AWS re:Invent

AIM404

Build, Train, and Deploy ML Models with Amazon SageMaker

David Arpin
Data Science Practice Manager
AWS, Professional Services

Featuring: Prasad Prabhu
Principal Architect
Intuit, Data Platform

Agenda

Review Amazon SageMaker

- Build, train, and deploy
- Algorithms, frameworks, bring your own, and automatic model tuning

Realtime deployment at scale

- Creating and updating endpoints
- Reduced risk deployments
- Automatic scaling

Customer story

- ML at Intuit
- Data science workflows
- Architecture and demo

The Amazon Machine Learning Stack

AI SERVICES

Vision

AMAZON REKOGNITION IMAGE

AMAZON REKOGNITION VIDEO

Speech

AMAZON POLLY

AMAZON TRANSCRIBE

Language

AMAZON TRANSLATE

AMAZON COMPREHEND

Chatbots & Contact Centers

AMAZON LEX

ML SERVICES

AMAZON SAGEMAKER

ML FRAMEWORKS & INFRASTRUCTURE

Frameworks

TensorFlow

mxnet

PYTORCH

Chainer

HOROVOD

Interfaces

GLUON

K Keras

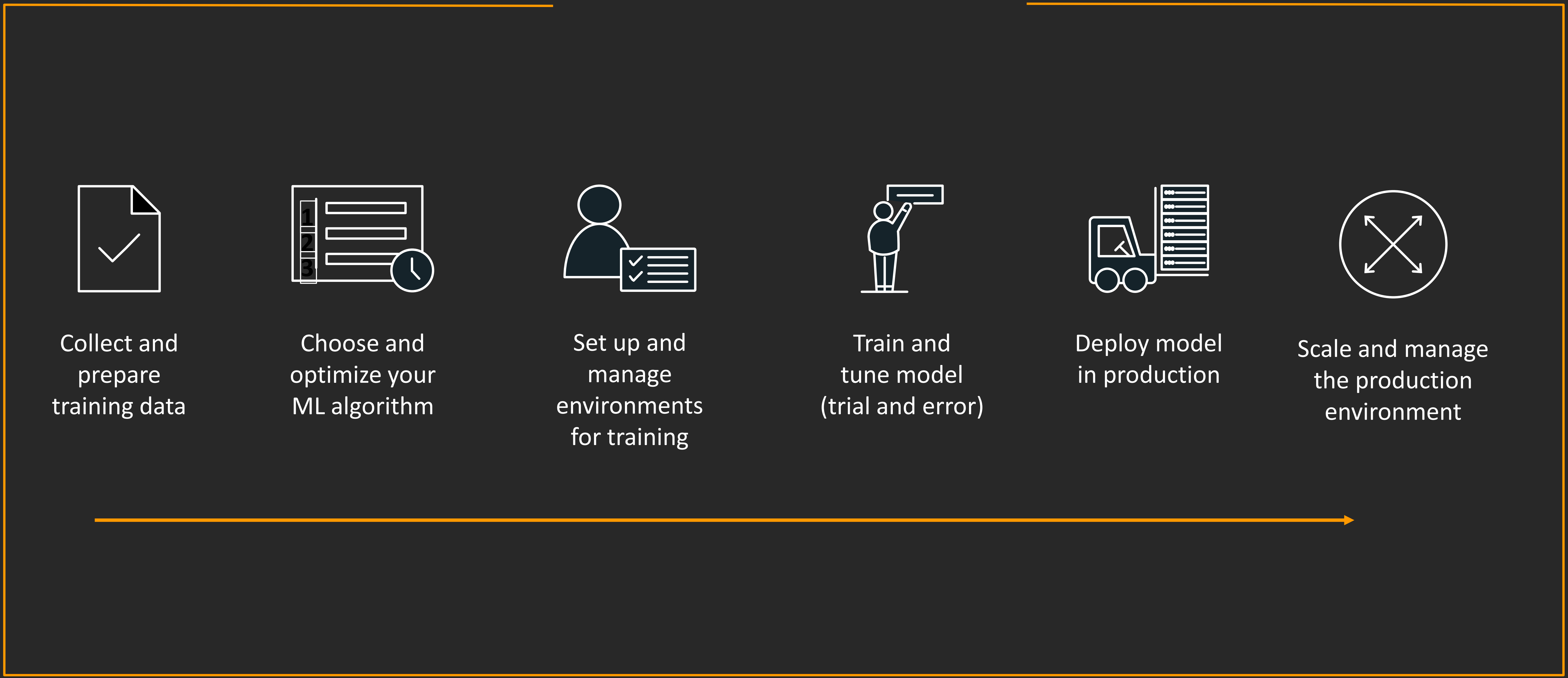
Infrastructure

AMAZON EC2
P3 Instances

AMAZON EC2
C5 Instances

FPGAs

ML is still too complicated for everyday developers



Collect and prepare training data

Choose and optimize your ML algorithm

Set up and manage environments for training

Train and tune model (trial and error)

Deploy model in production

Scale and manage the production environment

Machine Learning Made Simple

ML SERVICES

AMAZON
SAGEMAKER

One-click

model training
& deployment

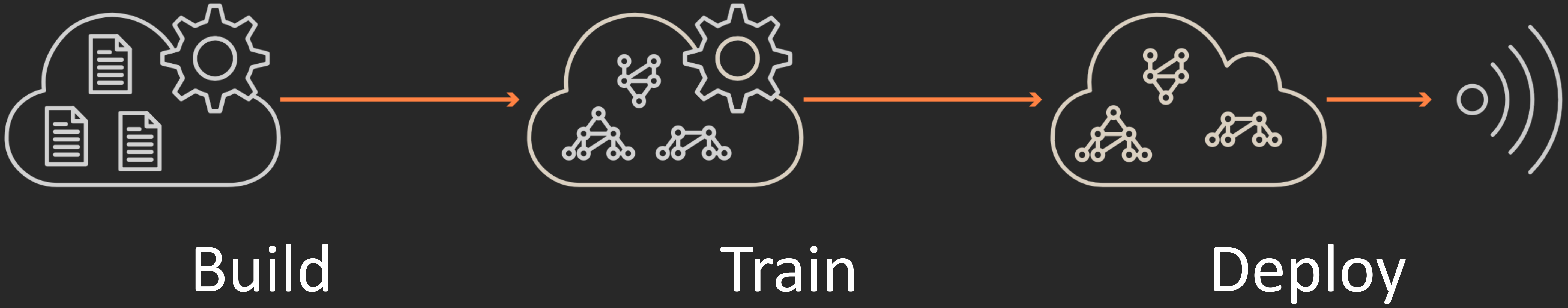
10x

better algorithm
performance

Predictive insights

to improve business
decision making

Amazon SageMaker simplifies Machine Learning

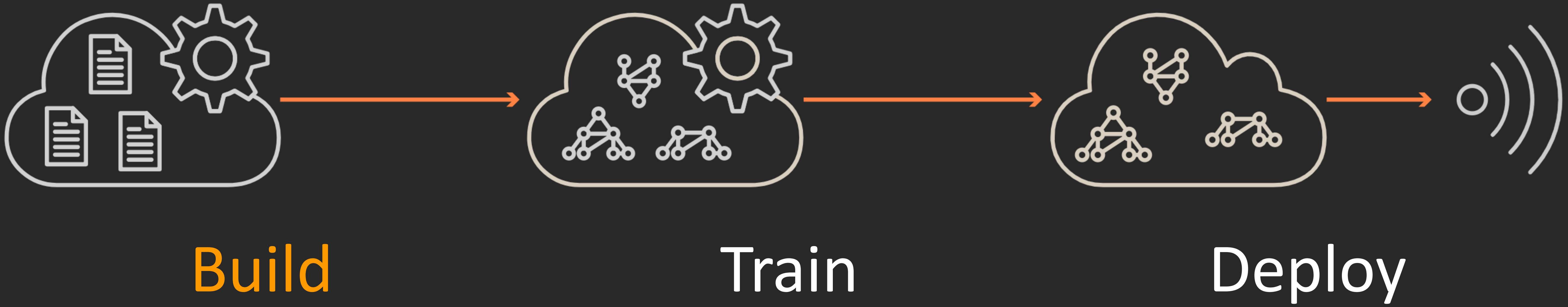


Build

Train

Deploy

Amazon SageMaker modules



- Notebook instances
- Call APIs from your device

Amazon SageMaker modules

Build

Train

Deploy

- Managed
- Distributed
- High performance I/O

Amazon SageMaker modules

Build

Train

Deploy

- Real-time endpoints
- Batch transform
- AWS Greengrass
- AWS DeepLens

Amazon SageMaker features

SageMaker algorithms

TensorFlow™

PYTORCH

Chainer

mxnet

Frameworks

docker

Bring your own

Automatic model tuning

Amazon SageMaker features

SageMaker algorithms

- Designed for speed and scale
- Supervised, unsupervised, computer vision, and NLP

TensorFlow™
PYTORCH
Chainer
mxnet

Frameworks

docker

Bring your own

Automatic model tuning

Amazon SageMaker features

SageMaker algorithms

TensorFlow™
PYTORCH
Chainer
mxnet

Frameworks

- 20 lines of Python
- Open sourced
- Local mode for testing

docker

Bring your own

Automatic model tuning

Amazon SageMaker features

SageMaker algorithms

TensorFlow™
PYTORCH
Chainer
mxnet

Frameworks

docker

Bring your own

- Publish to a container registry
- R, Java, Julia, etc.

Automatic model tuning

Amazon SageMaker features

SageMaker algorithms

TensorFlow™

PYTORCH

Chainer

mxnet

Frameworks

docker

Bring your own

Automatic model tuning

- Efficient meta-model hyperparameter tuning
- Works with algorithms, frameworks, and BYO

Real-time deployment at scale

Creating endpoints

Easy deployment to production REST API

Scalable, high throughput, and high reliability

Creating endpoints

Model

```
aws sagemaker create-model
  --model-name model1
  --primary-container ‘{“Image”: “123.dkr.ecr.amazonaws.com/algo”,
                        “ModelDataUrl”: “s3://bkt/model1.tar.gz”}’
  --execution-role-arn arn:aws:iam::123:role/me
```

Creating endpoints

Model

```
aws sagemaker create-model
  --model-name model1
  --primary-container ‘{“Image”: “123.dkr.ecr.amazonaws.com/algo”,
                        “ModelDataUrl”: “s3://bkt/model1.tar.gz”}’
  --execution-role-arn arn:aws:iam::123:role/me
```

Endpoint configuration

```
aws sagemaker create-endpoint-config
  --endpoint-config-name model1-config
  --production-variants ‘{“InitialInstanceCount”: 2,
                          “InstanceType”: “ml.m4.xlarge”,
                          ”InitialVariantWeight”: 1,
                          ”ModelName”: “model1”,
                          ”VariantName”: “AllTraffic”}’
```

Creating endpoints

Model

```
aws sagemaker create-model
  --model-name model1
  --primary-container ‘{“Image”: “123.dkr.ecr.amazonaws.com/algo”,
                        “ModelDataUrl”: “s3://bkt/model1.tar.gz”}’
  --execution-role-arn arn:aws:iam::123:role/me
```

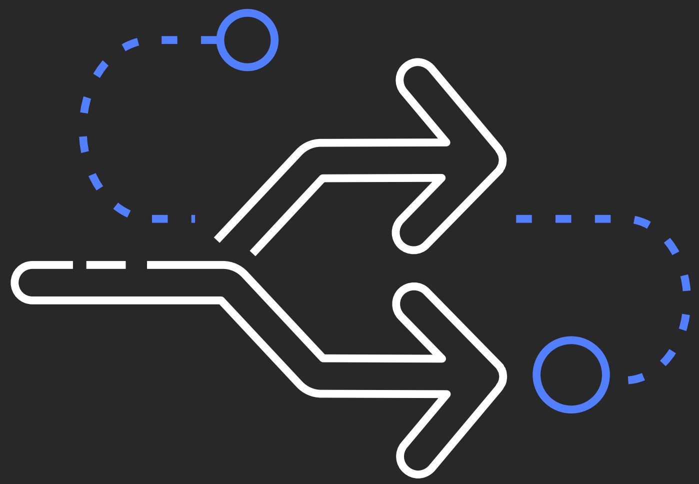
Endpoint configuration

```
aws sagemaker create-endpoint-config
  --endpoint-config-name model1-config
  --production-variants ‘{“InitialInstanceCount”: 2,
                          “InstanceType”: “ml.m4.xlarge”,
                          ”InitialVariantWeight”: 1,
                          ”ModelName”: “model1”,
                          ”VariantName”: “AllTraffic”}’
```

Endpoint

```
aws sagemaker create-endpoint
  --endpoint-name my-endpoint
  --endpoint-config-name model1-config
```

Updating endpoints



Blue-green deployments mean no scheduled downtime

Deploy one or more models behind the same endpoint

Updating endpoints

New model

```
aws sagemaker create-model
  --model-name model2
  --primary-container ‘{“Image”: “123.dkr.ecr.amazonaws.com/algo”,
                        “ModelDataUrl”: “s3://bkt/model2.tar.gz”}’
  --execution-role-arn arn:aws:iam::123:role/me
```

Updating endpoints

New model

```
aws sagemaker create-model
  --model-name model2
  --primary-container ‘{“Image”: “123.dkr.ecr.amazonaws.com/algo”,
                        “ModelDataUrl”: “s3://bkt/model2.tar.gz”}’
  --execution-role-arn arn:aws:iam::123:role/me
```

New endpoint configuration

```
aws sagemaker create-endpoint-config
  --endpoint-config-name model2-config
  --production-variants ‘{“InitialInstanceCount”: 2,
                          “InstanceType”: “ml.m4.xlarge”,
                          ”InitialVariantWeight”: 1,
                          ”ModelName”: “model2”,
                          ”VariantName”: “AllTraffic”}’
```

Updating endpoints

New model

```
aws sagemaker create-model
  --model-name model2
  --primary-container ‘{“Image”: “123.dkr.ecr.amazonaws.com/algo”,
                        “ModelDataUrl”: “s3://bkt/model2.tar.gz”}
  --execution-role-arn arn:aws:iam::123:role/me
```

New endpoint configuration

```
aws sagemaker create-endpoint-config
  --endpoint-config-name model2-config
  --production-variants ‘{“InitialInstanceCount”: 2,
                          “InstanceType”: “ml.m4.xlarge”,
                          ”InitialVariantWeight”: 1,
                          ”ModelName”: “model2”,
                          ”VariantName”: “AllTraffic”}’
```

Same endpoint

```
aws sagemaker update-endpoint
  --endpoint-name my-endpoint
  --endpoint-config-name model2-config
```

Reduced risk deployments

Incrementally retrain models with new data

Try new models and improved algorithms

Reduced risk deployments

Two-model endpoint configuration

```
aws sagemaker create-endpoint-config
  --endpoint-config-name both-models-config
  --production-variants ‘[{“InitialInstanceCount”: 2,
                           “InstanceType”: “ml.m4.xlarge”,
                           ”InitialVariantWeight”: 95,
                           ”ModelName”: “model1”,
                           ”VariantName”: “model1-traffic”},
                          {“InitialInstanceCount”: 2,
                           “InstanceType”: “ml.m4.xlarge”,
                           ”InitialVariantWeight”: 5,
                           ”ModelName”: “model2”,
                           ”VariantName”: “model2-traffic”}]’
```

Reduced risk deployments

Two-model endpoint configuration

```
aws sagemaker create-endpoint-config
  --endpoint-config-name both-models-config
  --production-variants ‘[{“InitialInstanceCount”: 2,
                           “InstanceType”: “ml.m4.xlarge”,
                           ”InitialVariantWeight”: 95,
                           ”ModelName”: “model1”,
                           ”VariantName”: “model1-traffic”},
                          {“InitialInstanceCount”: 2,
                           “InstanceType”: “ml.m4.xlarge”,
                           ”InitialVariantWeight”: 5,
                           ”ModelName”: “model2”,
                           ”VariantName”: “model2-traffic”}]’
```

Same endpoint

```
aws sagemaker update-endpoint
  --endpoint-name my-endpoint
  --endpoint-config-name both-models-config
```

Reduced risk deployments

Two-model endpoint configuration

```
aws sagemaker create-endpoint-config
  --endpoint-config-name both-models-config
  --production-variants ‘[{“InitialInstanceCount”: 2,
                           “InstanceType”: “ml.m4.xlarge”,
                           ”InitialVariantWeight”: 95,
                           ”ModelName”: “model1”,
                           ”VariantName”: “model1-traffic”},
                          {“InitialInstanceCount”: 2,
                           “InstanceType”: “ml.m4.xlarge”,
                           ”InitialVariantWeight”: 5,
                           ”ModelName”: “model2”,
                           ”VariantName”: “model2-traffic”}]’
```

Same endpoint

```
aws sagemaker update-endpoint
  --endpoint-name my-endpoint
  --endpoint-config-name both-models-config
```

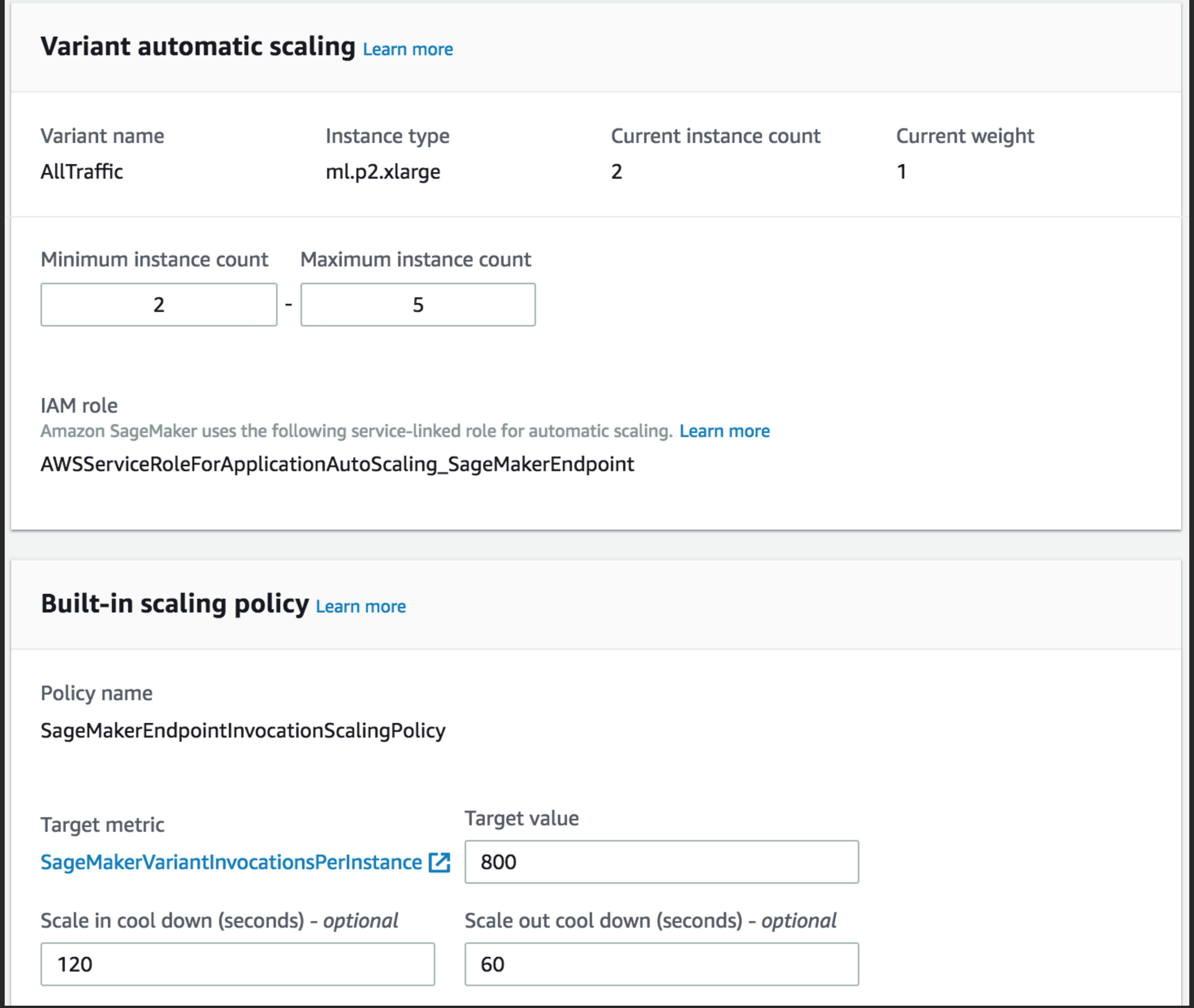
Swap

```
aws sagemaker update-endpoint-weights-and-capacities
  --endpoint-name my-endpoint
  --desired-weights-and-capacities ‘{“VariantName”: ”model1”,
                                     “DesiredWeight”: 5}’
```

Automatic scaling endpoints

SageMaker console settings:

- Min and max instances
- Target invocations per instance
- Scaling cooldowns



Variant automatic scaling Learn more

Variant name	Instance type	Current instance count	Current weight
AllTraffic	ml.p2.xlarge	2	1

Minimum instance count: 2 - Maximum instance count: 5

IAM role
Amazon SageMaker uses the following service-linked role for automatic scaling. Learn more
AWSServiceRoleForApplicationAutoScaling_SageMakerEndpoint

Built-in scaling policy Learn more

Policy name
SageMakerEndpointInvocationScalingPolicy

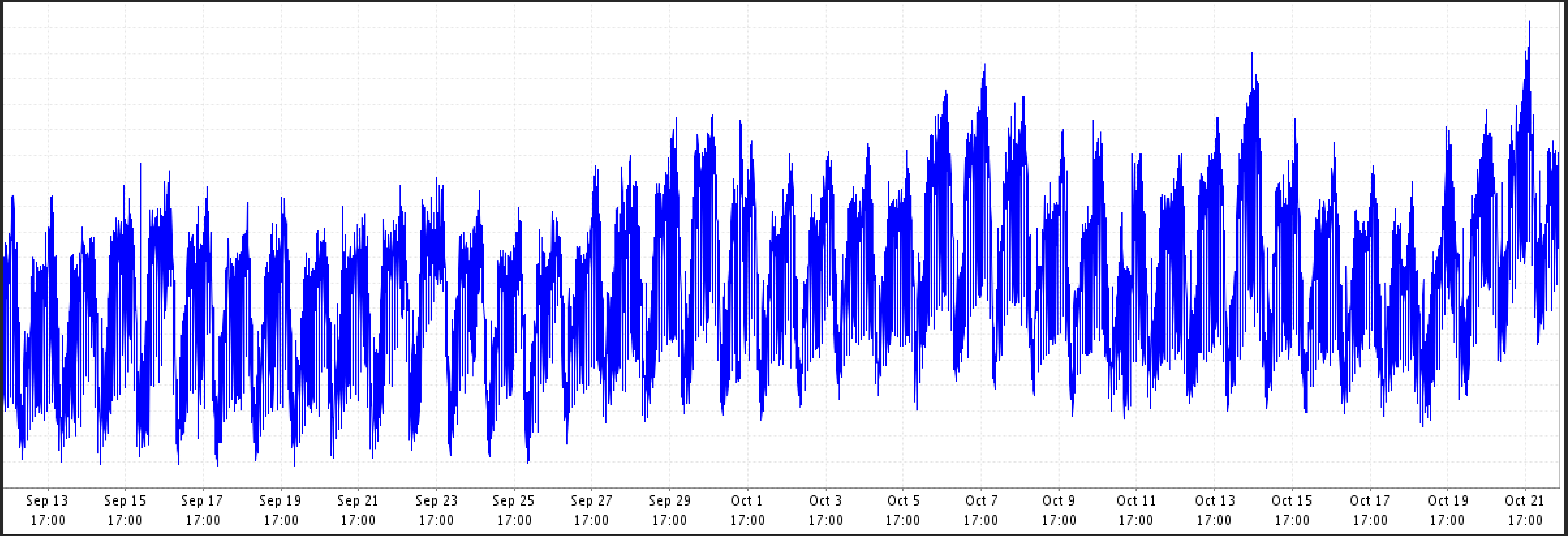
Target metric: SageMakerVariantInvocationsPerInstance

Target value: 800

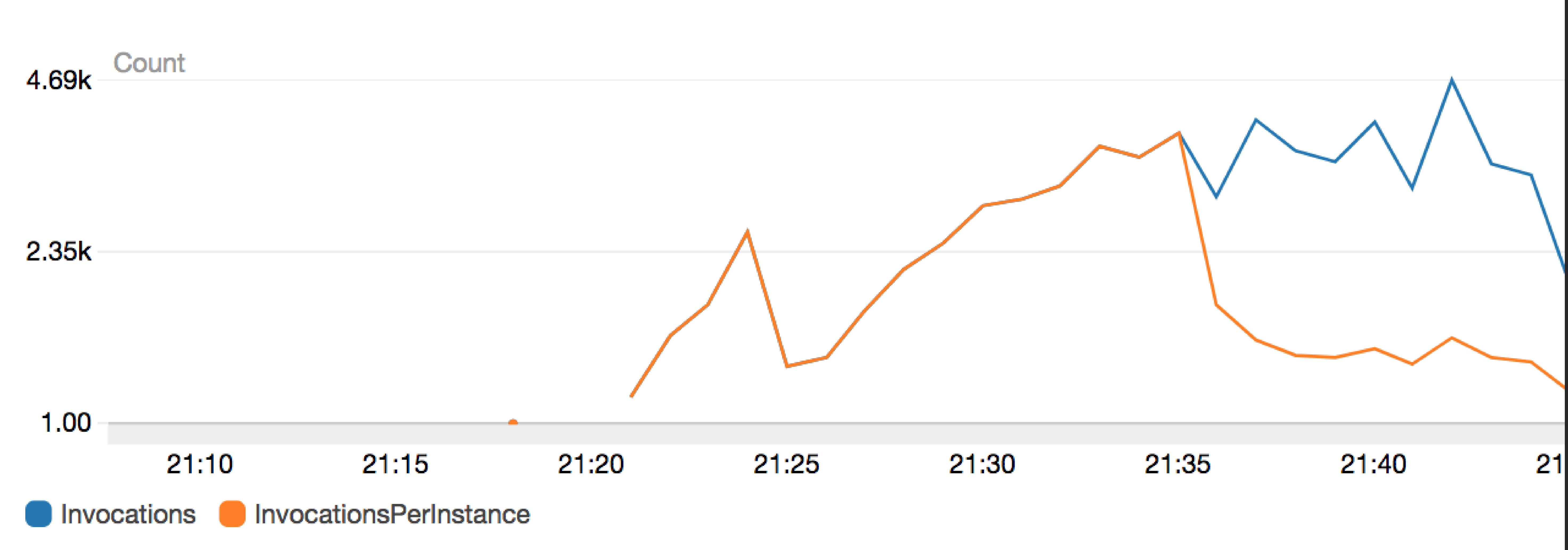
Scale in cool down (seconds) - *optional*: 120

Scale out cool down (seconds) - *optional*: 60

Why automatic scaling?



Automatic scaling in action



Scaling criteria

Algorithms have different memory, CPU, or GPU requirements

Automatically scale based on endpoint instance's Amazon CloudWatch metrics

Creating an automatic scaling policy

Variant

```
aws application-autoscaling register-scalable-target
  --service-namespace sagemaker
  --resource-id endpoint/my-endpoint/variant/model2
  --scalable-dimension sagemaker:variant:DesiredInstanceCount
  --min-capacity 2
  --max-capacity 5
```

Creating an automatic scaling policy

Variant

```
aws application-autoscaling register-scalable-target
  --service-namespace sagemaker
  --resource-id endpoint/my-endpoint/variant/model2
  --scalable-dimension sagemaker:variant:DesiredInstanceCount
  --min-capacity 2
  --max-capacity 5
```

Policy

```
aws application-autoscaling put-scaling-policy
  --policy-name model2-scaling
  --service-namespace sagemaker
  --resource-id endpoint/my-endpoint/variant/model2
  --scalable-dimension sagemaker:variant:DesiredInstanceCount
  --policy-type TargetTrackingScaling
  --target-tracking-scaling-policy-configuration
    ‘{"TargetValue": 50,
      "CustomizedMetricSpecification":
        {"MetricName": "CPUUtilization",
         "Namespace": "/aws/sagemaker/Endpoints",
         "Dimensions":
           [{"Name": "EndpointName", "Value": "my-endpoint"},
            {"Name": "VariantName","Value": ”model2"}],
        "Statistic": "Average",
        "Unit": "Percent”}}’
```

Creating an automatic scaling policy

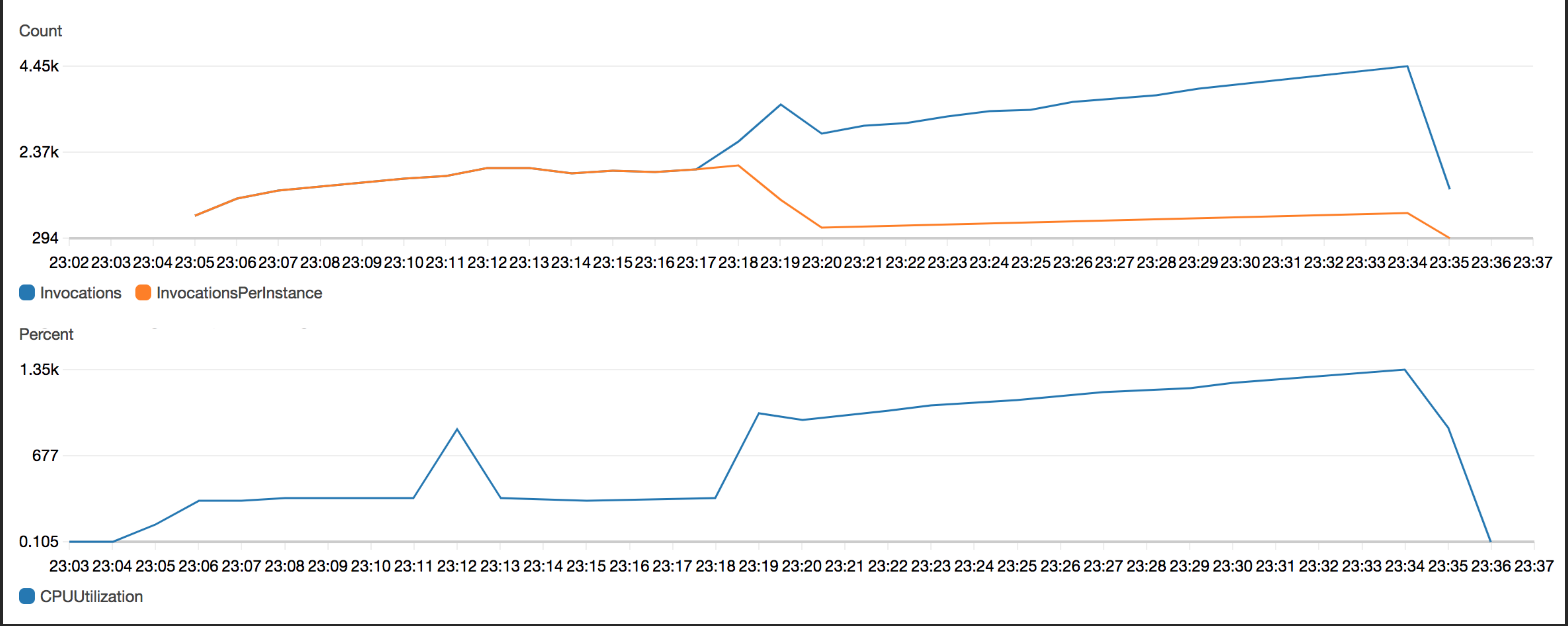
Variant

```
aws application-autoscaling register-scalable-target
  --service-namespace sagemaker
  --resource-id endpoint/my-endpoint/variant/model2
  --scalable-dimension sagemaker:variant:DesiredInstanceCount
  --min-capacity 2
  --max-capacity 5
```

Policy

```
aws application-autoscaling put-scaling-policy
  --policy-name model2-scaling
  --service-namespace sagemaker
  --resource-id endpoint/my-endpoint/variant/model2
  --scalable-dimension sagemaker:variant:DesiredInstanceCount
  --policy-type TargetTrackingScaling
  --target-tracking-scaling-policy-configuration
    ‘{"TargetValue": 50,
      "CustomizedMetricSpecification":
        {"MetricName": "CPUUtilization",
         "Namespace": "/aws/sagemaker/Endpoints",
         "Dimensions":
           [{"Name": "EndpointName", "Value": "my-endpoint"},
            {"Name": "VariantName","Value": ”model2"}],
        "Statistic": "Average",
        "Unit": "Percent”}}’
```

Scale by utilization



Intuit

Agenda

Who is Intuit?

Data lake functional architecture

Model development workflow

Key benefits of Amazon SageMaker

Standardizing model development for speed

Demo

intuit® powering prosperity

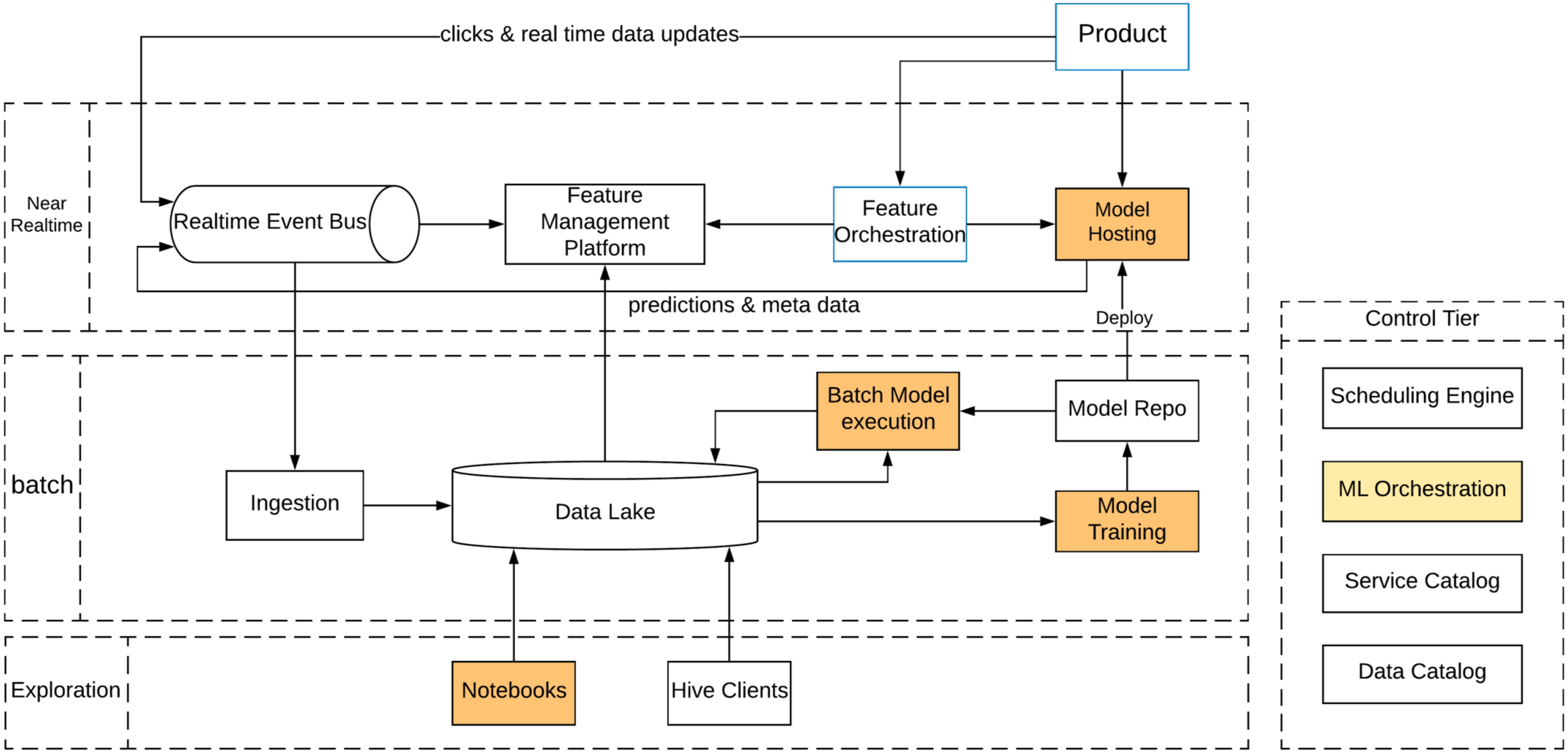
turbotax quickbooks proconnect mint

Consumers

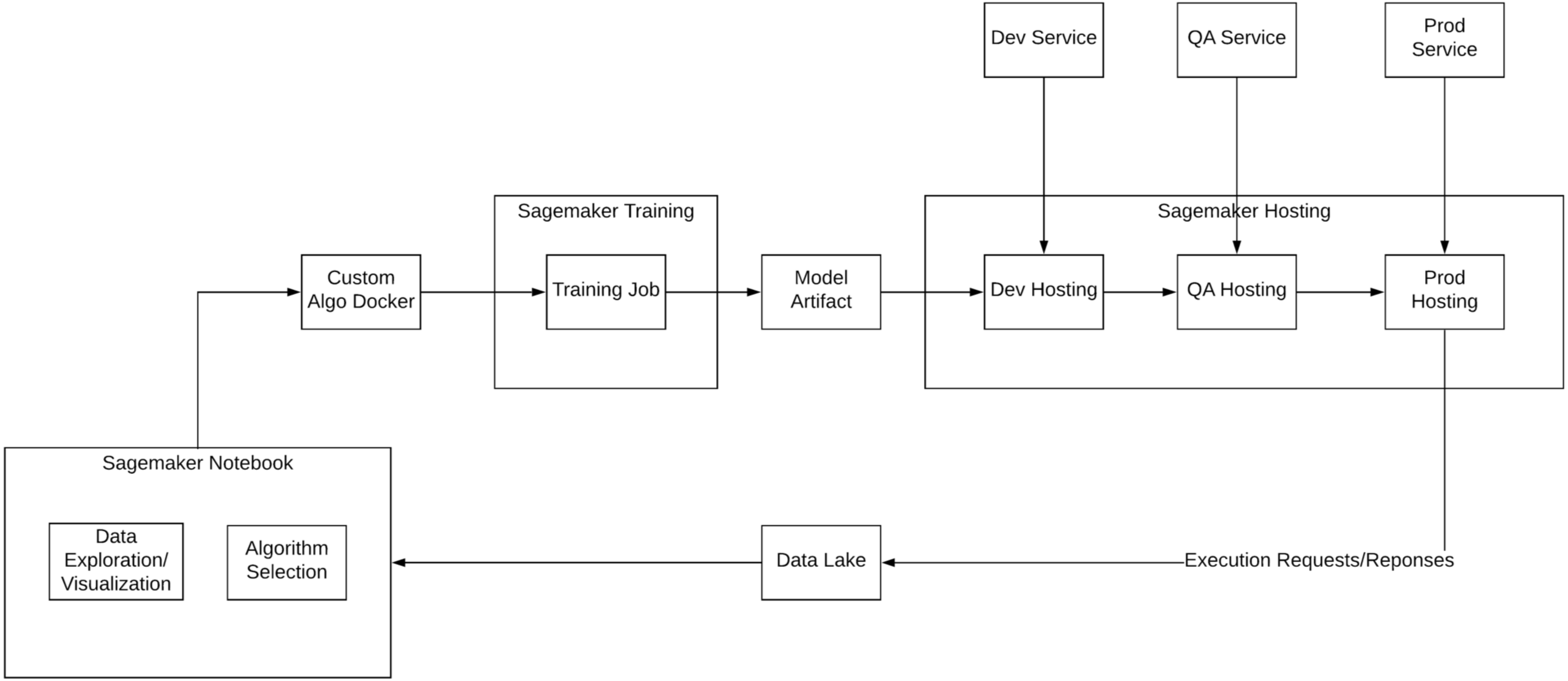
Small Businesses

Self-Employed

Overall data lake architecture



Model development workflow



AWS re:Invent

Key benefits of Amazon SageMaker

Functional

- Many algorithms supported with more being added constantly
- Custom algorithm supporting using docker
- Highly customizable
- Out of the box model parameter tuning

Security

- Integration with AWS Identity and Access Management (IAM) and AWS Key Management Service (AWS KMS)
- Good model for authentication and authorization

Scalability of Amazon cloud

Standardizing model development for speed

Standardized notebooks

- Standardized the security model around Amazon SageMaker
- Added functional integrations with Hive and our data marts

Training

- Python library to make docker images that work with Amazon SageMaker training
- We use out-of-the-box Amazon SageMaker training

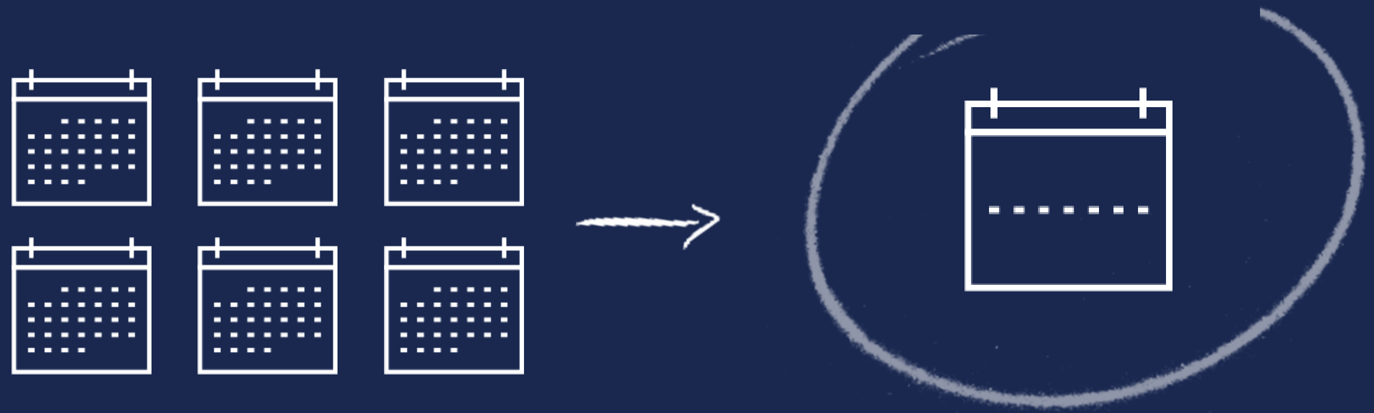
Hosting

- Python library to make Docker images that work with Amazon SageMaker hosting
- Integration with our internal Amazon API Gateway/Services Gateway

Model deployment tool

- Tracking trained model versions and taking it through MDLC intuit

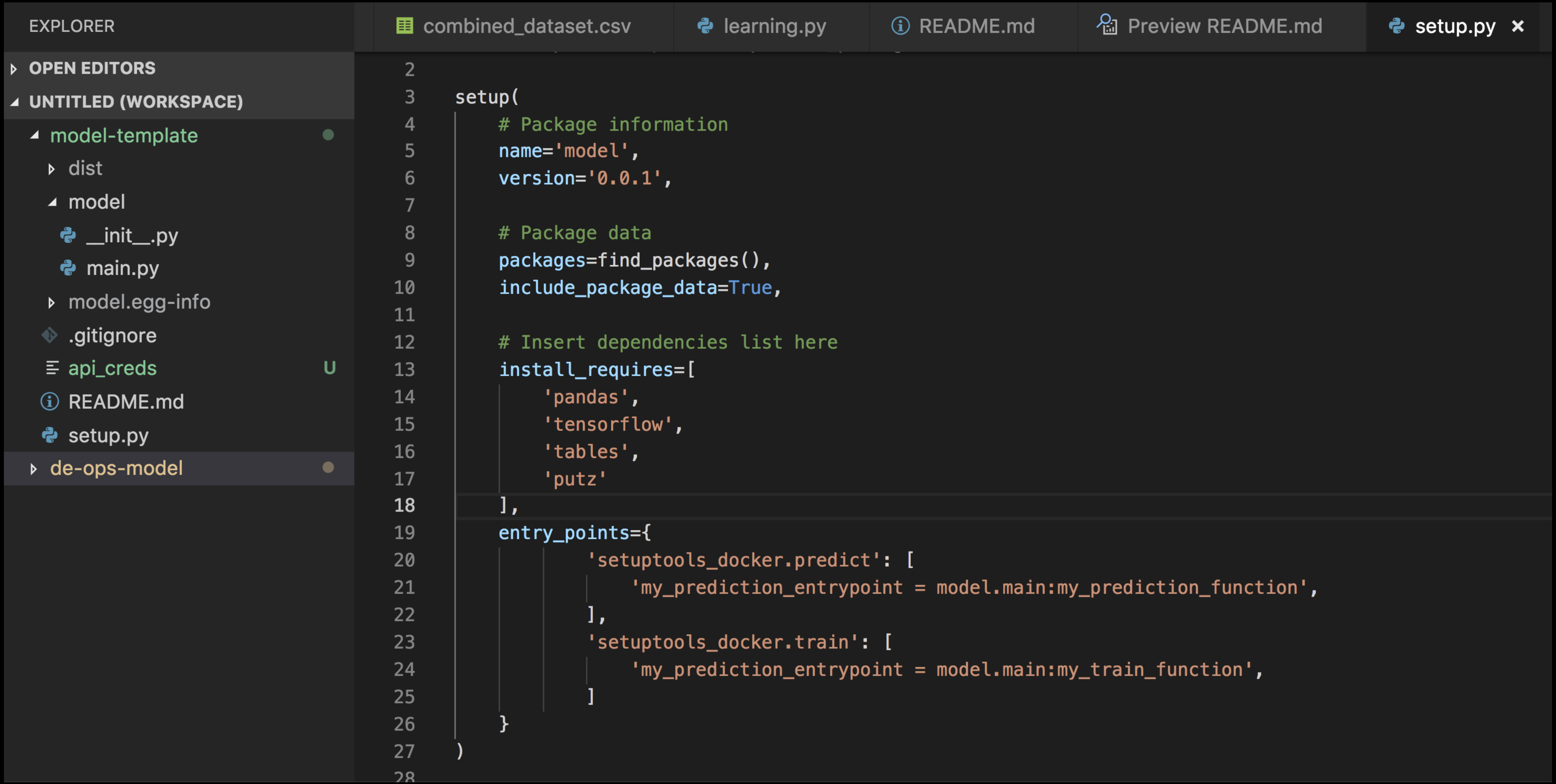
Deployment time down by 90% with Amazon SageMaker



6 MONTHS

< 1 WEEK

Demo



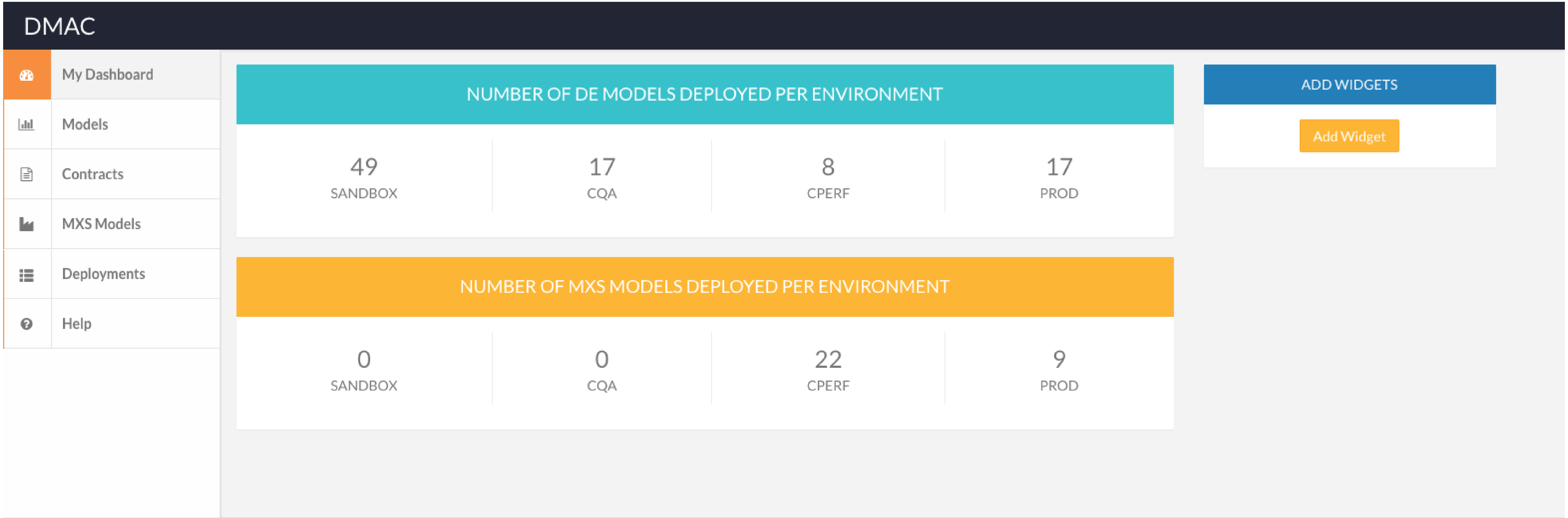
```python

setup(
    # Package information
    name='model',
    version='0.0.1',

    # Package data
    packages=find_packages(),
    include_package_data=True,

    # Insert dependencies list here
    install_requires=[
        'pandas',
        'tensorflow',
        'tables',
        'putz'
    ],
    entry_points={
            'setuptools_docker.predict': [
                'my_prediction_entrypoint = model.main:my_prediction_function',
            ],
            'setuptools_docker.train': [
                'my_prediction_entrypoint = model.main:my_train_function',
            ]
    }
)
```

Decision Model Automation Console (DMAC)



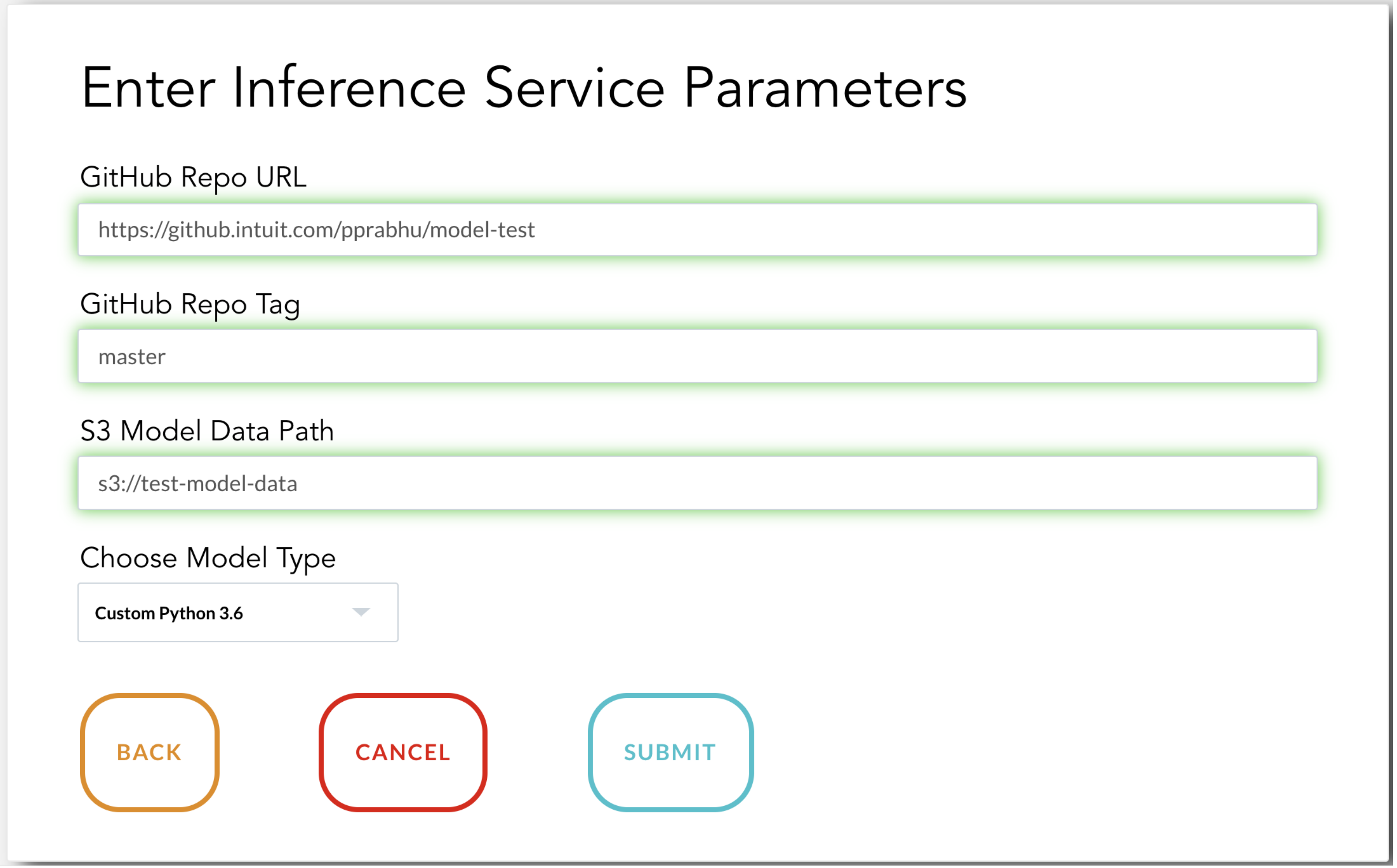
NUMBER OF DE MODELS DEPLOYED PER ENVIRONMENT

SANDBOX	CQA	CPERF	PROD
49	17	8	17

NUMBER OF MXS MODELS DEPLOYED PER ENVIRONMENT

SANDBOX	CQA	CPERF	PROD
0	0	22	9

DMAC: post training



Enter Inference Service Parameters

GitHub Repo URL

https://github.intuit.com/pprabhu/model-test

GitHub Repo Tag

master

S3 Model Data Path

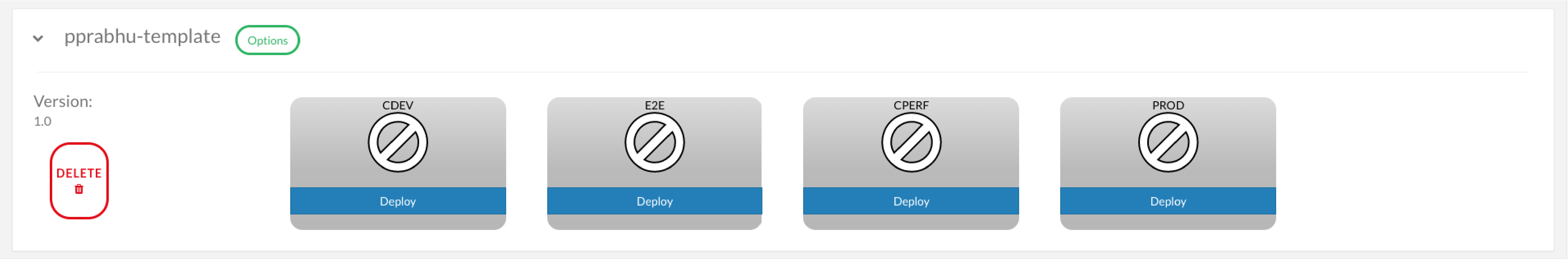
s3://test-model-data

Choose Model Type

Custom Python 3.6

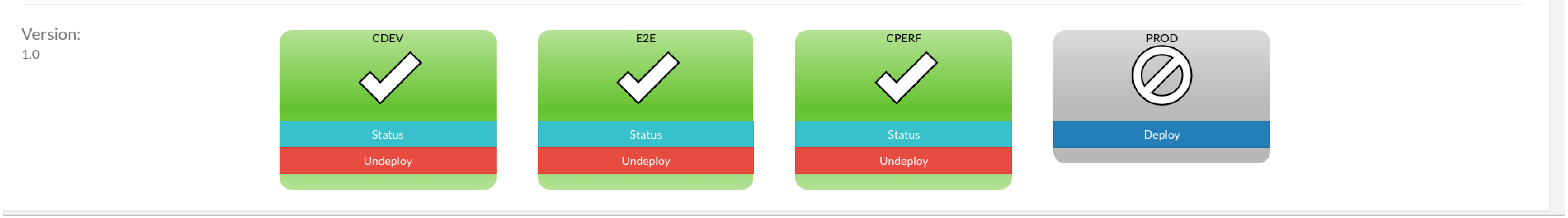
AWS re:Invent

Model deployment workflow



pprabhu-template

Version:
1.0



Version:
1.0

Conclusion

- Amazon SageMaker is a versatile platform to build, train, and deploy machine learning models at scale.

- Customers like Intuit are benefiting from integrating Amazon SageMaker into their data science workflows.

- Explore Amazon SageMaker (free tier eligible*) and build models of your own.

*https://aws.amazon.com/free/

Thank you!

David Arpin
Prasad Prabhu

AWS re:Invent

aws

Please complete the session survey in the mobile app.

AWS re:Invent

aws