視] 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をトレーニングし始めます。

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は、完了までに約 45 分かかります。モデルトレーニングが完了すると、ステータスが「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と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、トレーニングしたばかりのモデル (sample_fraud_detection_model) を選択し、次に 1.0 を選択します。これは、Amazon Fraud Detector がお客様のモデルに対して作成したバージョンです。

2. モデルパフォーマンスの総合スコアと、Amazon Fraud Detector がこのモデルについて生成したその他すべての指標を見てください。

   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ックの詳細については、「」モデルスコア と「」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 Amazon Fraud Detector モデルには、このチュートリアルのモデルに表示される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メトリクスが期待できます。

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整ったら、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] を選択してから、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、ステータスにはデプロイのステータスが表示されます。デプロイを完了すると、ステータスが [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ており、不正予測を生成できることを示しています。パート B: 不正予測を生成するに進んで、不正予測を生成する手順を完了します。

パート B: 不正予測を生成する

不正予測は、事業活動 (イベント) における不正行為の評価です。Amazon Fraud Detector は、検出器を使用して不正行為の予測を生成します。ディテクターには、不正を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、以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を構築し、デプロイしたモデルを追加し、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のバージョンを作成して有効化します。

ステップ 1: ディテクターの構築

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. ディテクタ詳細の定義ページで、`sample_detector`ディテクタ名を入力します。必要に応じて、ディテクターの説明 (など) を入力します`my sample fraud detector`。
4. [イベントタイプ] で [サンプル登録] 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、Amazon Fraud Detector モデルをディテクターに追加できる可能性があります。まだモデルを作成していない場合は、パート A に進み、モデルの作成、トレーニング、展開の手順を完了してから、パート B に進んでください。

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、以前にデプロイした Amazon Fraud Detector の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、モデルのパフォーマンススコアの解釈方法を Amazon Fraud Detector に指示する条件です。このチュートリアルでは、、、の 3`high_fraud_risk` つのルールを作成します`low_fraud_risk`。`medium_fraud_risk`

1. [ルールの追加] ページの [ルールの定義] にルール名を入力し`high_fraud_risk`、[説明-オプション]**`This rule captures events with a high ML model score`** にルールの説明としてを入力します。
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する例です。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てください。

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、「[Rules] (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によって、定義したすべてのルールを評価するか、または最初にルールが一致したところでルールの評価を停止するかが決まります。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。

ルールの実行順序は、得られ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、次の手順に従って、ルールの順序を変更して目的の順序で実行します。

high_fraud_riskルールがルールリストの一番上にない場合、[順序] を選択してから、[1] をクリックします。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。

FIRST_MATCHEDこのチュートリアル用にを選択し、[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では、不正予測を生成するため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、および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検出器をテストします。

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にする

Amazon Fraud Detector コンソールでは、テストの実行機能を使用して、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] では、イベントが発生した時刻のタイムスタンプと、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一覧から日付を選択し、エンティティ ID に「1234」と入力します。
3. [イベント変数] では、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、`ip_address`と`email_address`フィールドのみが必要です。これは、これらが AmazonFraud Detector モデルのトレーニングに使用される入力だからです。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アドレス:`205.251.233.178`
   - Eメールアドレス:`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のものとなります。最初のルールは、優先順位が最も高いルールです。True と評価されればマッチします。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のものとなります。つまり、それらはすべて真実と評価されているということです。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。サンプルデータセットの ip_address と email_address の値をテストに使用して、結果が期待どおりかどうかを確認できます。
6. 検出器の動作に満足できたら、`Draft`ディテクタをからに昇格させます`Active`。これにより、検出器をリアルタイムの不正検出に使用できるようになります。

   [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、ディテクタのステータスが [ドラフト] から [アクティブ] に変わります。

   この時点で、お使いの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、オンラインでの不正行為をリアルタイムで評価する準備が整いました。CSV 入力ファイルと`CreateBatchPredictionJob` API を使用して、オフラインでイベントを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。不正予測の取得

このチュートリアルを完了すると、次のようになります。

- サンプルイベントデータセットを 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 の不正検出モデルを作成し、トレーニングしました。
- Amazon Fraud Detector が生成したモデルのパフォーマンススコアとその他のパフォーマンス指標を確認しました。
- 不正検出モデルを導入しました。
- 検出器を作成し、展開したモデルを追加しました。
- ルール、ルールの実行順序、および結果をディテクタに追加しました。
- さまざまな入力を行い、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して、検出器をテストしました。
- 検出器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終了すると、次のアクションを完了します。

- Amazon Fraud Detector モデルの構築、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は次のとおりです。

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

  すでにお持ちの場合はAWS SDK をセットアップする、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。
- サンプルデータセットの取得とアップロードこのチュートリアルに必要なファイルの指示に従いました。

開始方法

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、および管理できます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。

インストールしたらAWS SDK for Python (Boto3)、以下の Python コマンドを実行して、環境が適切に設定されていることを確認します。環境が適切に設定されている場合、レスポンスに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、リストは空になります。

```
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、およびルールを作成するために使用する、データセットからのデータ要素です。

次の例では、CreateVariableAPI を使用して 2 つの変数を作成します。`email_address`と`ip_address`変数はとです。これらを対応する変数タイプ (`EMAIL_ADDRESS`および`IP_ADDRESS`) に割り当てますこれらの変数は、アップロードしたサンプルデータセットの一部です。変数のタイプを指定する場合、Amazon Fraud Detector はモデルトレーニングおよび予測を取得するとき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の作成

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、PutEntityTypeAPIsample_customer を使用して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ことを学習します。

次の例では、Putlabel APIfraud を使用してと 2legit つのラベルを作成しています。

```
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'
)
```

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、個々のイベントについて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、PutEventTypeAPI 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email_address、前のステップで作成した変数 (,ip_addresssample_customer)、エンティティタイプ (legit)fraud、およびラベル (,) 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: モデルの作成、トレーニング、およびデプロイ

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べるよう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、トレーニングします。このモデルは、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。

以下のサンプルコードを使用して、オンライン詐欺インサイトモデルを作成して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 S3sample_registration 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

次の例では、CreateModelAPI 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、CreateModelVersionAPI を使用してモデルをトレーニングします。trainingDataSourceと、サンプルデータセットを保存した Amazon S3 RoleArnの場所、の Amazon S3 バケットを指定します'EXTERNAL_EVENTS'externalEventsDetail。trainingDataSchemaパラメータには、Amazon Fraud Detector がサンプルデータをどのように解釈するかを指定します。具体的には、含める変数とイベントラベルの分類方法を指定します。

```
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://your-S3-bucket-name/your-example-data-
filename.csv',
            'dataAccessRoleArn' : 'role_arn'
}
)
```

モデルは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、モデルのバージョン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とその他のモデルパフォーマンスメトリックを確認できます。

モデル性能のレビュー

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。これにより、モデルのパフォーマンススコアとその他のパフォーマンスメトリクスを生成します。

DescribeModelVersionsAPI を使用してモデルのパフォーマンスを確認します。モデルパフォーマンスの総合スコアと、このモデルについて Amazon Fraud Detector によって生成されたその他すべての指標を見てください。

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 Amazon Fraud Detector モデルには、このチュートリアルのメトリクスに似た実世界の不正検出パフォーマンスメトリクスが含まれていることが期待できます。

モデルをデプロイする

トレーニングしたモデルのパフォーマンスメトリックスを確認したら、モデルをデプロイして Amazon Fraud Detector で使用できるようにして、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、UpdateModelVersionStatusAPI を使用します。次の例では、これを使用してモデルのバージョンステータスを 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'
)
```

ステップ 5: ディテクター、結果、ルール、および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、不正行為について評価する特定のイベントを対象としています。ルールは、予測時に変数値の解釈方法を Amazon Fraud Detector に指示するために指定する条件です。結果は、不正予測の結果です。ディテクタには複数のバージョンがあり、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、それに関連するルールが少なくとも 1 つ必要です。

次のサンプルコードを使用して、ディテクタ、ルール、結果を作成し、ディテクタを公開します。

検出器を作成

次の例では、PutDetectorAPIsample_detectorsample_registration を使用してイベントタイプの検出器を作成しています。

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

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

成果を創出

結果は、可能性のある不正予測結果ごとに作成されます。次の例では、PutOutcomeAPI を使用して、verify_customerreview、および 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'
)
```

ルールを作成

ルールは、データセットからの 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ます。

次の例では、CreateRuleAPI を使用して、、high_risk、medium_riskと
いう 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、sample_fraud_detection_model_insightscoreモデルのパフォーマンススコア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判断し、前のステップで定義した結果を割り当てるためです。

```
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']
     )
```

検出器バージョンの作成

検出器バージョンでは、不正行為を予測するために使用されるモデルとルールが定義されています。

次の例では、CreateDetectorVersionAPI を使用して検出器バージョンを作成します。これは、モデルバージョンの詳細、ルール、およびルール実行モード FIRST_MATCHED を提供することで実現されます。ルール実行モードは、ルールを評価する順序を指定します。ルール実行モード FIRST_MATCHED 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。

```
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、sample_detector前のステップで作成した検出器を使用して、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。検出器は、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、一致したルールに関連する結果だけでなく、モデルのパフォーマンススコアも含まれます。

次の例では、GetEventPredictionAPI 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、アカウント登録の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。一番上のヘッダー行の後の各行 (行) は、1 つ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すると、次の操作を行いました。

- Amazon S3 にサンプルイベントデータセットを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、および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 が生成したモデルのパフォーマンススコアとその他のパフォーマンス指標を確認しました。
- 不正検出モデルを導入しました。
- 検出器を作成し、展開したモデルを追加しました。
- ルール、ルールの実行順序、および結果をディテクタに追加しました。
- 探知器バージョンが作成されました。
- さまざまな入力を行い、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して、検出器をテストしました。

(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 API の使用方法に関するその他の例については、「aws-fraud-detector-samples GitHub リポジトリ」を参照してください。ノートブックの対象となるトピックには、Amazon Fraud Detector API を使用したモデルとディテクターの構築、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

モデルと検出器を作成できたので、さらに深く掘り下げてモデルと検出器を作成し、不正予測を生成できます。

Amazon Fraud Detector ユーザーガイドの以下のセクションでは、お客様のビジネスまたは組織が Amazon Fraud Detector を使用して不正行為を検出する方法について説明しています。

- 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
- ディテクターの作成
- 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) を管理します
- セキュリティとコンプライアンス目標を達成するように Amazon Fraud Detector を設定します

- Amazon Fraud Detector を監視し、Amazon Fraud Detector の API 呼び出しを口グに記録します
- Amazon Fraud Detector の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の構造

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。Amazon Fraud Detector では、ビジネスユースケースや不正検出モデルの作成とトレーニングに使用したモデルタイプに応じて、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 ファイルのヘッダーにイベントメタデータを指定するときは、Amazon Fraud Detector で指定されている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てください。
- 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したいイベント固有のデータ要素を表します。ビジネスユースケースや、不正検出モデルの作成とトレーニングに使用した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または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たい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、メールアドレス、IP アドレス、電話番号などがあります。CSV ファイルのヘッダーにイベント変数名を指定するときは、任意の変数名を使用し、小文字のみを使用してください。
- 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の各行は、1 つのイベントのデータ要素で構成されます。たとえば、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ています。行の各データ要素は、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

モデルを作成するため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 は、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、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。データセットが要件を満たさない場合、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すると、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、特定したモデルタイプのデータセット要件を把握できます。

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、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る Amazon Fraud Detector コンソールのツールです。データモデルエクスプローラーでは、Amazon Fraud Detector が不正検知モデルを作成するために必要なデータ要素に関する洞察も得られます。イベントデータセットの準備を始める前に、データモデルエクスプローラーを使用して Amazon Fraud Detector がビジネス用途に推奨するモデルタイプを確認し、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確認してください。

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、詐欺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 は、お客様のビジネスユースケースに合った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のため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。

   推奨モデルタイプを書き留めておきます。モデルの作成について後で必要になります。

> **Note**
>
> ビジネスユースケースが見つからない場合は、説明の「お問い合わせ」リンクを使用して、ビジネスユースケースの詳細をお知らせください。ビジネスユースケースに適した不正検出モデルの作成には、どのモデルタイプを使用するかをおすすめします。

5. データモデルインサイトペインでは、ビジネスユースケースに合わせた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、データセットを作成するためにデータモデルエクスプローラーが提供したデータ要素のリストに留意してください。必須 (イベントメタデータ) データをすべて収集し、モデル作成の目標に基づいて、どの推奨データ要素とオプションデータ要素 (イベント変数) を含めるか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、推奨イベントメタデータを指定するための形式に関するガイダンスを示しています。

> **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	[Format] (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。 • そのイベントはユニークである。 • ビジネスにとって有意義な情報を表している。 • 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	[Format] (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	[Format] (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。 • %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	はい

メタデータ名	[Format] (形式)	必須

	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。	

メタデータ名	[Format] (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_.@+-]+$` • 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

イベント変数について詳しくは、「変数」を参照してください。

> **Important**
> アカウントテイクオーバーインサイト (ATI) モデルを作成する場合は、データの準備と選択の詳細についてを参照してくださいデータの準備。

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。

- 100 件以上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 50 以上のイベント (行) を含める必要があります。

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、データセットに次のものを含めることをお勧めします。

- 最低 3 週間分の履歴データ。ただし、理想的には 6 か月分のデータ。
- イベントデータの合計は最小 10K にしてください。
- 不正として分類された 400 以上のイベント (行) と、正当なものとして分類された 400 以上のイベント (行) を含めてください。
- モデルタイプで 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てください。

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

AWS CloudFormation自動データプロファイラーはスタックを使用して構築されており、数回のクリック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 未満である。
- 不正または正当と特定されたイベントの数が 50 未満である。
- 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、「モデルタイプの選択」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。ラベルを一度追加すると、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

不正検知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、不正行為の有無を評価するビジネスアクティビティ（イベント）を定義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特定し、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する必要があります。

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、以下を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このツールを使用して、Amazon Fraud Detector が必要とするデータ要素に関する洞察を得て、不正検出モデルを作成しました。
- データモデルエクスプローラーから得た洞察を使用してイベントデータセットを作成し、そのデータセットを Amazon S3 バケットにアップロードしました。
- が作成され可変エンティティLabels、Amazon Fraud Detector をこのイベントの不正検出モデルの作成に使用させたいと考えています。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てください。

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 詐欺検出器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」に、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の定義方法の選択」で、
     - このイベントのイベント変数をすでに作成している場合は、変数リストから [変数を選択] 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、Amazon Fraud Detector にデータセットを含む Amazon S3 バケットへのアクセスに使用させたい IAM ロールを選択します。
       - データロケーションに、データセットのロケーションへのパスを入力します。S3 URI次のようなパスを使用してください:`S3://your-bucket-name/example dataset filename.csv`.
       - [Upload] (アップロード) を選択します。
       - 変数には、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。

         不正行為の検出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」を選択すると、変数が不正検出の対象から削除されます。リスト内の変数ごとにこの手順を繰り返します。

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。不正なイベントと合法的なイベントには、必ずラベルを1つずつ選択してください。
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 でのイベントオーケストレーション EventBridge-オプション」で「Amazon でのイベントオーケストレーションを有効にする」をオンにしてください。EventBridge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。

AWS SDK for Python (Boto3) を使用したイベントタイプの作成

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. AWS Management Console にサインインして、Amazon Fraud Detector コンソールを https://console.aws.amazon.com/frauddetector で開きます。
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、削除操作が完了してから最大30秒間、イベントデータを検索できます。

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イベントの数によっては、保存されているすべてのイベントを削除するの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、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。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。

次の表に、モデルタイプとそのモデルがサポートするデータソースの詳細を示します。

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
アカウント乗に関するインサイト	内部ストレージ

Amazon Simple Storage Service を使用してデータセットを外部に保存する方法については、を参照してくださいAmazon S3 でのイベントデータの外部保存 。Amazon Fraud Detector を使用してデータセットを内部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社内に保存する。

Amazon S3 でのイベントデータの外部保存

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**
>
> トランザクション不正インサイトとアカウント乗インサイトでは、Amazon S3 で外部に保存されたデータセットをサポートしていません

CSV ファイルの作成

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の取得とアップロード」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履歴登録イベントを表します。

```
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

Amazon S3 バケットにイベントデータをアップロードする

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. AWS Management Console にサインインし、Amazon S3 コンソール https://console.aws.amazon.com/s3/ 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. [Region] (リージョン) で、バケットを格納する 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社内に保存する

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。1 つのイベントは、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 コンソールでバッチインポート機能を使用するか、CreateBatchImportJobAPI を使用して、大規模なデータセットを Amazon Fraud Detector にすばやく簡単にデータセットの

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の準備

Amazon Fraud Detector によって内部的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。そのため、同じイベントからのすべてのイベントデータは 1 つに保存されます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 不正検出コンソール、API、またはSendEvent API を使用してデータを Amazon Fraud Detector に保存するたびに、Amazon Fraud Detector はデータを保存する前に検証します。CreateBatchImportJob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 を使用してデータを社内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ことを確認するには、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。まだ行っていない場合は、指示に従ってイベントタイプを作成してください。

スマートデータ検証

データセットを Amazon Fraud Detector コンソールにアップロードしてバッチインポートすると、Amazon Fraud Detector はスマートデータ検証 (SDV) を使用してデータをインポートする前にデータセットを検証します。SDV はアップロードされたデータファイルをスキャンし、データの欠落、形式やデータタイプの誤り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解決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 によって特定された問題の一部は重大な場合があり、Amazon Fraud Detector がデータセットを正常にインポートする前に解決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。SDV はファイルレベルでデータファイルをスキャンし、ファイルへのアクセス権限が不十分、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) が正しくない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、データ長、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 コンソールでのみ利用可能で、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 にスマート

データ検証を使わせたくない場合は、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てください。

API またはAWS SDK を使用する場合の保存データの検証

、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SendEventGetEventPrediction、Amazon Fraud Detector は次のことを検証します。

- EventIngestion そのイベントタイプの設定は ENABLED です。
- 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たイベントデータの保存

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できるデータがインポートされます。

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の準備」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 ギガバイト (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。 • 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。 • 正規表現パターンに従う必要があります (例えば、`^[0-9a-z_-]+$.`) • 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。
		• 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。
		  • %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。

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。
		• 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

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。 • 正規表現のパターンを満たす必要があります: `^[0-9A-Za-z_.@+-]+$` • 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。 • 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. AWS Management Console にサインインし、Amazon S3 コンソール https://console.aws.amazon.com/s3/ 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. [Region] (リージョン) で、バケットを格納する 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 コンソールでイベントデータをBatch インポートする

`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-新規設定は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権限と出力 Amazon S3 バケットへの書き込み権限があることを確認してください。

8. [Start] (開始) を選択します。

9. [イベントデータのインポート] ペインの [ステータス] 列には、検証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され、次にインポートが行われるときのステータスの大まかな説明が表示されます。
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監視する。

データセットの検証とインポートジョブの進行状況を監視する

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する場合、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 コンソールの新規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監視できます。ページの上部にあるバナーには、検証結果とインポートジョブのステータスについて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が成功するように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操作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	どうすればよいですか
データ検証が開始されました	検証中	SDV はデータセットの検証を開始しました	ステータスが変わるのを待つ
データセットにエラーがあるため、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	検証に失敗	SDV はデータファイル内の問題を特定しました。データセットのインポートを正常に行うには、この問題に対処	インポートイベントデータページでJob ID を選択し、検証レポートを表示します。レポートに記載されている推奨事項に従って、記載されているすべてのエラーに対処してください。詳細については、「検証レポートを使用する」を参照してください。

バナーメッセージ	ステータス	意味	どうすればよいですか
を参照してください		する必要があります。	
データの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	データセットは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	ステータスが変わるのを待つ
検証は警告付きで完了しました。データのインポートが開始されました	インポート中	データセット内のデータの一部が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監視し、ステータスが変わるのを待ちます

バナーメッセージ	ステータス	意味	どうすればよいですか
データが部分的にインポートされました。一部のデータが検証に失敗し、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。	輸入品。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が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はインポートされました。	インポートイベントデータページでJob 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、記載されている警告に対処してください。すべての警告に対処する必要はありません。ただし、インポートが成功するためには、データセットのデータの 50% 以上が検証に合格してい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、「検証レポートを使用する」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が失敗しました。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	輸入品	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Job ID を選択して詳細を表示し、モデルトレーニングを続行します。

> **Note**
>
> データセットが Amazon Fraud Detector に正常にインポートされた後、システムによって完全に取り込まれるようにするために、10 分待つ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解決するための推奨アクションが記載されています。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、問題の重大度、および修正方法を判断できます。検証レポートは検証が正常に完了しても作成されます。この場合は、レポートを表示してリストされている問題があるかどうかを確認し、問題がある場合は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、バッチインポートが失敗する原因となる可能性のある問題がないか、データセットをスキャンします。検証とバッチインポートが成功しても、データセットにまだ問題があり、モデルトレーニングが失敗する可能性があります。モデルトレーニングを成功させるには、検証とインポートが成功した場合でも検証レポートを確認し、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。問題を解決したら、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

検証が完了した後は、次のいずれかのオプションを使用していつでも検証レポートにアクセスできます。

1. 検証が完了したら、インポートジョブの進行中に、上部のバナーで [検証レポートを表示] を選択します。
2. インポートジョブが完了したら、インポートイベントデータペインで、完了したばかりのインポートジョブのJob ID を選択します。

検証レポートを使用する

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、重大なエラーが見つかった場合はそのリスト、データセット内の特定のイベント（行）が見つかった場合の警告のリスト、および無効な値や各変数の欠損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。

- ジョブの詳細をインポートする

  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] を選択すると、インポートに失敗したイベントのエラーログが表示されます。

- 重大なエラー

  SDV によって特定されたデータセットで最も影響の大きい問題の詳細が表示されます。このペインに表示されている問題はすべて重大であり、インポートを続行する前に解決する必要があります。重大な問題を解決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、各警告に記載されている推奨事項に従ってください。重大なエラーペインに表示されているすべての問題を解決したら、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) に関する警告の概要が表示されます。データレベルの警告ペインにデータ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」列には問題が発生したイベントの数が表示されます。また、サンプルイベントIDにはサンプルイベントIDの一部が記載されており、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てください。また、出力ファイルのエラーログを使用して、問題に関する追加情報も参照してください。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ついて生成されます。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。[警告の数] 列に 0 件を超える警告が表示される場合は、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。警告の数列に 0 と表示されている場合は、引き続きモデルのトレーニングを続けてください。

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およびが含まれている必要がありますiamRoleArn。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 iamRoleArn とは必ず存在しなければなりません。IAM ロールは、Amazon S3 バケットを入力するための読み取り権限と、Amazon S3 バケットを出力するための書き込み権限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のキャンセル

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たイベントデータの保存

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 EventIngestion オペレーションを使用して値を DISABLED に更新します。`GetEventPrediction` API オペレーションの詳細については、「不正予測の取得」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たイベントデータの保存

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

SendEventAPI を使用してイベントデータを保存する場合、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の取得

Amazon Fraud Detector にイベントデータを保存した後、GetEventAPI を使用して、イベントに保存された最新のデータをチェックできます。次のコード例では、sample_registrationイベントについて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の表示

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

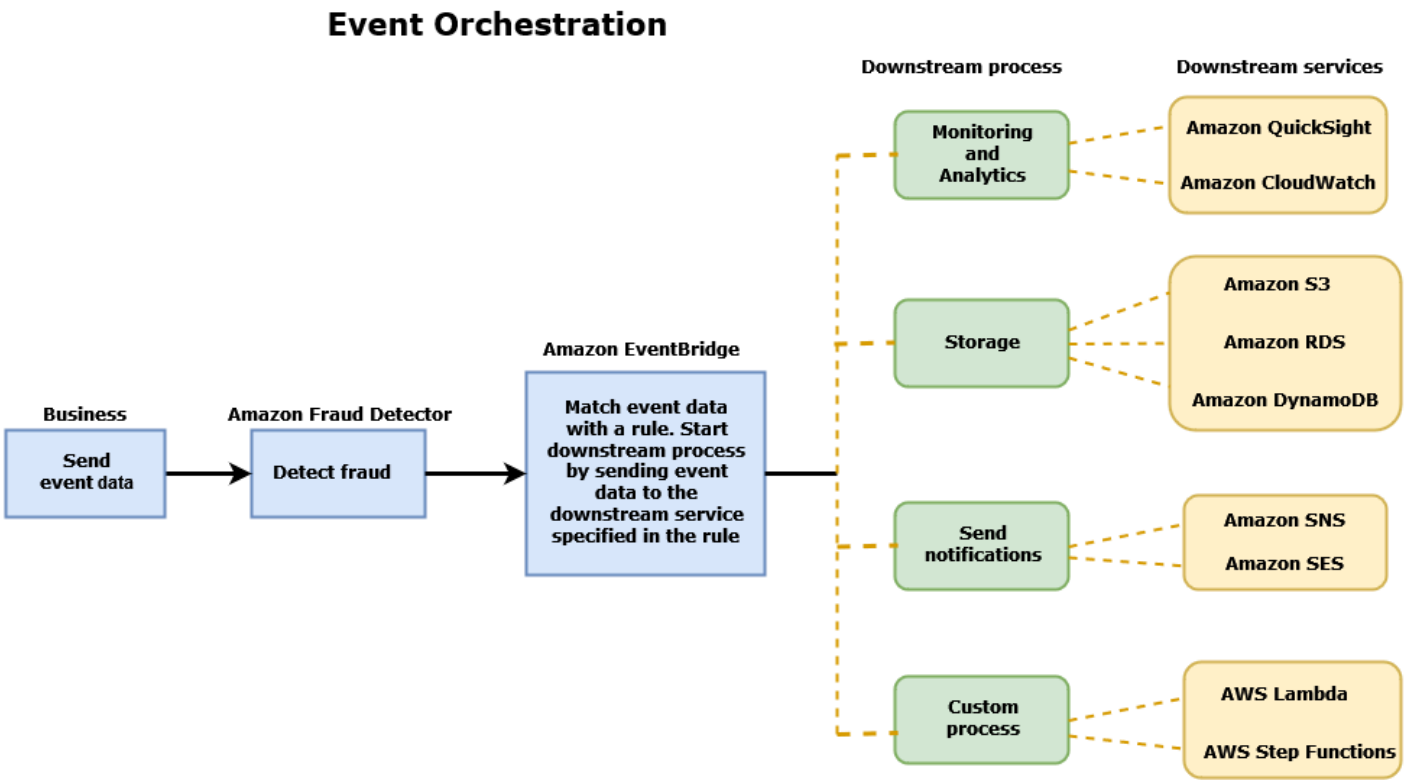
イベントオーケストレーションを使用すると、Amazon EventBridge を使用して、ダウンストリーム処理AWS のサービスのために にイベントを簡単に送信できます。Amazon Fraud Detector には、不正検出後のイベントの処理を自動化するために使用できる簡単なルールが用意されています。イベントオーケストレーションを使用すると、イベントデータからインサイトを取得するためのダッシュボードへのイベントの送信、不正検出の結果に基づく通知の生成、不正検出からの学習に基づくラベルによるイベントの更新など、ダウンストリームイベントプロセスを自動化できます。

イベントオーケストレーションにより、Amazon を通じてAWS環境内のサービスに簡単にアクセスできます EventBridge。API 送信先 に直接イベントを送信するかAWS のサービス、間接的にイベントを送信する EventBridge ように Amazon を設定できます。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オーケストレーション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 — 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 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 EventBridge、「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 EventBridge するように Amazon を設定し、ダウンストリームプロセスを開始する条件 EventBridge を指定するルールを Amazon 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て、Amazon の使用方法を確認してください EventBridge。ユースケース EventBridge に合わせて Amazon で ルールを作成する方法を必ず確認してください。
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する必要があり、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 でイベントパターンを構築する場合は EventBridge、`aws.frauddetector`必ずソースフィールドと`Event Prediction Result Returned`詳細タイプフィールドに を指定してください。

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

イベントタイプの作成時または作成後に、イベントのイベントオーケストレーションを有効にできます。イベントオーケストレーションは、Amazon Fraud Detector コンソールで、 `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て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。

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 でイベントオーケストレーションを有効にする EventBridgeをオンにします。
5. の手順 3 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有効に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、変数 ip_addressおよび email_address、ラベル legitおよび fraud、エンティティタイプ を作成したと仮定しますsample_customer。これらのリソースを作成する方法については、「 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、いつでもイベントのイベントオーケストレーションを無効にできます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 によるイベントオーケストレーションの有効化 EventBridge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

  アカウント乗っ取りインサイトモデルタイプは、アカウントがフィッシングや別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。ログインに使用され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モデルタイプは、オンライン、または card-not-presentトランザクション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インサイト

アカウント乗っ取りインサイト (ATI) モデルタイプは、悪意のある乗っ取り、フィッシング、または認証情報の盗難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、オンライン上の不正な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アカウント乗っ取りインサイト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、さまざまな認証および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のログイン、API ベースの認証、および single-sign-on (SSO) が含まれます。アカウント乗っ取りインサイト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。

アカウント乗っ取りインサイトモデルを使用して、アカウントログインを非同期的に評価し、リスクの高い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、高リスクの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。

アカウント乗っ取りインサイトモデルは、ビジネスのログインイベントの履歴を含むデータセットを使用してトレーニングされます。このデータを指定します。オプションで、アカウントを正当なものとして、または不正なもの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。アカウント乗っ取りインサイトモデルは、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を高める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、同じ一連のデバイスと IP アドレスからログインするユーザーなどです。不正行為者は通常、別のデバイスと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リスクスコアを生成します。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。

アカウント乗っ取りインサイトモデルをトレーニングする前に、Amazon Fraud Detector は機械学習技術を組み合わせて、データの強化、データ集約、および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 データ要素の例としては、IP アドレスやユーザーエージェントなどがあります。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、ジオロケーション入力が含まれます。Amazon Fraud Detector は、指定したログインデータも使用して、過去のユーザーの動作を説明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アグリゲートを使用すると、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。

アカウント乗っ取りインサイトモデルは、悪意のあるアクターが人間かロボットかにかかわらず、悪意のあるアクターによって正当なアカウントが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。侵害された可能性が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。高リスクのアカウントは、2 つの方法のいずれかで処理できます。または、追加の ID 検証を適用することもできます。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

アカウント乗っ取りインサイトモデルは、Amazon Fraud Detector の内部に保存されているデータセットでトレーニングされます。Amazon Fraud Detector を使用して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、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 を使用したイベント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。 Amazon Fraud Detector を使用してイベントデータを社内に保存する

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) ファイルで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。

アカウント乗っ取りインサイトモデルでは、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 メタデータを含めて、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る場合は問題ありません。ラベルを指定すると、Amazon Fraud Detector はラベルを使用してアカウント乗っ取り検出レートを自動的に計算し、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

アカウント乗っ取りインサイトモデルには、必須の (必須の) 変数とオプションの変数の両方を指定する必要があります。変数を作成するときは、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ている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

アカウント乗っ取りインサイト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	有効	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

アカウント乗っ取りインサイトモデルのトレーニングでは、次の変数はオプションです。

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。
タイムスタンプ	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の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、アカウント乗っ取りインサイトモデルのトレーニングのためのこれらの変数は含まれません。

データの選択

データを収集することは、アカウント乗っ取りインサイト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ます。
- 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「不正」または「正当」の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で履歴データを準備し、100K 個のエンティティを含めます。

すでに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

アカウント乗っ取りインサイト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 モデルのトレーニング用にデータセットを検証すると、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、「ステップ 3: モデルを作成する」を参照してください。

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。

AWS SDK for Python (Boto3) を使用したモデルのトレーニングとデプロイ

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」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、具体的にはどのイベント変数を含めるか、およびイベントラベルを分類する方法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。このコード例ではunlabeledEventsTreatment、 AUTOに を使用して、Amazon Fraud Detector がラベルなし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異なる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、モデルスコアを生成するために集計値 (一連の raw 変数を組み合わせて計算された値) のみを使用します。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、同じエンティティおよび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 から 1000 までのモデルスコアを生成します。0 は低不正リスクを示し、1000 は高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 の範囲で指定できます。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、モデルスコアは偽陽性率 (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 — モデルに期待される 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場合 (>0.1)、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

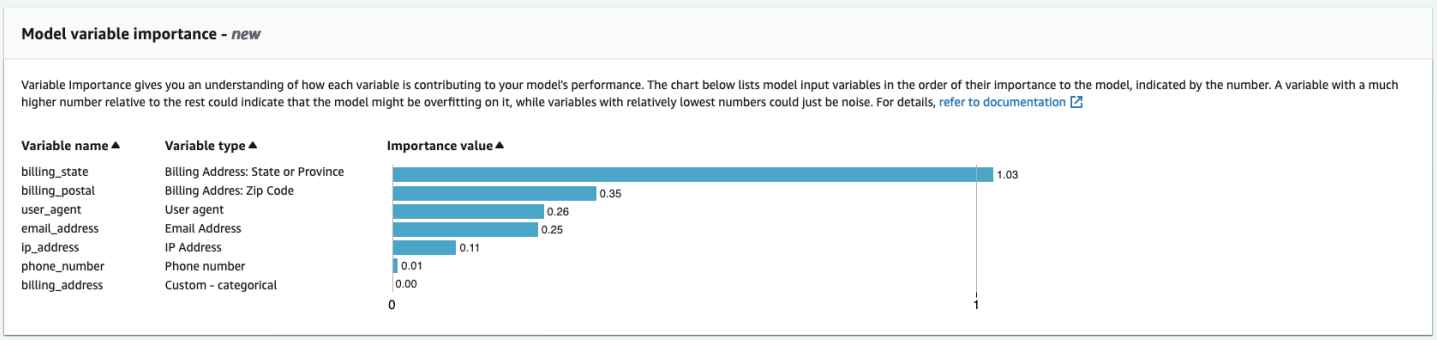
低くなります。モデル変数重要度値を使用すると、モデルのパフォーマンスを左右するのはどのような入力かを把握することができます。

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 コンソールまたは DescribeModelVersion API を使用して表示できます。

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未加工または集計された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) モデルのいずれかについて、モデル変数重要度値が次のように表示されます。アカウント乗っ取りインサイト (ATI) モデルは、未加工の変数重要度値に加えて、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、Amazon Fraud Detector コンソールまたは DescribeModelVersion API を使用して、トレーニングしたモデルバージョンのモデル変数重要度ランキングを表示できます。

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。

モデルのインポート SageMaker

オプションで、 SageMakerホストモデルを Amazon Fraud Detector にインポートできます。モデルと同様に、 `GetEventPrediction` API を使用して SageMaker モデルをディテクターに追加し、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 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 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 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 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 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 モデルをインポートする AWS SDK for Python (Boto3)

SageMaker 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 エンドポイント`sagemaker-transaction-model`がデプロイされ、 `InService` ステータスであり、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。は、GetEventPredictionリクエストの一部として送信される変数から CSV 入力を構築する方法 csvInputTemplate 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 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 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。

ディテクターのバージョンに関連付けられていない場合は、Amazon SageMaker モデルも削除できます。 SageMaker モデルを削除すると Amazon Fraud Detector から引き続き使用できます SageMaker。

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. AWS Management Console にサインインして、Amazon Fraud Detector コンソールを https://console.aws.amazon.com/frauddetector で開きます。
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. [Actions] (アクション) を選択してから、[Delete] (削除) をクリックします。
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 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. SageMaker 削除する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. モデル名を入力してから、[SageMakerモデルを削除する] を選択します。

ディテクター

検出器とは、不正行為の有無を評価したい特定のビジネスイベントを対象とした、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。まず、定義済みのイベントを指定して検出器を作成し、オプションで Amazon Fraud Detector によってその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、ディテクタにルールとルールの実行順序を追加して、ディテクタのバージョンを作成します。検出器バージョンはルールを定義し、オプションで不正予測を生成するリクエストの一部として実行されるモデルも定義します。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任意のモデルを検出器バージョンに追加することもできます。検出器には複数のバージョンがあり、各バージョンには異なるルールとルールの実行順序を設定して、複数のユースケースに対応できます。

各ディテクタのバージョンには、次のステータスが必要です。DRAFT、ACTIVE、または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 詐欺検出器は以下の検出器バージョンを使用しますACTIVE不正行為の予測を行うためのステータス。

ディテクターの作成

検出器を作成するには、定義済みのイベントタイプを指定します。オプションで、Amazon Fraud Detector によって既にトレーニングおよびデプロイされているモデルを追加できます。モデルを追加すると、Amazon Fraud Detector によって生成されたモデルスコアをルール作成時のルール表現で使用できます (たとえば、`$model score < 90`)。

Amazon 詐欺検出器コンソールでは、次の方法で検出器を作成できます。PutDetectorAPI、を使用するプットディテクターコマンド、またはAWSSDK。API、コマンド、または SDK を使用して検出器を作成している場合は、検出器を作成した後に、次の指示に従ってください。ディテクターバージョンの作成。

Amazon 詐欺検出器コンソールで検出器を作成する

この例では、イベントタイプを作成し、不正行為の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たことを前提としています。

ステップ 1: 検出器をビルドする

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. に検出器の詳細を定義ページ、入力sample_detector検出器名用。オプションで、次のような検出器の説明を入力します。my sample fraud detector。
4. にとってイベントタイプで、不正行為予測用に作成したイベントタイプを選択してください。
5. [次へ] をクリックします。

ステップ 2: デプロイし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。検出器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. にモデルを追加-オプション、選択モデル追加。
3. にモデル追加ページ、用モデル選択で、以前にデプロイした Amazon 不正検知器のモデル名を選択してください。にとってバージョンを選択で、デプロイし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. [次へ] をクリックします。

ステップ 3: ルールを追加する

ルールとは、Amazon Fraud Detector に不正予測を評価する際に変数値をどのように解釈するかを指示する条件です。この例では、モデルスコアを変数値として使用して 3 つのルールを作成します。high_fraud_risk、medium_fraud_risk、およびlow_fraud_risk。独自のルール、ルール表現、ルールの実行順序、および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てください。

1. にルールを追加ページ、下ルールを定義、入力high_fraud_riskルール名とその下用説明-オプション、入力**This rule captures events with a high ML model score**ルールの説明として。
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. ユースケースのルールをすべて作成したら、[次へ]。

   ルールの作成と記述の詳細については、「[Rules] (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: ルール実行とルール順序の設定

検出器に含まれるルールの実行モードによって、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するかが決まります。また、ルールの順序によって、ルールの実行順序が決まります。

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行為の予測結果に影響する可能性があります。ルールを作成したら、次の手順に従ってルールの順序を変更し、目的の順序で実行します。

もし、あなたのhigh_fraud_riskルールがまだルールリストの一番上にない場合は注文を選択し、1。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。

[選択]FIRST_MATCHEDこのチュートリアルでは、次に[次へ]。

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

検出器のバージョンは、詐欺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. にレビューと作成ページで、設定した検出器の詳細、モデル、および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。

あなたは使うドラフト検出器をテストするためのバージョン。

AWS SDK for Python (Boto3) を使用したディテクターの作成

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

検出器バージョンは、不正予測を生成するリクエストの一部として使用されるルール、ルールの実行順序、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。

AWS SDK for Python (Boto3) を使用したディテクターバージョンの作成

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. AWS Management Console にサインインして、Amazon Fraud Detector コンソールを https://console.aws.amazon.com/frauddetector で開きます。
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、およびディテクターは、変数、結果、ラベル、リスト、およびエンティティを使用して、不正リスクに関するイベントを評価します。このセクションでは、リソースの作成および管理について説明します。

トピック

- 可変
- Labels
- [Rules] (ルール)
- リスト
- 結果
- エンティティ
- AWS CloudFormation を使用した Amazon Fraud Detector リソースの管理

可変

変数は、不正行為の予測に使用したい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 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した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。

不正行為の予測に使用する変数は、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タイプ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 (BIN)、電話番号など、ユーザーが入力する変数の一部を強化して、これらの変数を使用するモデルの入力を増やし、パフォーマンスを向上させます。

データ型

変数には、その変数が表すデータ要素のデータ型が必要で、変数タイプオプションで定義済みのデータ型のいずれかを割り当てることができます。変数型に割り当てられた変数の場合、データ型は事前に選択されています。使用可能なデータ型には次の種類があります。

データ型	説明	デフォルト値	値の例
文字列	文字、整数、またはその両方の任意の組み合わせ	<空>	abc、123、13B
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 フォーマットでのみ指定されている日付と時刻	<空>	2019-11-30 13:01:01 Z
浮動小数点	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 が不正行為の予測を生成すると、Amazon Fraud Detector が変数の値を受け取らない場合に、このデフォルト値を使用してルールまたはモデルが実行されます。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、ブール値、`false`浮動小数点数、文字列には (空) `0.0` のデフォルト値を割り当てます。`0`これらのデータ型のいずれにも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、オプションでその変数を変数型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、変数の強化、機能エンジニアリング、およびリスクスコアリングを実行します。

Amazon Fraud Detector には、変数への割り当てに使用できる以下の変数タイプがあらかじめ定義されています。

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たIPアドレス	文字列	192.0.2.0 注:Amazon詐欺検出器はこのデータを強化します。詳細については、「位置情

カテゴリ	変数タイプ	説明	データ型	例
				報エンリッチメント」を参照してください。
	USERAGEN	イベント中に収集されたユーザーエージェント	文字列	Mozilla 5.0 (Windows NT 10.0、Win64、x64、rv:68.0) Gecko 20100101
	FINGERPRINT	イベントに使用されるデバイスの固有識別子	文字列	sadfow987u234

カテゴリ	変数タイプ	説明	データ型	例
	SESSION_ID	イベントのアクティブセッションのセッション ID	文字列	123456789
	資格情報は有効ですか	イベント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	ブール値	True
ユザ	EMAIL_ADDRESS	イベント中に収集されたメールアドレス	文字列	abc@domain.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	文字列	+1 555-0100 注:Amazon 詐欺検出器はこのデータを強化します。詳細については、「電話番

カテゴリ	変数タイプ	説明	データ型	例
				号の充実」を参照してください。
請求	BILLING_NAME	請求先住所に関連する名前	文字列	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	文字列	+1 555-0100 注:Amazon 詐欺検出器はこのデータを強化します。詳細については、「電話番

カテゴリ	変数タイプ	説明	データ型	例
				号の充実」を参照してください。
	BILLING_ADDRESS_L	請求先住所の最初の行	文字列	どの通りでも
	請求先住所_L2	請求先住所の 2 行目	文字列	任意のユニット123

カテゴリ	変数タイプ	説明	データ型	例
	ビリング・シティ	請求先住所にある都市	文字列	すべての都市
	ビリング・ステート	請求先住所にある州または都道府県	文字列	すべての州または省

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	文字列	どの国でも 注:Amazon 詐欺検出器はこのデータを強化します。詳細については、「	

カテゴリ	変数タイプ	説明	データ型	例	
				位置情報エンリッチメント」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	請求先住所の郵便番号	文字列	01234 注:Amazon 詐欺検出器はこのデータを強化します。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。
配送	SHIPPING_NAME	配送先住所に関連する名前	文字列	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	文字列	+1 555-0100 注:Amazon 詐欺検出器はこのデータを強化します。詳細については、「電話番

カテゴリ	変数タイプ	説明	データ型	例
				号の充実」を参照してください。
	SHIPPING_ADDRESS_1	配送先住所の最初の行	文字列	123 Any Street
	SHIPPING_ADDRESS_2	配送先住所の 2 行目	文字列	Unit 123
	SHIPPING_CITY	配送先住所の都市	文字列	どの都市でも

カテゴリ	変数タイプ	説明	データ型	例
	SHIPPING_STATE	配送先住所にある州または都道府県	文字列	すべての州

カテゴリ	変数タイプ	説明	データ型	例	
	SHIPPING_COUNTRY	配送先住所に記載されている国	文字列	どの国でも 注:Amazon 詐欺検出器はこのデータを強化します。詳細については、「	

カテゴリ	変数タイプ	説明	データ型	例
				位置情報エンリッチメント」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	配送先住所の郵便番号	文字列	01234 注:Amazon 詐欺検出器はこのデータを強化します。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。
支払い	ORDER_ID	トランザクションの固有識別子	文字列	LUX60
	料金	注文金額の合計	文字列	560.00
	CURRENCY	ISO 4217 通貨コード	文字列	USD

カテゴリ	変数タイプ	説明	データ型	例
	PAYMENT_TYPE	イベント中の支払いに使用される支払い方法	文字列	クレジットカード
	AUTH_COD	クレジットカード発行会社または発行銀行から送信される英数字コード	文字列	0000
	AVS	カードプロセッサからの住所検証システム (AVS) 応答コード	文字列	Y
製品	PRODUCT_ATEGORY	注文商品の製品カテゴリ	文字列	キッチン
Custom	NUMERIC	実数として表現できる任意の変数	浮動小数点	1.224
	CATEGORICAL	カテゴリ、セグメント、またはグループを記述する変数	文字列	ラージ

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 (カスタマーレビューやコメントなど)	文字列	フリーフォームテキスト入力の例

変数型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がある場合は、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。変数型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が変数を使用している場合は、後で割り当てを変更するのが非常に難しくなります。

変数には、定義済みの変数タイプまたはカスタム変数タイプ
(、FREE_FORM_TEXT、CATEGORICALまたは) のいずれかを割り当てることができます。NUMERIC

変数を適切な変数型に割り当てる際の重要な注意事項

1. 変数が定義済みの変数型のいずれかに一致する場合は、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てください。たとえば、変数タイプに ip_address 変数を割り当てても、ip_address EMAIL_ADDRESS 変数には ASN、ISP、位置情報、リスクスコアなどのエンリッチメントは追加されません。詳細については、「可変エンリッチメント」を参照してください。
2. 変数が定義済みの変数タイプのいずれにも一致しない場合は、以下の推奨事項に従ってカスタム変数タイプの 1 つを割り当ててください。

3. 通常、自然な順序付けがなく、カテゴリ、セグメント、またはグループに分類できる変数には、CATEGORICAL変数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)。次のデータを含む変数には、CATEGORICAL 変数タイプを割り当てる必要があります-
   - 顧客ID、セグメントID、色ID、部門コード、製品IDなどのデータを含む変数。
   - true、false、または NULL 値の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、紹介媒体などのグループまたはカテゴリに入れることができる変数。

   > **Note**
   >
   > ENTITY_IDは Amazon 詐欺検出器が ENTITY_ID 変数に割り当てるために使用する予約済み変数タイプです。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。トランザクション詐欺インサイト (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。アクションを開始するエンティティを一意に識別するデータ内の変数を決定し、それを ENTITY_ID 変数として渡す必要があります。CATEGORICAL 変数タイプをデータセット内のその他すべての ID に割り当てます (存在する場合やモデルトレーニングに使用している場合)。データセットのエンティティではない他のIDの例としては、Merchant_ID、Policy_ID、Campaign_IDなどがあります。

4. テキストブロックを含む変数に変数タイプを割り当てますFREE_FORM_TEXT。FREE_FORM_TEXT 変数タイプの例としては、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。FREE_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。区切り文字には、英数字とアンダースコア記号以外の任意の文字を使用できます。たとえば、ユーザーのレビューやコメントは「スペース」で区切ったり、日付や紹介コードではハイフンを区切ってプレフィックス、サフィックス、中間部分を区切ることができます。Amazon 詐欺検出器は区切り文字を使用して FREE_FORM_TEXT 変数からデータを抽出します。
5. 実数で固有の順序を持つ変数には NUMERIC 変数型を割り当てます。数値変数の例としては、曜日、インシデントの重大度、顧客評価などがあります。これらの変数には CATEGORICAL 変数型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 変数型に割り当てることを強くお勧めします。

可変エンリッチメント

Amazon Fraud Detector は、IP アドレス、銀行識別番号 (BIN)、電話番号など、ユーザーが提供した未加工のデータ要素の一部を強化して、これらのデータ要素を使用するモデルの追加入力やパフォーマンスの向上を図ります。このエンリッチメントは、疑わしいと思われる状況を特定し、モデルがより多くの不正行為を捉えるのに役立ちます。

電話番号の充実

Amazon Fraud Detector は、位置情報、元の配送業者、電話番号の有効性に関連する追加情報を電話番号データに追加します。電話番号のエンリッチメントは、2021 年 12 月 13 日以降にトレーニングを受け、電話番号に国コード (+xxx) が含まれる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て、2021 年 12 月 13 日より前にトレーニングを行った場合は、この拡張機能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。

データが正常にエンリッチ化されるように、電話番号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	[Format] (形式)	説明
PHONE_NUMBER	E.164 スタンダード	電話番号には必ず国コード (+xxx) を含めてください。
請求先電話と配送先電話番号	E.164 スタンダード	電話番号には必ず国コード (+xxx) を含めてください。

位置情報エンリッチメント

2022 年 2 月 8 日以降、 Amazon Fraud Detector は、イベントについて指定した IP_ADDRESS、BILLING_ZIP、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、不正検知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、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、BILLING_ZIPとSHIPPING_ZIPの各値には、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に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 値に関連するロケーションを特定できない場合は、代わりに特別なプレースホルダー値が使用されます。たとえば、あるイベントの IP_ADDRESS と BILLING_ZIP の値は有効だが、SHIPPING_ZIP の値は有効ではないとします。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。IP_ADDRESS—>SHIPPING_ZIPとBILLING_ZIP—>SHIPPING_ZIPについてはエンリッチメントは行われません。代わりに、プレースホルダー値が代わりに使用されます。モデルで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関係なく、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。

位置情報変数形式

位置データを正しくエンリッチ化するために、位置情報変数には以下の形式を使用することを強くお勧めします。

変数	[Format] (形式)	説明
IP_ADDRESS	IPv4 アドレス	例えば-1.1.1.1
請求先_郵便番号と配送_郵便番号	指定された国の ISO 3166-1 アルファ-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。
請求先の国と配送先の国	ISO 3166-1 アルファ-2 の 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その ISO 3166-1 2 文字の標準

変数	[Format] (形式)	説明
		国コードと一致させようとします。ただし、これらが正しく一致することは保証できません。

国と地域のコード

次の表は、Amazon Fraud Detector が位置情報エンリッチメントをサポートしている国と地域の一覧です。それぞれの国と地域には、国コード (具体的には 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ています。

郵便番号の形式

- 9-数字
- A-レター
- [X]-X はオプションです。たとえば、ガースニーの「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 つの形式を使用してください。
- [X/XX]-X または XX のいずれかを使用できます。たとえば、バミューダ諸島の「aa [aa/99]」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、両方は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。たとえば、アンドラの郵便番号は AD999 です。つまり、国コードはADの文字で始まり、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	999
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	アリゾナ州 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/9]
BY	ベラルーシ	999999
CA	カナダ	a9
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	99 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] 9 [a/9] aa
GG	ガーンジー代官管轄区	GY9 [9] aa
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] aa
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] aa
JP	日本	999-999
KR	大韓民国	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	9999
LV	ラトビア	9999
MC	モナコ	99999
MD	モルドバ共和国	9999
MH	マーシャル諸島	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 999
MX	メキシコ	99999
MY	マレーシア	99999
NL	オランダ	999
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	再会	99999
RO	ルーマニア	999999
RU	ロシア連邦	999999
SE	スウェーデン	99 99
SG	シンガポール	999999
SI	スロベニア	9999
SK	スロバキア	99 99
SM	サンマリノ	99999
TH	タイ	99999
TR	トルコ	99999
UA	ウクライナ	99999
US	アメリカ	99999
UY	ウルグアイ	99999
VI	米領バージン諸島	99999
WF	ワリス・フツナ	99999
YT	マヨット	99999
ZA	南アフリカ	9999

ユーザーエージェントエンリッチメント

アカウントテイクオーバーインサイト (ATI) モデルを作成する場合は、useragentデータセットに変数タイプの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、、、user_agent_familyOS_familyなどの追加情報をユーザーエージェントデータに追加します。device_family

変数の作成

変数は、Amazon 詐欺検出器コンソールで作成することができます。変数を作成するには、create-variable コマンドを使用するか、またはを使用します。CreateVariableAWS SDK for Python (Boto3)

Amazon 詐欺検出コンソールを使用して変数を作成する

この例では、と 2 email_address つの変数を作成しip_address、それらを対応する変数タイプ (EMAIL_ADDRESSと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てください。可変エンリッチメント変数を作成する前に変数タイプ、必ずその内容と内容をお読みください。

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. 「新規変数」ページで、email_address変数名として入力します。オプションで、変数の説明を入力します。
4. 「変数タイプ」で、「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タイプを自動的に選択します。変数に自動的に変数タイプが割り当てられない場合は、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. 変数に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、この手順をスキップしてください。
7. [作成] を選択します。
8. email_address の概要ページで、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] を選択して更新内容を入力します。[変更を保存] をクリックします。

9. ip_addressこのプロセスを繰り返して別の変数を作成し、変数タイプとして [IP Address] 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、どの変数をモデルにトレーニングして不正行為を検出したり不正行為を検出したりするかを決めることができます。

AWS SDK for Python (Boto3) を使用した変数の作成

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てください。可変エンリッチメント変数を作成する前に変数タイプ、必ずその内容と内容をお読みください。

必ず変数ソースを指定してください。変数値の派生元を特定するのに役立ちます。変数ソースが EVENT の場合、GetEventPrediction変数値は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 詐欺検出器によって入力されます。もしそうならEXTERNAL_MODEL_SCORE、SageMaker変数値はインポートされた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を削除することはできません。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 SageMaker モデル出力変数と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 詐欺検出器コンソールで変数を削除するには、delete-variable CLI コマンドを使用するか、DeleteVariableAPI を使用するか、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. AWS Management Console にサインインして、Amazon Fraud Detector コンソールを https://console.aws.amazon.com/frauddetector で開きます。
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. [Actions] (アクション) を選択してから、[Delete] (削除) をクリックします。
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、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'

)
```

Labels

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。オンライン詐欺インサイト (OFI) モデルまたはトランザクション詐欺インサイト (TFI) モデルのトレーニングを計画している場合、トレーニングデータセット内の少なくとも400のイベントを不正または正規のイベントとして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、Fraud、legit、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がイベントに不正行為がないか評価し、これらの値を使用してイベントを不正または正規のイベントとして分類します。

まず、トレーニングデータセットで使用されている値を使用してラベルを作成し、次にそのラベルを不正検出モデルの構築とトレーニングに使用するイベントタイプに関連付ける必要があります。

ラベルの作成

Amazon Fraud Detector コンソールでは、put-label コマンド、PutLabelAPI、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]]]]]]]]]]]]]]]]]]]]]]]]
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 つ目のラベルを作成し、正規イベント用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てください。

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

AWS SDK for Python (Boto3)次のコード例では、PutLabelAPI を使用して、2 つのラベル((、、、、、、、、、、、、、、、、、、、、、、、、、ラベルの作成後、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'
)
```

ラベルを更新

イベントデータセットが Amazon Fraud Detector に保存されている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 にイベントデータセットを、Amazon Fraud Detector に保存イベントのイベントデータセットを追加または更新する必要がある場合があります。

update-event-label保存されているイベントのラベルを追加または更新するには、コマンド、UpdateEventLabelAPI、またはAWS SDK for Python (Boto3)

AWS SDK for Python (Boto3)次のコード例では、UpdateEventLabel API を使用してイベントタイプ登録に関連するラベル詐欺を追加します。

```
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データラベルを更新する

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。 UpdateEventLabel API コールの例を以下に示します。

```
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 のイベントタイプタイプタイプタイプタイプタイプタイプタイプタイプタイプタイプタイプ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は削除できません。まず、関連するイベント ID を削除する必要があります。

ラベルを削除Fraud Detector delete-label コマンド、DeleteLabelAPI、またはAWS SDK for Python (Boto3)

コンソールを使用してラベルの削除

ラベルを削除するには

1. AWS Management Console にサインインして、Amazon Fraud Detector コンソールを https://console.aws.amazon.com/frauddetector で開きます。
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. [Actions] (アクション) を選択してから、[Delete] (削除) をクリックします。
5. ラベル名を入力してから、[ラベルの削除] を選択します。

を使用してラベルの削除AWS SDK for Python (Boto3)

AWS SDK for Python (Boto3)次のコード例では、DeleteLabelAPI を使用してラベルを合法的に削除します。

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

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

[Rules] (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。ルールは検出器ロジックの一部で、次の要素で構成されています。

- 変数またはリスト — 変数は、不正行為の予測に使用したい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の変数に対する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、評価対象のイベントタイプで事前に定義されている必要があり、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、可変およびリストを参照してください。
- 表現 — 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する場合、変数、>、<、<=、>=. == などの比較演算子、および値を使用して簡単なルール式が作成されます。リストを使用する場合、ルール表現はリストエントリ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 — 結果とは、ルールが一致した場合に Amazon Fraud Detector から返される応答です。結果は不正行為の予測結果を示します。起こり得る不正行為の予測ごとに結果を作成して、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、ディテクタには最大 30 のリストを含めることができます。ディテクタ作成プロセスの一部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
```

リストを使う

変数型に関連付けられていて、エントリが入力されているリストならどれでもルール式の一部として使用できます。リストエントリの値を示すにはドル記号を使用してください。

```
$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	<
以下と同等かそれ以上	>=
以下	<=
中	中
および	and
または	または
以外	!

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] (乗算)	*
/ (除算)	/
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)
エスケープ特殊文字	\

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 詐欺検出器では、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機能

関数	説明	例
現在の日付/時刻を取得 ()	ルール実行の現在の時刻を ISO8601 UTC 形式で表示します。getepoch ミリ秒 (getcurrentdatetime ()) を使うと追加の操作を実行することができます	getcurrentdatetime () ==「2023-03-28T 18:34:02 Z」
以前です (DateTime1, DateTime 2)	DateTime呼び出し元の 1 が 2 より前の場合は、ブール値 (True/False) を返します DateTime	以前です (getcurrentdatetime (),「2019-11-30T 01:01:01 Z」) ==「False」 以前です (getcurrentdatetime (),「2050-11-30T 01:05:01 Z」) ==「True」
シフター (DateTime1, DateTime 2)	DateTime呼び出し元の 1 が 2 の後の場合は、ブール値 (True/False) を返します DateTime	isafter (現在の日付時刻を取得 (),「2019-11-30T 01:01:01 Z」) ==「True」 isafter (getcurrentdatetime (),「2050-11-30T 01:05:01 Z」) ==「False」
GET エポックミリ秒 () DateTime	a を受け取りDateTime、DateTime それをエポックミリ秒単位で返します。日付に数学演算を実行するのに便利です	getepochミリ秒 (「2019-11-30T 01:01:01 Z」) = 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	# 自分のコメント

ルールを作成する

ルールは、Amazon 詐欺検出器コンソールで create-rule コマンド、CreateRuleAPI、またはを使用して作成できます。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、事前に定義され、変数タイプに関連付けられ、エントリが入力されている必要があります。

次の例では、`high_risk`既存の検出器のルールを作成しますpayments_detector。`verify_customer`ルールは式と結果をルールに関連付けます。

前提条件

下記の手順を実行するには、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

ユースケースに合わせて検出器、ルール、結果を作成する場合は、例の検出器名、ルール名、ルール式、結果名を、ユースケースに関連する名前と式に置き換えてください。

Amazon 詐欺検出器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 [検出器] を選択し、ユースケース用に作成した検出器 (payments_detectorなど) を選択します。
3. payments_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. 「新規ルール」ページで、次のように入力します。
   a. 「名前」に、ルールの名前 (例) を入力します **high_risk**
   b. 「説明 (オプション)」に、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. エクスプレッションには、エクスプレッション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. アウトカムで、ユースケース用に作成したアウトカムを選択します (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. [ルールを保存] を選択します

検出器の新しいルールを作成しました。これは Amazon Fraud Detector が自動的に検出器で使用できるようにするルールのバージョン 1 です。

AWS SDK for Python (Boto3) を使用したルールの作成

次のコード例では、CreateRuleAPI `high_risk` を使用して既存の検出器のルールを作成しますpayments_detector。サンプルコードでは、verify_customerルール式と結果をルールに追加しています。

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以下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

ユースケースに合わせて検出器、ルール、結果を作成する場合は、例の検出器名、ルール名、ルール式、結果名を、ユースケースに関連する名前と式に置き換えてください。

```
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']
)
```

これで、Amazon Fraud Detector が自動的に検出器で使用できるようにするルールのバージョン 1 が作成されました。

ルールを更新

ルールの説明を追加または更新したり、ルール式を更新したり、ルールの結果を追加または削除したりすることで、いつでもルールを更新できます。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、update-rule-versionコマンド、UpdateRuleVersionAPI、または AWS SDK を使用してルールを更新できます。

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検出器のバージョンを更新してください。

Amazon 詐欺検出器コンソールのルールを更新

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、[Detectors] を選択します。
3. 検出器ペインで、更新するルールに関連付けられている検出器を選択します。
4. ディテクタページで [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ていることに注意してください。更新された説明、表現、または結果を入力します。
7. [新しいバージョンを保存] を選択します

を使用してルールを更新 AWS SDK for Python (Boto3)

次のコード例では、UpdateRuleVersionAPI 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。このルールは検出器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']
)
```

リスト

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。入力データは、ディテクタに関連するルールで使用します。ルールは、不正検出時に入力データをどのように解釈するかを Amazon Fraud Detector に対して指示する条件です。たとえば、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含まれている場合は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、$ip_address_value@list_nameの形式で表現されます。

Amazon Fraud Detector を使用すると、関連するルールを更新しなくても、データを追加または削除して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 個の固有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。ルールで使用するリストはすべて、デフォルトで Amazon 詐欺検出器の変数タイプ FREE_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。1 つのルールには最大 3 つのリストを使用できます。

API、、またはAWS SDK を使用して、Amazon Fraud Detector コンソールでリストを作成したり、リストにエントリを追加したり、リストを削除したり、リスト内の 1 つ以上のエントリを削除したり、リストに変数タイプを割り当てたりすることができます。AWS CLI

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 Amazon Fraud Detector変数タイプ がサポートするリストにリストを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 FREE_FORM_TEXT 変数タイプであると見なします。

Amazon Fraud Detector コンソールで、API または SDK を使用して、API またはAWS SDK を使用して、Amazon Fraud Detector コンソールでAWS CLI、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」で

   a. [リスト名] に、リストの名前を入力します。

   b. [説明] に説明を入力します。

   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 FREE_FORM_TEXT 変数タイプであると見なします。

4. [リストデータの追加] に、リストエントリを 1 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 つの同一のエントリを入力した場合、1 つだけが追加されます。

5. [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)

リスト名を指定してリストを作成します。オプションで、リストを作成するときに、説明を入力したり、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、後でリストに変数タイプを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 FREE_FORM_TEXT 変数タイプであると見なします。

次の例では、CreateListAPI 操作を使用して説明と変数タイプを指定し、4 つのリストエントリを追加することでリストを作成します。`allow_email_ids`

```
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、リストに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 個の固有のエントリを含めることができ、各エントリには最大 320 文字まで入力できます。

API または SDK を使用して、Amazon Fraud Detector コンソールで、API またはAWS SDK を使用してAWS CLI、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、[リストデータ] タブを選択し、[データを追加] 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。コンマを使用してエントリを区切らないようにしてください。
6. [Add] (追加) を選択します。

を使用してリストにエントリを追加しますAWS SDK for Python (Boto3)

次の例では、UpdateListAPIallow_email_ids オペレーションを使用してリストに 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てください。

```
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

ルールで使用するリストはすべて、Amazon Fraud Detector変数タイプ の変数タイプに関連付けられている必要があります。デフォルトでは、Amazon Fraud Detector はリストが FREE_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には IP_ADDRESS 変数タイプを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後から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。すでにリストを変数型に関連付けていて、後で変更したい場合は、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は変更できません。

API または SDK を使用して、Amazon Fraud Detector コンソールで、API または SDK を使用して、API または SDK を使用して、Amazon Fraud Detector コンソールでAWS CLI、API またはAWS SDK を使用して、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リストに変数タイプを割り当てる

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、「リストを編集」を選択します。
5. 編集リストボックスで、リストの変数タイプを選択します。
6. [Save] (保存) 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ますAWS SDK for Python (Boto3)

次の例では、UpdateListAPIallow_ip_address オペレーションを使用して変数タイプをリストに割り当てます。

```
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は削除できます。リストを削除すると、Amazon Fraud Detector はそのリストとリストのすべてのエントリを完全に削除します。

API または SDK を使用して、Amazon Fraud Detector コンソールで、APIAWS 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、「リストを削除」を選択します。
5. [リストを削除] を選択します。

を使用してリストを削除しますAWS SDK for Python (Boto3)

次の例では、DeleteListAPIallow_email_ids オペレーションを使用して削除します。

```
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、いつでも削除できます。リスト内のエントリを削除しても、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。

API または SDK を使用して、Amazon Fraud Detector コンソールのリストから、APIAWS CLI またはAWS SDK を使用して、エントリを削除できます。

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、削除するエントリが含まれている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] を選択して確定します。

を使用してリストからエントリを削除しますAWS SDK for Python (Boto3)

次の例では、UpdateListAPIallow_email_ids オペレーションは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。リストにある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。

API または SDK を使用して、Amazon Fraud Detector コンソールのリストから、APIAWS CLI またはAWS SDK を使用して、エントリを削除できます。

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、削除するエントリが含まれている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、[すべて削除] を選択します。
5. [すべて削除] ボックスに確認を入力しdelete all、[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しますAWS SDK for Python (Boto3)

次の例では、UpdateListAPIallow_email_ids オペレーションは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。可能性のある不正予測結果ごとに結果を作成できます。例えば、結果にリスクレベル (high_risk、medium_risk、およびlow_risk) やアクション (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します。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 コンソールでは、put-outcome コマンド、PutOutcomeAPI、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た結果の作成

1 つ以上の結果を作成するには、1 つ以上の結果を作成します。

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. 左のナビゲーションペインの [結果で [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、以下を入力します。
   a. [結果名] に、結果の名前を入力します。
   b. [結果の説明) [説明を入力します。
5. [結果の保存] を選択します。
6. 手順 2 ~ 5 を繰り返して、さらに結果を増やします。

AWS SDK for Python (Boto3) を使用した結果の作成

次の例では、PutOutcome API を使用して 3 つの結果を作成しています。それらはverify_customer、、review、approveです。結果を作成したら、それらを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 コンソールで結果を削除するには、delete-outcome コマンド、DeleteOutcomeAPI、またはAWS SDK for Python (Boto3)

Amazon Fraud Detector コンソールの結果を削除します。

結果を削除するには

1. AWS Management Console にサインインして、Amazon Fraud Detector コンソールを https://console.aws.amazon.com/frauddetector で開きます。
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. [Actions] (アクション) を選択してから、[Delete] (削除) をクリックします。
5. 結果名を入力してから、[結果の削除] を選択します。

を使用した結果を削除します。を使用した結果を削除します。AWS SDK for Python (Boto3)

次の例では、DeleteOutcomeAPIverify_customer を使用して結果を削除します。結果を削除すると、それをルールに割り当てることはできなくなります。

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

fraudDetector.delete_outcome(
name = 'verify_customer'
)
```

エンティティ

エンティティとは、イベントを実行する人またはモノのことを指します。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、お客様、マーチャント、ユーザー、アカウントなどがあります。イベントを実行した特定のエンティティを示すエンティティタイプ (ENTITY_TYPE) とエンティティ識別子 (ENTITY_ID) をイベントデータセットの一部として指定します。

Amazon Fraud Detector は、イベントの不正予測を生成する際にエンティティタイプを使用して、誰がイベントを実行したかを示します。詐欺予測に使用するエンティティタイプは、最初に Amazon Fraud Detector で作成し、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。

エンティティタイプの作成

エンティティタイプは、put-entity-typeコマンド、PutEntityTypeAPI、またはを使用して Amazon Fraud Detector コンソールで作成できますAWS SDK for Python (Boto3)。次の例では、Amazon Fraud Detector コンソールされ、SDK for Python (Boto3) を使用して、`customer`エンティティタイプになっています。不正検出モデルのトレーニング用に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、イベントデータセットのユースケースに適したエンティティタイプを使用してください。

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、左のナビゲーションで [エンティティ] を選択してから、[作成] をクリックします。
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。

AWS SDK for Python (Boto3) を使用したエンティティタイプの作成

AWS SDK for Python (Boto3)次のコード例では、`PutEntityType` API を使用してエンティティタイプを作成します`customer`。不正検出モデルのトレーニング用に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、イベントデータセットのエンティティからユースケースに適したものを使用してください。

```
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。最初に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、次に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 コンソールで、delete-entity-typeコマンド、DeleteEntityTypeAPI、または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. AWS Management Console にサインインして、Amazon Fraud Detector コンソールを https://console.aws.amazon.com/frauddetector で開きます。
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。

を使用したエンティティタイプを削除するAWS SDK for Python (Boto3)

AWS SDK for Python (Boto3)次のコード例では、DeleteEntityTypeAPI を使用してエンティティタイプ customer を削除します。

```
import boto3
```

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

fraudDetector.delete_entity_type (

name = 'customer'

)
```

AWS CloudFormation を使用した Amazon Fraud Detector リソースの管理

Amazon Fraud Detector は AWS CloudFormation と統合されています。これは、リソースとインフラストラクチャの作成と管理の所要時間を短縮できるように Amazon Fraud Detector リソースをモデル化して設定するためのサービスです。お客様が必要なすべての Amazon Fraud Detector リソース (Detector、Variables、、、、、、、、、 EntityType、 EventType、、、Label など) を説明するテンプレートを作成し、AWS CloudFormationがそれらのリソースを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 の使用に関して別途料金が発生することはありません CloudFormation。

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または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には、AWS CloudFormation スタックにプロビジョニングしたいリソースを記述します。JSONやYAMLに不慣れな方は、AWS CloudFormation Designerを使えば、 AWS CloudFormation テンプレートを使いこなすことができます。詳細については、AWS CloudFormation ユーザーガイドの「AWS CloudFormationDesigner とは」を参照してください。

Amazon Fraud Detector リソースは、AWS CloudFormation テンプレートを使用して作成、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。

を既に使用している場合は CloudFormation、追加の IAM CloudTrail ポリシーや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

Amazon Fraud Detector スタックを作成、 CloudFormation 更新、削除AWS。

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、 CloudFormation すべての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 CloudFormation つまたは両方を含めない場合、以下に示すデフォルト値が使用されます。

パラメータ	値	[Default Value] (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 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

パラメータ	値	[Default Value] (デフォルト値)
インライン	TRUE CloudFormation : 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。 FALSE: CloudFormation 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。	TRUE

Amazon Fraud Detector 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サンプル AWS CloudFormation YAML テンプレートです。

```
# 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'
```

AWS CloudFormation の詳細はこちら

AWS CloudFormation の詳細については、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

不正予測の取得

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。

リアルタイム予測の取得

`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。

AWSコマンドラインインターフェイス (AWSCLI) または Amazon Fraud Detector SDK の 1 つを使用して GetEventPredictionAPI オペレーションを呼び出すことにより、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。

必要に応じて、フィールドにデータを渡すことで、 SageMaker データを送信してモデルを呼び出すことができます`externalModelEndpointBlobs`。

AWS SDK for Python (Boto3) を使用した不正予測の取得

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`aws-fraud-detector-samples GitHub リクエストのその他の例はリポジトリにあります。

```
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'
}
)
```

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proof-of-concept、オフラインで実行したり、イベントのリスクを遡及的に評価したりできます。

バッチ予測ジョブは、Amazon Fraud Detector コンソールを使用するか、AWS Command Line Interface (AWSCLI) または Amazon Fraud Detector SDK の 1 つを使用して CreateBatchPredictionJobAPI 操作を呼び出すことで作成できます。

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
- AWS SDK for Python (Boto3) を使用したバッチ詐欺の予測の取得

バッチ予測の仕組み

-CreateBatchPredictionJobAPI 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. AWS Management Console にサインインし、https://console.aws.amazon.com/frauddetector で Amazon Fraud Detector 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権限と出力 S3 バケットへの書き込み権限が必要です。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. [Start] (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権限と出力 S3 バケットへの書き込み権限が必要です。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 入力および出力バケットです。
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。

AWS SDK for Python (Boto3) を使用したバッチ詐欺の予測の取得

次の例は、CreateBatchPredictionJob API のサンプルリクエストを示しています。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。

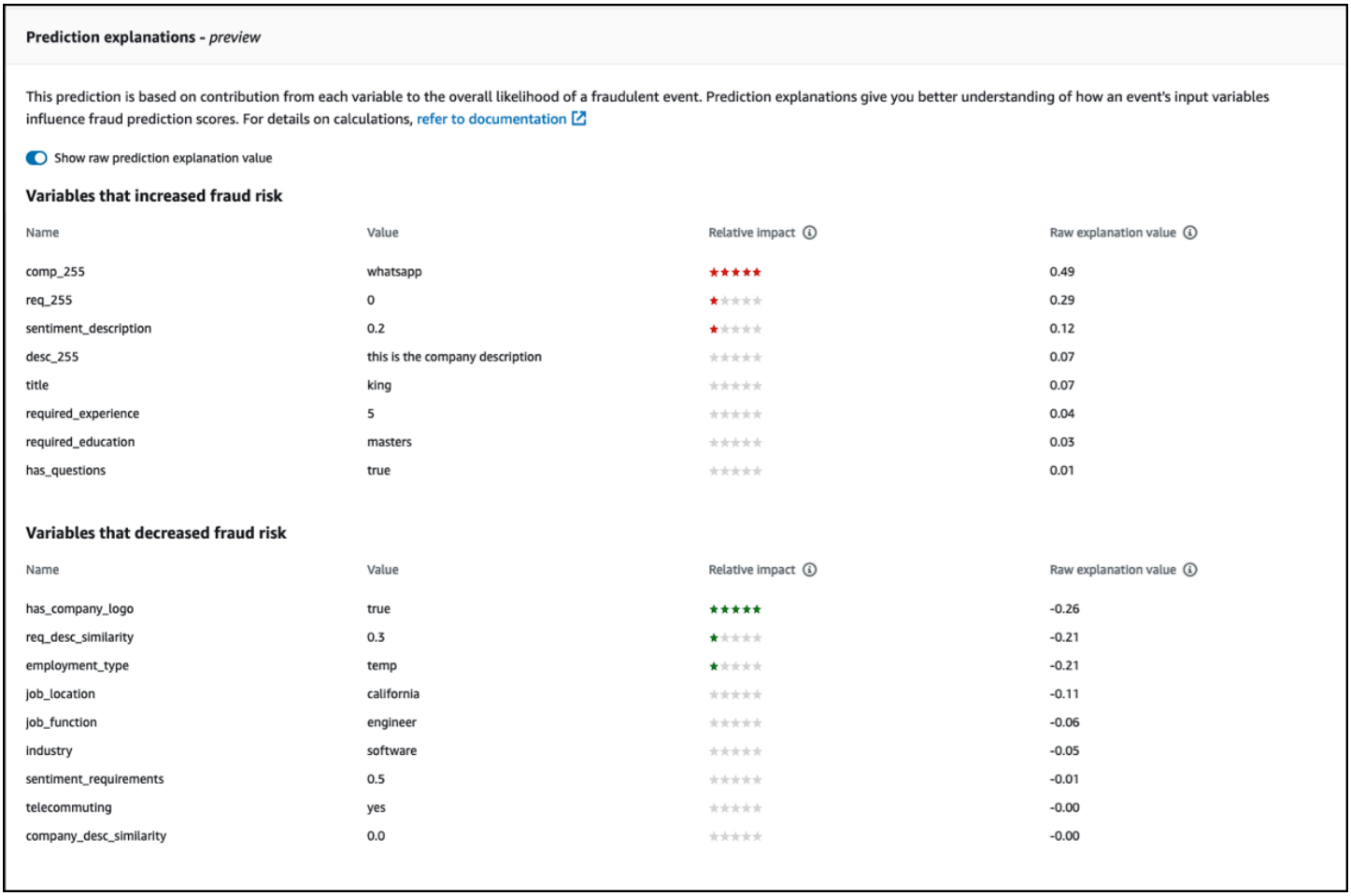
- 不正リスクが減少した変数は、緑色の星で示されます。緑の星の数が多いほど、変数が不正リスクスコアを下げて、不正の可能性が低くなります。
- すべての変数の星がゼロの場合、どの変数もそれ自体では不正リスクを大きく変えなかったことを示しています。

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 から APIs を使用して予測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、指定した期間に生成された、指定したイベント ID、イベントタイプ、ディテクター ID、および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 を呼び出して取得されます。

```
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

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 とユーザーの間で共有される責任です。責任共有モデルではこれを、クラウドのセキュリティ、およびクラウド内で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 — 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責任 AWS は AWS にあります。 AWS また、 は、安全に使用できるサービスも提供します。コンプライアンスAWS プログラムコンプライアンスプログラム の一環として、サードパーティーの監査者は定期的にセキュリティの有効性をテストおよび検証。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、お客様は、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や保護に役立つ他の AWS の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。お客様は、このインフラストラクチャでホストされてい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のよくある質問」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記事「AWS 責任共有モデルおよび GDPR」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 は必須であり TLS 1.3 がお勧めです。
- 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。
- 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 などの高度なマネージド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 140-2 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) 140-2」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は、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配置し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 名前に使用する自由記述のテキストフィールドやタグに入力したデータは、課金や診断ログに使用される場合があります。外部サーバーへの URL を提供する場合は、そのサーバーへのリクエストを検証するための認証情報を URL に含めないように強くお勧めします。

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、 AWS KMS 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。
- 単一リージョンの 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 AWS「KMS 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

Amazon Fraud Detector の PutKMSEncryptionKey API を使用して、カスタマーマネージド KMS キーを使用して保管中の Amazon Fraud Detector データを暗号化します。`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。

- イベント 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

モデルをトレーニングし、予測を生成するために提供した履歴イベントデータは、サービスを提供および維持する目的でのみ使用されます。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、お客様に対する当社のコミットメントを確実に順守します。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て、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。

> **Note**
>
> オプトアウトポリシーを使用するには、AWS アカウントを AWS Organizations によって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 AWS のサービス 使用できる 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しておくと、管理者に適切な許可をリクエストするうえで役立ちます。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 機能や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（ にサインイン 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 の認証情報は、フェデレーション ID の例です。フェデレーティッドアイデンティティ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、「 ユーザーガイド」の「 にサインインする方法 AWS アカウントAWS サインイン 」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 Software Development Kit (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に暗号で署名します。 AWS ツールを使用しない場合は、リクエストに自分で署名する必要があります。推奨される方法を使用してリクエストを自分で署名する方法の詳細については、IAM ユーザーガイドの API AWS リクエストの署名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。例えば、 AWS では、アカウントのセキュリティを強化するために多要素認証 (MFA) を使用する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「Multi-factor

authentication」(多要素認証) および『IAM ユーザーガイド』の「AWSにおける多要素認証 (MFA) の使用」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、アカウントのすべての AWS のサービス およびリソースへの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 内のアイデンティティです。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権限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、ロールでは一時的な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 (ロールではなく) の作成が適している場合」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 内のアイデンティティです。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。ロール を切り替える

AWS Management Console ことで、 で IAM ロールを一時的に引き受けることができます。ロールを引き受けるには、 または AWS API AWS CLI オペレーションを呼び出すか、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「IAM ロールの使用」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アイデンティティに権限を割り当てるには、ロールを作成してそのロールの権限を定義します。フェデレーティッドアイデンティティが認証されると、そのアイデンティティはロールに関連付けられ、ロールで定義されている権限が付与されます。フェデレーションの詳細については、『IAM ユーザーガイド』の「サードパーティーアイデンティティプロバイダー向けロールの作成」 を参照してください。IAM アイデンティティセンターを使用する場合、権限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。権限セットの詳細については、『AWS IAM Identity Center ユーザーガイド』の「権限セット」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) リソースにポリシーを直接アタッチできます。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 ロールとリソースベースのポリシーとの相違点」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権限、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 でアクションを実行する場合 AWS、ユーザーは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 へのリクエストのリクエスト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 のサービスに権限を委任するロールの作成」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、サービスにリンクされたロールの権限を表示できますが、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。 AWS ロールを EC2 インスタンスに割り当て、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 ロールを使用して権限を付与する」を参照してください。

IAM ロールと IAM ユーザーのどちらを使用するかについては、『IAM ユーザーガイド』の「(IAM ユーザーではなく) IAM ロールをいつ作成したら良いのか?」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、 AWS ID またはリソースにアタッチします。ポリシーは AWS 、ID 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 AWS JSON 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んなリソースにどんな条件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継ぐことができます。

IAM ポリシーは、オペレーションの実行方法を問わず、アクションの権限を定義します。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、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 権限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「IAM ポリシーの作成」を参照してください。

アイデンティティベースポリシーは、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「マネージドポリシーとインラインポリシーの比較」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権限を持つかをコントロールします。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、 AWS WAF、および Amazon VPC は、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、一般的ではない追加のポリシータイプをサポートします。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のポリシーによって 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。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) に対する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、 AWS アカウント ビジネスが所有する複数の をグループ化して一元管理するサービスです。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「SCP の仕組み」を参照してください。
- 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。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、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許可を持つ 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 AWS JSON 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んなリソースにどんな条件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権限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権限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。

単一ステートメントに複数アクションを指定するには、次のようにカンマで区切ります:

```
"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 AWS JSON 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どのような条件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、リソースレベルの権限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 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 AWS JSON 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んなリソースにどんな条件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することで、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 要素を指定するか、1 つの Condition 要素に複数のキーを指定すると、 AWS 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AWS 」の「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管理するには、`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 ロールを引き受ける、またはクロスアカウントロール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 などの AWS STS API オペレーションを呼び出しますGetFederationToken。

Amazon Fraud Detector は一時的な認証情報の使用をサポート

サービスリンクロール

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。

サービスロール

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ロールを引き受けることが許可されます。このロールにより、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

- ポリシーの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

ポリシーのベストプラクティス

ID ベースのポリシーは、ユーザーのアカウントで誰かが 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を付与するには、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して、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「AWS ジョブ機能の管理ポリシー」を参照してください。
- 最小特権を適用する – IAM ポリシーで権限を設定するときは、タスクの実行に必要な権限のみを付与します。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、最小特権権限とも呼ばれています。IAM を使用して権限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 でのポリシーと権限」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、 などの特定の を介してサービスアクションが使用される場合に 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 policy elements: Condition] (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 ポリシーの検証」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、セキュリティを強化するために MFA を有効にします。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 保護 API アクセスの設定」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、どのアクセス許可が必要かを調査する必要がなくなります。詳細については、「 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。 AWS Identity and Access Management

アカウントのユーザーにアタッチできる次の AWS マネージドポリシーは、Amazon Fraud Detector に固有です。

`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 のすべてのモデルエンドポイントを一覧表示して説明する SageMaker
- 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、 のユーザーに、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 フルアクセスを許可します。

```
{
    "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権を の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

リソースレベルのポリシーのこの例では、特定の 1 つの 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 には、リストオペレーションと説明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 が用意されています。パラメータなしで呼び出されると、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。リソースには、モデル、変数、イベントタイプ、またはエンティティタイプを使用できます。

デュアルモード APIsIAM ポリシーの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。例えば、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、変数名を指定しない場合、すべての変数が返され、情報漏えい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、Amazon Fraud Detector のすべてのリソースへのアクセスをユーザーに許可します。ただし、NotResourceポリシー要素は、プレフィックス 、user*、job_*および を持つ変数名のみに GetVariables API コールを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、レスポンスは以下の動作を示します。

- 変数名を含まない GetVariables 呼び出しは、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 エラーになります。
- 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。例えば、可変名を持つ 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と一致する変数名を含む GetVariables 呼び出しは、期待どおりに返されます。例えば、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 AWS アカウント 、Team1 と Team2 という Team2 つの異なるグループを定義しました。
- 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) にアカウントのリソースへのアクセスを提供する場合に、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 (呼び出したサービス) を呼び出すときに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、 aws:SourceArn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 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*:* です。アクセス許可ポリシーで使用できる Amazon Fraud Detector の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:*"
        }
      }
    }
  ]
}
```

次のロールの信頼ポリシーは、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 を実行する権限がありません。PassRole
- 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

から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*####*ター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* リソースへのアクセスが許可されるように、管理者にポリシーの更新を依頼します。

iam を実行する権限がありません。PassRole

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。Mary には、ロールをサービスに渡す権限がありません。

```
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) へのアクセス権限」を参照してください。

- クロスアカウントアクセスでのロールとリソースベースのポリシーの使用の違いの詳細については、『IAM ユーザーガイド』の「IAM ロールとリソースベースのポリシーとの相違点」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 で実行してい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。の詳細については CloudWatch、「Amazon ユーザーガイド CloudWatch 」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。の詳細については CloudTrail、「 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 AWS 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、コンプライアンスプログラムAWS のサービス による対象範囲内のコンプライアンスプログラムを参照し、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 「コンプライアンスプログラム」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、「 でのレポートのダウンロード AWS Artifact」の」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 を提供しています。

- セキュリティとコンプライアンスのクイックスタートガイド – これらのデプロイガイドでは、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に重点を置いたベースライン環境 AWS を にデプロイする手順について説明します。
- アマゾン ウェブ サービスにおける HIPAA セキュリティとコンプライアンスのアーキテクチャ — このホワイトペーパーでは、企業が AWS を使用して HIPAA 対象アプリケーションを作成する方法について説明します。

> **Note**
>
> すべて AWS のサービス HIPAA の対象となるわけではありません。詳細については、「HIPAA 対応サービスのリファレンス」を参照してください。

- 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。このガイドでは、ガイダンスを保護し AWS のサービス 、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にマッピングするためのベストプラクティスをまとめています。
- 「 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、リソース設定が社内プラクティス、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確認できます 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リストについては、「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 – これにより AWS アカウント、疑わしい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監視することで、、、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 は、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への対応に役立ちます。
- 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。

が AWS 公開している 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 は必須で TLS 1.3 がお勧めです。
- 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 で実行してい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 ユーザーガイド CloudWatch 」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 による Amazon Fraud Detector のモニタリング CloudWatch
- を使用した Amazon Fraud Detector API コールのログ記録 AWS CloudTrail

Amazon による Amazon Fraud Detector のモニタリング CloudWatch

を使用して Amazon Fraud Detector をモニタリングできます。これにより CloudWatch、raw データを収集し、読み取り可能なほぼリアルタイムのメトリクスに加工します。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 ユーザーガイド CloudWatch 」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。API または list-metrics コマンドを使用して CloudWatch ListMetrics 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のに役立つ名前と値のペアです。例えば、ディメンション名に`DetectorId`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、Amazon AWS Software Development Kit (SDKsまたは CloudWatch API ツールのいずれかを使用して使用することもできます。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。

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 CloudWatch。詳細については、「Amazon の認証とアクセスコントロール CloudWatch」を参照してください。

Amazon Fraud Detector メトリクスへのアクセス

次の手順は、 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. https://console.aws.amazon.com/cloudwatch/ で CloudWatch コンソールを開きます。
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

CloudWatch アラームの状態が変更されたときに Amazon Simple Notification Service (Amazon SNS) メッセージを送信するアラームを作成できます。指定した期間中、1 つのアラームが 1 つのメトリクスを監視します。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、持続する状態の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理由だけ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. By Detector ID を選択し、メトリクスを選択しますGetEventPrediction。
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 では、しきい値タイプに静的、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 E メールサブスクリプションリストの名前と E 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。リスト CloudWatch を保存して フィールドに表示し、将来のアラームの設定に使用できます。

> **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] をクリックします。
12. [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 に送信します CloudWatch。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。 有効なディメンション: `DetectorID`
`GetEventPredictionLatency`	GetEventPrediction リクエストからクライアントリクエストに応答するのにかかる時間間隔。

メトリクス	説明
	有効なディメンション: DetectorID 単位: ミリ秒
GetEventPrediction4XXError	Amazon Fraud Detector が 4xx HTTP レスポンスコードを返した 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。 有効なディメンション: DetectorID
GetEventPrediction5XXError	Amazon Fraud Detector が 5xx HTTP レスポンスコードを返した 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。 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。 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。 有効なディメンション: DetectorID 、DetectorVersionID 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。 有効なディメンション: DetectorID 、DetectorVersionID

メトリクス	説明
VariableUsed	評価の一部として変数が使用された GetEventPrediction リクエストの数。 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、評価中に変数のデフォルト値が使用された GetEventPrediction リクエストの数。 有効なディメンション: DetectorID 、DetectorVersionID 、VariableName
RuleNotEvaluated	以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。 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。 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。 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 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。 有効なディメンション: DetectorID 、DetectorVersionID 、OutcomeName
ModelInvocation (Amazon SageMaker model endpoint)	評価の一部として SageMaker モデルエンドポイントが呼び出された GetEventPrediction リクエストの数。 有効なディメンション: DetectorID 、DetectorVersionID 、ModelEndpoint
ModelInvocationError (Amazon SageMaker model endpoint)	呼び出した 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。 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。 有効なディメンション: DetectorID 、DetectorVersionID 、ModelEndpoint 単位: ミリ秒
ModelInvocation	評価の一部としてモデルが呼び出された GetEventPrediction リクエストの数。 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。 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。 有効なディメンション: DetectorID 、DetectorVersionID 、ModelType 、ModelID 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 AWS のサービスによって実行されたアクションを記録するサービスである と統合されています。は、Amazon Fraud Detector コンソールからの呼び出しや、Amazon Fraud Detector API へのコードからの呼び出しなど、Amazon Fraud Detector のすべての API 呼び出しをイベントとして CloudTrail キャプチャします。 APIs

証跡を作成する場合は、Amazon Fraud Detector の CloudTrail イベントなど、Amazon S3 バケットへのイベントの継続的な配信を有効にすることができます。 Amazon S3 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。によって収集された情報を使用して CloudTrail、Amazon Fraud Detector に対するリクエスト、リクエスト元の IP アドレス、リクエスト者、リクエスト日時などの詳細を確認できます。

の詳細については CloudTrail、「AWS CloudTrail ユーザーガイド」を参照してください。

の Amazon Fraud Detector 情報 CloudTrail

CloudTrail AWS アカウントを作成すると、 が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 CloudTrail イベント履歴 の他の

AWS サービスイベントとともに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、「イベント履歴を使用した CloudTrail 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。証跡により CloudTrail 、 はログファイルを 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 ログで収集されたイベントデータをより詳細に分析し、それに基づいて処理するように、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 サポートされているサービスと統合
- の Amazon SNS 通知の設定 CloudTrail
- 複数のリージョンからの CloudTrail ログファイルの受信と複数のアカウントからのログファイルの受信 CloudTrail

Amazon Fraud Detector 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することをサポートしています。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

証跡は、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できるようにする設定です。 CloudTrail ログファイルには 1 つ以上のログエントリが含まれます。イベントは任意のソース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が含まれます。 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。

次の例は、 GetDetectorsオペレーションを示す CloudTrail ログエントリを示しています。

```
{
    "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ている行は 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 50 を超えていることを確認するとともに、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 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 件の不正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、より長い期間をカバーすることで、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ている行は 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 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、より長い期間をカバーすることで、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、不正行為者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小さなエンティティグループによって実行された場合、モデルはうまく一般化されない可能性があります。

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。データセット全体に 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 件のイベント (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 を超えるイベントから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を考慮します。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

$IP_variable_name 値の 50% 以上が IPv4 または IPv6 アドレス https://digitalfortress.tech/tricks/top-15-commonly-used-regex/ の正規表現と一致しません。パフォーマンスを向上させるために、データセット中の **$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、アマゾン ウェブ サービスごとにデフォルトのクォータ (以前は制限と呼ばれていました) があります。特に明記されていない限り、クォータはリージョンごとに存在します。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、Amazon Fraud Detector クォータをコンポーネント別にまとめたものです。

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数	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
ディテクターごとのリスト数	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 1 秒あたりの API コール数	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 には、有用な情報を抽出するために使用できる新しい変数タイプとデータタイプが導入されています。	2023 年 6 月 5 日
イベントオーケストレーション	イベントオーケストレーションにより、Amazon AWS のサービス を使用してイベントを簡単に送信してダウンストリーム処理を行うことができます。EventBridge	2023 年 5 月 30 日
リスト	Lists リソースでは、IP アドレスやメールアドレスなどの値のセットをルールの一部として参照できます。ルール内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	データモデルエクスプローラーでは、Amazon Fraud Detector が不正検知モデルを作成するのに必要なデータ要素を詳しく調べることができます。イベントデータセットを準備する前に、データモデ	2022 年 12 月 15 日

	ルエクスプローラーを使用してください。	
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、または認証情報の盗難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
チャプター更新	Amazon 詐欺検出器に関する追加情報を追加して、入門の章を更新しました	2022 年 4 月 11 日
可変エンリッチメント	提供した未加工データの一部をエンリッチメントして、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てください。	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、イベントデータが Amazon Fraud Detector の開発または品質向上に使用されることを拒否してください。	2022 年 1 月 6 日
混乱を招く副予防	ポリシーを作成して、サードパーティまたはクロスサービスエンティティが権限を持つエンティティを操作して、そのエンティティに代わってアカウントのリソースにアクセスすることを防ぎます。	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	予測の説明を使用すると、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	トランザクション詐欺インサイト（TFI）モデルを使用して、card-not-presentオンライン詐欺や取引詐欺を検出します。	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、保存し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すると、モデルのパフォーマンスを向上または低下させている要因や、どのモデル変数が最も影響を与えているかを把握できます。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	Amazon AWS CloudFormation 詐欺検出器リソースの管理に使用します。	2021 年 5 月 10 日
バッチ予測	バッチ予測を使用すると、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。	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。