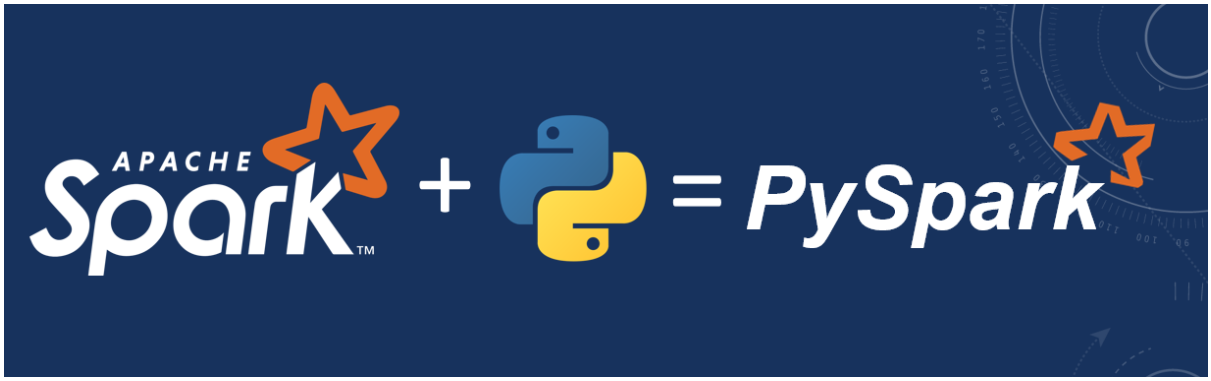


ESG Lab - Getting started with PySpark

For more information contact @Shrivats Agrawal - shriv9@seas.upenn.edu

> 💡 This document serves as a starting point on getting information about how to set up an EMR cluster on AWS, connecting a PySpark Notebook to the cluster and thereafter running some sample PySpark commands to get started.

Table of Contents:

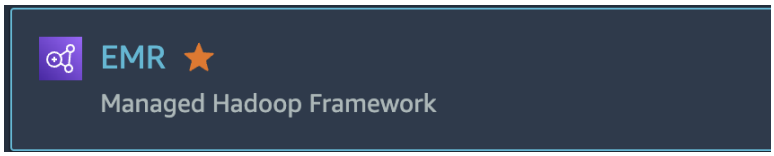
Table of Contents:
1. Setting up an EMR cluster:
2. Creating and attaching PySpark notebook to EMR cluster
3. PySpark Hello World and running some sample Python commands
Additional Resources:

---

1. Setting up an EMR cluster:

STEP 1: Log into ESG lab AWS account.

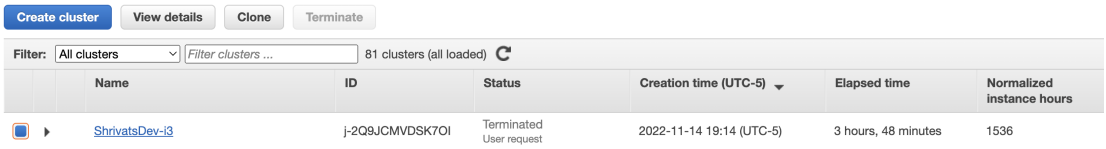
STEP 2: Navigate to EMR service as follows:

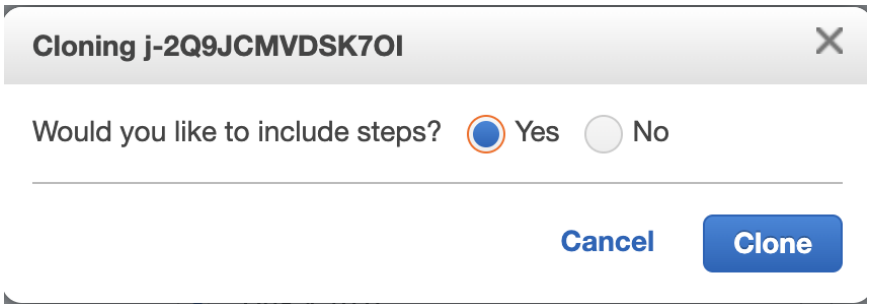


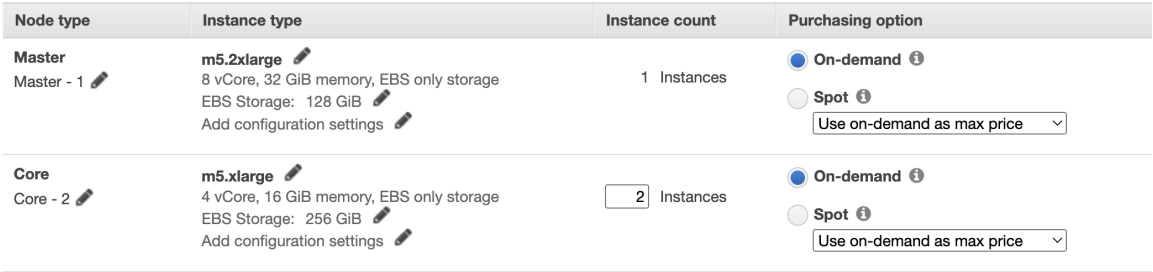
STEP 3: Create an EMR cluster by **one** of the following two ways:

a. Clone existing cluster by selecting the cluster and clicking on clone and select 'yes' when prompted to include steps and click on clone again.

> 💡 Please make sure to change the name of the cluster to your own unique name in **Step 3: General Cluster Settings**, and change hardware requirements to the exact number and type of instances required in **Step 2: Hardware** of the set-up. For this tutorial select **1** `m5.2xlarge` as the **Master Node** and **2** `m5.xlarge` as the **Core Nodes.** (The existing clusters might instantiate a large number of instances which might not be required for most use cases.)







Node type	Instance type	Instance count	Purchasing option
Master Master - 1	**m5.2xlarge** 8 vCore, 32 GiB memory, EBS only storage EBS Storage: 128 GiB Add configuration settings	1 Instances	On-demand Spot Use on-demand as max price
Core Core - 2	**m5.xlarge** 4 vCore, 16 GiB memory, EBS only storage EBS Storage: 256 GiB Add configuration settings	2 Instances	On-demand Spot Use on-demand as max price

b. Create a new cluster by clicking on “Create Cluster”.

A setup wizard will open after clicking on Create Cluster with the options to choose the distributed engine, hardware, bootstrap actions and much more.

(If this is your first time setting up a cluster, try step a. instead and change the number of instances to one of each kind during setup.)

Helpful Resources:

1. Setting up EMR Cluster
2. EMR Instance Types and Costs

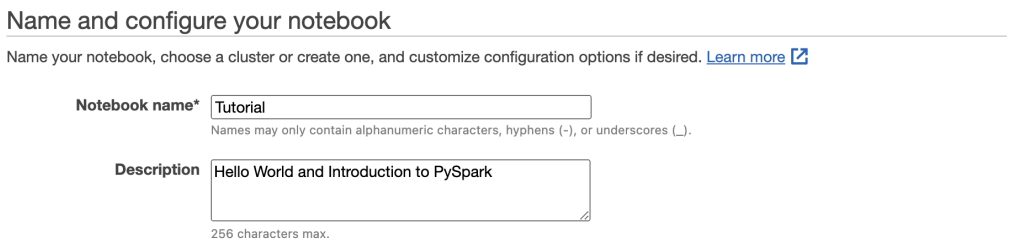
2. Creating and attaching PySpark notebook to EMR cluster

After making sure the cluster has been created in the previous step (**1.**) a PySpark or Python notebook can be created and attached to the cluster to run programs in a distributed fashion as follows:

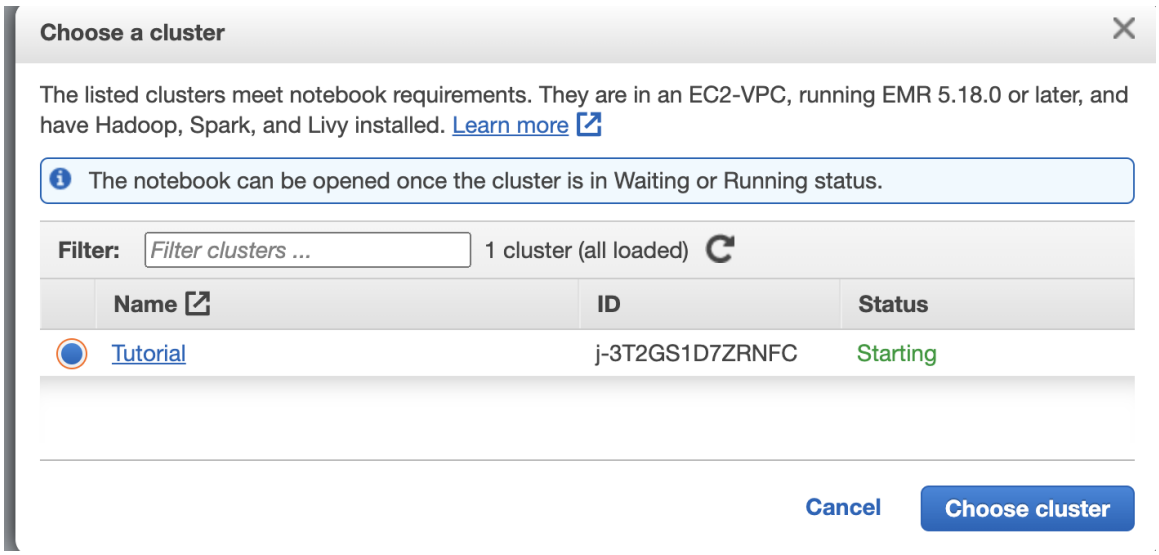
STEP 1: Head to EMR service dashboard by using the search bar, and click on **Notebooks** under the section **EMR on EC2.**

STEP 2: If you have already created a notebook and want to re-use it skip to STEP 3. Otherwise, click on **Create Notebook.**

a. Choose a Notebook Name and (optional) write a description for the purpose the notebook will accomplish.

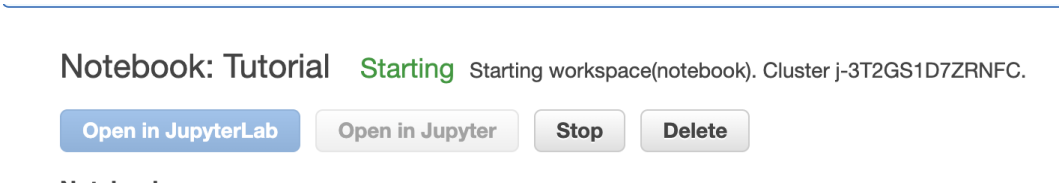


b. Click on **Choose an Existing Cluster** and select the cluster created in the previous steps as follows:



c. Click on **Create Notebook.**

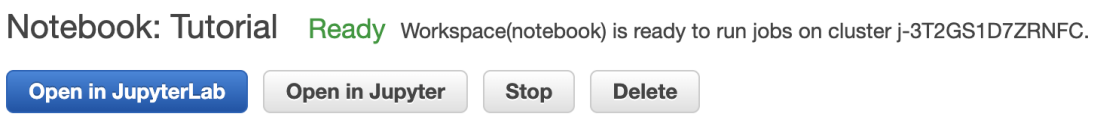
d. You should see a screen as follows indicating the status to be Starting or Pending. Please wait and refresh periodically until the status changes to "Ready" and then skip to the next step: **PySpark Hello World.**



STEP 3: If you have already created a notebook in the past, start a cluster as outlined in the previous steps and the click on ***Start*** instead of *Create Notebook.* Attach the notebook to the cluster created in the previous steps and proceed to the next step: **PySpark Hello World.**

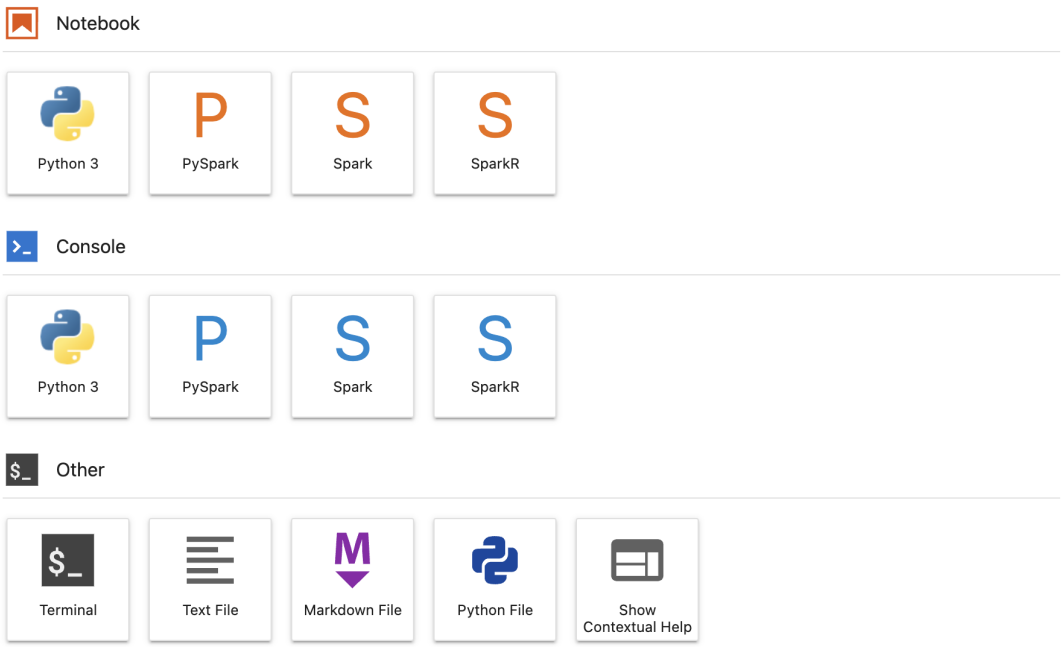
3. PySpark Hello World and running some sample Python commands

If you have successfully completed the previous steps by creating a cluster, and attaching a notebook to the cluster the notebook should be in 'Ready' status as follows:



Step 1: Open in JupyterLab

Step 2: The following menu contains the different types of files that can be created. Most prominently Python3 and PySpark notebooks are used. For the purpose of this tutorial click on "**PySpark**" under the **Notebook** sub-heading.



Step 3: Starting a Spark kernel and loading a dataframe.

a. The following list of imports prove to be very handy in performing a large number of data-processing operations with PySpark. Feel free to copy-paste them and run the cell which will also start the Spark kernel.

```
from pyspark import SparkContext
from pyspark.sql import SQLContext
from pyspark import SparkConf
from pyspark.sql import SparkSession
from pyspark.sql.functions import year, month, dayofmonth
from pyspark.sql.functions import udf
from pyspark.sql.types import StringType, ArrayType, MapType, IntegerType, DoubleType
from pyspark.sql.types import *
from pyspark.sql import functions as F
from collections import Counter
import pyspark
```

b. Start a spark session with the following command.

```
spark = SparkSession.builder.getOrCreate()
```

c. Reading sample data from a S3 location into a Spark dataframe.

```
df = spark.read.option("header",True) \
               .csv(f"s3://sector-classification/FIPS_Countries/fips_countries.csv")
```

d. Viewing the first few rows of the data.

```
df.show(5)
```

[5]:
```
df.show(5)
```

Last executed at 2022-11-16 00:04:33 in 767ms

Spark Job Progress

```
+--------------------+----+-----+-----+-----+
|         Countryname|fips|iso2c|ccode|iso3c|
+--------------------+----+-----+-----+-----+
| Antigua and Barbuda|  AC|   AG| 58.0|  ATG|
|United Arab Emirates|  AE|   AE|696.0|  ARE|
|         Afghanistan|  AF|   AF|700.0|  AFG|
|             Algeria|  AG|   DZ|615.0|  DZA|
|          Azerbaijan|  AJ|   AZ|373.0|  AZE|
+--------------------+----+-----+-----+-----+
only showing top 5 rows
```

Additional Resources:

- https://spark.apache.org/docs/latest/api/python/getting_started/index.html - Getting Started with PySpark
- https://sparkbyexamples.com/pyspark-tutorial/ - PySpark Tutorial