Spark Notebooks in Cloud

Databricks, AWS, Google

Comparing PySpark cloud offerings

- AWS & Microsoft Azure
  - Free tier doesn’t provide cluster for PySpark
- Google Cloud Platform
  - Free tier Dataproc does provide cluster for PySpark
- Simplest
  - Databricks
- Best, inexpensive managed cluster for PySpark
  - GCP Dataproc

Managed and unmanaged infrastructure

- Managed infrastructure
  - Underlying computing resources (CPUs, cluster, network) managed by technology, not by user (data scientist)
  - E.g., Databricks, Google Colab, AWS SageMaker, Azure studio
- Unmanaged infrastructure
  - User must create cluster and install libraries (e.g., Spark)
  - E.g.,
    - Google Colab, Azure simple notebook (for PySpark setup)
    - AWS SageMaker, Azure studio for connecting to clusters defined elsewhere in technology stack

Development phases

- Common to all environments
  - Build your model
  - Train your model
  - Deploy your model
    - AWS supports deployment into Docker images, which run in their clusters

Summary of environments for PySpark notebooks

- Google Colab
  - Free, simple environment
  - See also Dataproc, AI Platform
- Amazon SageMaker
  - Can be free and simple
  - Many associated tools for more complex usage
    - Define your own cluster
    - Use Amazon Spark libraries
    - Deploy models to Docker images as endpoints on network API
- Microsoft Azure
  - Simple notebook
  - Azure Databricks
  - Cluster + notebook
  - Studio
    - Drag & drop interface
    - Design for non-data scientist

Google Colab

- https://colab.research.google.com/
- Jupyter notebook stored in Google Drive
- Simple for Python
- Managed infrastructure
  - Linux with 12G memory
- Setup required for PySpark
- Tips
  - https://www.kdnuggets.com/2018/02/essential-google-colaboratory-tips-tricks.html
  - https://www.geeksforgeeks.org/how-to-use-google-colab/
  - https://dev.to/kriyeng/8-tips-for-google-colab-notebooks-to-take-advantage-of-their-free-of-charge-12gb-ram-gpu-be4

Example PySpark in colab https://towardsdatascience.com/pyspark-in-google-colab-6821c2faf41c

Google Colab setup required

```
!wget -q http://apache.cs.utah.edu/spark/spark-2.4.4/spark-2.4.4-bin-hadoop2.7.tgz
!tar xf spark-2.4.4-bin-hadoop2.7.tgz
!pip install -q findspark
!pip install pyspark
```

```
Requirement already satisfied: pyspark in /usr/local/lib/python3.6/dist-packages (
Requirement already satisfied: py4j==0.10.7 in /usr/local/lib/python3.6/dist-packa
```

```
import os
os.environ["JAVA_HOME"] = "/usr/lib/jvm/java-8-openjdk-amd64"
os.environ["SPARK_HOME"] = "/content/spark-2.4.4-bin-hadoop2.7"
```

```
from pyspark.sql import SparkSession
from pyspark.sql import SQLContext
import findspark
#findspark.init()

APP_NAME = "Housing"
SPARK_URL = "local[*]"

spark = SparkSession\
        .builder\
        .appName(APP_NAME)\
        .master(SPARK_URL)\
        .getOrCreate()
sc = spark.sparkContext
sqlContext = SQLContext(sc)
```

```
!ls sample_data
```

```
anscombe.json                mnist_test.csv
california_housing_test.csv  mnist_train_small.csv
california_housing_train.csv README.md
```

```
df = spark.read.csv('sample_data/california_housing_test.csv',
df.printSchema()
```

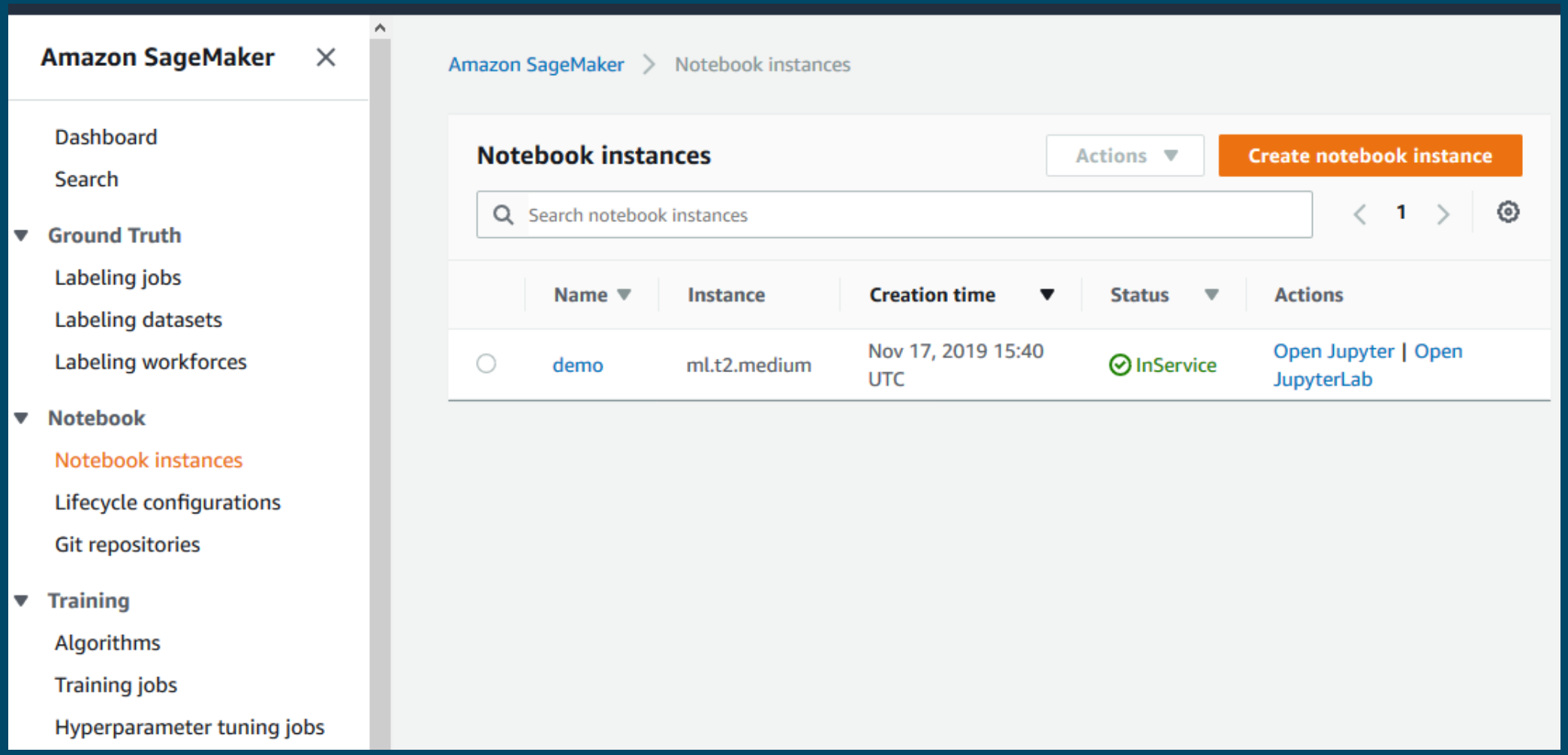
https://medium.com/@rmache/big-data-with-spark-in-google-colab-7c046e24b3

Amazon SageMaker

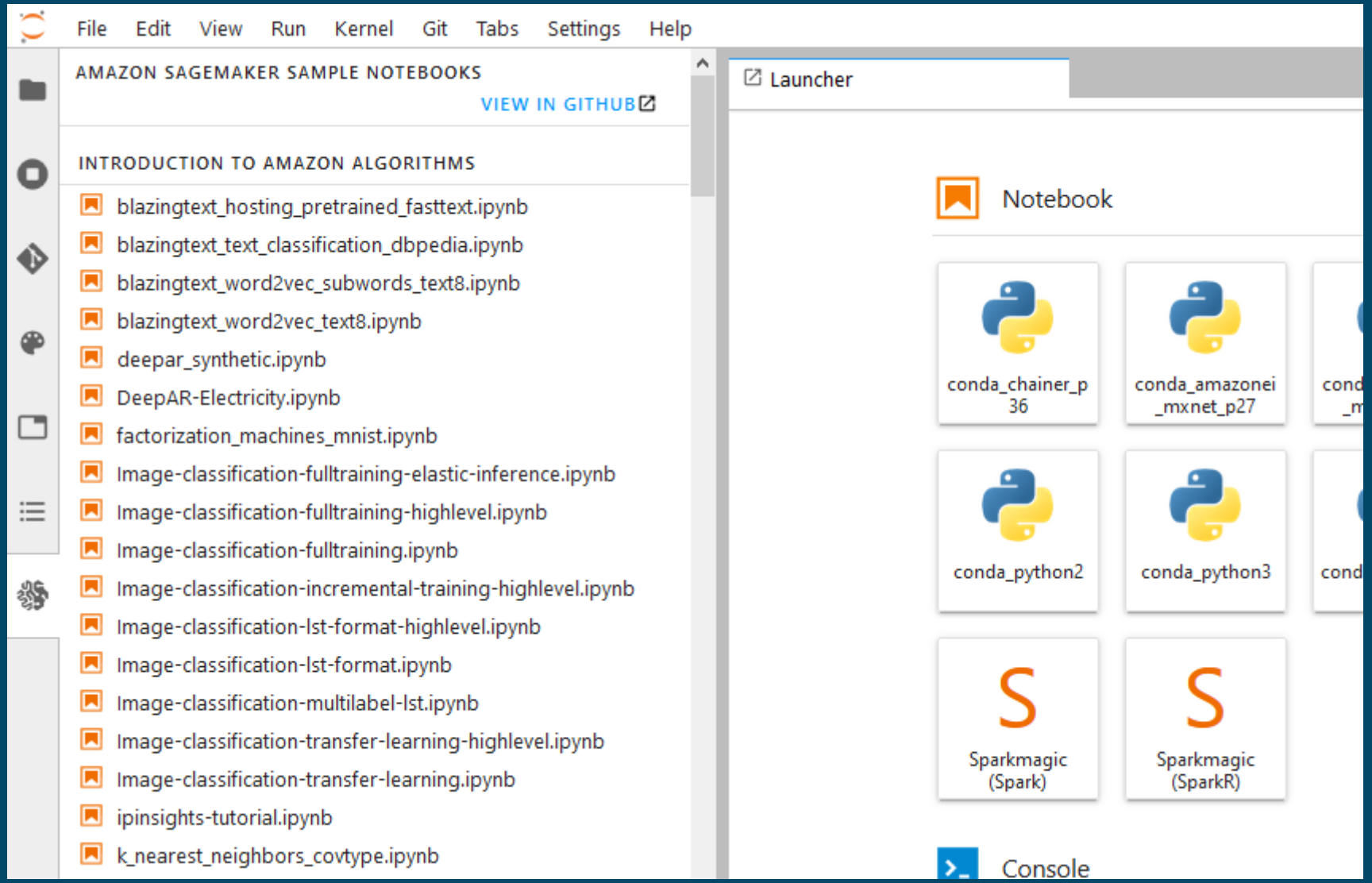
- https://aws.amazon.com/sagemaker/
- Managed infrastructure
  - Select from different size computing environments
  - Can also create EC2 spark cluster for environment
- Works with deployment
  - Using Amazon models (e.g., XGBoost), can use pre-built Docker image to create Docker image with model to create an endpoint where others can access your model via a standard API (over the network)
  - Can also use your own custom algorithms, create your own custom Docker image to deploy your model

https://towardsdatascience.com/how-to-decide-between-amazon-sagemaker-and-microsoft-azure-machine-learning-studio-157a08af839a

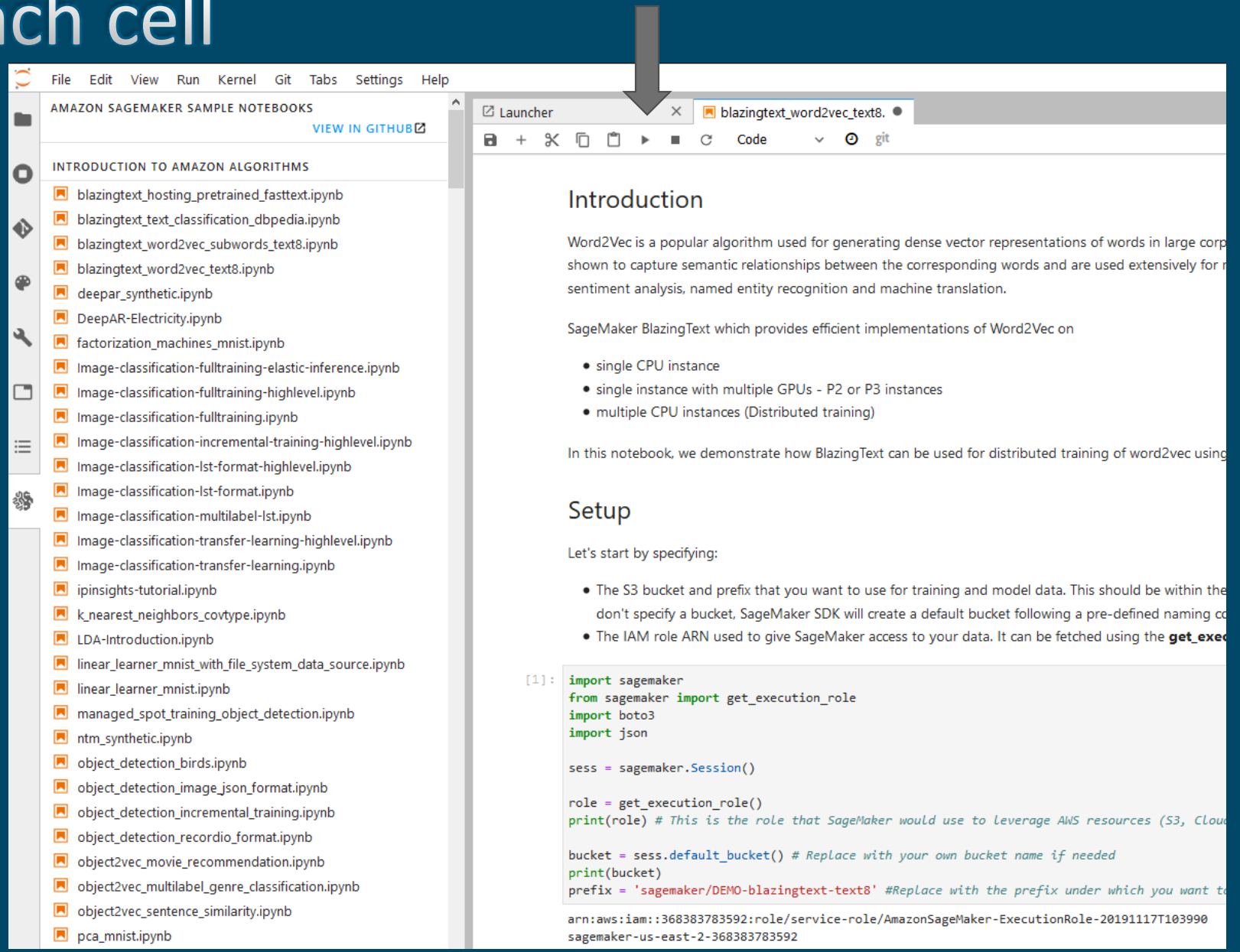
Create a notebook instance



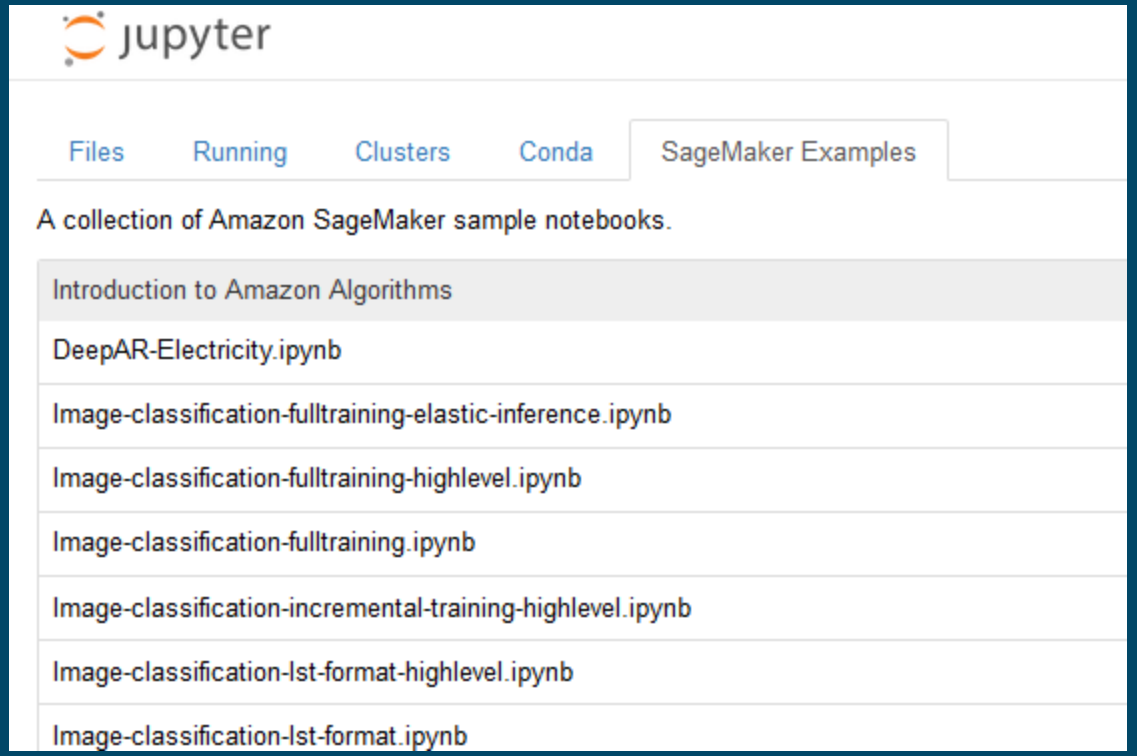
Open notebook (Jupyter lab tab) select from examples

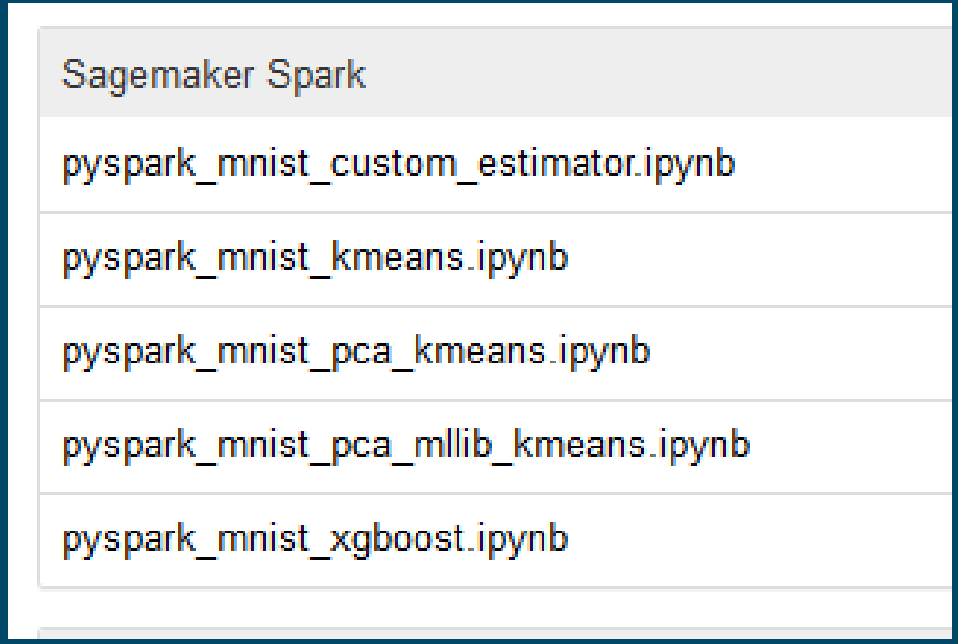


Word2Vec example
run each cell

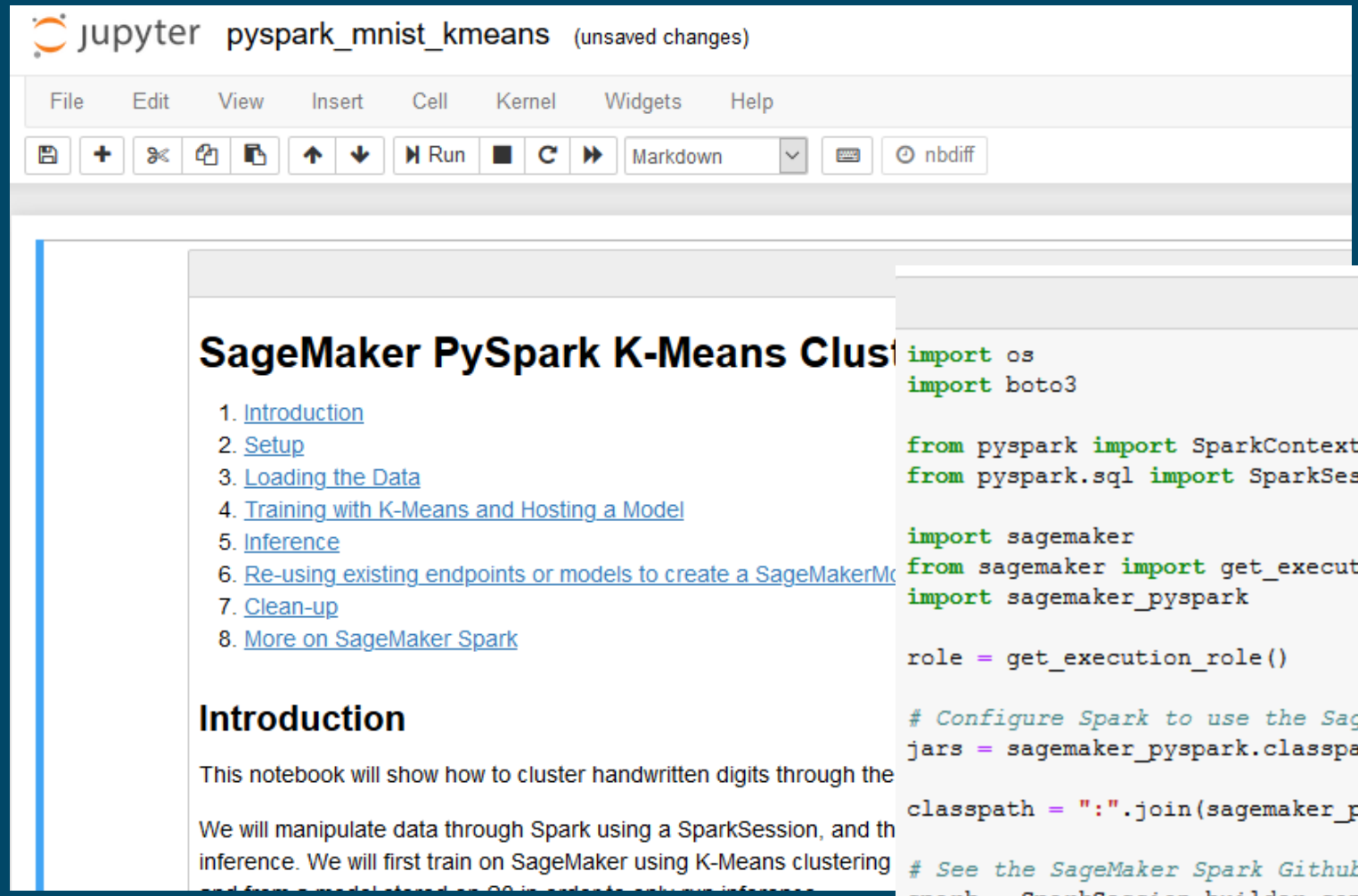


Open from Jupyter notebook tab





Running PySpark in Amazon SageMaker




```
import os
import boto3

from pyspark import SparkContext, SparkConf
from pyspark.sql import SparkSession

import sagemaker
from sagemaker import get_execution_role
import sagemaker_pyspark

role = get_execution_role()

# Configure Spark to use the SageMaker Spark dependency jars
jars = sagemaker_pyspark.classpath_jars()

classpath = ":".join(sagemaker_pyspark.classpath_jars())

# See the SageMaker Spark Github to learn how to connect to EMR from a notebook instance
spark = SparkSession.builder.config("spark.driver.extraClassPath", classpath)\
    .master("local[*]").getOrCreate()

spark
```

Amazon SageMaker

- Specialized managed environment
  - Supports creation and deployment of containers from PySpark notebooks
  - Depends on AWS API
- Similar to
  - Google AI platform
  - Microsoft Azure ML studio

Microsoft Azure notebook environments

- https://notebooks.azure.com/
- Managed infrastructure
  - 4-core CPU having a 17 GB of RAM
  - 2-core CPU having a 14 GB of RAM plus a GPU
- Notebook environments
  - Basic, like Colab, https://notebooks.azure.com/
  - Azure Databricks, https://azure.microsoft.com/en-us/services/databricks/
  - Studio, like SageMaker, https://studio.azureml.net/

Basic notebook

```
!pip install pyspark
```

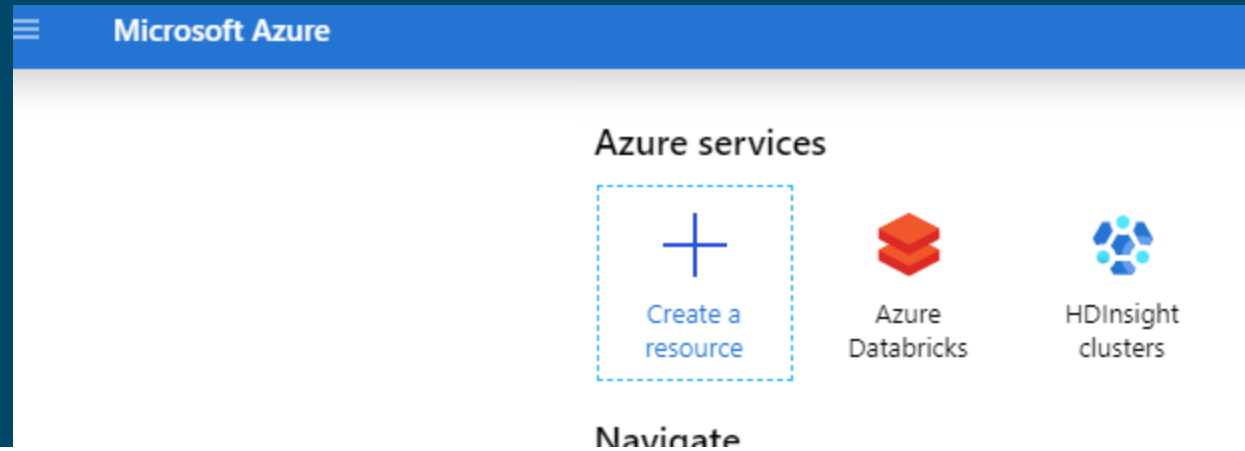
```
Requirement already satisfied: pyspark in
Requirement already satisfied: py4j==0.10.
```

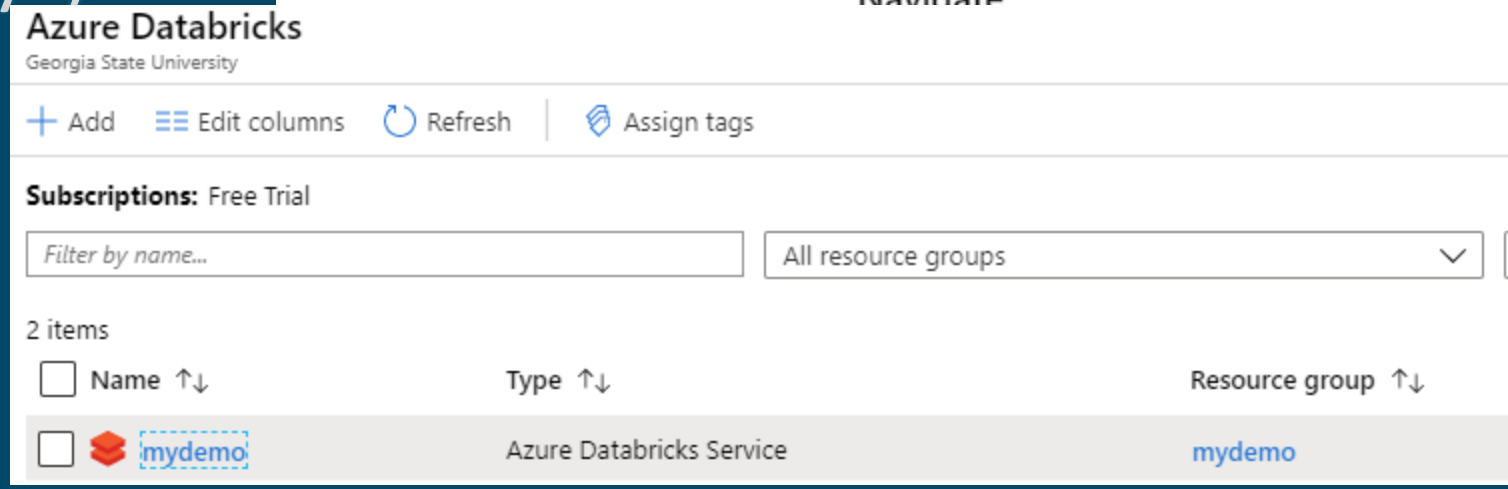
```
spark = SparkSession.builder.appName(name="PySpark Intro").master("local[*]").getOrCreate().newSession()
# .config("spark.jars", "hadoop-aws-2.7.3.jar")
spark
```

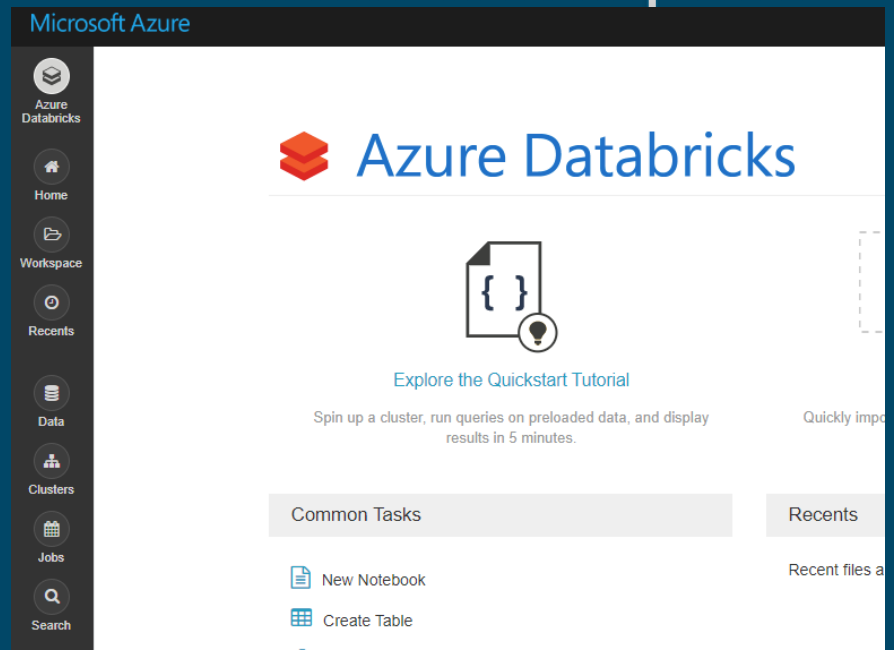
https://notebooks.azure.com/loldja/projects/pyspark-intro

Azure Databricks notebook

1. Create an Azure Databricks service
2. Select (after it deploys)
3. Launch workspace

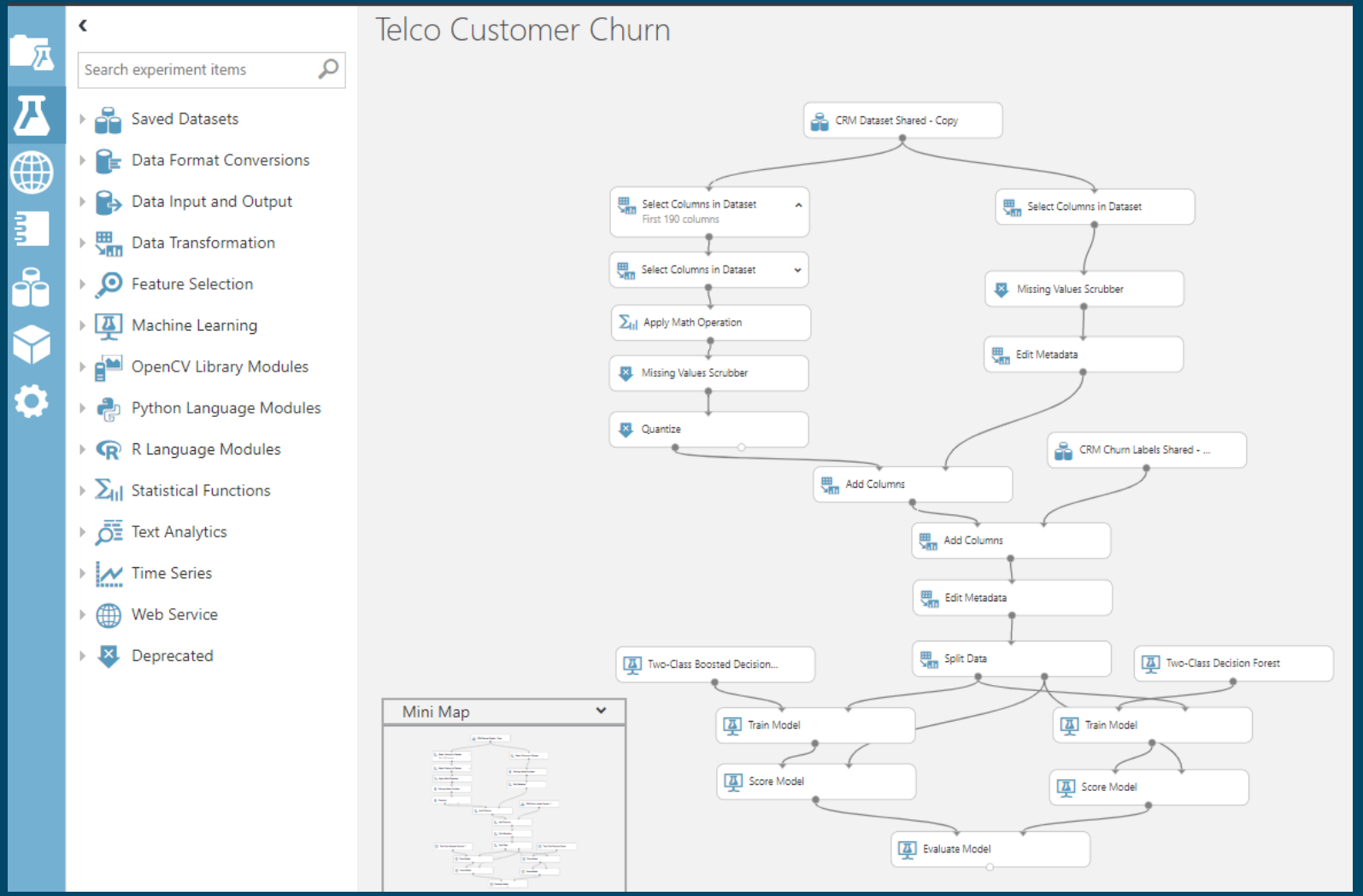






Azure ML studio

- http://studio.azureml.net
- Drag & drop



Important to remember

- Providers offer managed & unmanaged clusters
  - Managed has no need to specify node details, such as number, memory, CPU, pre-installed software etc.
  - Unmanaged requires that user configure cluster