# (7082CEM)

# Coursework

Big Data Analytics and Visualization Using PySpark

MODULE LEADER: Dr. Marwan Fuad

Student Name: Johnson Michael

SID: 11432809

Online News Popularity Analysis using Pyspark

Docker

I can confirm that all work submitted is my own: Yes

Introduction

In this paper, I am going to explain and demonstrate how to set up and use Spark in python with Jupyter notebook user interface using Docker and Machine Learning techniques are implemented in this project. Furthermore, the visualization and results of this project are discussed in detail below. And we will look at how Docker is very much easy to set up in a machine without any environment dependencies and to create a docker image that will have all the required libraries and our Machine learning project in it. Which will help us to run our work on any machine without any dependencies.

In the realm of Big Data advanced analytics, Apache Spark is the most extensively used in-memory parallel distributed processing framework. The ease of use of its API, as well as a wide set of features spanning from SQL querying of the data lake to distributed training of complicated Machine Learning models utilising the most popular algorithms, are the key reasons for its success.

Docker has become widely used in a number of contexts. Docker allows users to establish basic environment specs, making it simple to create, ship, and scale applications. Users can also create Docker containers that can run simultaneously on a single server while keeping separated from one another. Docker provides an abstraction layer known as the Docker Engine that ensures interoperability between machines that can execute Docker, resolving the age-old problem of "it works on my system, but not on yours."

Apache Spark provides the analytics engine for crunching the numbers, while Docker provides a consistent environment and quick, scalable deployment. These two technologies save a lot of time and energy, especially in the Big data world.

The dataset for this project compiles a variety of characteristics from articles posted by Mashable over the course of two years. The idea is to predict how many will share the article on social media (popularity). Mashable (www.mashable.com) published the articles, and they own the content as well as the rights to republish it. As a result, while this dataset does not contain the original content, it does have some statistics related to it. Using the specified urls, the original content can be accessed and obtained by the general public.

Implementation

In this section, I will explain how Pyspark along with Jupyter notebook is implemented using Docker image. Installing Docker in Linux machine and pulling the required Docker image from the Docker repository are explained detail in this process. All of the codes that will be used will be described in depth, including their functions and value within the project.

SPARK

Spark can work with incredibly huge datasets over a cluster of servers because it is distributed. Spark is available in Scala, Java, Python, and R, among other languages. This sounded like a significant success because it doesn't limit options to a single language. Spark has its own DSL (Domain Specific Language) that is consistent across all implementations, implying that there is a shared language regardless of implementation language. But in this project Python is used to work with Apache Spark. The JupyterLab IDE, the Spark master node, and two Spark workers nodes are the four major components of the cluster. The user connects to the master node and issues Spark commands via the Jupyter notebooks' user interface. The master node receives the data and distributes it to the workers, who then relay the results back to the IDE. Cluster is not used in this project, only local node is used.

PYSPARK

Pyspark is a Big Data collaboration between Apache Spark and Python. Python is a high-level programming language, while Apache Spark is an open-source cluster-computing platform for large-scale data processing written in Scala and designed at UC Berkeley's AMP Lab. Spark was originally created in Scala, however owing to industry adoption, its Framework PySpark was converted to Python using Py4J. It's a Java library included with PySpark that allows Python to communicate with JVM objects dynamically; as a result, in addition to Python and Apache Spark, you'll need Java to run PySpark.

DOCKER

To overcome the above JAVA dependencies docker is used. It allows you a means to package things and distribute in a fairly portable fashion. Docker is the only dependency I need to install on my own PC machine.

*Pyspark docker image*

Docker container

Client

Spark kernal

Jupyter

notebook

Docker Installation

It's possible that the Docker installation package in the official Ubuntu repository isn't the most recent version. We'll instal Docker from the official Docker repository to guarantee we obtain the most recent version. To do so, we'll create a new package source, add the Docker GPG key to verify the downloads, and then install the package. To install docker,

Step 1:

Update your existing list of packages:



Step 2:

Install a few prerequisite packages



Step 3:

Add the GPG key for the official Docker repository



Step 4:

Docker repository to APT



Step 5:

Update the package



Step 6:

Install from the Docker repo



Step 7:

Docker installed, the daemon started. Check status



Pulling Docker Image

After install the docker the required image has to be pulled from the docker repository. To work with this project an image with Jupyter notebook and pyspark is pulled from the Docker repository. Docker image can be pulled with simple command as follows,

*Docker pull command*

*sudo docker pull jupyter/all-spark-notebook*

jupyter/all-spark-notebook image will be pulled to your machine and this image will have all the dependent files to run jupyter notebook and Pyspark in your machine. To view the image type this command in the terminal (sudo docker images). This will list all the available images in your machine if you have pulled earlier.

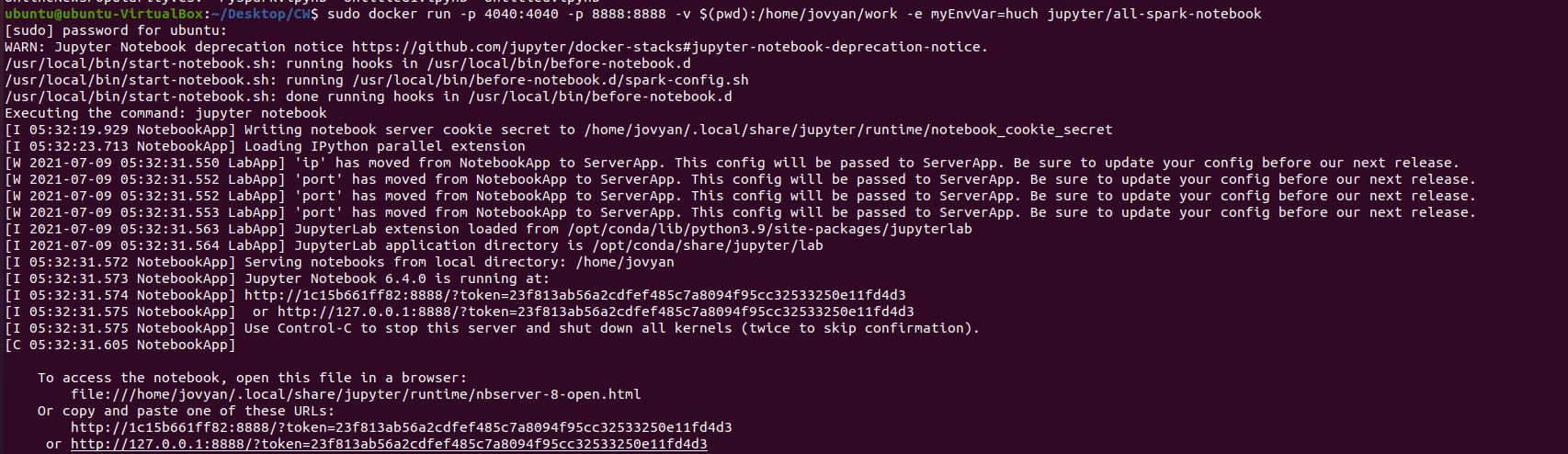
Persistent volume

Before running the jupyter/all-spark-notebook image in your machine we have to setup our volume. Volume is the one which will hold all the data (example: Saved Jupyter ipython file). If the volume was not configured then our data will be lost once the image is restarted. To avoid this conflict volumes are mapped. There are different volumes which we can use, but in this project local volume is used and mapped to the Docker image.

When we run the image the data in our local volume will be synced to the volume which is in the docker container. Therefore, if the image restarted or corrupted our data won’t be lost, it will be there in our local folder path.

Running Docker Image

Before running the docker command change the working directiory where you have your working file. Then enter the following command,



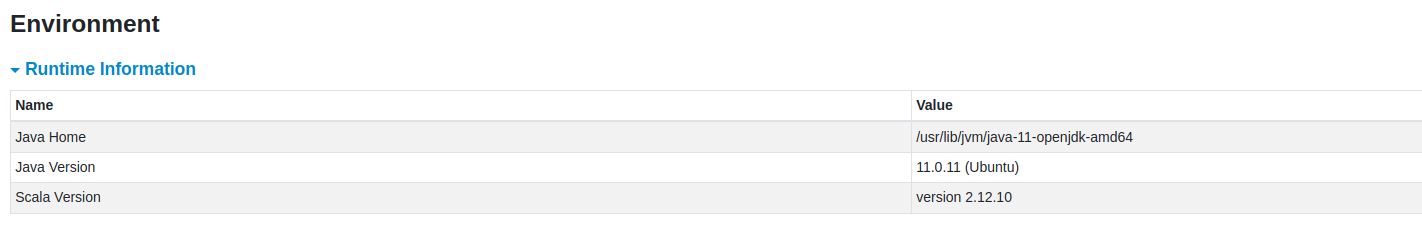
*sudo docker run -p 4040:4040 -p 8888:8888 -v $(pwd):/home/jovyan/work -e myEnvVar=huch jupyter/all-spark-notebook*

In the above command –p represent the port and Apache spark is running locally and exposed to the port 4040.

And jupyter notebook will be running in the port 8888 with access token in it. This access token will allow us to run the jupyter notebook in our localhost.

*-v $(pwd):* here –v is the volume which map our present working directory (pwd) to the docker container.

Version of JAVA and Scala running in the container

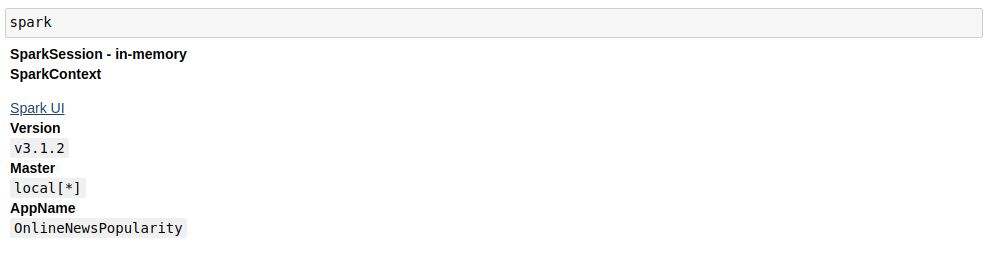


Jupyter notebook

After running the command we can access jupyter notebook using the access token. And due to the volume mapping our data in our local folder will be displayed in the jupyter notebook. Now our jupyter notebook will have all the dependent libraries in it. We just need to import the required package to this project and we can load our data.



SPARKSESSION



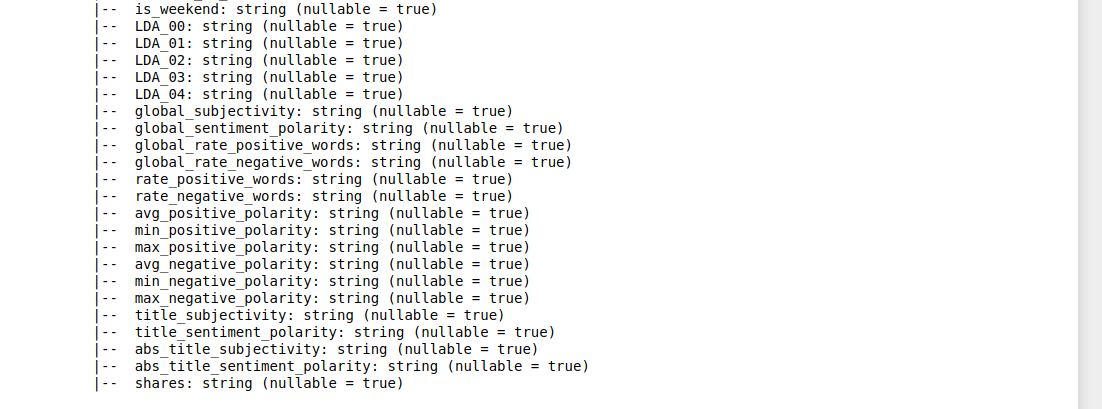
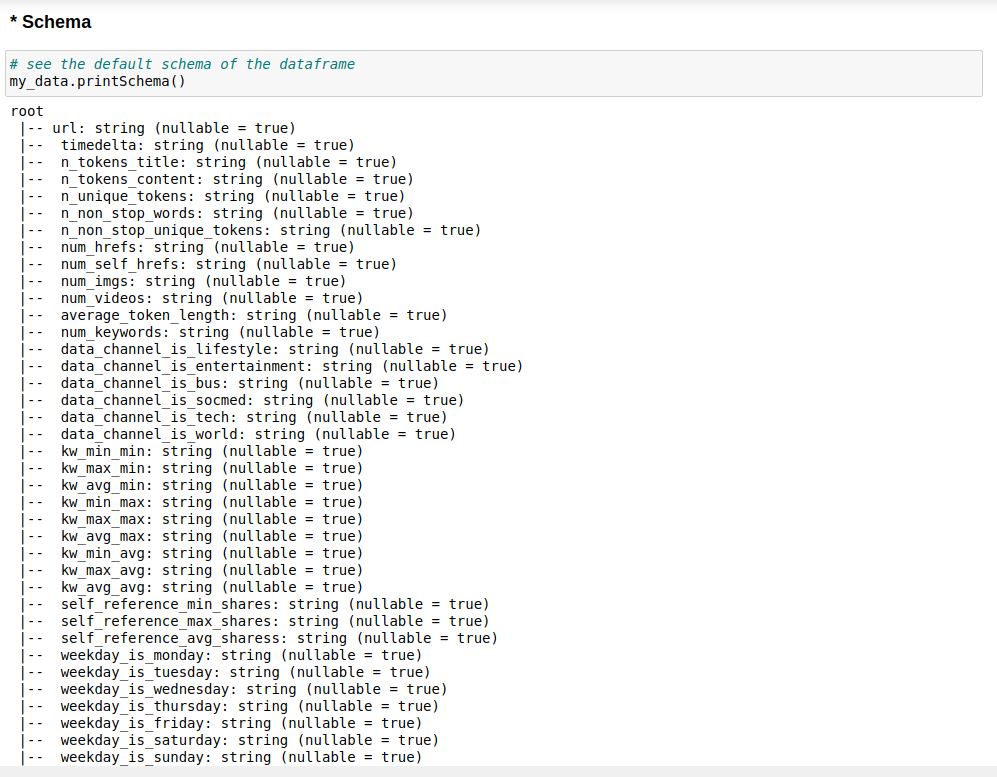
DATA SCHEMA

The dataset consist of 61 attributes (58 predictive attributes, 2 non-predictive and 1 target attribute). The 2 non-predictive attributes are (url and timedelta).

The attribute url consist of actual article link. Which is filtered to find the article name with top sharing score.

Timedelta is the days between the article publication and the dataset acquisition.

The Schema for 61 attributes are printed and show in the below image.



DATA DIMENSION

The dimension of the dataset is (39644, 61). It consist of 39644 instances and 61 attributes in it.

The command used to display dataset shape are given below.



SHOW

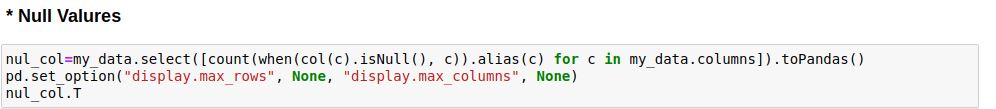


The .show() command will display the data for top 10 rows.

NULL VALUES

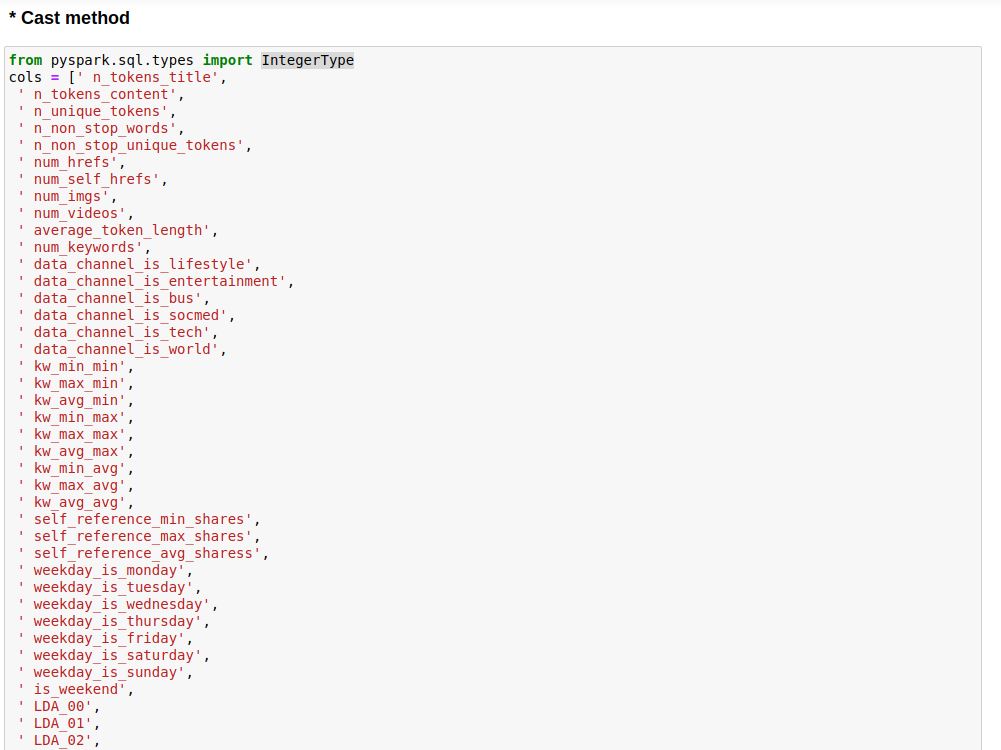
To find null values in the data select is used along with isNull() function. This helps us to find whether the data is having null or not and then imputation is done accordingly. In our data there is no null value in it.

The command used to find the null values are attached below

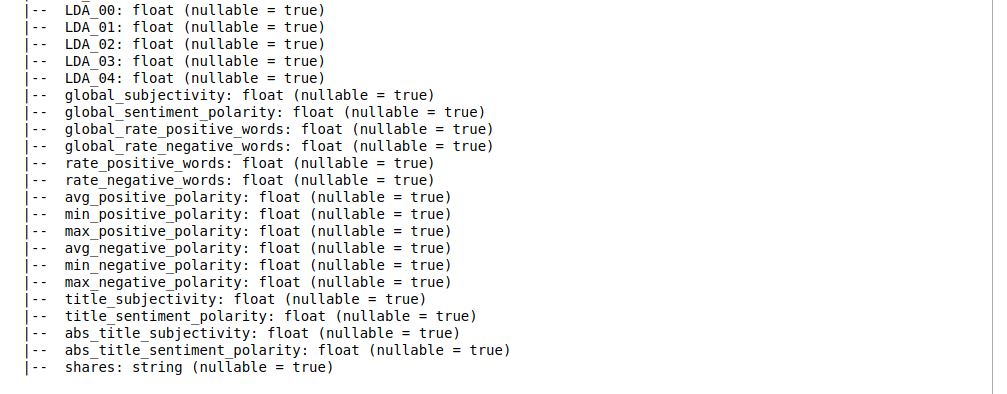


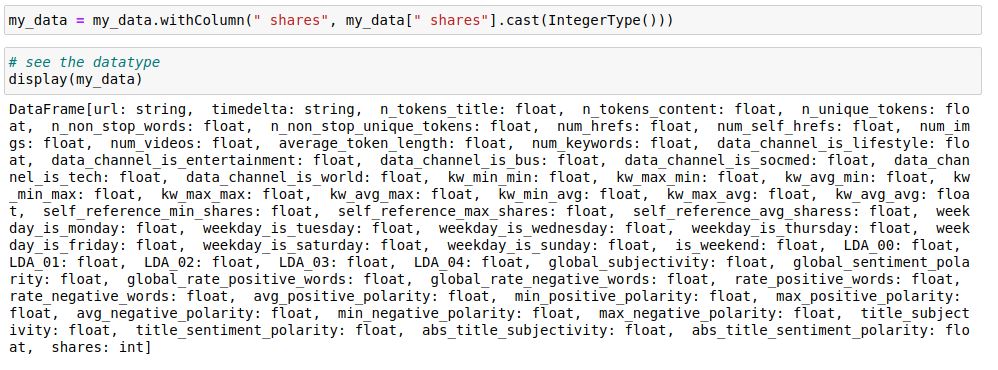
CAST

Cast is used to change the datatype of a column. In this dataset most of the columns are float and the target attribute is an integer so both float and interger casting is done to change the datatype.



All features except target variable is converted and displayed below.

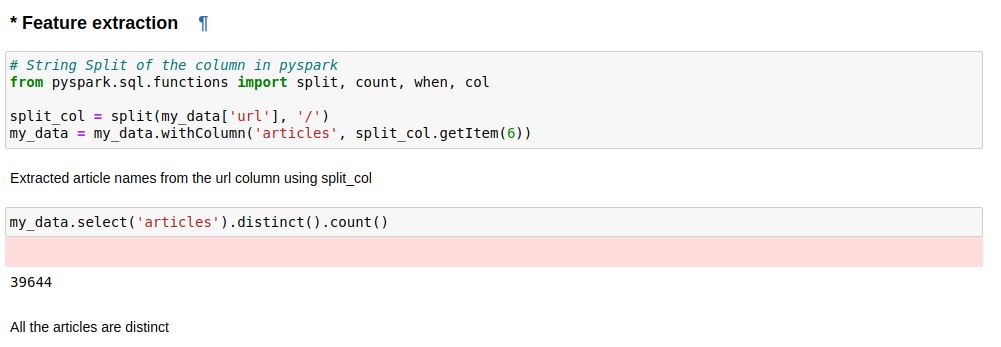




Display shows all the attributes datatype. Here you can see the target is converted into integer using IntegerType().

FEATURE EXTRACTION

Article feature is extracted for the url attribute using split in pyspark.

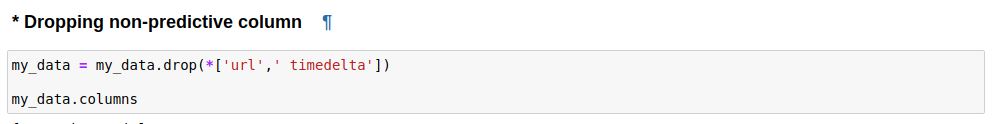


Then the extracted feature is compared with Shares and the article low-cost-iphone was the one which was shared a lot, next is dove-ad-beauty-sketches.



DROPPING NON-PREDICTIVE COLUMN

Using drop the non-predictive column url and timedelta is droped from the dataset. \* is used for multiple column to be dropped.



GROUPBY

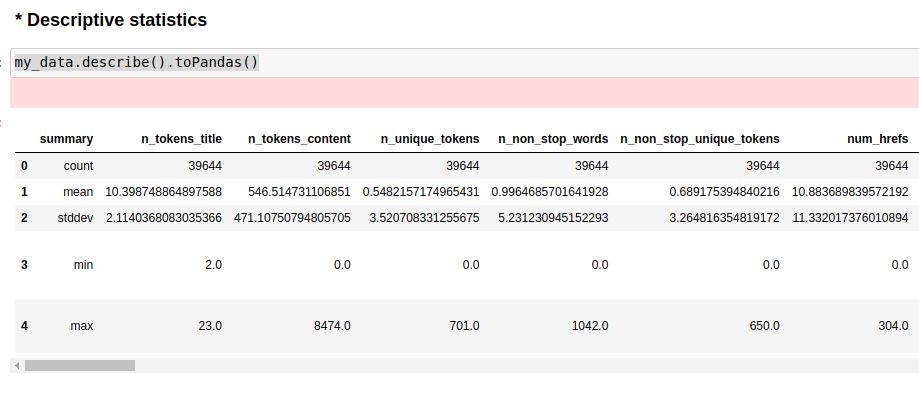
The data is grouped by shared and their values are displayed.



DESCRIPTIVE STATISTICS

The command describe helps to describe about the data it will display all the descriptive statistics values like min, max and mean in numerical attributes. This helps to interpret the data.

The command for descriptive statistics are shown below.

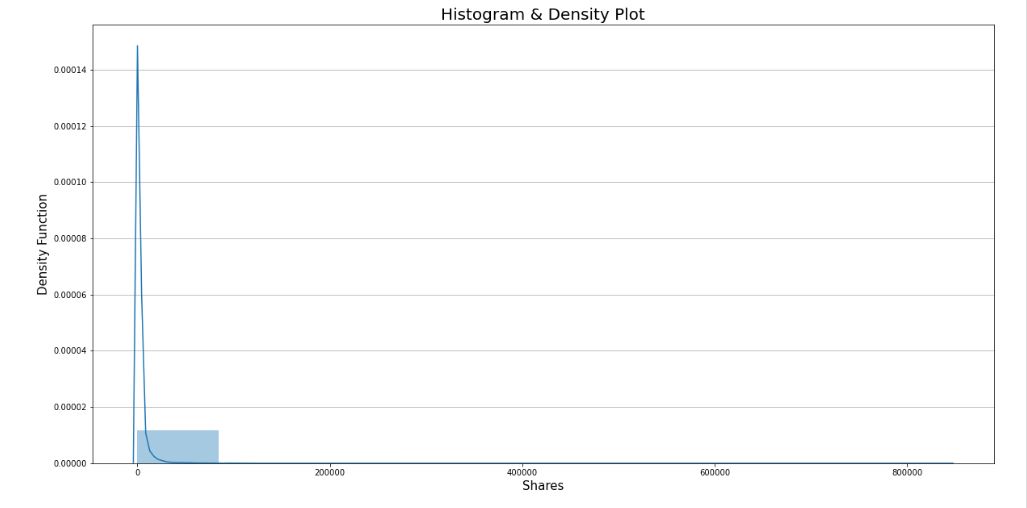


DISTRIBUTION

Distribution plot is used to find the skeweness and outlier in the data.



The distribution plot below shows that most of the data in shares attribute falls between 1400. This helps to group those value and to change it to a classification problem. If the shares are more than 1400 then it is popular article.



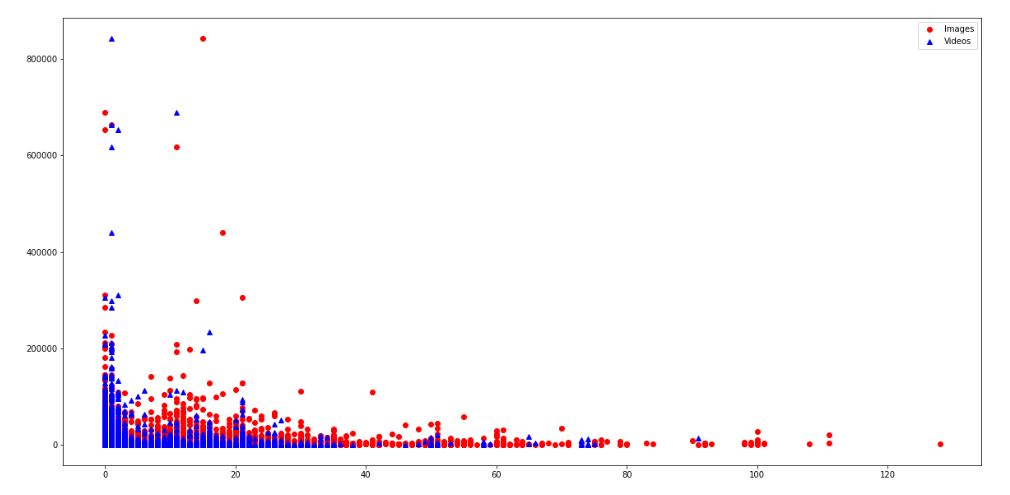
The graph above depicts the density distribution of shares throughout our whole dataset. As can be seen, the maximum examples of shares range from 0-10,000 shares, and the number of shares steadily decreases as the number of shares increases. Outliers are shares with a value greater than 20,000, while in our project, anomalies are defined as shares with a value greater than 20,000. According to our project, if an article has received the most shares, then the elements of the piece have been so unique that it has received the largest number of shares. As a result of the anomaly condition, we identify these unusual data as anamolies rather than outliers.

DATA ANALYSIS

To find the relationship between the total number of shares and images and videos scatter plot is used.



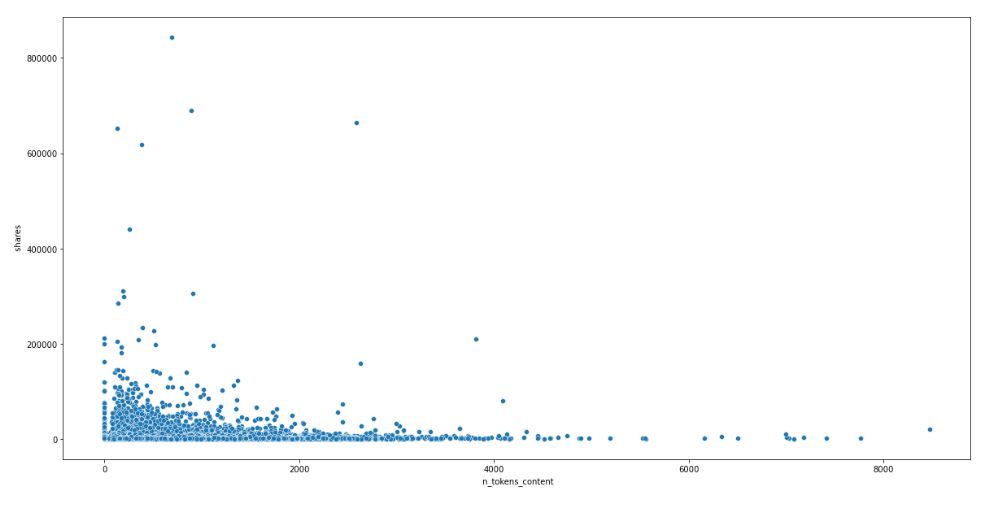
Relation between number of videos and images in an article with their corresponding shares. The number of shares is high if the article consist of images and videos between 0 to 20.



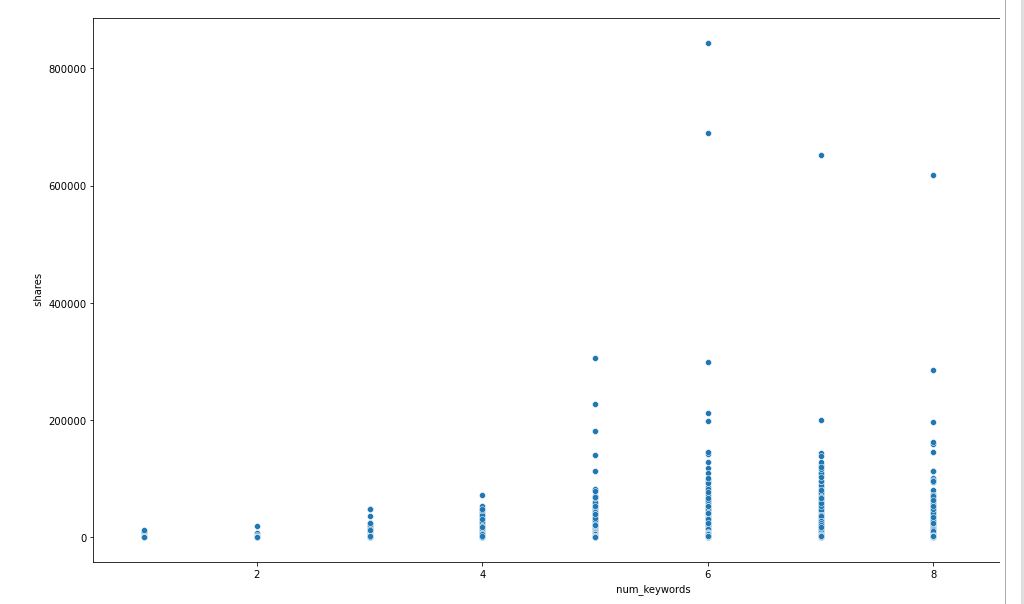
SCATTER PLOT

Total words in the content between 0-2000 are getting the higher response.Above 2000 articles have not been shared more number of times.

Data is right skewed

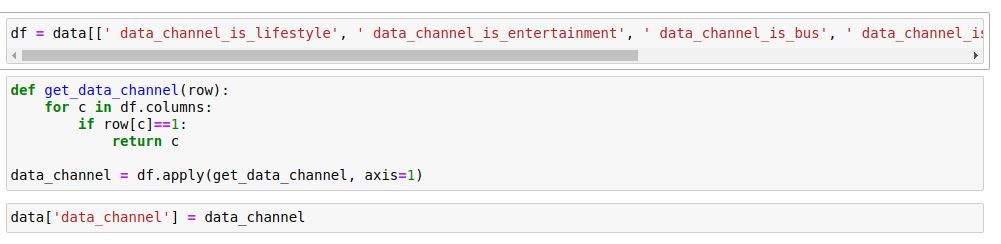


Number of share is less when the number of keywords in the metadata is between 0 to 4.

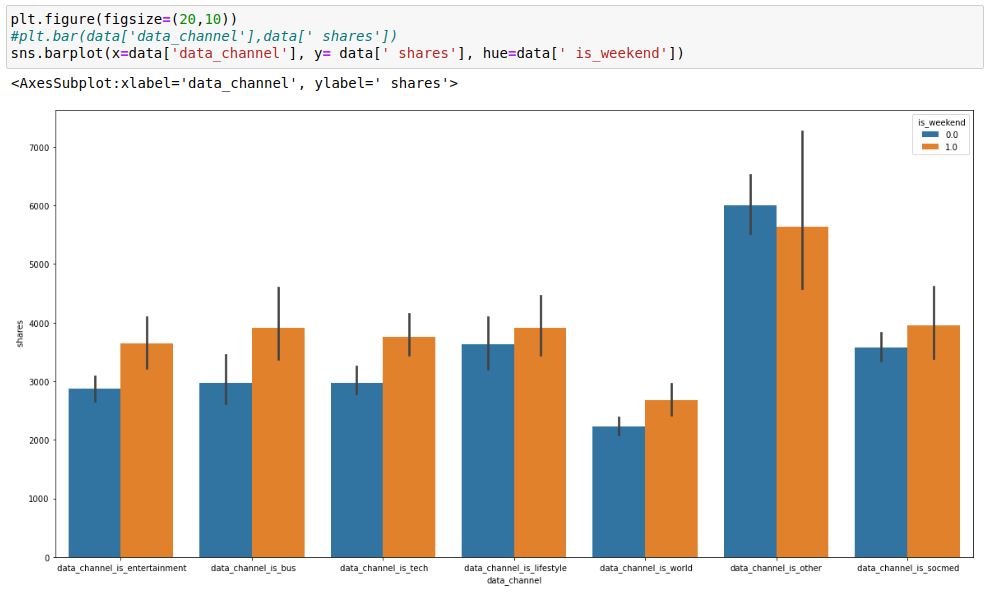


REVERSE ONE-HOT ENCODING

Reverse one-hot encoading is used to combine all the attributes which is already encoded to find insights from the bar plot.



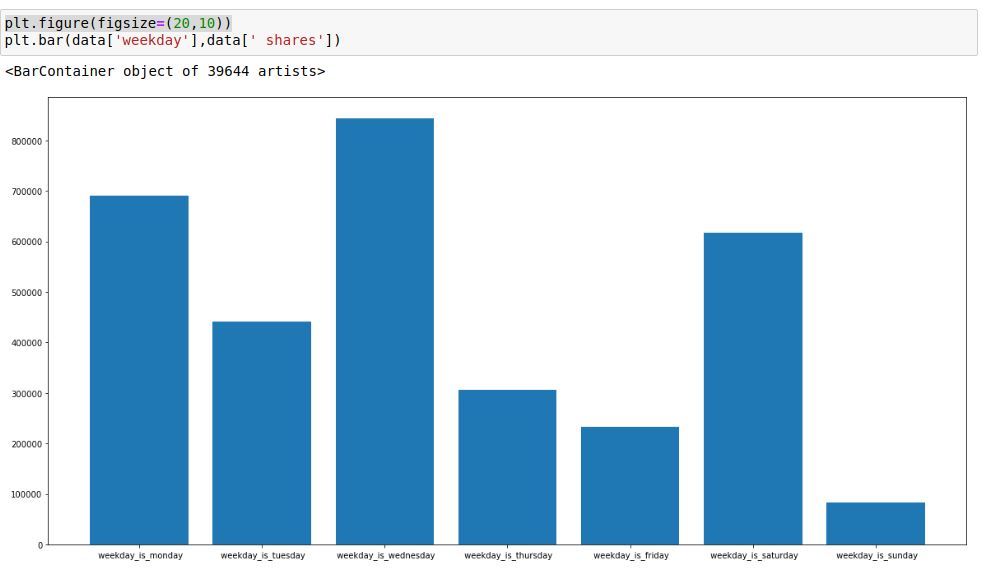
Most of the articles are shared at the weekend.



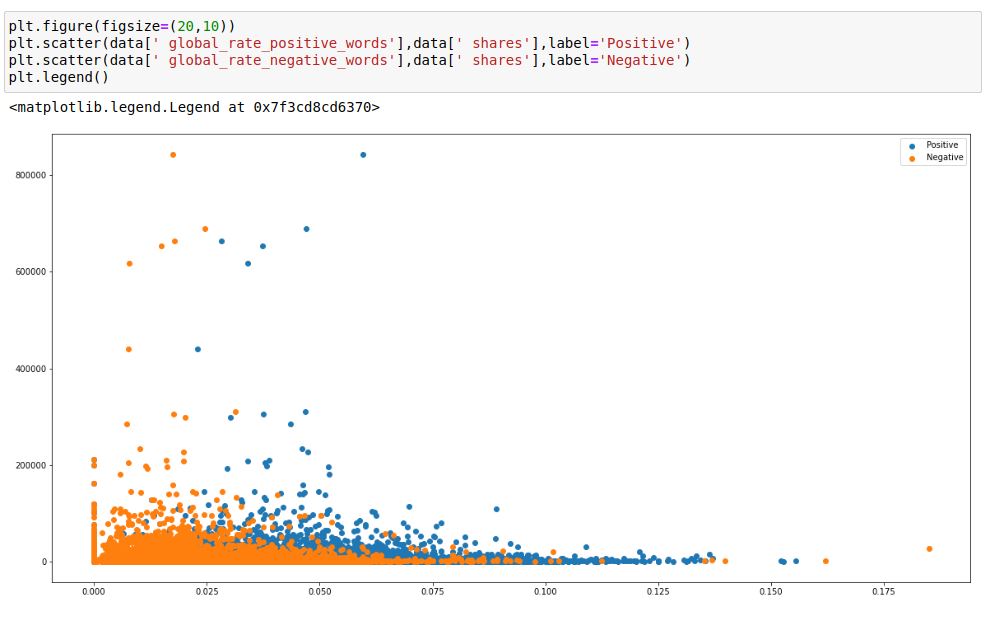
Weekdays are combined to plot and find the day at which most of the sharing is done.



Number of shares are high at monday, wednesday and Saturday.



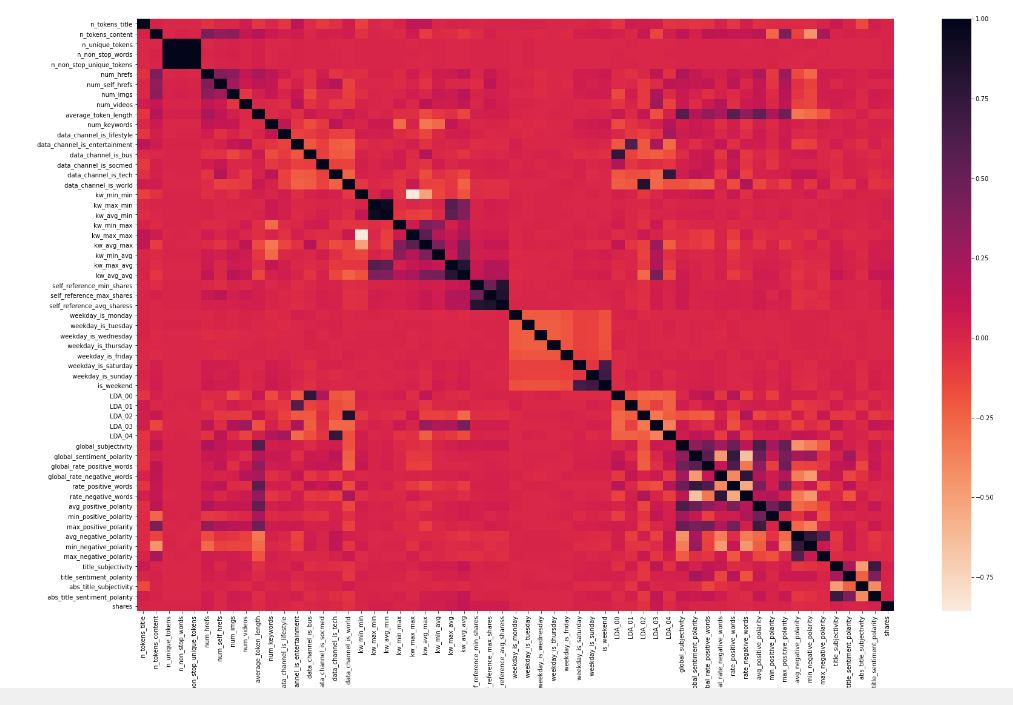
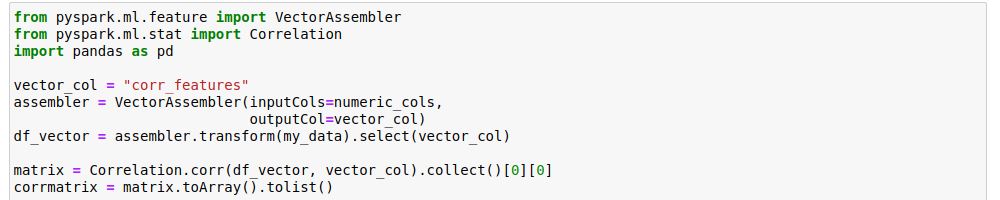
Relation between global rate and shares are shown below using scatter plot



From the above scatter plot it is clear that the rate of positive words is slightly higher than negative words.

CORRELATION HEATMAP

This helps to find the correlation between the variables and the higher correlated attributes are dropped form the dataset to reduce overfitting.

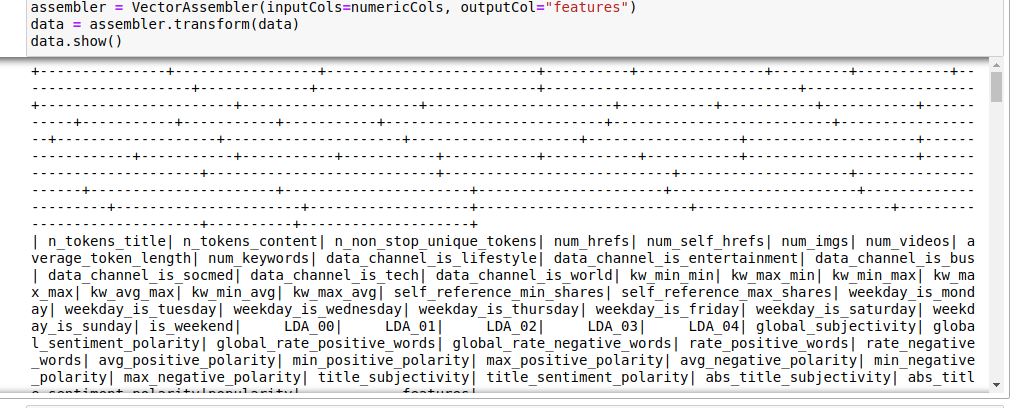
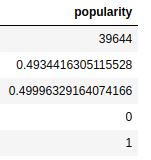
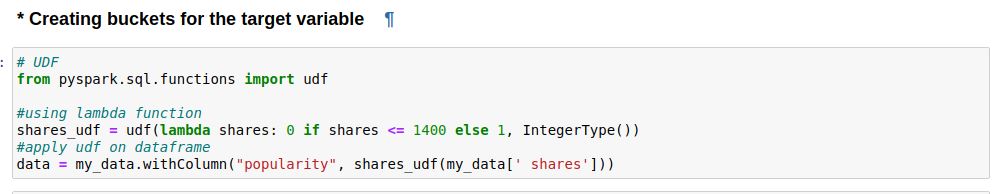


The number of unique words and number of non-stop-words and number of non-stop-unique tokens are strongly correlated which implies that they are strongly linearly dependent on each other. Same as the above case Kw-avg- min and kw-max-min are also strongly corelated.



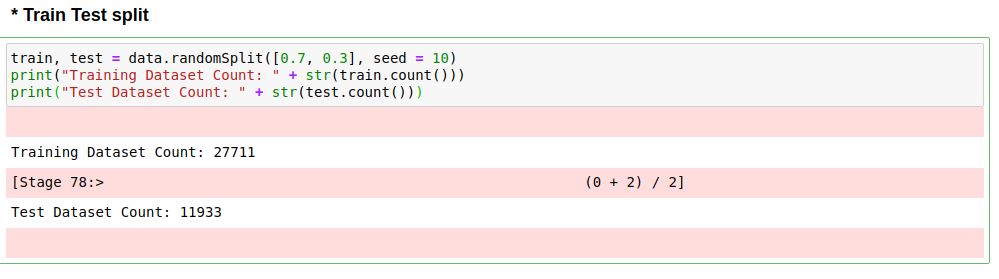
GROUPING TARGET VARIABLE

The target variables are grouped and put in a bucket based on the above mentioned distribution plot. If the value is below 1400 then it is 0 else 1.



TRAIN TEST SPLIT

The dataset is splitted into training and testing data. The training data consist of 70 % of the data and testing is done on remaining 20% with the seed rate 10 so that the values won’t change if we retrain.



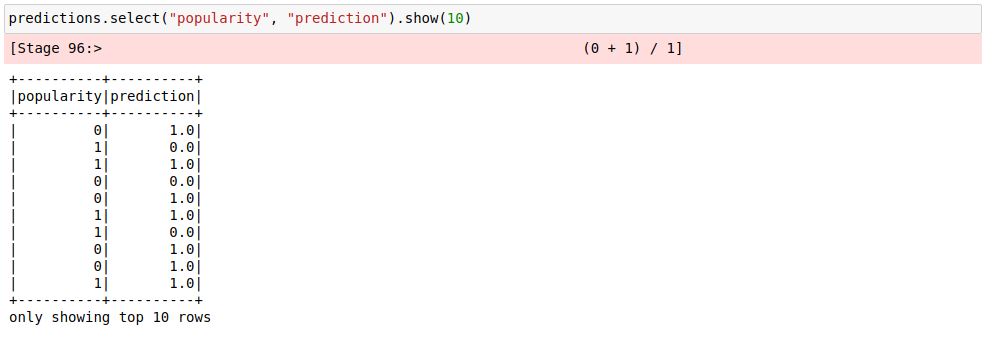
RANDOM FOREST

Random forest is used in this data because of the high variance which was found by plotting and visualizing the data. Decision tree will overfit with the data if the data has high variance. Moreover, our dataset also has bias. Random forest will perform well in these situation by reducing the variance.



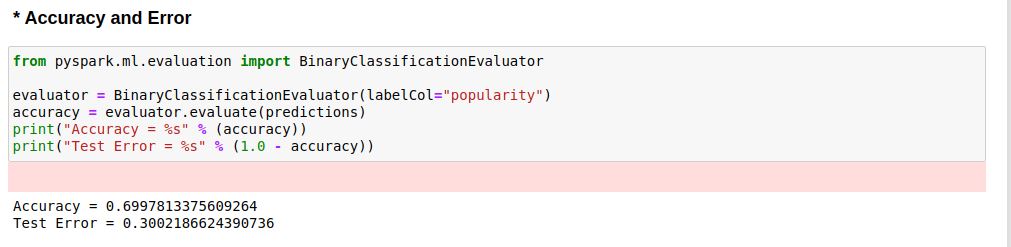
PREDICTIONS

The below table shows the actual and predicted value.



ACCURACY AND ERROR

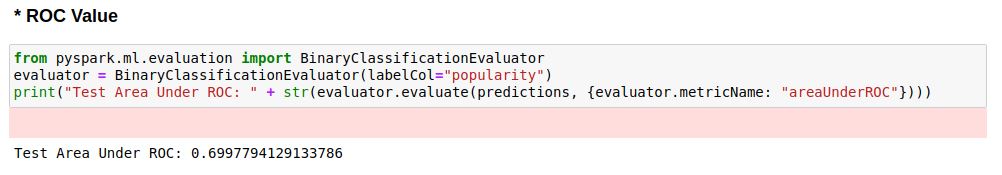
In Pyspark BinaryClassificationEvaluator is used to measure the performance of binary class target variable. The accuracy and error is displayed below.



The accuracy of the model is 0.69 and the test error is 0.30 for the base model.

ROC VALUE

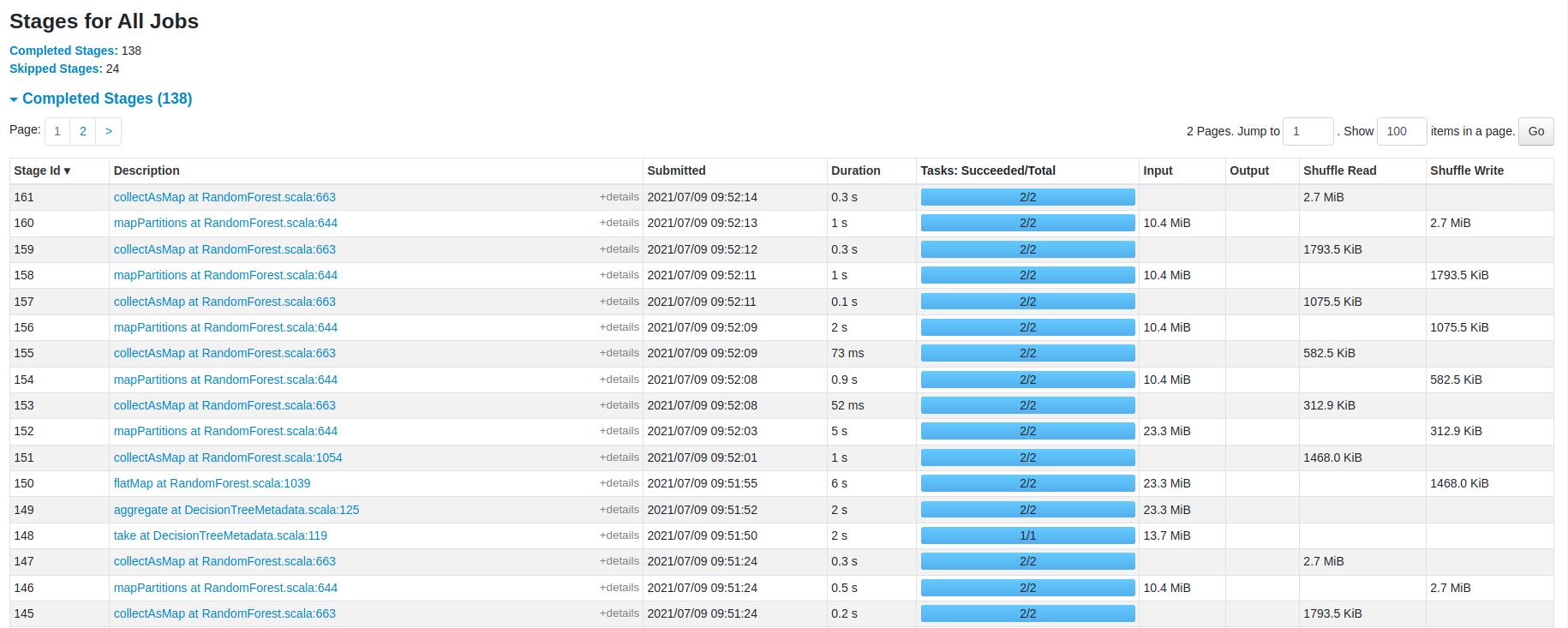
ROC and Area Under the Curve is used to define how the model will perform.



The Test area under ROC is 0.69 for a base model.

SPARK STAGES

All the jobs ran for this project can be seen in the spark stages.



DISCUSSION

In this project different statistic technique and Visualization techniqes are used to gain insights from the data. Because of that we were able to find the algorithm best suited for the data. The algorithm can be tuned for better accuracy. The depth of the tree can be pruned to find the best depth of the model.

Insights found in this project are as follows,

* The Histogram and density plot shows that the column [Shares] is right skewed. And the range for popularity is found.
* Relation between number of videos and images in an article with their corresponding shares. The number of shares is high if the article consist of images and videos between 0 to 20.
* Total words in the content between 0-2000 are getting the higher response. Above 2000 articles have not been shared more number of times. Data is right skewed.
* Number of share is less when the number of keywords in the metadata is between 0 to 4.
* Most of the articles are shared at the weekend.
* Number of shares are high at monday, wednesday and Saturday.
* Rate of positive words is slightly higher than negative words.
* The number of unique words and number of non-stop-words and number of non-stop-unique tokens are strongly correlated which implies that they are strongly linearly dependent on each other. Same as the above case Kw-avg- min and kw-max-min are also strongly corelated.

Moreover, Docker works well with this project and the setup time for the project is reduced. And there is no dependency libraries we need to install.

CONCLUSION

The best model can be found by further analysis of the data. The computation for 50+ attributes are faster in pyspark due to its parallel processing. In addition to that docker helps to reduce the setup time and enables portability of the code without dependencies. And the image is committed so that the committed image will have same code and dependent imports in it. Coming to the model the performance of the model can be further improved using hyper parameter tuning and feature selection method.

CODE LINK

https://github.com/Johnson-28/OnlineNewsPopularity-Pyspark

REFERENCES

K. Fernandes, P. Vinagre and P. Cortez. A Proactive Intelligent Decision Support System for Predicting the Popularity of Online News. Proceedings of the 17th EPIA 2015 - Portuguese Conference on Artificial Intelligence, September, Coimbra, Portugal.

Assefi.M, Behravesh.E, Liu.G and Tafti A. P,(2017) "Big data machine learning using apache spark MLlib," 2017 IEEE International Conference on Big Data (Big Data), Boston, MA, pp. 3492-3498

Merkel, D., 2014. **Docker**: lightweight linux containers for consistent development and deployment. Linux journal, 2014(239), p. 2.

Dean, J. and Ghemawat, S. MapReduce: Simplified data processing on large clusters. In *Proceedings of the Sixth OSDI Symposium on Operating Systems Design and Implementation* (San Francisco, CA, Dec. 6--8). USENIX Association, Berkeley, CA, 2004.

Sheshasaayee.A and Lakshmi. J. V. N,(2017) "An insight into tree based machine learning techniques for big data analytics using Apache Spark," International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT), Kannur, 2017, pp. 1740-1743