BETTER TOGETHER WITH intel

aws DEV DAY

Tokyo | Sep 29, 2021

F - 4

Amazon Elastic Container Service (Amazon ECS) Anywhere と機械学習 ハイブリッド環境でのモデルの構築と推論

林　政利
Solutions Architect, AWS Japan

呉　和仁
Solutions Architect, AWS Japan

自己紹介

林　政利
Specialist Solutions Architect, Containers

好きなサービス
Amazon Elastic Kubernetes Service (Amazon EKS)
AWS Certificate Manager

SIer

フリーランス
受託開発

Web系企業
サービス開発・運用

外資ベンダー
コンテナ担当SA・
サポートエンジニア

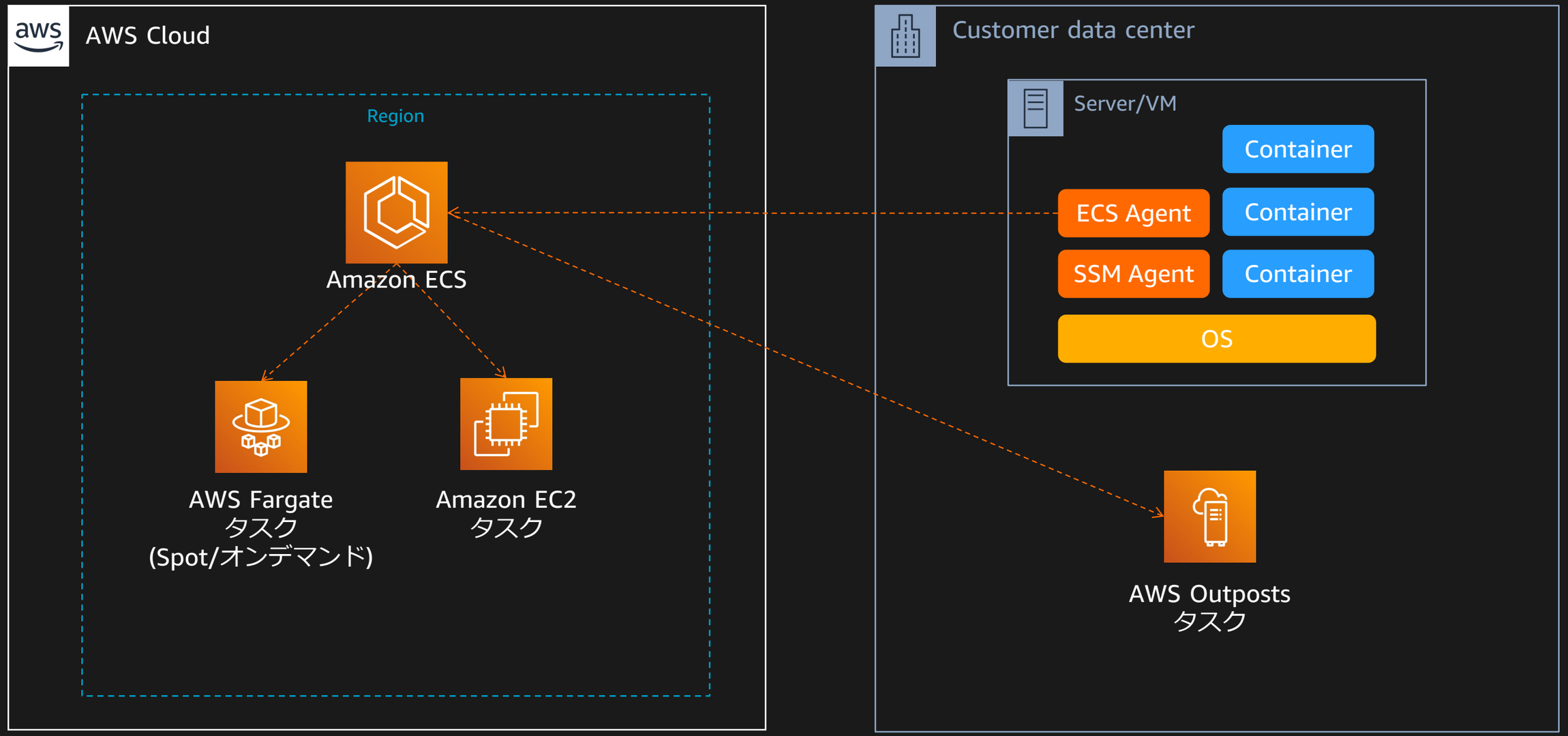
AWS Japan
コンテナ担当SA

自己紹介

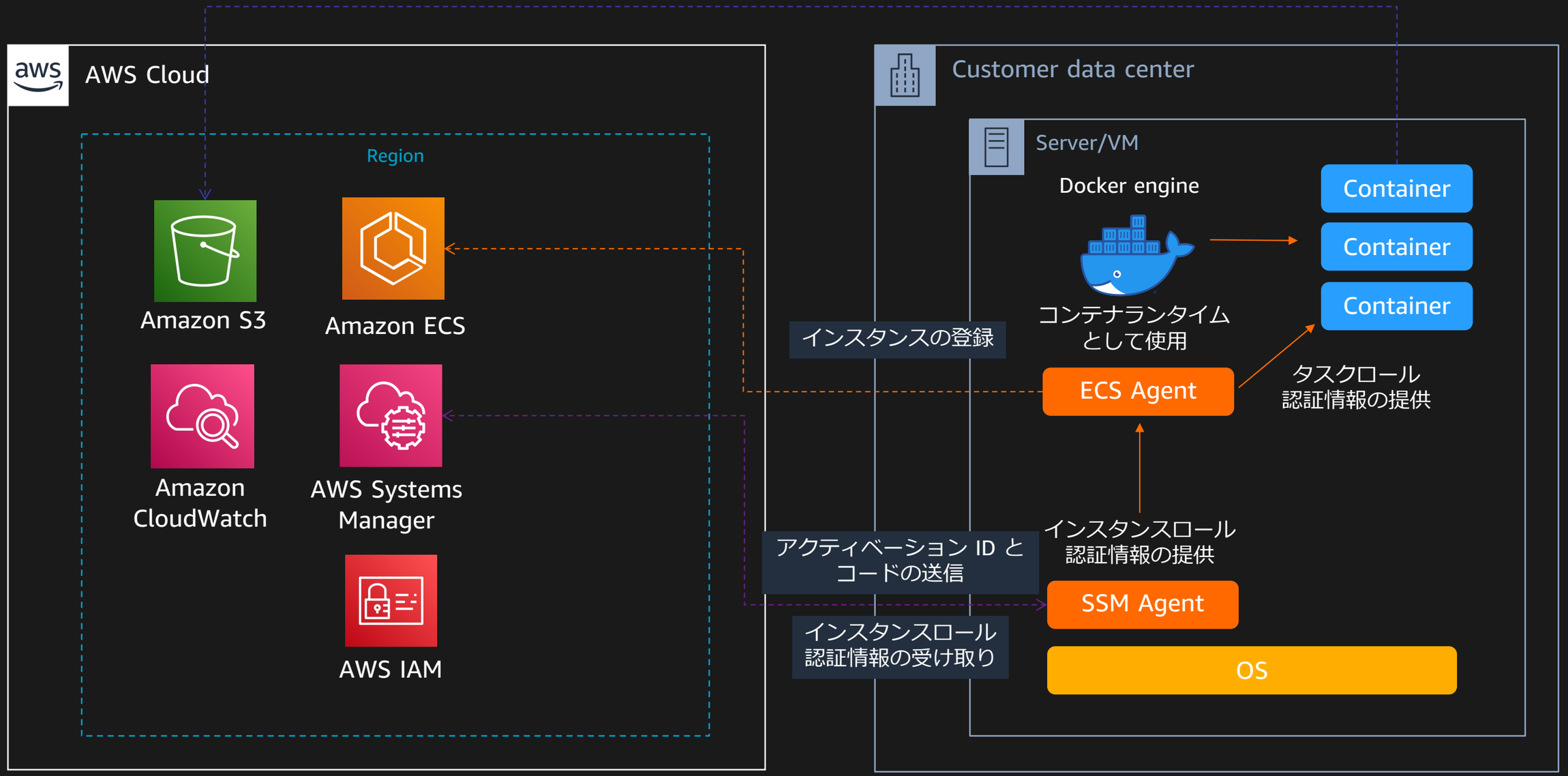
- 呉 和仁(ごう かずひと)
- 機械学習ソリューションアーキテクト
- 好きな AWS サービス
  - AWS の AI サービス全般
  - AWS Lambda
  - Amazon SageMaker
- Twitter
  - @kazuneet

Amazon ECS Anywhere

Amazon ECS と管理対象の Data Plane

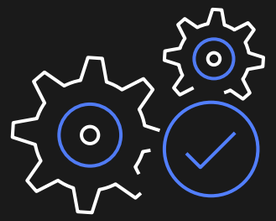


ECS Anywhere の仕組み - Deep Dive



デモ
外部インスタンス登録

ユースケース

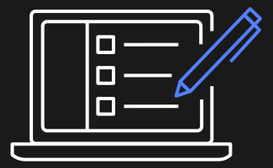


一貫性のあるオペレーション

オンプレミスとクラウドのそれぞれでアプリケーションを運用する際にオペレーションモデルを分けたくない

設備投資の回収

クラウドへ移行する前に償却する必要があるデータセンターに設備投資を行なっている



コンプライアンス要件

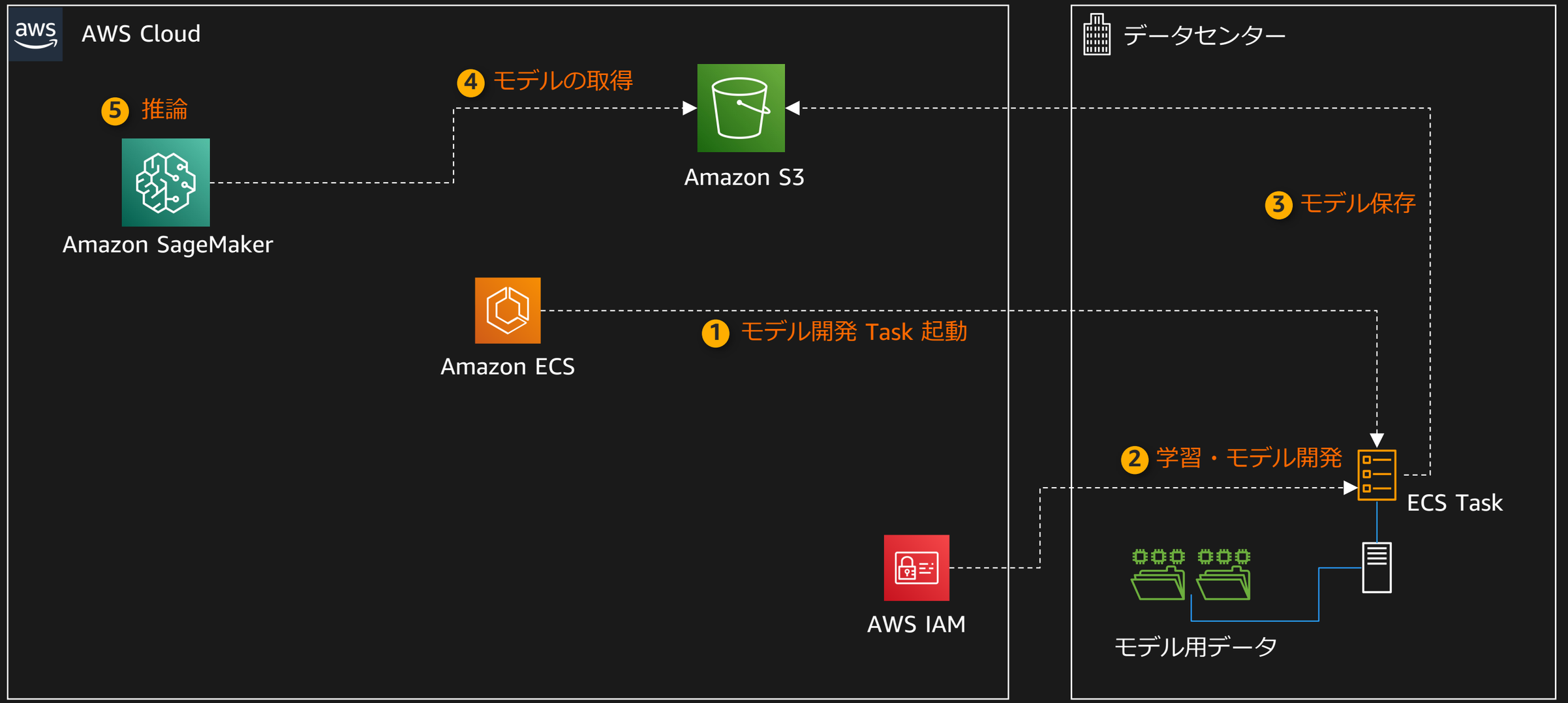
市場や業界の規制により、インフラストラクチャの大部分を自身で所有する必要がある

データに近接したワークロード

大規模なデータセットを低レイテンシで処理するために、データの近くにアプリケーションをデプロイする必要がある

ユースケースのデモ

機械学習のモデルをオンプレミスで構築し、推論はクラウドで実行する



ゆけ！Amazon ElastiCache 警察
～第一部コンテナを学習するの巻～

宣伝

記事リンクに飛びます

https://aws.amazon.com/jp/builders-flash/202109/elasticache-police/

- 誤ったサービス名の画像を機械学習で検出するワークロードを
  コンテナで動かしてみる記事を今月パブリッシュ

- この記事に書いてあるコードを題材に本セッションのデモを行う

正しい綴り

誤った綴り

デモ
オンプレミスでMLのモデルを構築

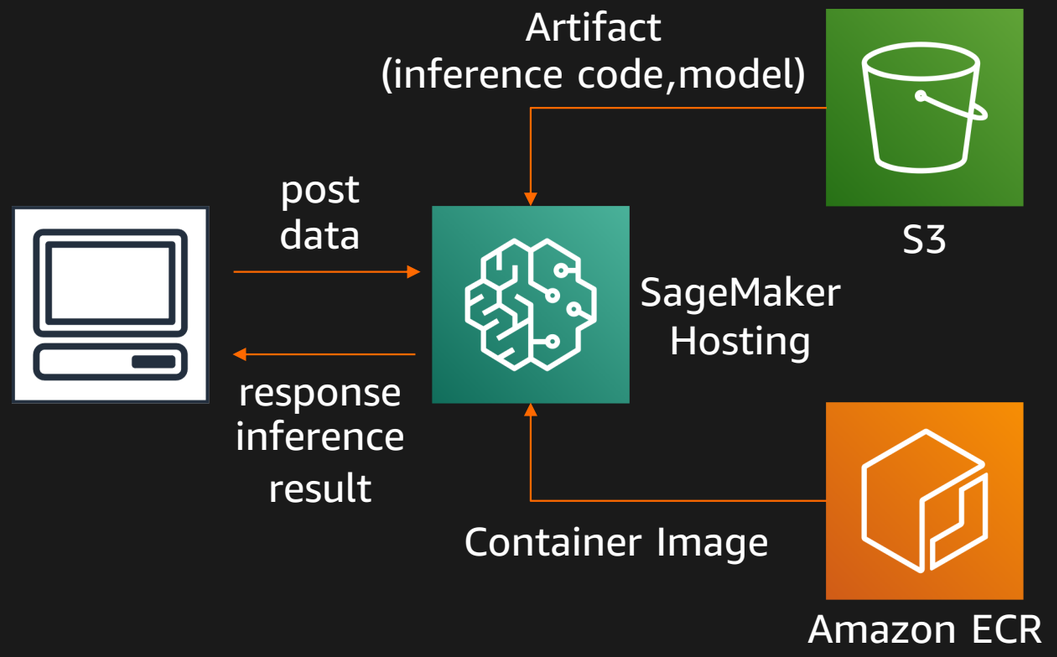
デモ
Amazon SageMaker で推論

Amazon SageMaker における推論方法

- リアルタイム推論
  - データを送ると推論結果を返す
  - 推論リクエストにデータを入れて推論し、推論結果はリクエストの中に含まれる
  - エンドポイントを常時立ち上げておくことで即座に推論処理を走らせられる
- 非同期推論
  - 推論対象データを予め S3 に配置しておき、推論リクエストは S3 の URI を指定し、レスポンスは推論結果を格納する S3 URI をレスポンスとしてすぐに受け取る
  - 推論自体は非同期で実行され、推論が完了したらレスポンスで指定された S3 URI に格納
  - オートスケールの機能を利用してインスタンス数を最小 0 台にすることで、コストを抑えられる（反面、推論リクエストを投げる際起動のオーバヘッドがかかる）
- バッチ変換
  - 推論対象データ群を予め S3 に配置しておき、まとめて推論する
  - コスト効率に優れる

リアルタイム推論

- 推論データをリクエストすると推論結果をレスポンスで受け取る
- ワークロードに応じてインスタンスタイプを設定
  - モデルがコンパイル可能な場合は
    高速かつ低コスト化できる場合がある
    Amazon SageMaker Neo, Inf1 インスタンス
- 負荷に応じてロードバランスや
  オートスケーリングが可能
- 推論エンドポイントを立ち上げている
  時間に対して課金



リアルタイム推論を実行するための準備
(TensorFlow で AWS マネジメントコンテナを使用する場合)

1. モデルを tar.gz で固めて S3 に配置（済）

2. SageMaker の CreateModel API で、SageMaker にモデルを登録
   モデルと使用するコンテナを指定

3. SageMaker の CreateEndpointConfig API で 推論エンドポイントの設定
   モデルとインスタンスタイプ、インスタンス数を指定

4. SageMaker の CreateEndpoint API で推論エンドポイントを作成

5. エンドポイントに推論データをリクエストして推論

リアルタイム推論を実行するための準備
(TensorFlow で AWS マネジメントコンテナを使用する場合)

1. モデルを tar.gz で固めて S3 に配置（済）

2. SageMaker の CreateModel API で、SageMaker にモデルを登録
   モデルと使用するコンテナを指定

3. SageMaker の CreateEndpointConfig API で 推論エンドポイントの設定
   モデルとインスタンスタイプ、インスタンス数を指定

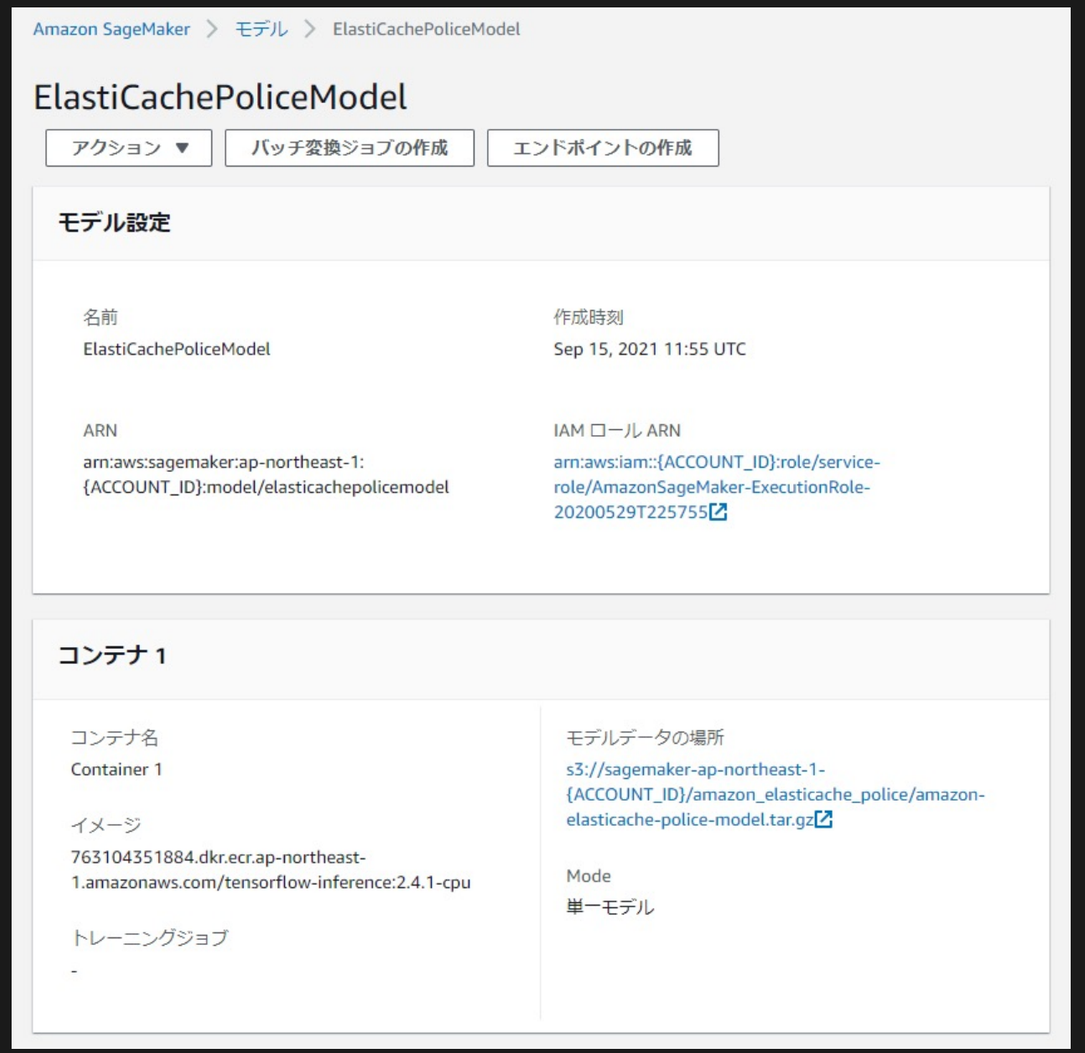
4. SageMaker の CreateEndpoint API で推論エンドポイントを作成

5. エンドポイントに推論データをリクエストして推論

CreateModel API

学習済のモデルと推論に使用するコンテナをひとまとめにして SageMaker に登録する

```
MODEL_NAME = 'ElastiCachePoliceModel'
response = sm_client.create_model(
    ModelName=MODEL_NAME, # モデルの任意の名前
    PrimaryContainer={
        'Image': CONTAINER_URI, # コンテナの URI
        'ModelDataUrl': MODEL_S3_URI, # モデルの URI
    },
    ExecutionRoleArn=SM_ROLE_ARN, # ロールの ARN
)
print(response)
```



リアルタイム推論を実行するための準備
(TensorFlow で AWS マネジメントコンテナを使用する場合)

1. モデルを tar.gz で固めて S3 に配置（済）

2. SageMaker の CreateModel API で、SageMaker にモデルを登録
   モデルと使用するコンテナを指定

3. SageMaker の CreateEndpointConfig API で 推論エンドポイントの設定
   モデルとインスタンスタイプ、インスタンス数を指定

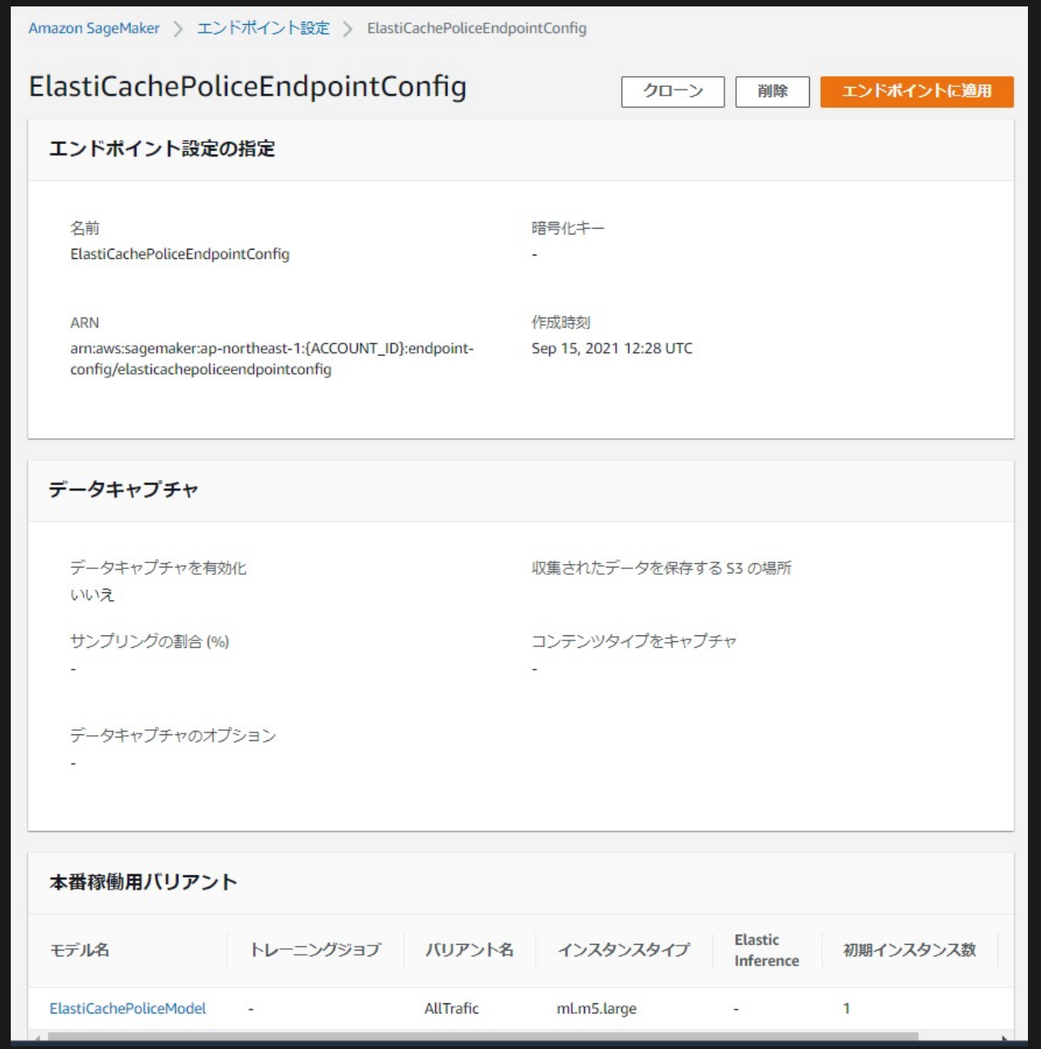
4. SageMaker の CreateEndpoint API で推論エンドポイントを作成

5. エンドポイントに推論データをリクエストして推論

CreateEndpointConfig API

開発済のモデルと推論に使用するコンテナをひとまとめにして SageMaker に登録する

```
ENDPOINT_CONFIG_NAME = 'ElastiCachePoliceEndpointConfig'
response = sm_client.create_endpoint_config(
    EndpointConfigName=ENDPOINT_CONFIG_NAME,
    ProductionVariants=[
        {
            'VariantName': 'AllTrafic',
            'ModelName': MODEL_NAME,
            'InitialInstanceCount': 1,
            'InstanceType': 'ml.m5.large',
        },
    ],
)
print(response)
```



リアルタイム推論を実行するための準備
(TensorFlow で AWS マネジメントコンテナを使用する場合)

1. モデルを tar.gz で固めて S3 に配置（済）

2. SageMaker の CreateModel API で、SageMaker にモデルを登録
   モデルと使用するコンテナを指定

3. SageMaker の CreateEndpointConfig API で 推論エンドポイントの設定
   モデルとインスタンスタイプ、インスタンス数を指定

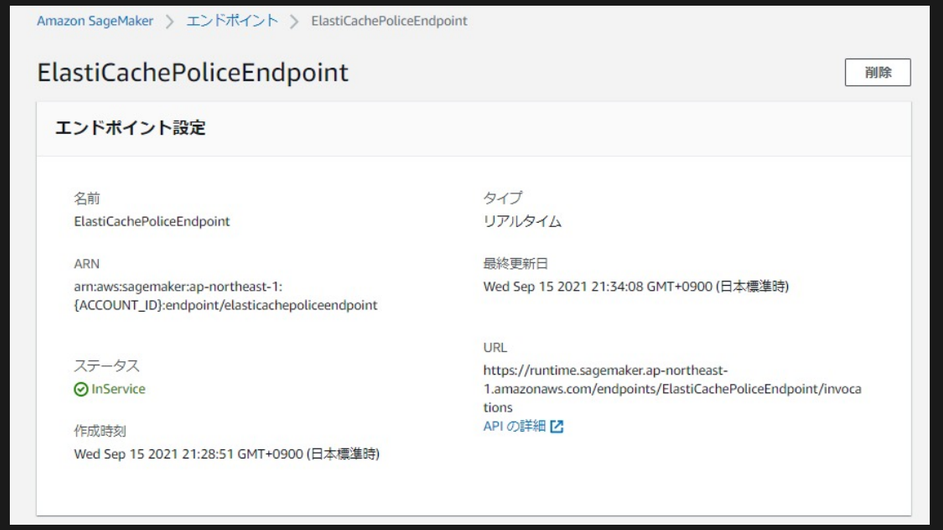
4. SageMaker の CreateEndpoint API で推論エンドポイントを作成

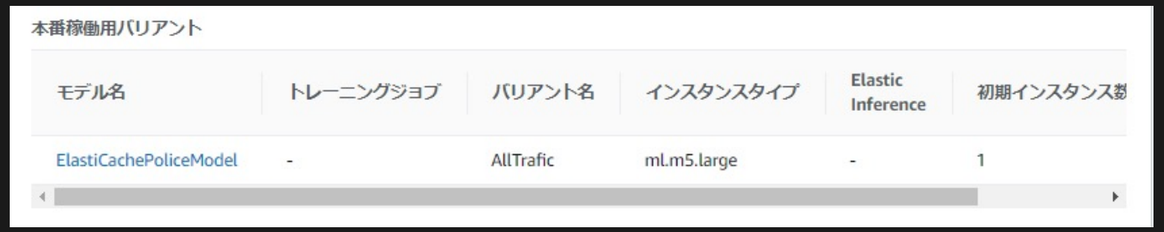
5. エンドポイントに推論データをリクエストして推論

CreateEndpoint API

作成済の endpoint config を使用して推論 endpoint を立ち上げる

```
ENDPOINT_NAME='ElastiCachePoliceEndpoint'
response = sm_client.create_endpoint(
    EndpointName=ENDPOINT_NAME,
    EndpointConfigName=ENDPOINT_CONFIG_NAME,
)
print(response)
```





リアルタイム推論を実行するための準備
(TensorFlow で AWS マネジメントコンテナを使用する場合)

1. モデルを tar.gz で固めて S3 に配置（済）

2. SageMaker の CreateModel API で、SageMaker にモデルを登録
   モデルと使用するコンテナを指定

3. SageMaker の CreateEndpointConfig API で 推論エンドポイントの設定
   モデルとインスタンスタイプ、インスタンス数を指定

4. SageMaker の CreateEndpoint API で推論エンドポイントを作成

5. エンドポイントに推論データをリクエストして推論

エンドポイントにデータを post して推論

```
# 画像ファイルを開く
img = Image.open('./test_data/AmazonElastiCache_ZenjidoJP-FeltPenLMT-TTF.ttf0.png')
# ndarray に変換
no_problem_arr=((np.array(img)-127.5)/127.5).reshape(1,50,700,1)
# request のデータを作成
request_args = {
    'EndpointName': ENDPOINT_NAME,
    'ContentType' : 'application/json',
    'Accept' : 'application/json',
    'Body' : str(no_problem_arr.tolist())
}
# 推論エンドポイントにリクエスト
response = smr_client.invoke_endpoint(**request_args)
print(response['Body'].read().decode('utf-8'))
```

```
{    "predictions": [[0.404227316]    ]}
```

まとめ

- コンテナを利用しておけばオンプレでもクラウドでも
機械学習ワークロードを簡単に再現可能

- 機械学習に携わるものは自分の書いたコードを動かす環境の
Dockerfile を書くところまでの責任を負いたい

- AWS のマネジメントコンテナイメージも便利なので
積極的に利用の検討を

【再宣伝】
本セッションの
元となった記事リンク

Thank you!

林　政利　　呉　和仁