aws SUMMIT

Building and running your first Machine Learning application on Amazon SageMaker

Constantin Gonzalez

Principal Solutions Architect, Amazon Web Services

What you'll get out of this session

- An overview of the Machine Learning (ML) process
- An overview of Amazon SageMaker
- Examples for:
  - Using Jupyter Notebooks
  - Feature extraction and data preparation in Python
  - ML algorithms available in Amazon SageMaker
  - Building, training and deploying ML models
- Hands-on
- Opportunities for Q&A

Overview

Long history of ML at Amazon

Personalized recommendations

Fulfillment automation and inventory management

amazon Prime Air

Drones

amazon alexa

Voice-driven interactions

amazon go

Inventing entirely new customer experiences

Machine Learning at AWS

Our mission:

Put machine learning in the hands of every developer and data scientist

The Amazon Machine Learning stack

Application Services

Platform Services

Frameworks & Infrastructure

Bottom layer: frameworks & interfaces

NVIDIA
Tesla V100 GPUs

5,120 tensor cores

128 GB of memory

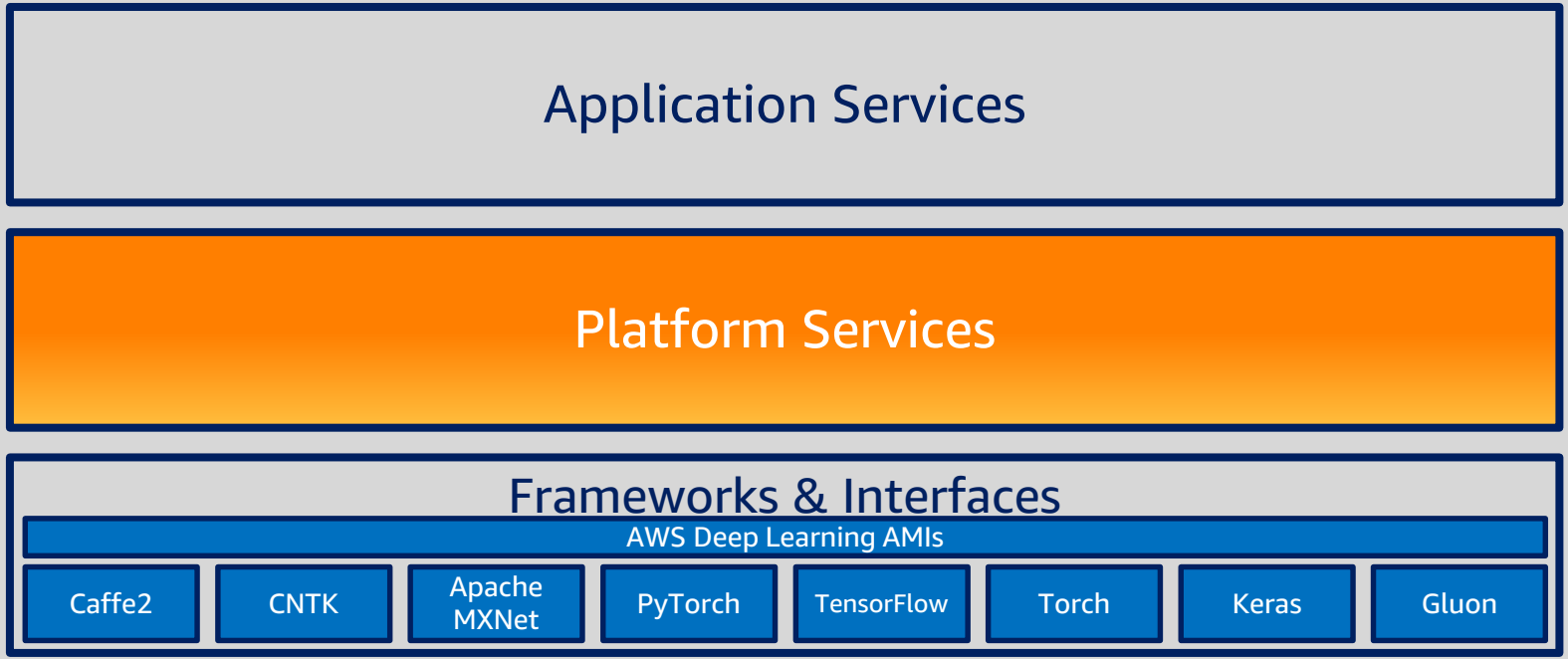
1 petaflop of compute

NVLink 2.0

~14X faster than P2

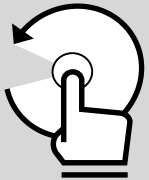
AWS Deep Learning AMI

The Amazon Machine Learning stack



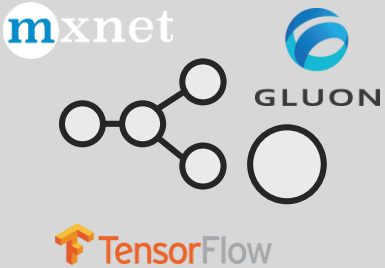
Amazon SageMaker

Build, train, and deploy machine learning models at scale



End-to-End Machine Learning Platform

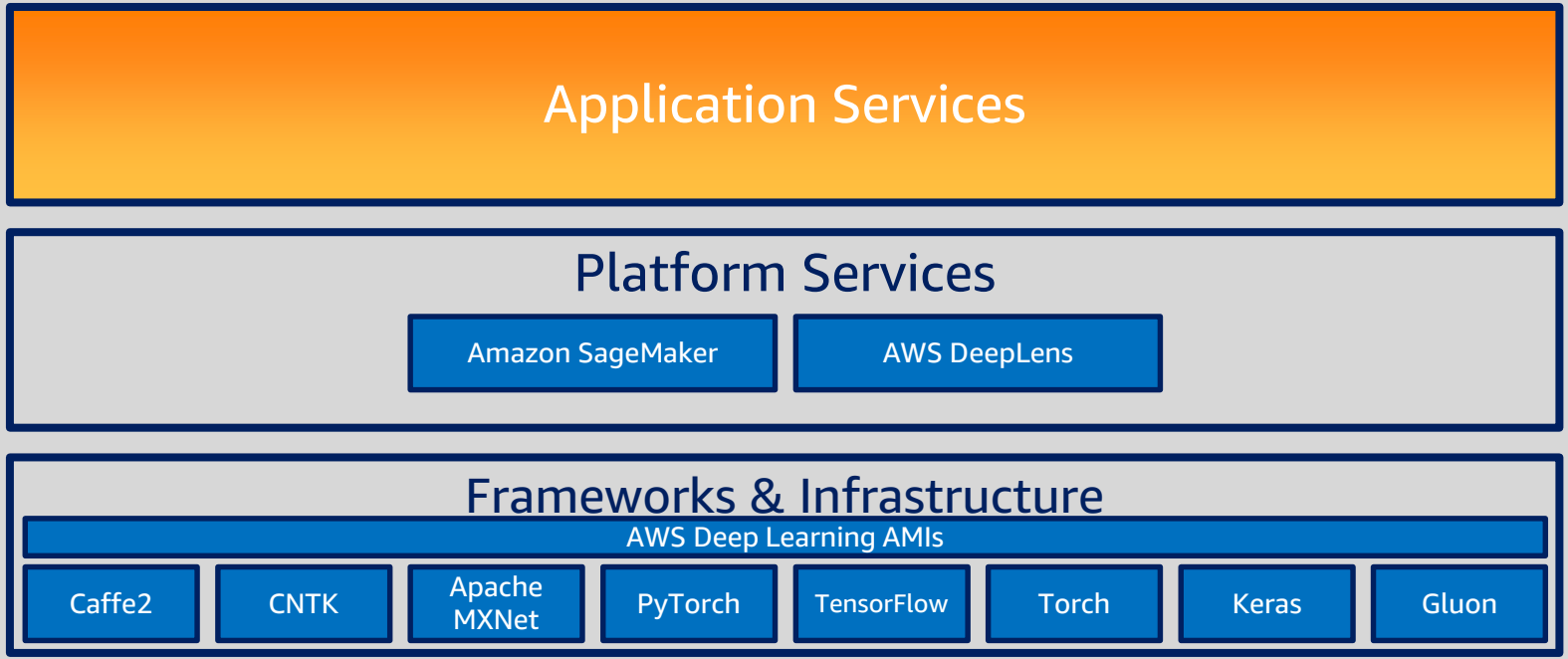
Zero setup



Flexible Model Training

Pay by the second

The Amazon Machine Learning stack



Amazon ML application services

Vision

Amazon Rekognition
Amazon Rekognition Video

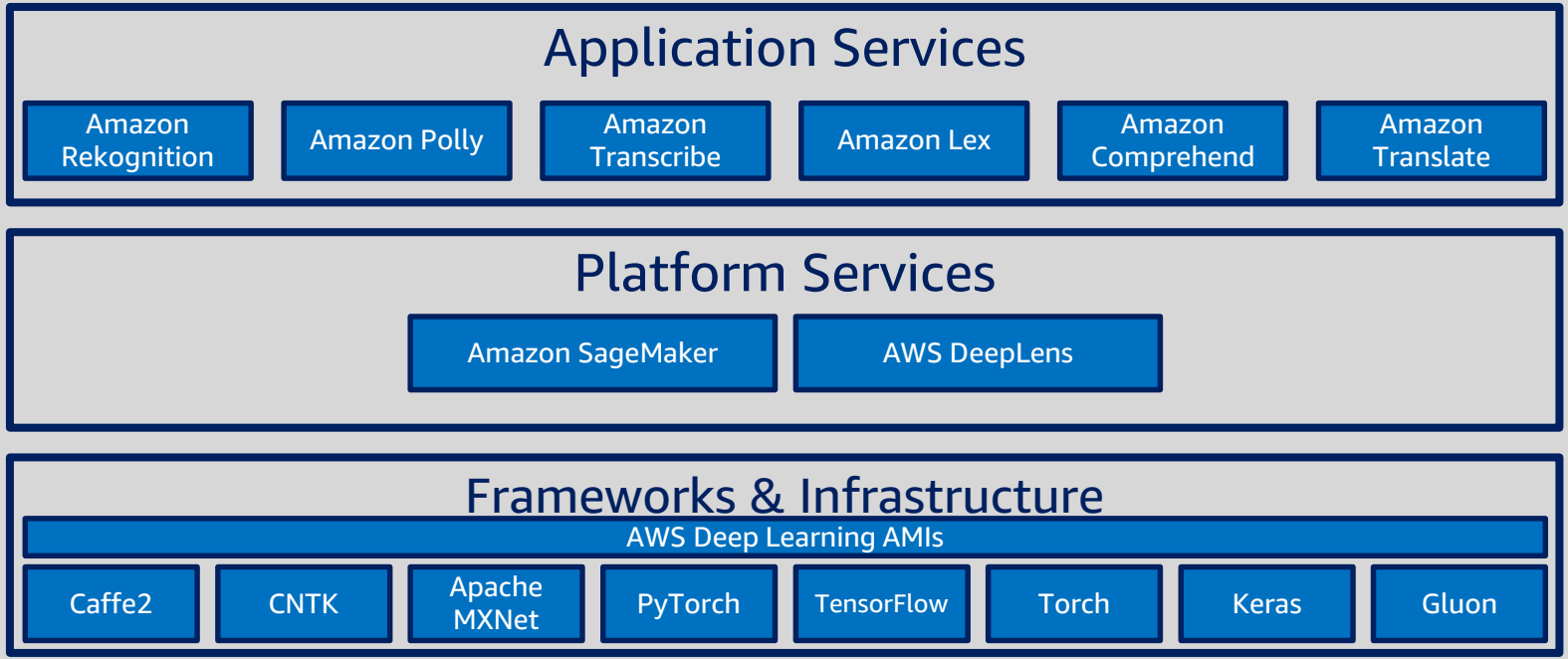
Speech

Amazon Polly
Amazon Transcribe

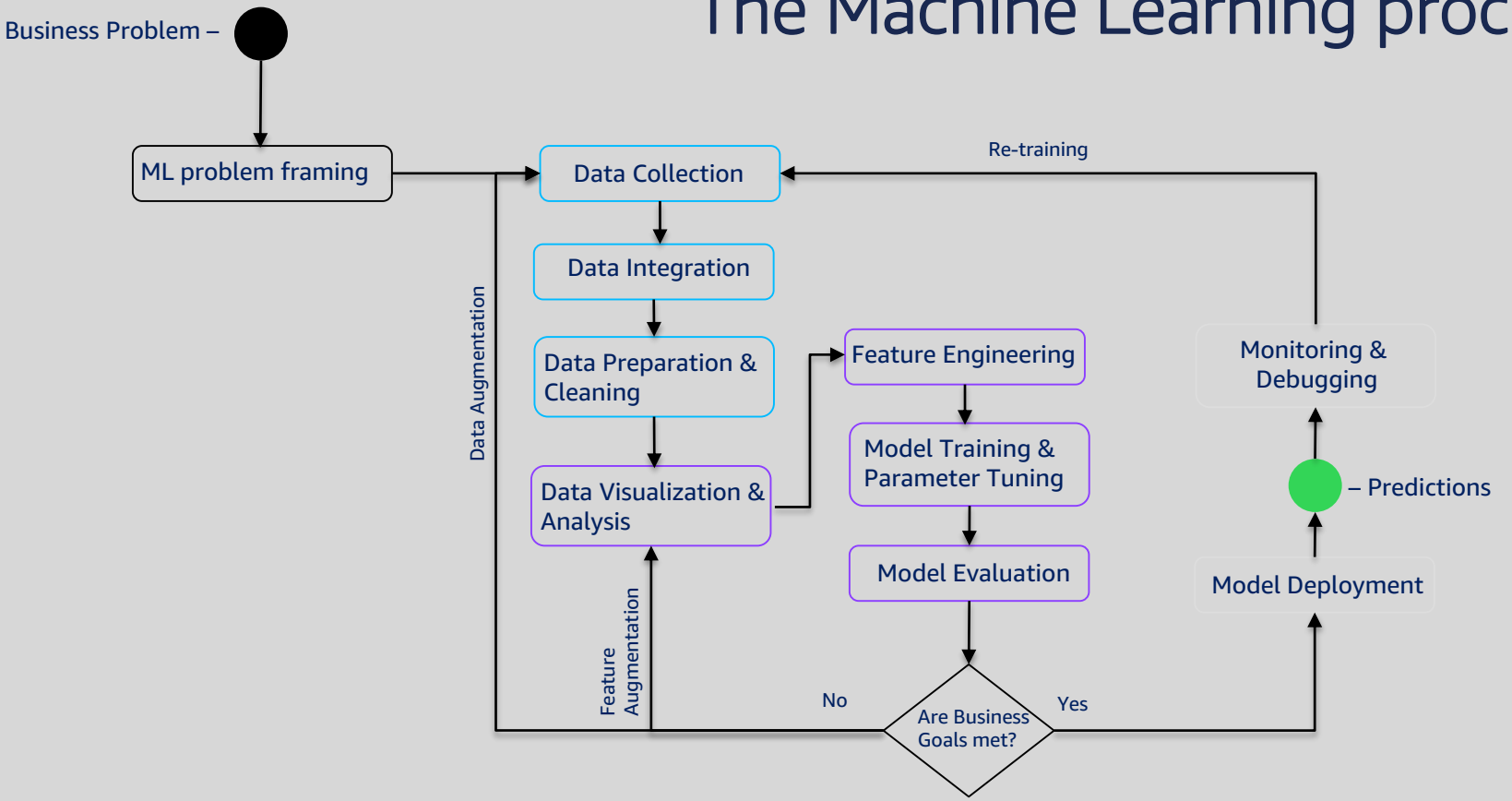
Language

Amazon Lex
Amazon Translate
Amazon Comprehend

The Amazon Machine Learning stack



The Machine Learning process

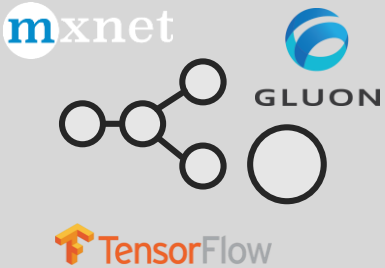


Amazon SageMaker

Build, train, and deploy machine learning models at scale

End-to-End Machine Learning Platform

Zero setup



Flexible Model Training

Pay by the second

Hands On!

Step 1: Create your notebook instance

aws Services Resource Groups PowerUser/cli-session @ glez Ireland Support

Amazon SageMaker

- Dashboard
- Notebook
- Jobs
- Inference
  - Models
  - Endpoint configurations
  - Endpoints

Amazon SageMaker > Notebook instances > Create notebook instance

Create notebook instance

Amazon SageMaker provides pre-built fully managed notebook instances that run Jupyter notebooks. The notebook instances include example code for common model training and hosting exercises. Learn More

Notebook instance settings

Notebook instance name

Maximum of 63 alphanumeric characters. Can include hyphens (-), but not spaces. Must be unique within your account in an AWS Region.

Notebook instance type

ml.t2.medium

IAM role

Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the AmazonSageMakerFullAccess IAM policy attached.

AmazonSageMaker-ExecutionRole-20171211T102774

VPC - *optional*

Your notebook instance will be provided with SageMaker provided internet access because a VPC setting is not specified.

No VPC

Lifecycle configuration - *optional*

Customize your notebook environment with default scripts and plugins.

No configuration

Encryption key - *optional*

Encrypt your notebook data. Choose an existing KMS key or enter a key's ARN.

No Encryption

Step 2: Load, visualize and prepare Data

jupyter linear_learner_mnist-Copy1 Last Checkpoint: 12/11/2017 (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Not Trusted conda_python3

Run Markdown

Data ingestion

Next, we read the dataset from an online URL into memory, for preprocessing prior to training. This processing could be done *in situ* by Amazon Athena, Apache Spark in Amazon EMR, Amazon Redshift, etc., assuming the dataset is present in the appropriate location. Then, the next step would be to transfer the data to S3 for use in training. For small datasets, such as this one, reading into memory isn't onerous, though it would be for larger datasets.

In [2]:
```
%%time
import pickle, gzip, numpy, urllib.request, json

# Load the dataset
urllib.request.urlretrieve("http://deeplearning.net/data/mnist/mnist.pkl.gz", "mnist.pkl.gz")
with gzip.open('mnist.pkl.gz', 'rb') as f:
    train_set, valid_set, test_set = pickle.load(f, encoding='latin1')
```

```
CPU times: user 876 ms, sys: 260 ms, total: 1.14 s
Wall time: 3.55 s
```

Data inspection ¶

Once the dataset is imported, it's typical as part of the machine learning process to inspect the data, understand the distributions, and determine what type(s) of preprocessing might be needed. You can perform those tasks right here in the notebook. As an example, let's go ahead and look at one of the digits that is part of the dataset.

In [3]:
```
%matplotlib inline
import matplotlib.pyplot as plt
plt.rcParams["figure.figsize"] = (2,10)


def show_digit(img, caption='', subplot=None):
    if subplot==None:
        _,(subplot)=plt.subplots(1,1)
    imgr=img.reshape((28,28))
    subplot.axis('off')
    subplot.imshow(imgr, cmap='gray')
    plt.title(caption)

show_digit(train_set[0][30], 'This is a {}'.format(train_set[1][30]))
```

This is a 3

Step 3: Train your model

aws Services Resource Groups PowerUser/cli-session @ glez Ireland Support

Amazon SageMaker

- Dashboard
- Notebook
  - Notebook instances
  - Lifecycle configurations
  - Jobs
- Inference
  - Models
  - Endpoint configurations
  - Endpoints

Amazon SageMaker > Jobs > Create training job

Create training job

When you create a training job, Amazon SageMaker sets up the distributed compute cluster, performs the training, and deletes the cluster when training has completed. The resulting model artifacts are stored in the location you specified when you created the training job. Learn More

Job settings

Job name

Maximum of 63 alphanumeric characters. Can include hyphens (-), but not spaces. Must be unique within your account in an AWS Region.

IAM role

Amazon SageMaker requires permissions to call other services on your behalf. Choose a role or let us create a role that has the AmazonSageMakerFullAccess IAM policy attached.

AmazonSageMaker-ExecutionRole-20171211T102774

Algorithm

Amazon SageMaker provides high-performance, scalable machine learning algorithms optimized for speed, scale, and accuracy for running on extremely large training datasets. See Using Built-in Algorithms with Amazon SageMaker You can also provide scripts for using deep learning frameworks, or use your own custom algorithms. See Using Your Own Algorithms with Amazon SageMaker

Choose an algorithm or custom training image...

Input mode

File

Resource configuration

Instance type	Instance count	Additional volume per instance (GB)
ml.m4.xlarge	1	1

Feedback English (US) © 2008 - 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

Step 4: Inference

aws Services Resource Groups PowerUser/cli-session @ glez Ireland Support

Amazon SageMaker

- Dashboard
- Notebook
  - Notebook instances
  - Lifecycle configurations
- Jobs
- Inference
  - Models
  - Endpoint configurations
  - Endpoints

Amazon SageMaker > Models

Models

Create endpoint | Create endpoint configuration | Actions | Create model

Search models

1 ...

Name	ARN	Creation time
DEMO-image-classification-model-constantin	arn:aws:sagemaker:eu-west-1:979400686812:model/demo-image-classification-model-constantin	Apr 19, 2018 14:48 UTC
DEMO-image-classification-model	arn:aws:sagemaker:eu-west-1:979400686812:model/demo-image-classification-model	Apr 19, 2018 14:47 UTC
DEMO-deepar-2018-04-13-09-38-10-826	arn:aws:sagemaker:eu-west-1:979400686812:model/demo-deepar-2018-04-13-09-38-10-826	Apr 13, 2018 09:48 UTC
test-image-classification-model	arn:aws:sagemaker:eu-west-1:979400686812:model/test-image-classification-model	Mar 14, 2018 09:45 UTC
ntm-2018-01-17-08-59-51-971	arn:aws:sagemaker:eu-west-1:979400686812:model/ntm-2018-01-17-08-59-51-971	Jan 17, 2018 08:59 UTC
pca-2018-01-11-09-21-02-429	arn:aws:sagemaker:eu-west-1:979400686812:model/pca-2018-01-11-09-21-02-429	Jan 11, 2018 09:21 UTC
kmeans-2018-01-11-09-14-26-872	arn:aws:sagemaker:eu-west-1:979400686812:model/kmeans-2018-01-11-09-14-26-872	Jan 11, 2018 09:14 UTC
sagemaker-mxnet-py2-cpu-2017-12-12-08-01-56-665	arn:aws:sagemaker:eu-west-1:979400686812:model/sagemaker-mxnet-py2-cpu-2017-12-12-08-01-56-665	Dec 12, 2017 08:34 UTC
xgboost-single-machine-regression-2017-12-11-17-10-17-model	arn:aws:sagemaker:eu-west-1:979400686812:model/xgboost-single-machine-regression-2017-12-11-17-10-17-model	Dec 11, 2017 17:18 UTC
factorization-machines-2017-12-11-16-50-38-313	arn:aws:sagemaker:eu-west-1:979400686812:model/factorization-machines-2017-12-11-16-50-38-313	Dec 11, 2017 16:50 UTC

https://eu-west-1.console.aws.amazon.com/sagemaker/home?region=eu-west-1#/models

 Privacy Policy Terms of Use

aws SUMMIT

Next Steps

- Explore Amazon SageMaker built-in algorithms
- Build you ML endpoint into your web/mobile/IoT app
- Deploy your ML model to IoT devices using Amazon Greengrass ML inference
- Collect new ground truth data from production
- Run A/B tests to find the best model
- Build automated data -> train -> deploy workflows
    - Amazon CodePipeline, AWS Step Functions, etc.
- Use Amazon Mechanical Turk to help label your data
- Combine multiple ML models into an ML pipeline

Build something cool on Amazon SageMaker

- Getting started with Amazon SageMaker: https://aws.amazon.com/sagemaker/

- Use the Amazon SageMaker SDK:
  - For Python: https://github.com/aws/sagemaker-python-sdk
  - For Spark: https://github.com/aws/sagemaker-spark

- SageMaker Examples: https://github.com/awslabs/amazon-sagemaker-examples

- Let us know what you build!

Please complete the session survey in the summit mobile app.

aws SUMMIT

Thank You!