Sagemakerの特徴：

1. インターフェース SageMakerでは、Jupyter Notebookを使用して、「インスタンスの作成」、「モデル構築」、「トレーニング」、「デプロイ」までのフローを実行することができる。対話型のブラウザで実行することで、実行結果を分かりやすい形で監視することができる。
2. 主要な機械学習フレームワーク Tesnsorflowやsklearnなどの機械学習モデルをDockerコンテナ上で実行するように構成されている。
3. 一般的な機械学習アルゴリズムが事前にインストールされている。 (他の機械学習サービスと比較して、高いパフォーマンスでアルゴリズムを実行できる可能性が高い)
4. ワンクリックトレーニング SageMakerのコンソールからワンクリックで高度なトレーニング他チューニングを実行できるようになっている。
5. フルマネージメントスケーリング機能 インフラ部分については自動的に管理されてスケーリング(規模の増減、面積の拡大縮小）されるので、ペタバイト規模のモデルトレーニングが簡単にスケーリングできる。

基本的な準備:

1. S3バケットの作成 名前の中で「sagemaker」の文字列を含める必要がある。
2. Amazon SageMakerのトップページから、ノートブックを作成する。 ノートブック作成の際にインスタンスを指定する。
3. IAMロールの作成

SageMakerのセットアップ

まずはコンソールログインする。

ノートブックインスタンスを選択

AWSサービスの概要

データ分析とデータ習得

AWSでデータ管理に使うことのできるサービス：

- Amazon SageMaker Ground Truth
    - データのラベル付けを簡単に行うことができるデータラベリングサービス
    - Amazon SageMaker Ground Truth では手動ラベリングと自動ラベリングを選択することができる。
        - 手動ラベリング：
          3 種類のチームを選択することができる。

          1. 社員などのプライベートのチーム
          2. Amazon Machanical Turk を使用したパブリックチーム
          3. ベンダーなどのサードパーティーのチーム

          チームのメンバーは Amazon SageMaker Ground Truth が用意した UI を通してラベリング作業を行う。
    - 自動ラベリング：
      手動ラベリングに追加して有効化でき、ビルトインアルゴリズムを利用した機械学習のモデル学習を行う。
    - ラベリングされたデータは安価で高耐久なオブジェクトストレージである Amazon S3 のバケットに格納されます。
- Amazon Athena
    - インタラクティブなクエリサービス. Amazon S3 内のデータを標準 SQL を使用してクエリを取得できる。
    - サーバーレスでインフラストラクチャの管理かつ実行したクエリに対してのみ料金が発生。
- Amazon Redshift

    - データウェアハウスサービスです。データウェアハウス（DWH）というのは、さまざまなデータ源からデータを収集・統合・蓄積し、分析のため保管しておくシステムのこと。伝統的なRDBMSとは違って、継続的な書き込みや更新には向いておらず、一括でデータを書き込み分析のため大容量データを読み出すという処理に最適化されている。その結果として、たとえばRDB設計における正規化はデータウェアハウスでは重視されず、読み出しの高速化のためにあえて正規化しないでデータを格納することもある。Amazon Redshiftでは、並列コンピューティングをサポートしており、大量のデータを短時間で読み出し・分析することが可能です。インターフェイスとしては、BIツールやPostgreSQLクライアントから操作することが可能。

前処理

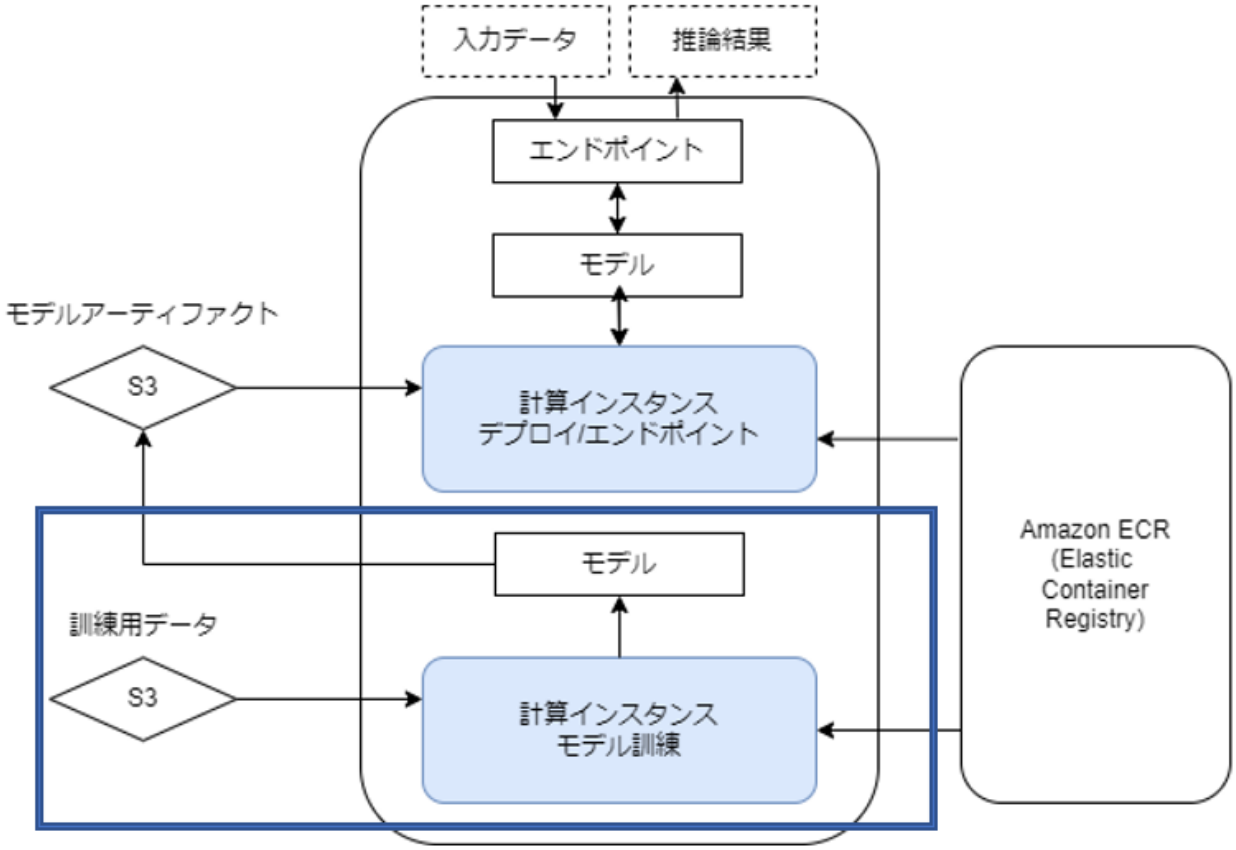
- Amazon SageMaker Processing Amazon SageMaker SDK を通して、コンテナイメージ、前処理のコード、入力と出力の Amazon S3 バケットを指定することで、自動でインフラの構築と処理を実行する。Amazon EC2 インスタンスの起動や停止、ライブラリのインストールなどを考える必要がない。
- AWS Batch Amazon SageMakerで一気通貫に行う必要がない場合に利用できる。
- AWS Glue より複雑なデータの変換などをしたい場合、Apache Sparkで大規模に処理したい場合などに利用すると良い。

モデルの構築・学習

- SageMaker
    - Sagemakerでは組み込みのアルゴリズムに加えて以下のようなフレームワークを利用できるコンテナを提供している。
        - TensorFlow
        - Pytorch
        - Apache MXNet
        - XGBoost
        - scikit-learn

機械学習モデルの作成

Sagemakerによる機械学習モデルの作成



学習時と推論時の環境の一致について

Sagemakerでは、基本的に学習用と推論用のコンテナを分けて運用することが前提となっており、それぞれについてコンテナイメージを用意する必要がある。また、SageMakerでは組み込みのコンテナを用いてモデルを学習させる、もしくは独自のモデルを含むコンテナを用意する必要があり、その際には学習用と推論用のコンテナイメージに含まれる環境をできる限り統一する必要がある。

推論器の稼働

SageMakerを利用することで、推論用の環境を独自で用意する必要はなくなる。基本的はモデルのデプロイ時に推論用のエンドポイントを作成して、そのエンドポイントを通じて推論用のサーバーに入力値を送信する。

- 大量のモデルを管理したい場合 SageMakerでは複数のモデルについていつでも本番環境で利用できるようにマルチモデルエンドポイントを利用することができる。また必要に応じて、1 つのエンドポイントに複数のトレーニング済みモデルをデプロイし、単一のサービングコンテナを使用して稼働させることでコストの削減を行うことができる。エンドポイントを呼び出す際には、ターゲットのモデル名を指定することで、特定のモデルに簡単にアクセスできる。

- 新しくモデルを追加する場合
  S3 ではモデルのどの階層でもエンドポイントを定義でき、マルチモデルエンドポイントにモデルを追加するには、新しいトレーニング済モデルのアーティファクトを S3 に追加して呼び出せば良い。

- すでに使用中のモデルを更新する場合
  S3 に新しい名前のモデルを追加して、新しいモデル名でエンドポイントの呼び出しを開始する。マルチモデルエンドポイントでデプロイされたモデルの使用を中止するには、モデルの呼び出しを停止し、S3 からモデルを削除すればよい。

- 具体的なユースケース

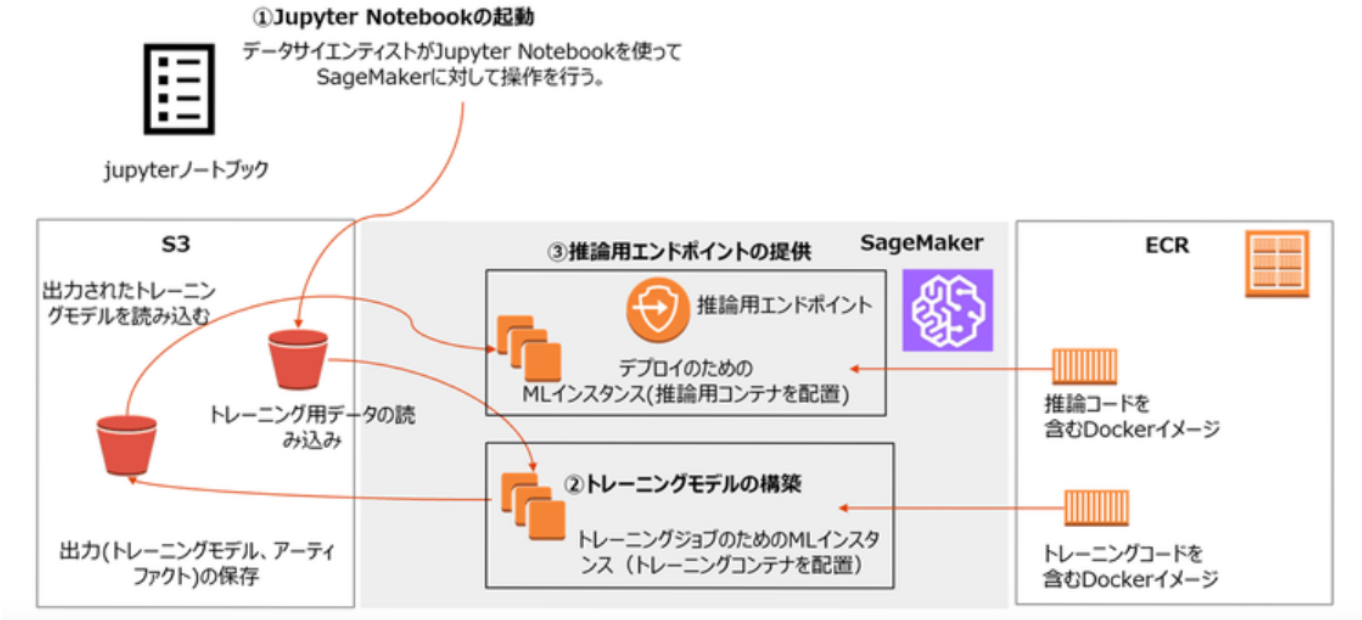
    - 法律関係のアプリケーション
      広範な規定上の管轄を完全にカバーする必要がありますが、それにはめったに使用されないモデルが多数含まれることになります。1 つのマルチモデルエンドポイントでは、使用頻度の低いモデルに対応し、コストの最適化と大量のモデルの管理を効率よく行うことができます。

また、SageMakerでは訓練用データ、出力先ファイルの他にモデルの学習に必要な環境変数（インスタンスの数など）を引き渡すことで、以下の訓練ジョブのフローを容易に行うことができる。

- 訓練ジョブの実行フロー
    1. S3からの入力データの読み込み
    2. データを利用したモデルの訓練
    3. モデルをS3に書き戻す

モデルロードパターン

SageMakerのシステムは「AIエンジニアのための機械学習システムデザインパターン」の3章で紹介されているモデルロードパターンに近い。



基本的にSageMakerはJupyter Notebook、もしくはJupyter Labを使用して操作を行うが、学習時、推論時にそれぞれ実行用のインスタンスを用意して、そこにモデルのDocker イメージをダウンロードする形式を取る。Dockerのイメージは、ダウンロード先の環境で動くイメージを予めECRに登録しておき、インスタンス起動時に使用できるようにしておく。

推論システムの作成

推論システムの運用

オプション	ユースケース
リアルタイム推論	トラフィックが多く、低レイテンシーが求められ、そこまでペイロードが大きくない場合。
非同期推論	ペイロードが大きく、処理時間も長い場合でコールドスタートが許容できる場合。
バッチ変換ジョブ	バッチで推論を実行。
Serverless Inference (preview)	トラフィックが断続的で予測できないが低レイテンシーが求められる場合。

- 運用について
  基本的に必要に応じてライブラリや学習済みモデルを外部からインストールするが、外部への依存度を考えると、これはセキュリティの観点や可用性あまり推奨されない。

  基本的には、これらのリソースをDocker イメージやS3バケットにコピーする形で利用する方が良い。

SageMakerの機能紹介（STEP1: ）

1の参考書についてまとめたのは以下の通り。

4.2 訓練データの収集と準備

4.2.1 SageMaker Data Wrangler

SageMaker Data Wranglerの基本的な機能：

- データ準備と機能エンジニアリングのプロセスを簡素化
- データ選択とクレンジング
- 探索
- 可視

上記のようなデータ準備の各ステップを単一のインターフェイスから実行できる。

SageMaker Data Wranglerを使用するメリット

- 300 を超える組み込みのデータ変換が含まれているため、コードを記述しなくてもデータをすばやく正規化、変換、結合できる。
- 視覚化テンプレートを完全統合開発環境 (IDE) である Amazon SageMaker Studio で使用でき、データの変換が意図した通りに完了したことを確認できる。

- Amazon SageMaker Pipelines を使用してワークフローを完全に自動化し、Amazon SageMakerで再利用できるように保存できる。SageMaker Pipelinesの詳細については後述するが、この機能を利用することで、他に人にワークフローを共有したり、本番環境への意向をスムーズに完結できる。またこのワークフローはノートブックもしくはコードスクリプトにワンクリックでエクスポートして本番環境に移行できる。
- Amazon S3、Amazon Athena、Amazon Redshift、AWS Lake Formation、Amazon SageMaker Feature Store などの複数のデータソースからデータを数回のクリックだけでデータを選択してクエリできる。また、データソースのクエリを記述し、CSV ファイル、Parquet ファイル、データベーステーブルなどのさまざまなファイル形式から SageMakerにデータを直接インポートすることもできる。

4.2.2 SageMaker Feature Store

Amazon SageMaker Feature Storeは機械学習 (ML) 特徴量を保存、更新、取得、共有するためのフルマネージド型のリポジトリサービス。

基本的な機能：

- チーム間で特定の特徴量に名前を付けたり、整理したり、再利用したりする作業を簡単に行うことができる。
- トレーニング中、リアルタイム推論を問わず、統合された特徴量の管理を行うことができ、この際に追加のコードを記述したり、機能の一貫性を保つために手動プロセスを作成したりする必要がない。
- 保存されている特徴量の情報 (特徴量名やバージョン番号など) を追跡するため、Amazon Athenaを使用してバッチまたはリアルタイムでクエリできます。
- トレーニングや推論中でも新しい特徴量が常に利用可能です。

使用するメリット：

- 多様なリソースからのデータ取り込み
    - Amazon SageMaker Data Wranglerなどのデータ準備ツールで機能を作成し、数回クリックするだけでSageMaker Feature Storeに直接保存することもできます。
- 検索と検出
    - SageMaker Studioのインターフェイスから簡単に特徴量の概要を確認でき特徴量が特定のモデルに役立つかどうかを素早く判断できる。
- 特徴量の一貫性
    - 学習と推論の両方で同じ特徴量を利用できるようにサポートされており、ストレージ要件がそれぞれ異なる場合においても対応できる。
- 特徴量を標準化する
    - 特徴量の定義を単一のリポジトリに保存することでチーム間の混乱を解消し、各特徴量がどのように定義されているかを明確にする。機能を明確に定義すると、さまざまなアプリケーションで機能を再利用しやすくなります。
- Amazon SageMaker Pipelinesとの統合
    - Amazon SageMaker Pipelinesと統合して自動化された機械学習ワークフローを作成できる。特徴量の検索、検出を行えるので、特徴量の再利用が簡単に実行できる。

モデルの訓練とチューニング

SageMaker Training

SageMaker Processing

参考：https://tech.connehito.com/entry/2021/12/01/164447#SageMaker-Processing%E3%81%A8%E3%81%AF

機械学習の前処理、後処理、モデル評価といったワークフローをSageMaker上で簡単に行うためのPython SDK。独自のコンテナ上で任意のpythonスクリプトを実行でき、処理が終了するとインスタンスが自動的に停止するというシンプルな機能を持ち、機械学習関連のデータ処理だけでなく、学習や推論にも使うことができる。

このPython SDKには主に4つのクラスが用意されていて、それぞれで使用方法が異なります。

- sagemaker.sklearn.processing.SKLearnProcessor
- sagemaker.processing.PySparkProcessor
- sagemaker.processing.Processor
- sagemaker.processing.ScriptProcessor

Batch Transform

Autopilot

Hyper-Parameter Tuning

SageMaker Debugger

SageMaker Experiment

モデルの評価と品質認定

SageMaker Processing

機能紹介（STEP2:推論システムの作成）

本番環境にモデルをデプロイ

SageMaker Hosting

SageMaker EndPoint

SageMaker Interface Pipeline

機能紹介（STEP3:推論システムの運用）

本番環境でモデルを監視

SageMaker Model Monitor

参考文献

https://www.acrovision.jp/service/aws/?p=1237

独自のコンテナを利用する方法

https://corp.logly.co.jp/blog/417

https://www.inoue-kobo.com/aws/sagemaker-with-mycontainer/index.html

Redshiftについての解説

https://techblog.nhn-techorus.com/archives/8232

SageMakerについての公式説明：

マルチエンドポイントについて https://aws.amazon.com/jp/blogs/news/save-on-inference-costs-by-using-amazon-sagemaker-multi-model-endpoints/

エンドポイントを使ったリリースについて https://aws.amazon.com/jp/blogs/news/load-test-and-optimize-an-amazon-sagemaker-endpoint-using-automatic-scaling/

SageMakerの仕組み

https://www.accenture.com/jp-ja/blogs/cloud-diaries/amazon-sage-maker

参考書の作成