**DIC LAB – 3**

Name: Rajiv Ranjan Ubid: 50249099

Partner Name: Pradeep Aitha

**Folder Structure provided: ranjanLab3.tar**

Inside it 3 folders and a file is there:

1. **Pre-Lab**: Contains the execution of pre-lab part of this assignment which is running the given word-count program on the the provided **doc.txt** file.

The script is in **wordcount.py**

The execution has been clearly shown in a Jupyter notebook file **word\_count.ipynb**

1. **Part1 Folder**: Titanic dataset execution:

The contents 3 files **train.csv**(training dataset) , **test.csv**(testing dataset) and **Titanicdataanalysis.ipynb.**

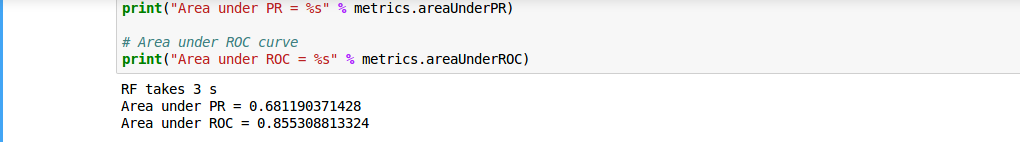
The details of the execution is below:

For the execution on the titanic dataset we have used the random Forest classifier.

It has been done in 3 ways:

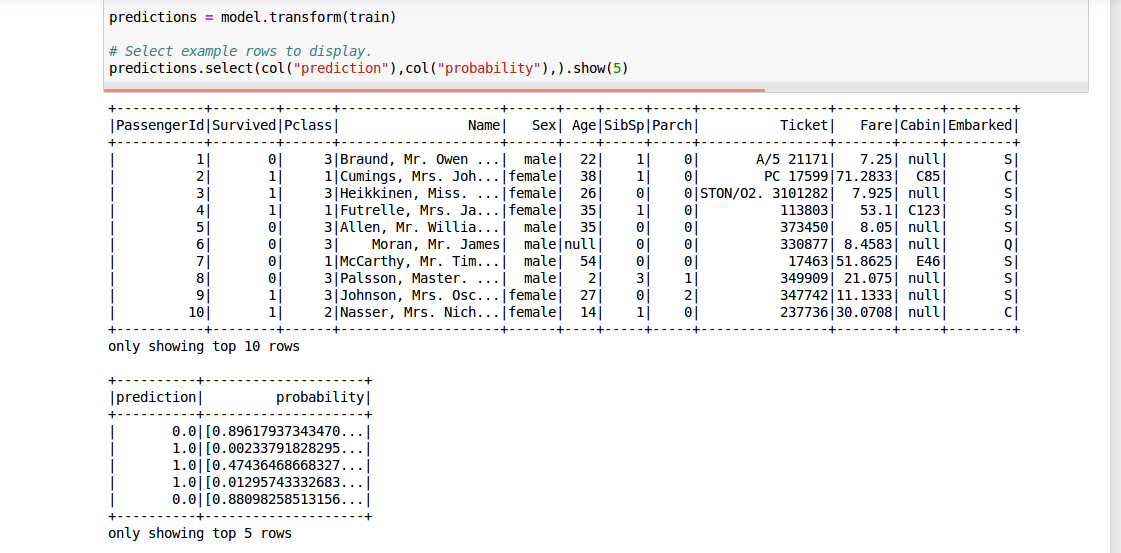
1. Using RDDs and MLLib module

**Screenshot of o/p:**



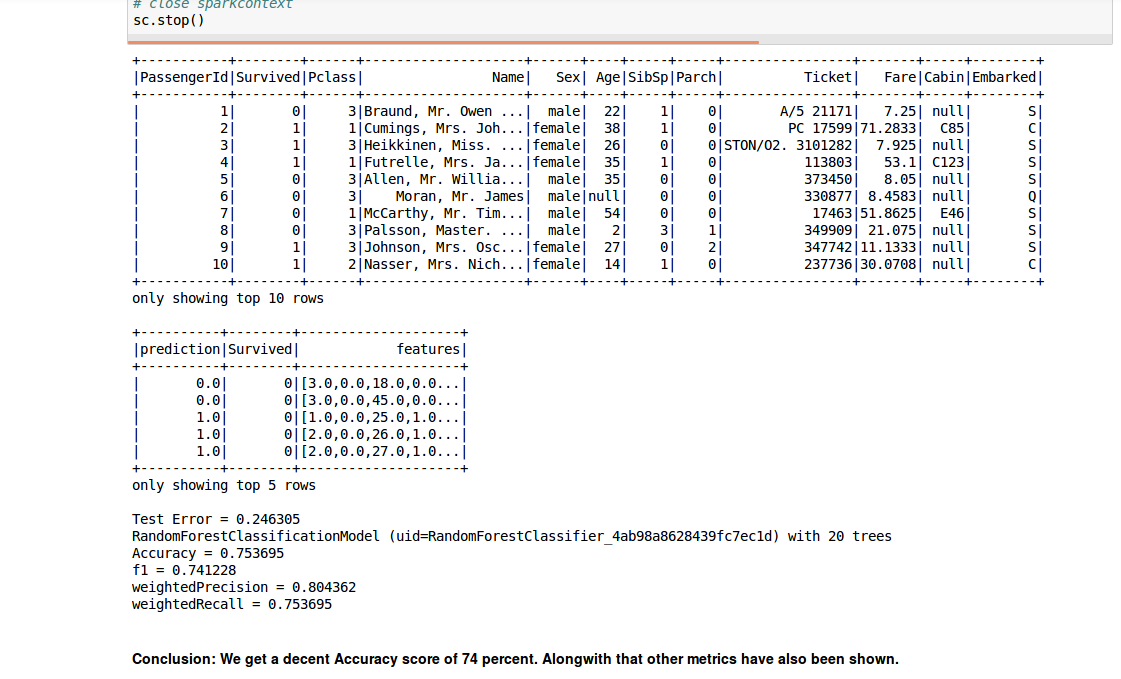
1. Using SQL dataframe and Ml module without using pipelines

**Screenshot of o/p:**

****

1. Using SQL dataframe and ML module using pipelines

**Screenshot of o/p:**

****

**3.** **article-classification Folder:**

**Contains 4 folders: Business, Music, Politics, Sports.** Now each of these folders contain 50 articles on the topic. Also we have 4 more files:

1) fetch**-**articles.py: used to fetch the articles

2) write\_csv.py: converts all the articles (all of them in a file name article.csv)

3)article.csv: all the articles accumulated in a csv file along with the category of that article

4)Classification.ipynb: Execution of multiclass document classification.

Details of the execution are below.

4)Readme.doc: Details of the lab. The execution and the other details are below.

5)unknown.csv: containing the unknown new data set

1. Readme file containing your name (First and last) and your partner’s name at the top with all the other details such as environment you have chosen etc. how to run your program and explanation of the output.

Ans: The name of partner has been provided at the top.

**Environment Chosen:**

For the execution of this lab following set-up was done:

1. Operating System: Ubuntu 16.04
2. Sypder3 for running the article collection code. It is the same script that was used in Lab-2. The script was written in python 3.
3. Feature Engineering was done using pyspark.
4. Spark version used: Spark-2.0.1
5. Scala version: 2.11.6
6. The code for Feature Engineering, Multi-class Classification has been run on Jupyter notebook.

The process in detail is summarized below:

1. Collect articles using NYtimes api. A total of 200 articles were 50 for each of the 4 categories:
2. Sports
3. Music
4. Business
5. Politics

Execution details and output of the execution steps:

**Data Collection phase:**

1. The script used for collecting data is in fetch\_articles.py
2. It is the same script that was used in the DIC lab-2 for collecting articles.
3. It uses beautiful soups for collecting data.

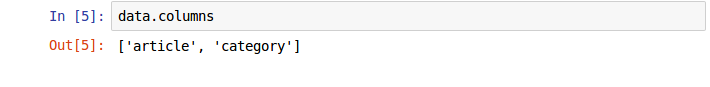
**Data Ingestion Phase:**

Data was ingested into the spark system in .csv format. For this we have used a script which has been provided **write\_csv.py**

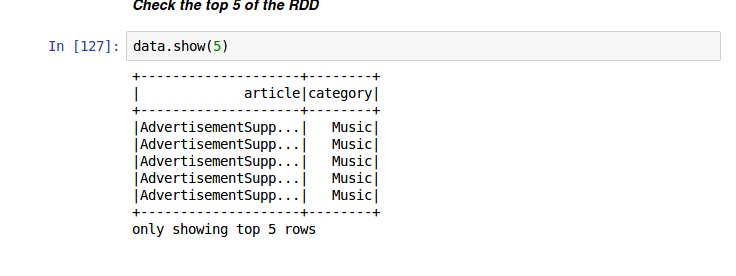
The output of this in a format where there are 2 columns one column of article containing the whole article and the other column is category column.

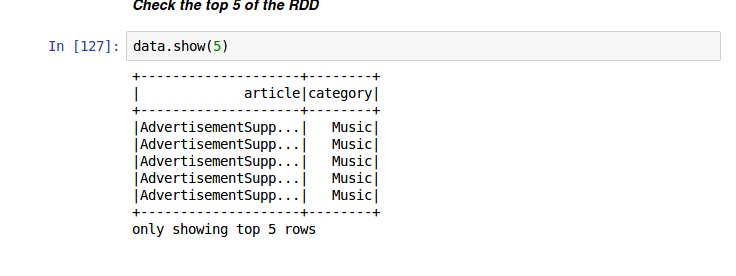
Now we create an instance of sqlContext to read the data. The data is read in the form of a dataframe.

The columns of the dataframe is as follows:

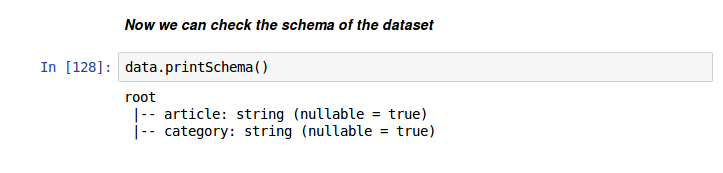


A sample of the dataframe is as follows:

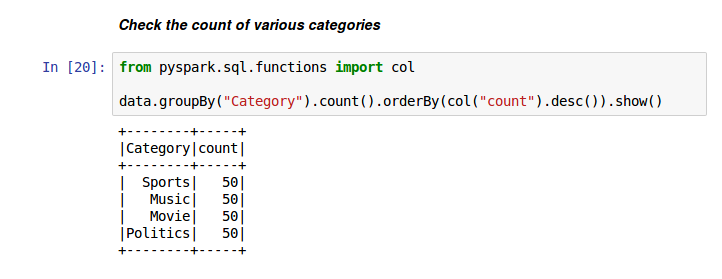




The schema of the dataframe is as follows:



The count of various categories are as follows:



**Data Processing phase:**

The data has be to passed through a series of steps before it could be fed to our classifier. All of these steps have been achieved through the creation of a pipeline:

HashingTF

Remove stop words

Model

Model

Naïve Bayes

Logistic regression

OneHotEncoder--StringIndexer

Word2Vec—Document-term vectors

Tokenizer

DataFrame

Imp: The important thing to note here

Let’s see the pipeline created in detail:

**Tokenizer**: Tokenizer is used to create tokens of documents. It basically creates individual words out of it.

**Remove Stop words**: It removes the unknown unnecessary words.

**HashingTF:** creates features

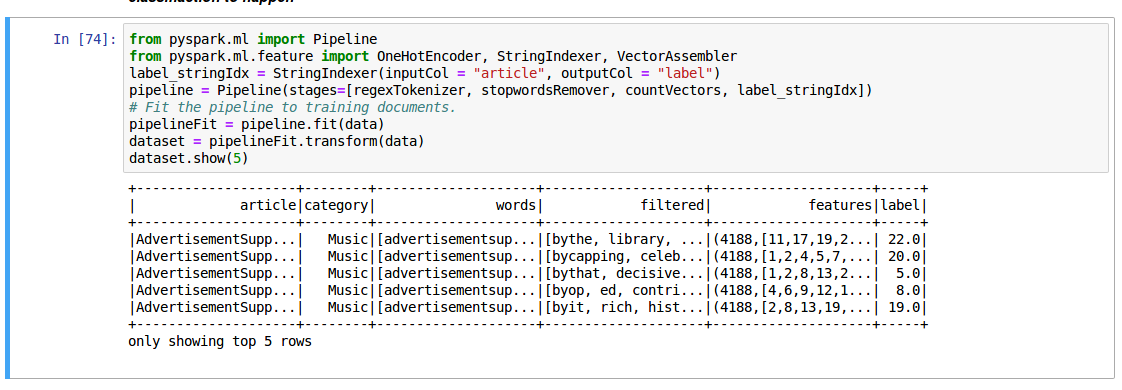
**Word2vec:** Count the vectors after the conversion in words to vector happens

**StringIndexer**: encodes a string column of labels to a column of label indices.

Till now we have extracted features from our data set.

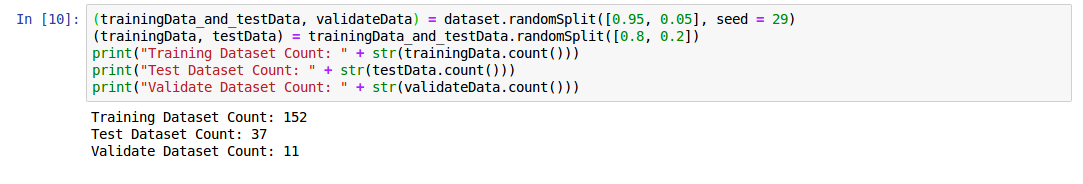
Now we will use our dataset for multi-class classification.

Let’s see a sample of the dataset first:



We have divided the whole dataset in 3 parts:

1. Training set
2. Testing set
3. Validate set: This is unseen data to the model and can be used to verify the accuracy of model on unknown data.



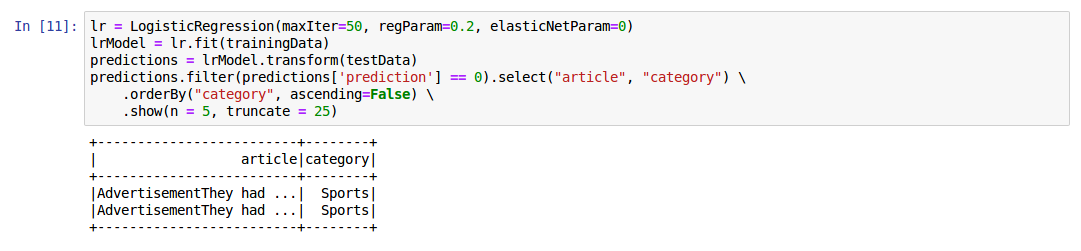
Two algorithms have been used for this.

1. Logistic regression:

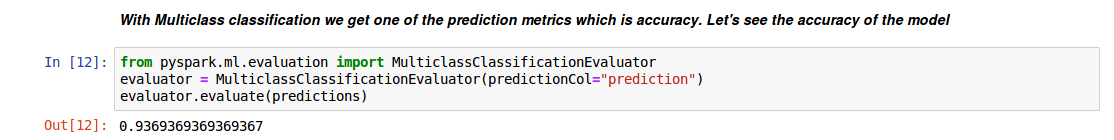
**Logistic regression** is a statistical method for analyzing a dataset in which there are one or more independent variables that determine an outcome. The outcome is measured with a dichotomous variable (in which there are only two possible outcomes).

We use our model on the training data and we see how it has performed on the testing data set.

The output is below:

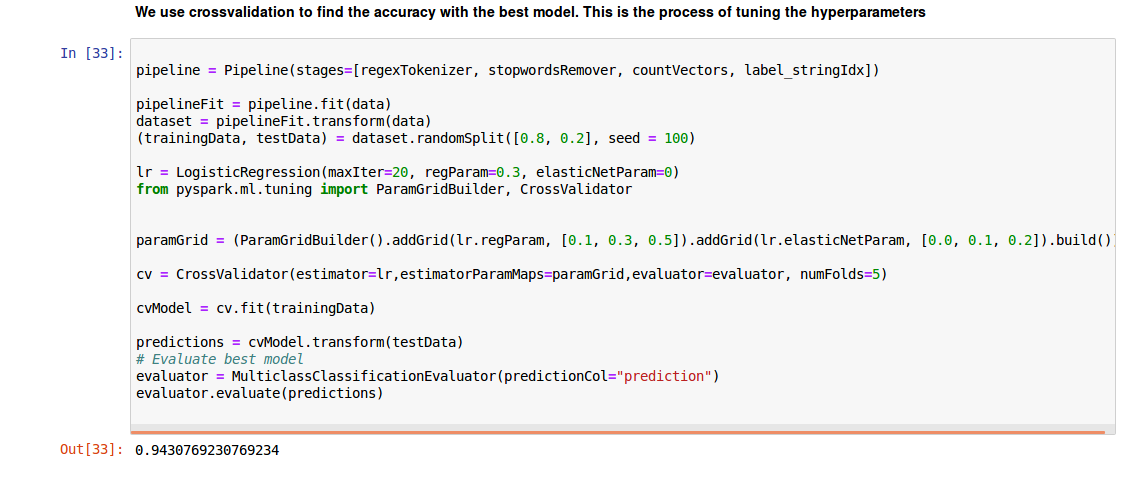


The accuracy is around 93 percent.



Using Cross-validation to tune the hyerparameters of the model. This will help us get the best model among the various models and then an accuracy score will be calculated on it. We see a slight improvement.

The accuracy is 94 percent.



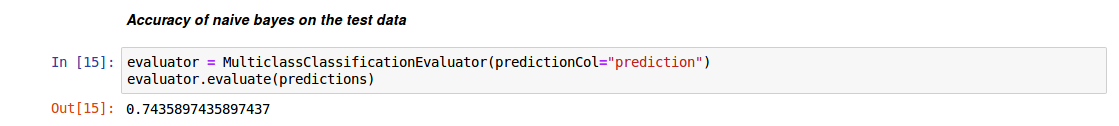
1. Naïve bayes: The second algorithm that we have used is as follows:

Naive Bayes classifiers are a family of simple "[probabilistic classifiers](https://en.wikipedia.org/wiki/Probabilistic_classifier)" based on applying [Bayes' theorem](https://en.wikipedia.org/wiki/Bayes%27_theorem) with strong (naive) [independence](https://en.wikipedia.org/wiki/Statistical_independence) assumptions between the features.

Let’s see the output of the model on the test data:



Let’s see the accuracy score which is around 74 percent:



**Last Phase : Testing the model on the new unknown dataset:**

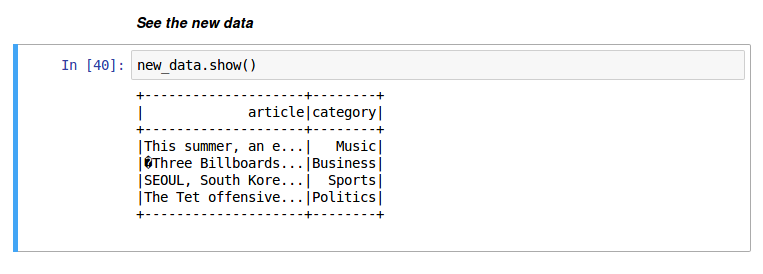
**We will do it in 2 ways:**

**1)Using entirely new data set**

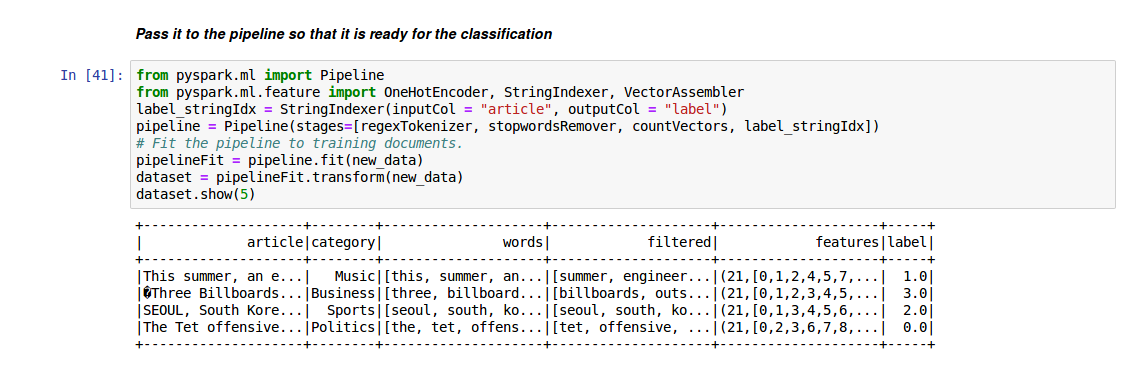
**2) Using the validate dataset that we had saved.**

**First method:**

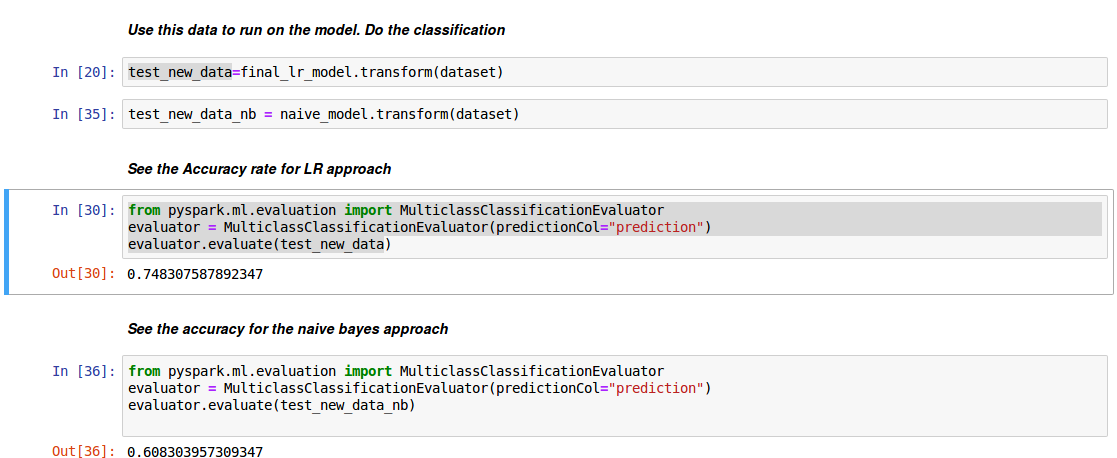
**Let’s check the data first:**

****

**Now we see that how the pipeline runs on the data set. The output is below:**

****

**The accuracy for both the classifiers is below:**

****

1. **Second method: Now we can see the second method with the validatedata. The accuracy score is below.**

