

Lab - Databricks Setup

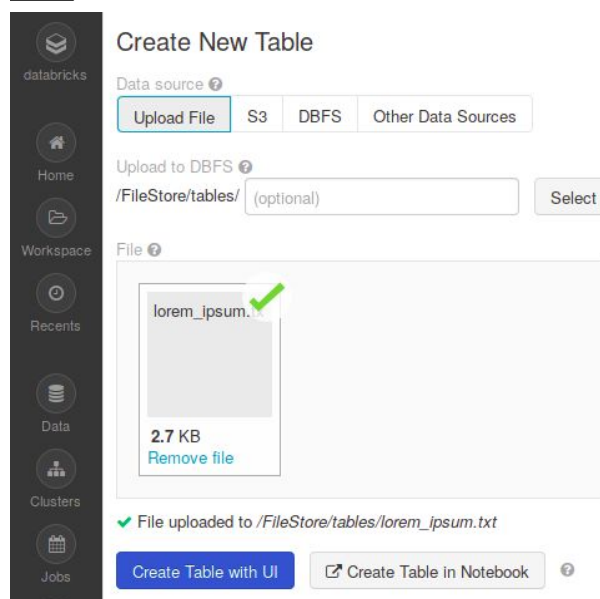
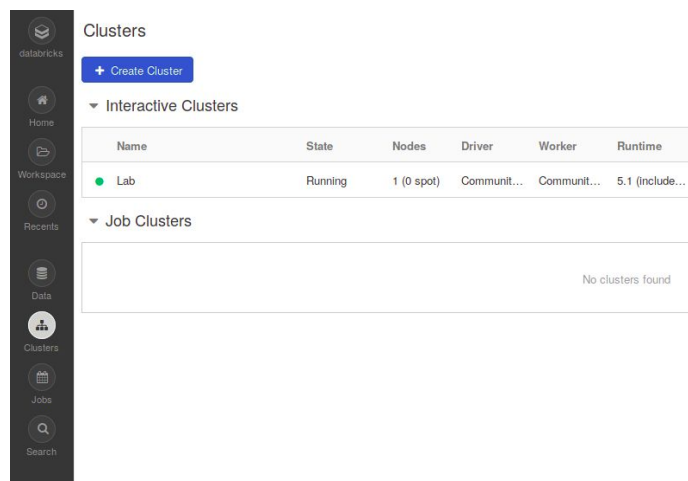
Objectives:

1. Running ML algorithms in Spark.
2. Using scikit-learn to perform computation on driver node.
3. Using scikit-learn and Spark to perform computation in distributed setting.
4. Understanding distinction between embarrassingly parallel vs native parallel algorithms - scikit-learn vs Spark MLlib.

Instructions:

Databricks

1. Databricks account creation: Create a community edition account in databricks:
<https://databricks.com/try-databricks>
2. Login to the databricks community edition cloud
3. Click the "Clusters tab" and create a cluster



4. Click data and upload a new dataset file. Click "Create Table in Notebook"
5. The notebook should be imported to databricks
6. Open notebook and run python/scala code
7. One can view the spark jobs, DAG structure, executor node status, etc.. in the same UI



The screenshot shows a Databricks notebook interface. At the top, there's a tab labeled 'Cmd 3'. Below it, a code editor contains the following Scala code:

```
%scala
val df2 = spark.read.textFile("/FileStore/tables/lorem_ipsum.txt").as[String];

val wordCount = df2.flatMap(_.split(" "))
  .map(_._stripSuffix(".").stripSuffix(","))
  .groupByKey(word => word)
  .count
  .orderBy(org.apache.spark.sql.functions.col("count(1)").desc);

display(wordCount)
```

Below the code editor, there's a section for Spark Jobs. It shows two jobs: 'df2: org.apache.spark.sql.Dataset[String] = [value: string]' and 'wordCount: org.apache.spark.sql.Dataset[(String, Long)] = [value: string, count(1): long]'. Below the jobs, there's a table displaying the results of the 'wordCount' job:

value	count(1)
in	11
vitae	9
vel	7
eget	7
volutpat	7

While Amabri is popularly used Hadoop solution that offers a number of services (which we will use later on). This setup is great for running spark code in a cluster setup.

You don't have to worry about credits and can experiment with this setup easily.