Yelp Data Analysis using Databricks

Spark MLlib Hands-On Lab:

## Overview

In the lab, you will practice spark MLlib (Naïve Bayes Classifier) to perform predictive analysis on yelp business data.

## Objectives

In this hands-on lab you will learn how to:

* Upload the dataset in Databricks File System(DBFS)
* Load Data & Data Preprocessing
* Explore Data using Spark SQL queries
* Create a multiclass Naive Bayes Classifier and Evaluation
* Analyze the result

## Prerequisites

The following are required to complete this hands-on lab:

* Databricks community Edition account
* A provisioned Apache Spark cluster
* A web browser
* Python notebook
* Sample data from yelp Dataset (Here we are taking Yelp-Business dataset for Arizona and Nevada state)

## Exercises

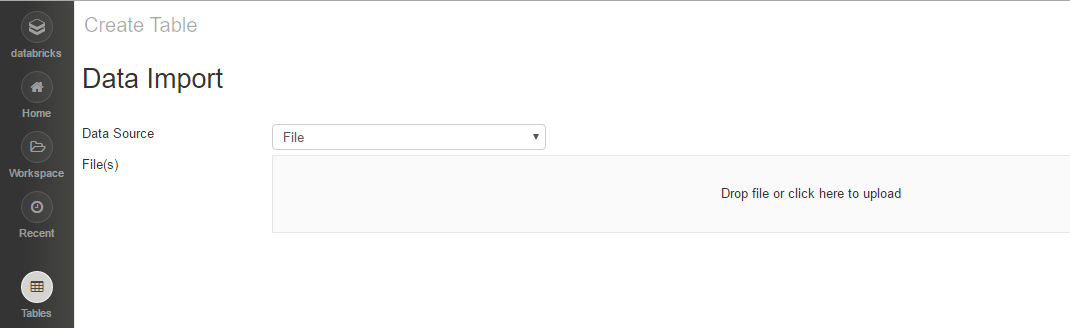
This hands-on lab includes the following exercises:

* Exercise 1: Upload a dataset file in the Databricks file system (DBFS).
* Exercise 2: Load and Preprocessing of data.
* Exercise 3: Explore the Data
* Exercise 4: Create a Multiclass Naive Bayes Classifier
* Exercise 5: Experimenting with Various Smoothing Parameters

## Exercise 1: Upload a dataset in the Databricks file system (DBFS).

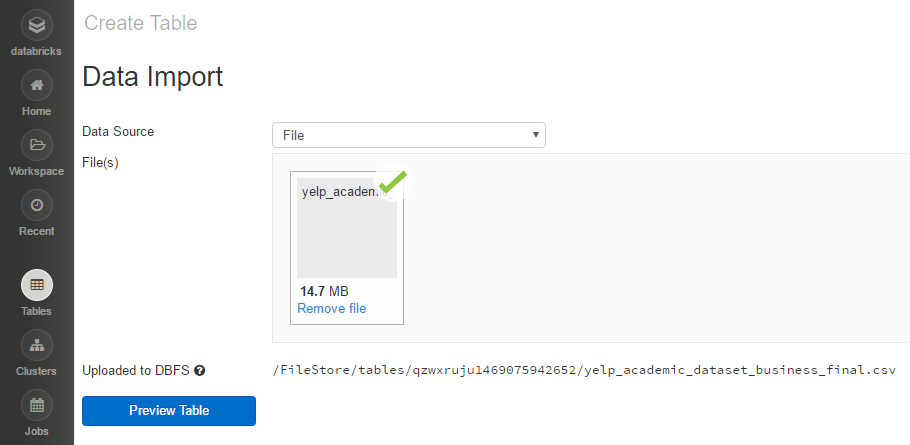
In this exercise, students will learn how to upload external dataset in the Databricks file system.

1. Access the create table tab on the left hand side in the Databricks home page.

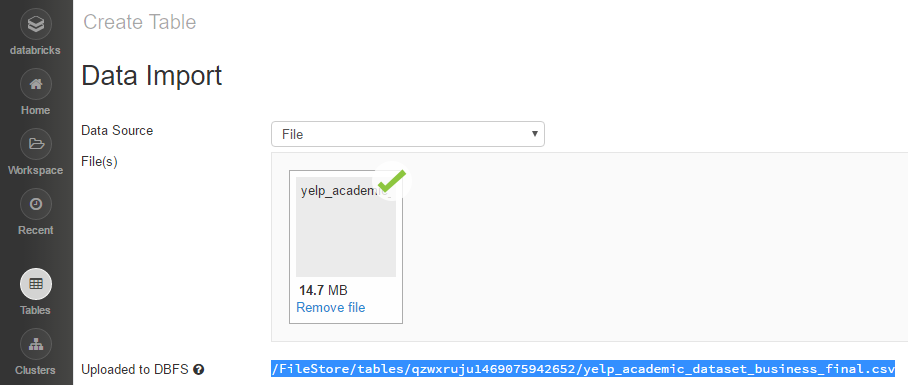


1. Select data to upload

From the above figure, either you can **upload file**, by clicking **Choose File** to select the data file from which to upload the data or you can drag and drop the file directly. Here I have uploaded the Yelp-Business csv file. For demo purpose we are uploading small file, otherwise we can upload any size.



1. Note down the location of uploaded file. We might need this in the later labs.



## Exercise 2: Load and Preprocessing of data.

Now in this Lab, we will be using the yelp dataset that is mounted on the DBFS in the above steps.

As shown below, open a python notebook, give it any suitable name and then provide the path copied before in the function dbutil.fs.ls.



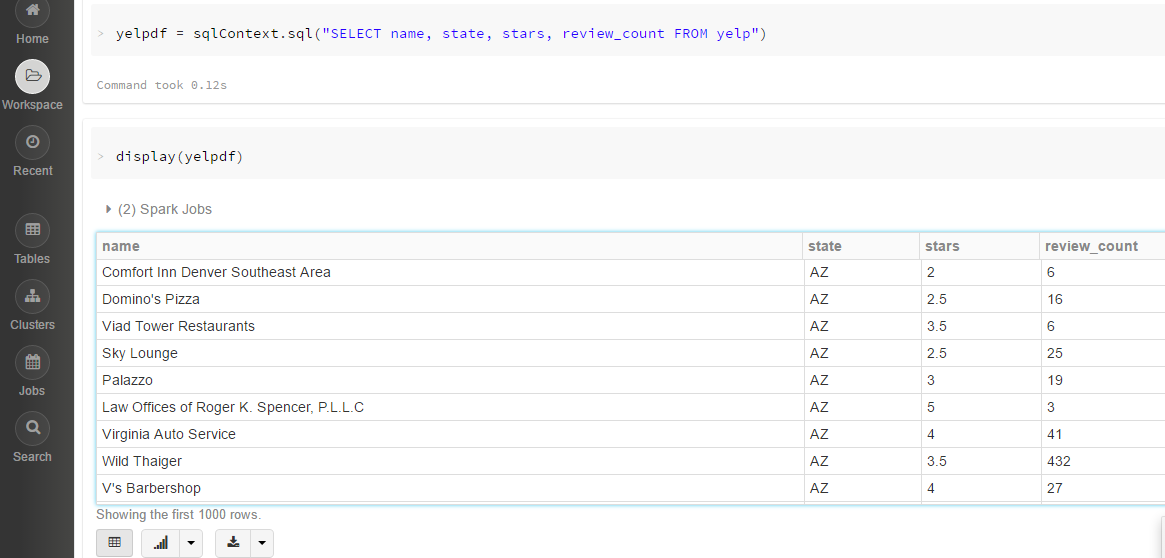
It will run a spark job and display the file uploaded with the size in bytes.

Read in the dataset using the spark-csv package. Note that the yelp dataset has dots in column names by default, such as "yelp.BusinessID" and "yelp.name". Spark-csv doesn't support dots in column names because it's usually the notation used for nested queries. To get by this, we will have to rename the columns when we read the data in using SQL

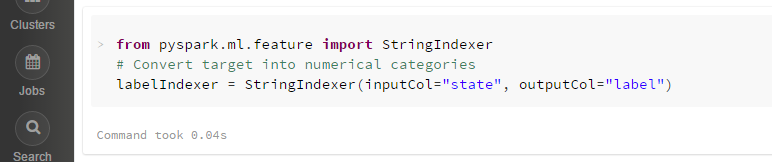




Since we do not need the first column of row indexes, we will only select the relevant columns that we need and convert it into a Data Frame. Here we have selected columns like name, review count, stars, state.

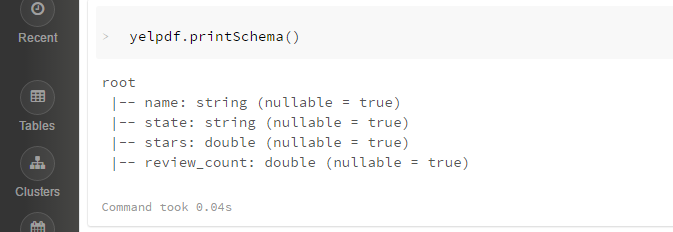


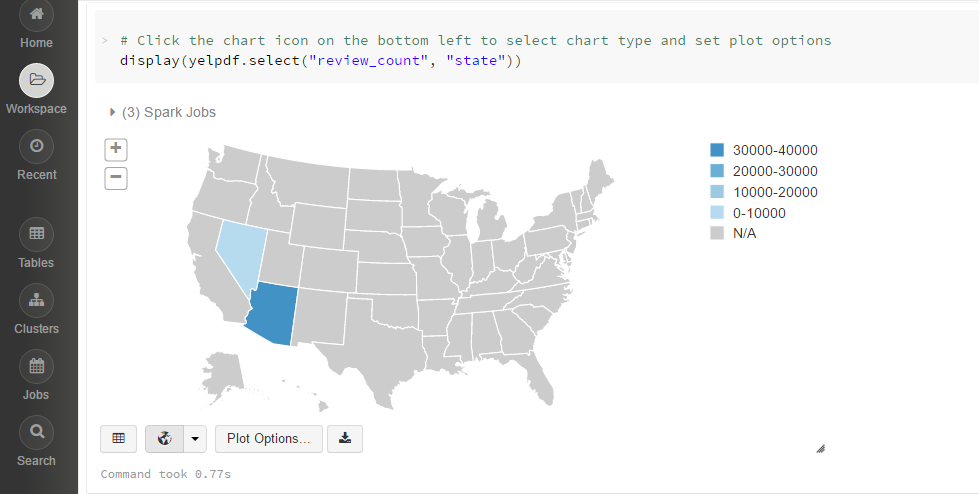
Since we have already removed the row index column and separated the header from the rest of the dataset, the next data preprocessing step we need to take is to convert our label into numerical categories. This can be easily done with the StringIndexer(). We won't transform the dataset just yet as we will pass the StringIndexer() into our ML Pipeline later.



## Exercise 3: Explore the data.

We can easily obtain some quick visualizations to better understand the data with the display() command.



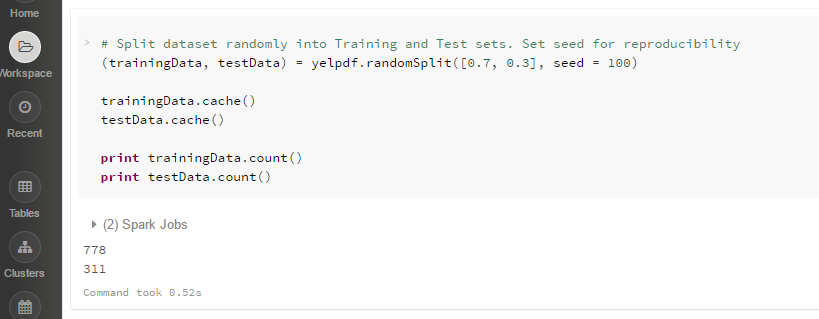


The above map suggests that Arizona is having more business review counts than Nevada.

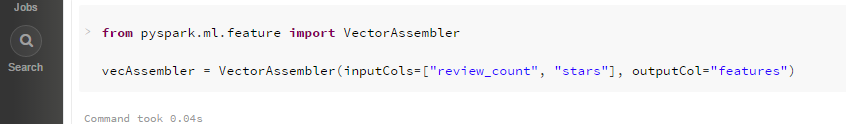
## Exercise 4: Create a Multiclass Naive Bayes Classifier.

In this tutorial, we will be demonstrating the use of the ML Pipeline API.

To proceed, we will first randomly split the dataset into training set (70%) and test set (30%). The training set will be used to build our models, and the test set will be used to evaluate models. We cache the datasets as we will be using them multiple times in this tutorial.

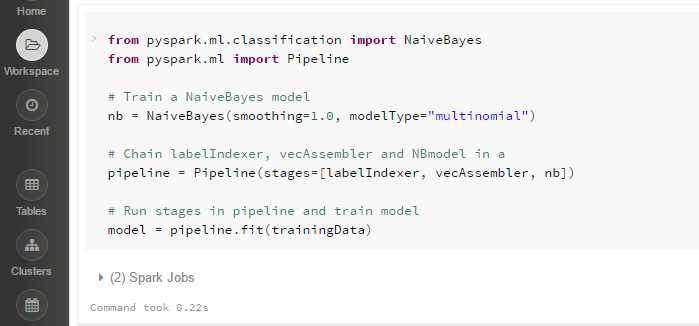


Next, we will use the VectorAssembler () to merge our feature columns into a single vector column, which we will be passing into our Naive Bayes model. Again, we will not transform the dataset just yet as we will be passing the VectorAssembler into our ML Pipeline.

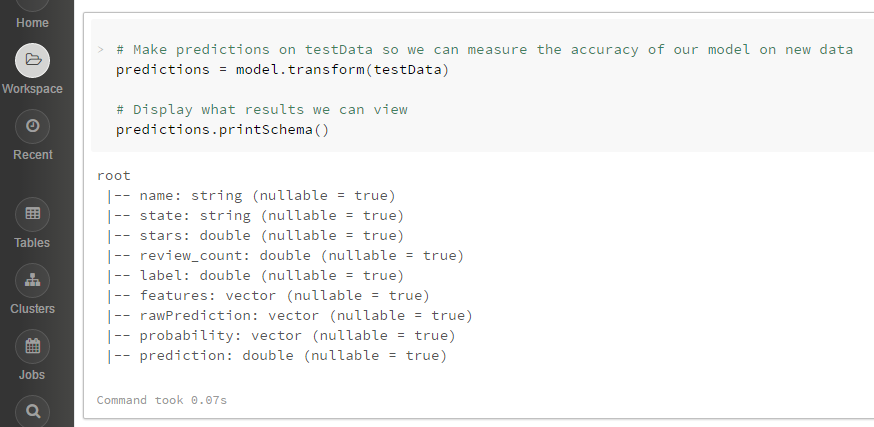


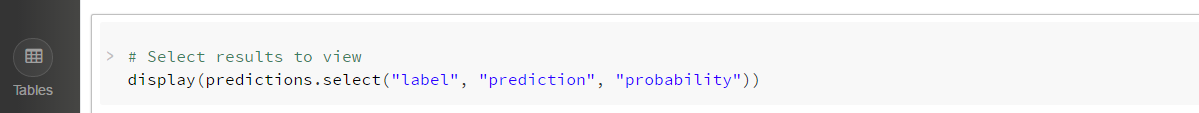
#### Create a Multiclass Naive Bayes Classifier

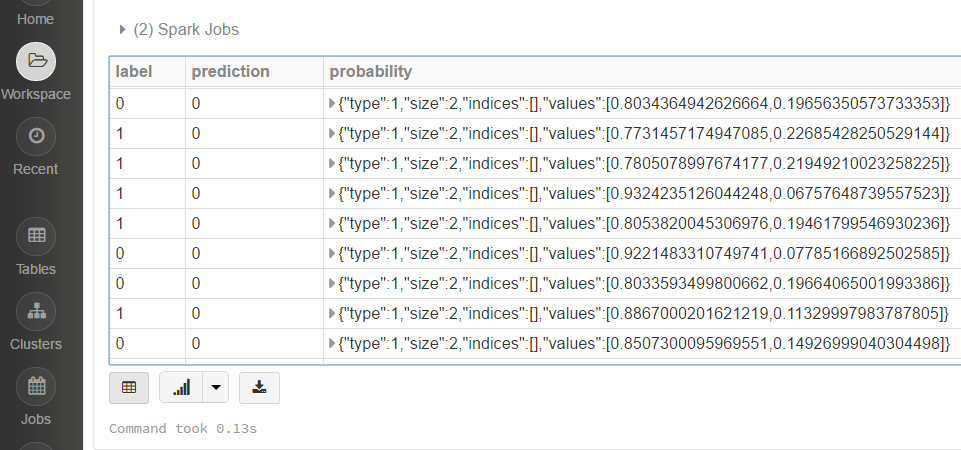
We will try to see how well Naive Bayes can predict the state of business using its 2 features – review\_count and stars. This is a multiclass problem as we have 2 different states using yelp in our dataset. Keep in mind that the Naive Bayes algorithm assumes independence between features, and requires your features to take on non-negative values.



We can now make predictions from our model and view results.







#### Model Evaluation

To evaluate our model, we will be making use of the Evaluator in MulticlassClassification. Note that f1-score is the default metric for the MulticlassClassificationEvaluator. There are other choices for evaluations metrics that can be found in the API.



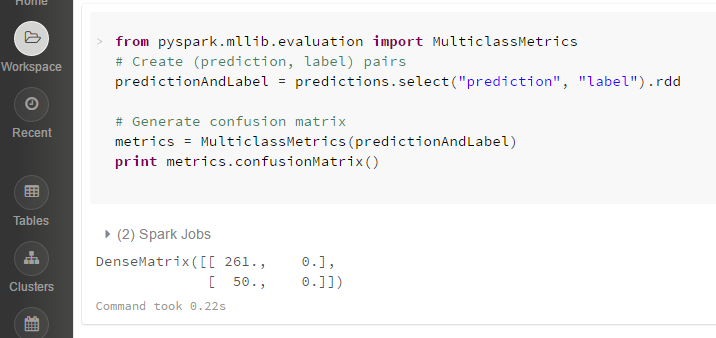
The Evaluator is able to use a few metrics such as f1-score, precision, recall, weightedPrecision and weightedRecall.

evaluator.setMetricName ("insert\_metric\_here") can be used to change the metric used to evaluate models.

We can also generate a Confusion Matrix to see the results of the predictions better. ConfusionMatrix () works only with RDDs, so we will have to convert our DataFrame of (prediction, label) into a RDD.

confusionMatrix() returns a DenseMatrix with the columns representing the predicted class ordered by ascending class label, and each row represents the actual class ordered by ascending class label. The diagonal from top left to bottom right represents the observations that were predicted correctly.

From the above confusion matrix, we observe that all Arizona (class 0) and Nevada (class 1) have been classified correctly.

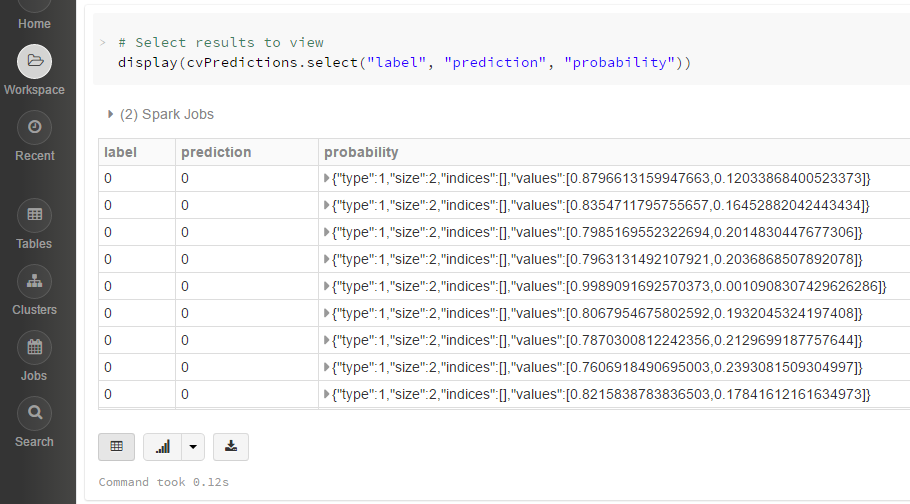


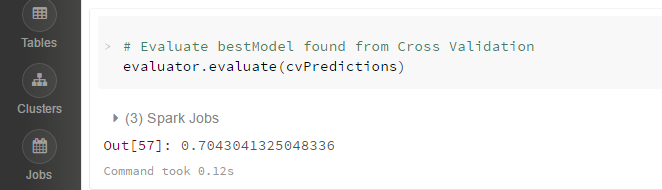
#### Experimenting with Various Smoothing Parameters

We can experiment with various smoothing parameters to see which returns the best result. This is easily done with the ParamGridBuilder and CrossValidator.

As we indicate 6 values for the smoothing parameter, this grid will provide 6 parameter settings for CrossValidator to model, evaluate and choose from.







Turns out that smoothing has no effect on this dataset!

## Summary

In this hands-on lab, you learned how to:

* Upload the dataset in the DBFS
* Use spark SQL to explore the data
* Use Naïve Bayes classifier to predict using the features taken into consideration.