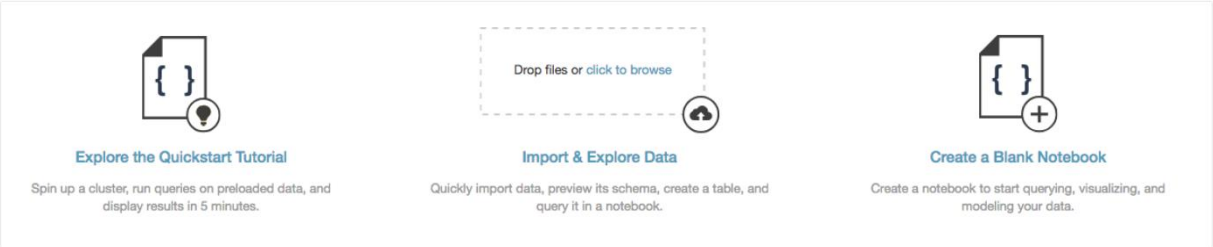
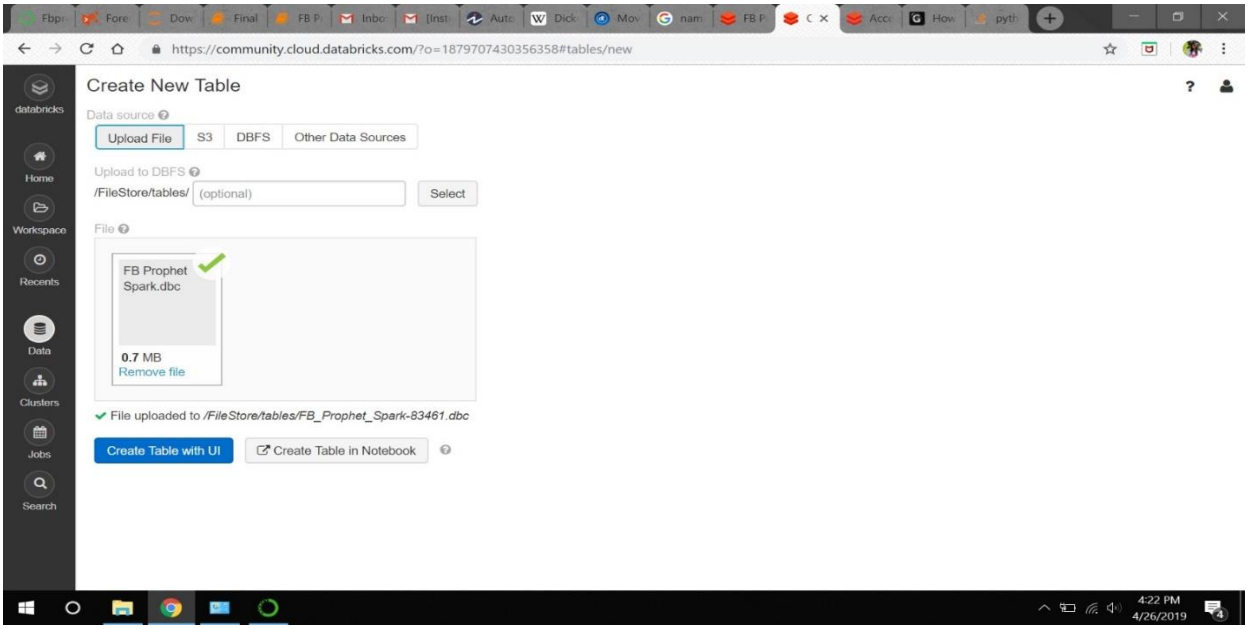
How to run the code in Databricks-

Databricks- Databricks is a managed platform for running Apache Spark and provides a host of features to help its users be more productive with Spark. It's a point and click platform for those that prefer a user interface. However, this UI is accompanied by a sophisticated API for those that want to automate aspects of their data workloads with automated jobs.

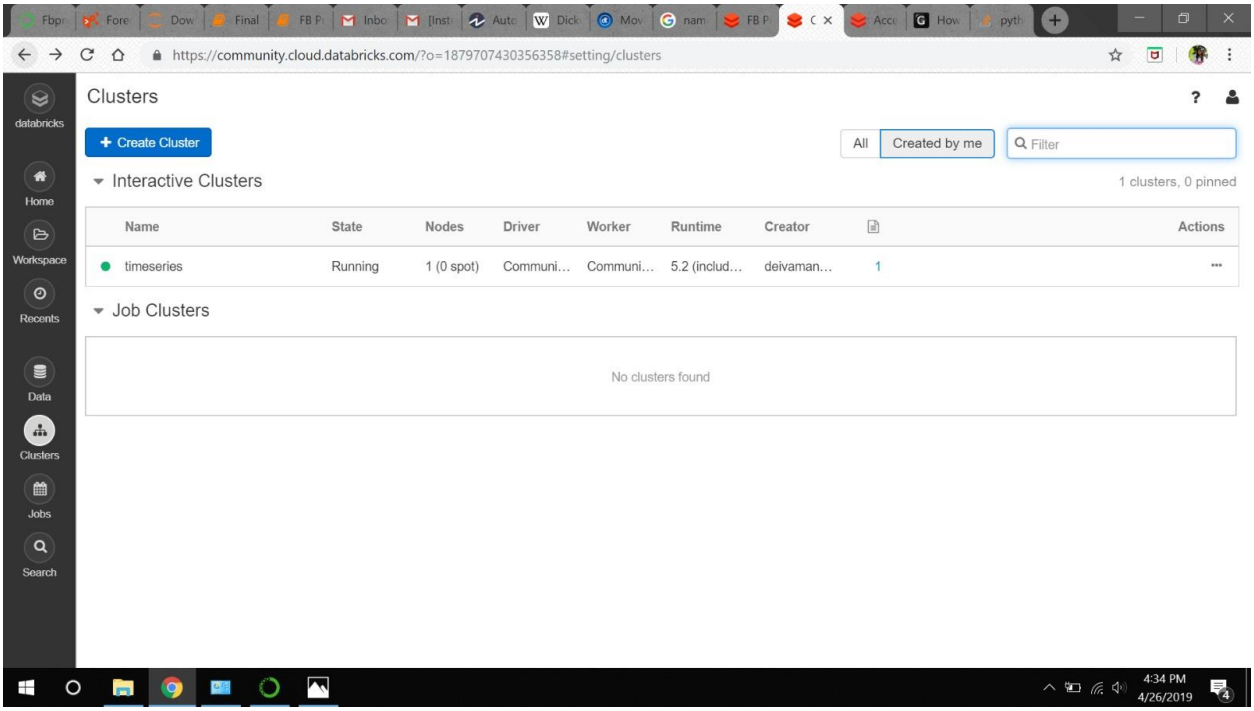
1. Import data- Drop files into or click to browse in Import & Explore data box on landing page.



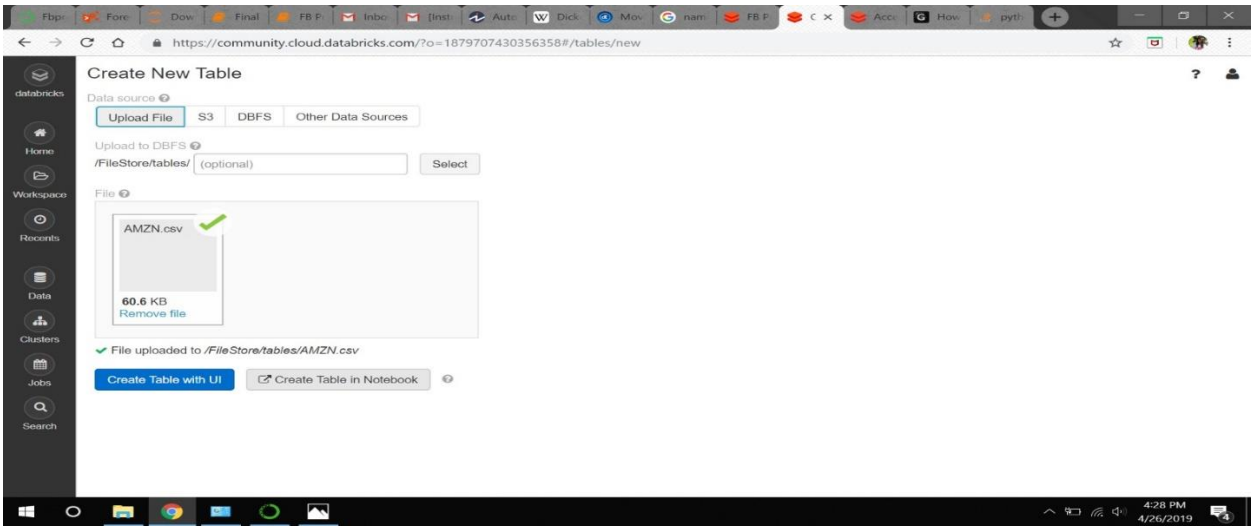


2. Creating Cluster:

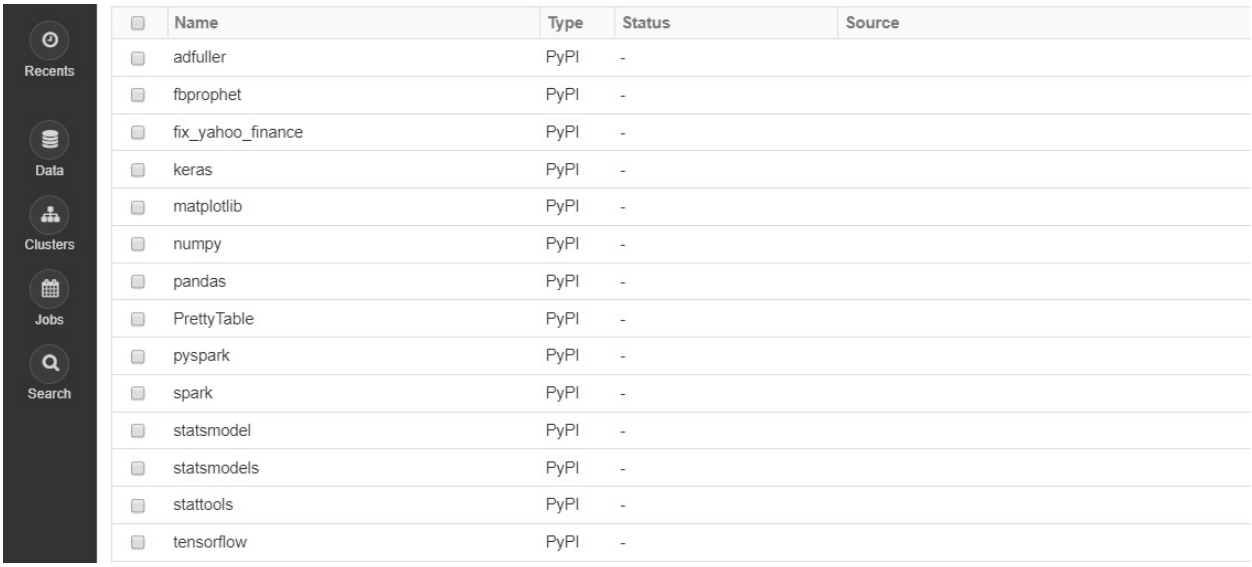
In Databricks you can create two different types of clusters: standard and high concurrency. Standard clusters are the default and can be used with Python, R, Scala, and SQL. High-concurrency clusters are tuned to provide the efficient resource utilization, isolation, security, and the best performance for sharing by multiple concurrently active users. High concurrency clusters support only SQL, Python, and R languages.



3 Importing CSV file-

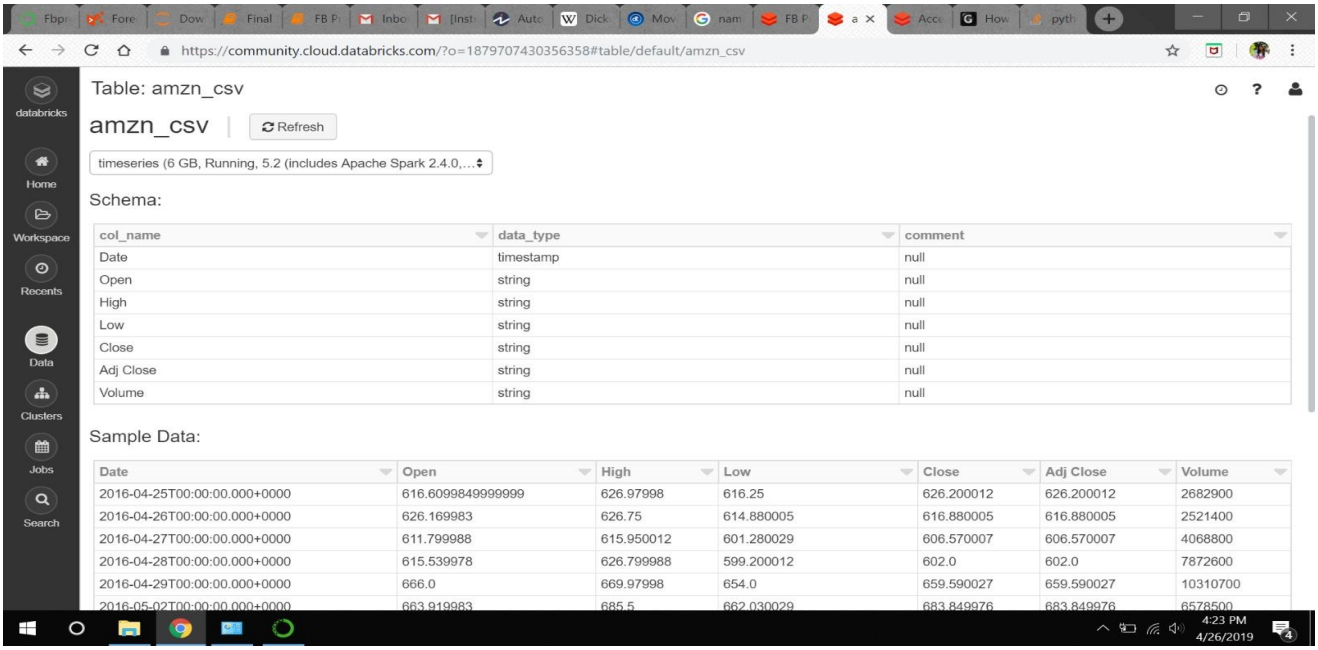


4. Importing Libraries- The below libraries are the required for our model. Steps to getting libraries installed- Select the running cluster which you have created earlier then click on the libraries you want to install for your model and done(required libraries installed)



Name	Type	Status	Source
adfuller	PyPI	-	
fbprophet	PyPI	-	
fix_yahoo_finance	PyPI	-	
keras	PyPI	-	
matplotlib	PyPI	-	
numpy	PyPI	-	
pandas	PyPI	-	
PrettyTable	PyPI	-	
pyspark	PyPI	-	
spark	PyPI	-	
statsmodel	PyPI	-	
statsmodels	PyPI	-	
stattools	PyPI	-	
tensorflow	PyPI	-	

5. View databases and tables- Click on the sidebar. Databricks selects any running cluster to which you have access. The Databases folder displays the list of databases with the default database selected. The Tables folder displays the list of tables in the default database.



col_name	data_type	comment
Date	timestamp	null
Open	string	null
High	string	null
Low	string	null
Close	string	null
Adj Close	string	null
Volume	string	null

Date	Open	High	Low	Close	Adj Close	Volume
2016-04-25T00:00:00.000+0000	616.6099849999999	626.97998	616.25	626.200012	626.200012	2682900
2016-04-26T00:00:00.000+0000	626.169983	626.75	614.880005	616.880005	616.880005	2521400
2016-04-27T00:00:00.000+0000	611.799988	615.950012	601.280029	606.570007	606.570007	4068800
2016-04-28T00:00:00.000+0000	615.539978	626.799988	599.200012	602.0	602.0	7872600
2016-04-29T00:00:00.000+0000	666.0	669.97998	654.0	659.590027	659.590027	10310700
2016-05-02T00:00:00.000+0000	663.919983	685.5	662.030029	683.849976	683.849976	6578500

6. Notebook in Databricks

Cmd 40

Tuning hyper parameters for ARIMA model

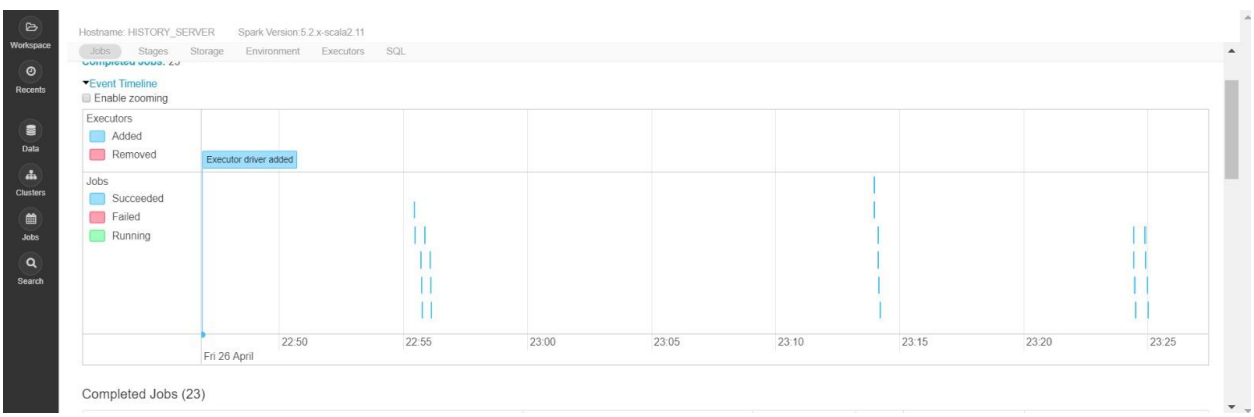
We are evaluating combinations of p, d and q values for ARIMA model to find the best combination of p, d, q values for our model

Cmd 41

```python
import warnings
from pandas import Series
from statsmodels.tsa.arima_model import ARIMA
from sklearn.metrics import mean_squared_error
from math import sqrt

order=(2,1,1)
# evaluate an ARIMA model for a given order (p,d,q) and return RMSE
def evaluate_arima_model(X, arima_order):
    # prepare training dataset
    X = X.astype('float32')
    train_size = int(len(X) * 0.50)
    train, test = X[0:train_size], X[train_size:]
    history = [x for x in train]
    # make predictions
    predictions = list()
    for t in range(len(test)):
        model = ARIMA(history, order=arima_order)
        # model_fit = model.fit(disp=0)
        model_fit = model.fit(trend='nc', disp=0)
        yhat = model_fit.forecast()[0]
```

7. Spark UI



8. Computational Time

SQL

Completed Queries

ID	Description		Submitted	Duration	Jobs
30	collectResult at OutputAggregator.scala:132	+details	2019/04/26 23:28:20	3 ms	
29	toDF at CustomDisplayTypes.scala:77	+details	2019/04/26 23:28:20	0 ms	
28	collectResult at OutputAggregator.scala:132	+details	2019/04/26 23:28:20	17 ms	
27	toDF at CustomDisplayTypes.scala:77	+details	2019/04/26 23:28:20	0 ms	
25	sql at SQLDriverLocal.scala:87	+details	2019/04/26 23:28:20	27 ms	
26	sql at SQLDriverLocal.scala:87	+details	2019/04/26 23:28:20	9 ms	
24	toPandas at <command-2828119668611524>:1	+details	2019/04/26 23:24:59	0.1 s	22
23	collectResult at OutputAggregator.scala:132	+details	2019/04/26 23:24:51	0.2 s	20
22	load at NativeMethodAccessorImpl.java:0	+details	2019/04/26 23:24:51	0.2 s	19
21	toPandas at <command-2828119668611524>:1	+details	2019/04/26 23:24:28	0.1 s	18
20	collectResult at OutputAggregator.scala:132	+details	2019/04/26 23:24:23	0.2 s	16
19	load at NativeMethodAccessorImpl.java:0	+details	2019/04/26 23:24:23	0.2 s	15
18	toPandas at <command-2828119668611524>:1	+details	2019/04/26 23:14:11	0.2 s	14