**STOCK PREDICTION USING LARGE TWITTER DATA**

Team Members:

Sahaja Adavelli(01)

Zeal Patel(18)

Sree Valli Tata(23)

Objective: The main objective of increment 2 is to clean the data and find the polarity values. If the polarity value is less than 0 then it is negative, if the value is greater than 0 then it is positive. Then train the model using logistic regression.

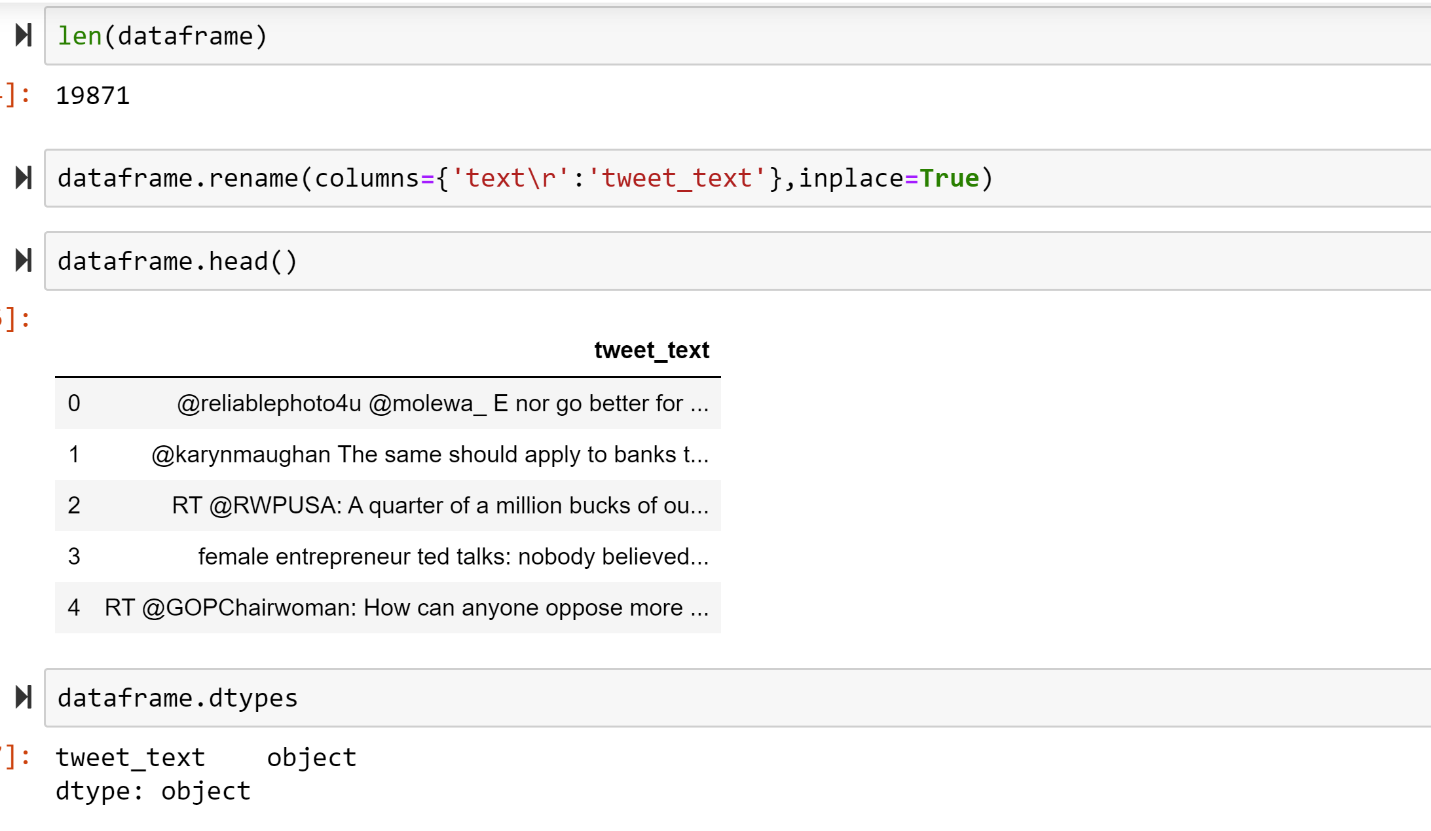
Implementation:

Using the twitter API we have collected 20000 tweets using the keywords related to stocks and convert to .csv file.

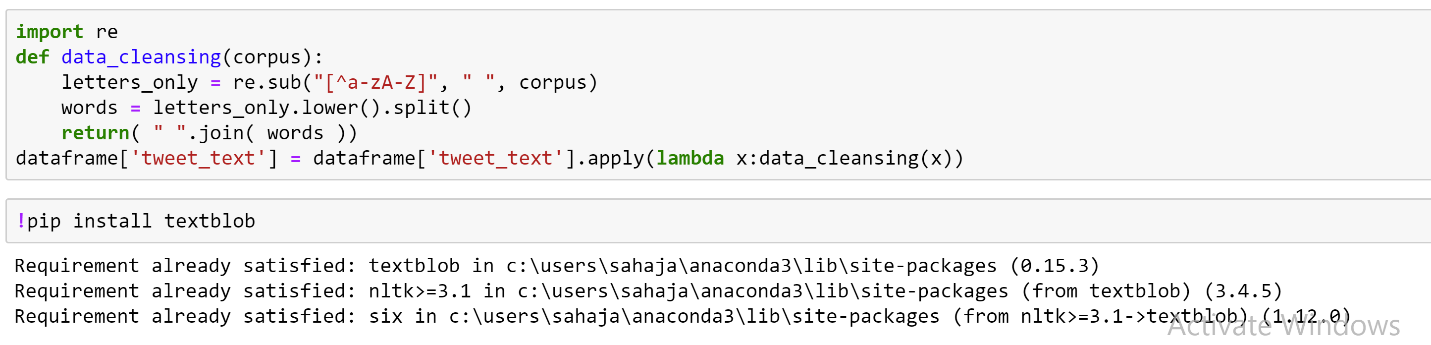
Import the required libraries and read the data using pandas .



Drop all the columns except tweets as they are not required and change the name of the column to tweet\_text

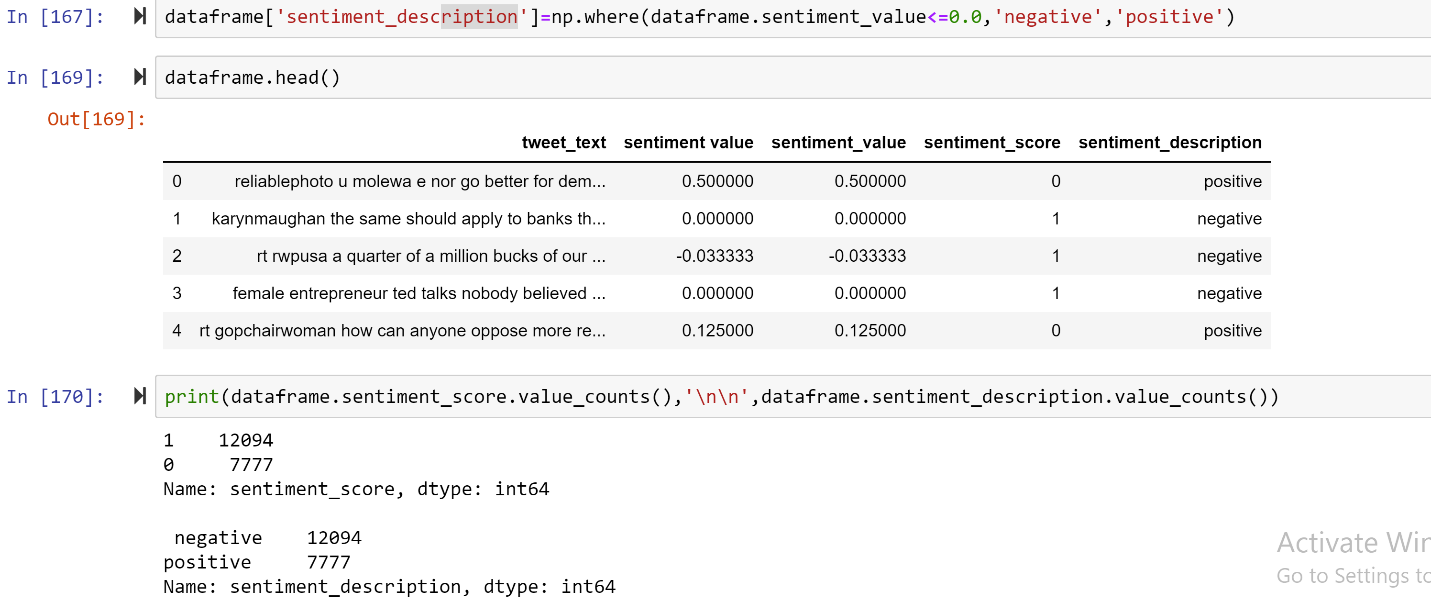


Data Cleaning: We are cleaning the data by removing all other characters except alphabets, consider all small and capital alphabets.

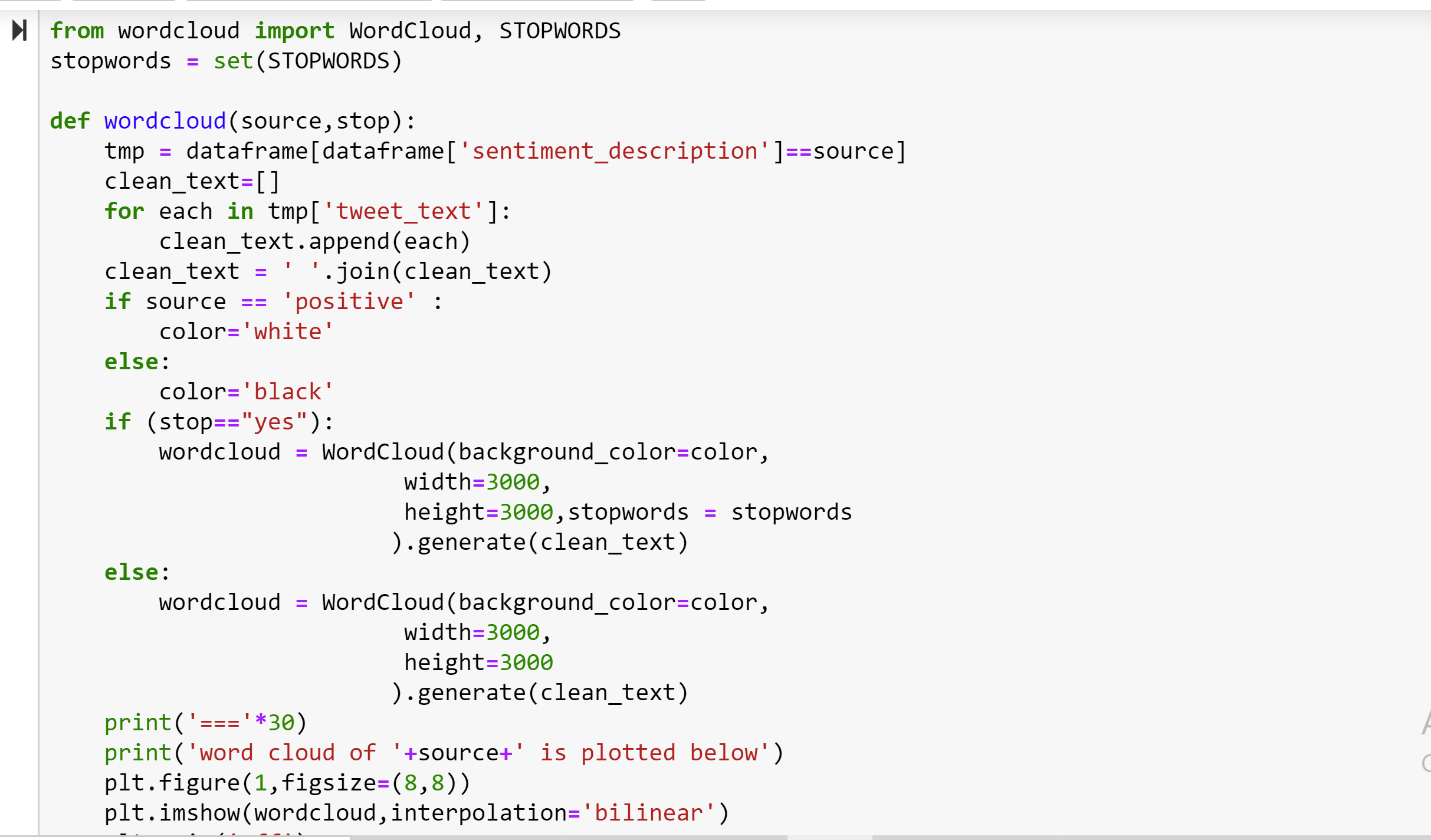


Textblob: Using textblob we are analyzing the sentiment of by finding polarity values. If polarity value is less than 0 then it is negative if the polarity value is greater than zero then it is positive.





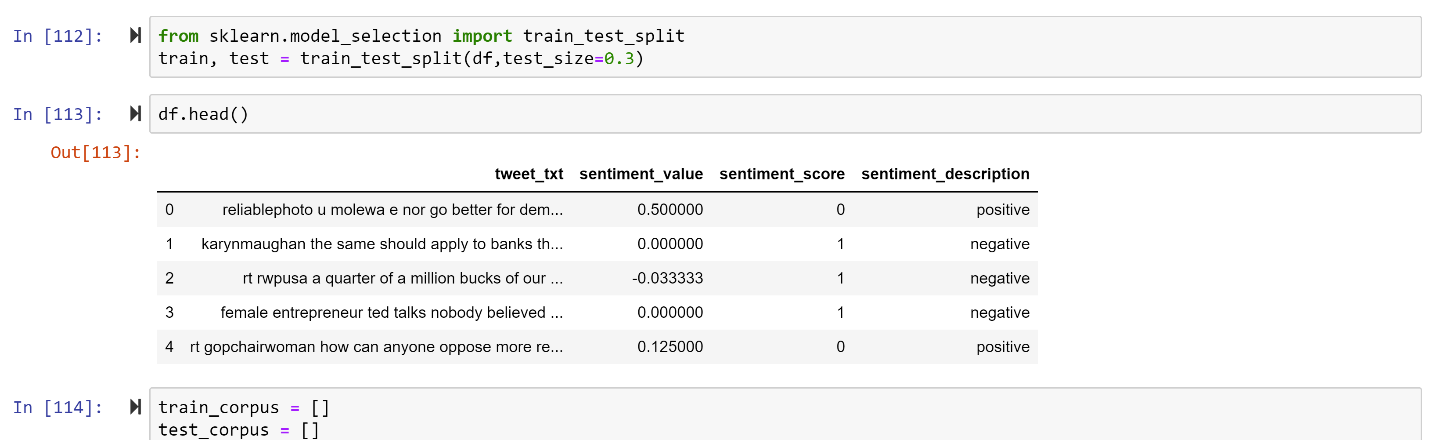
Using the WordCloud we are representing the negative and positive words







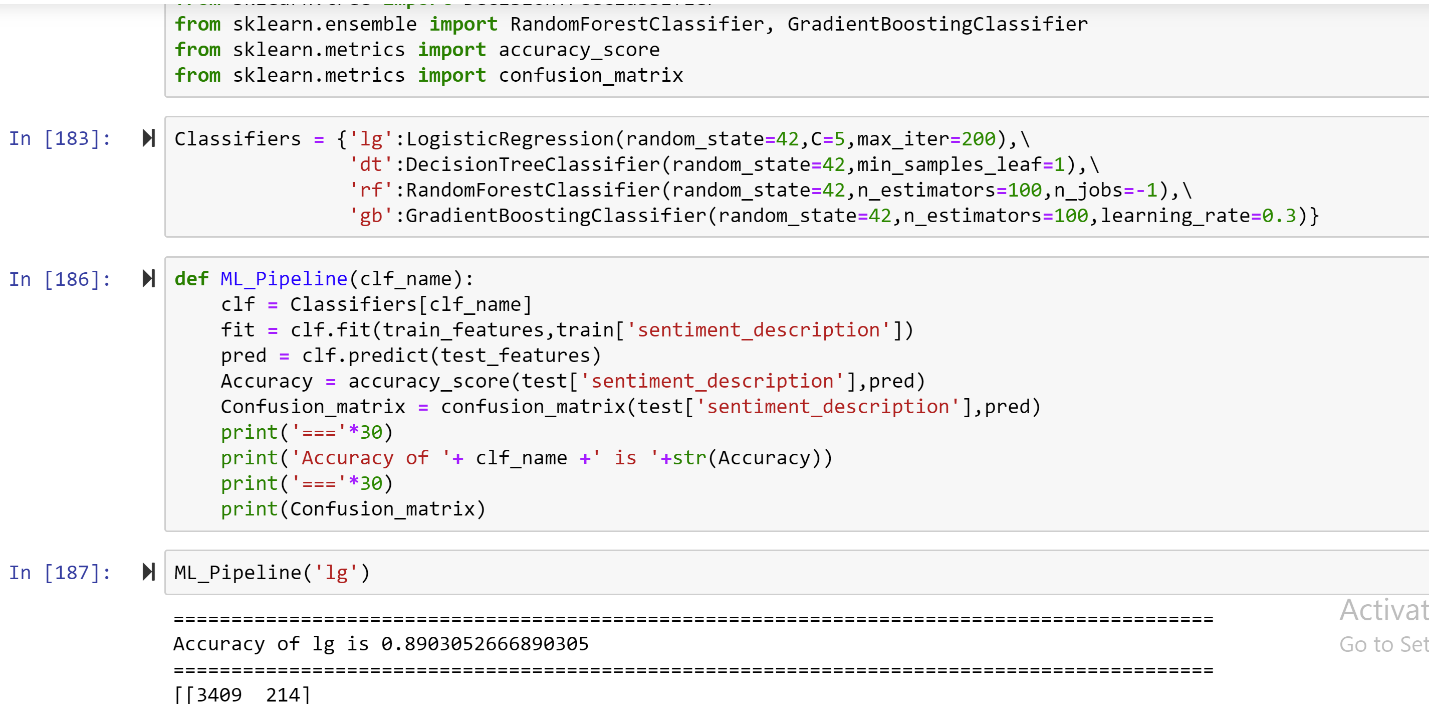
Split the data into train and test data using train\_test\_split



TfidVectorization: As it is difficult to train using words so we are Vectorize the words to get Unicode for each word and train the words using unicodes.



Train the model using Logistic regression

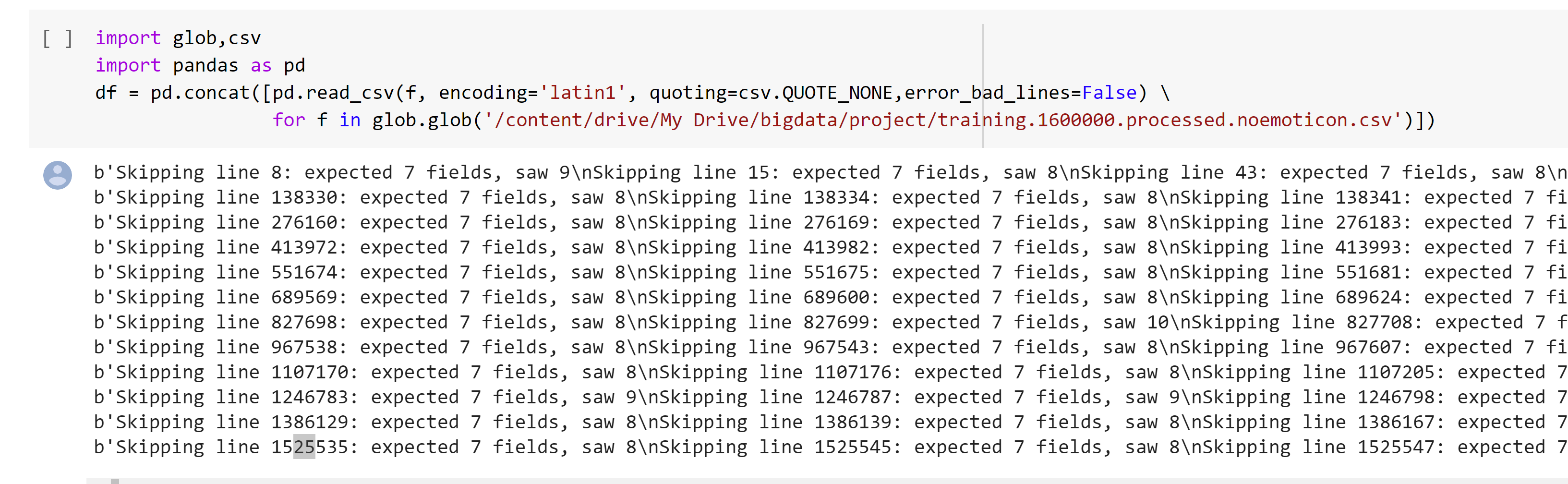


Find the words which contribute more for the analysis



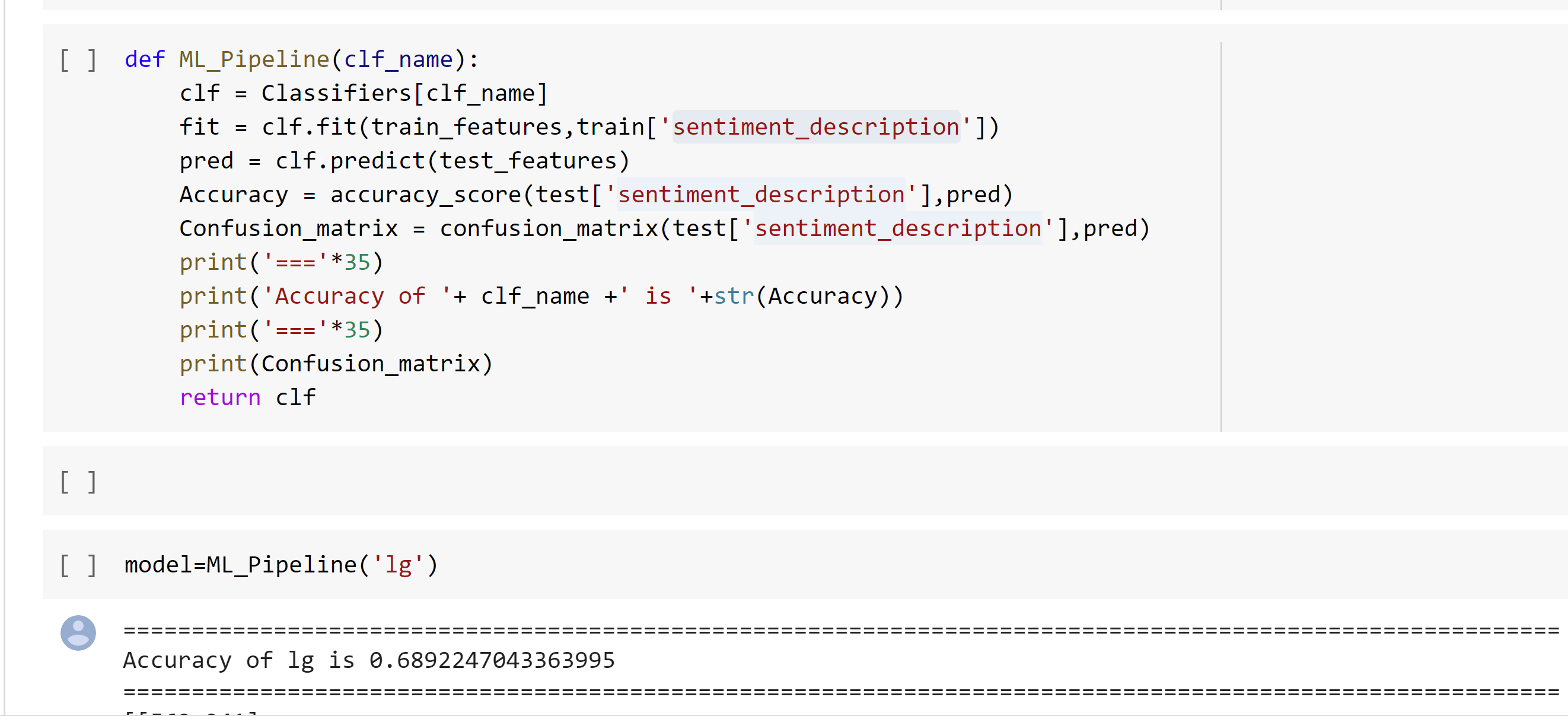
Comparison of trained model and already existing model:

Considered the dataset which contains sentiment score given manually which has 15,00,000 tweets.



Img13

Using logistic regression find the accuracy of the model



Img14

Using TextBlob find the sentiment polarity values



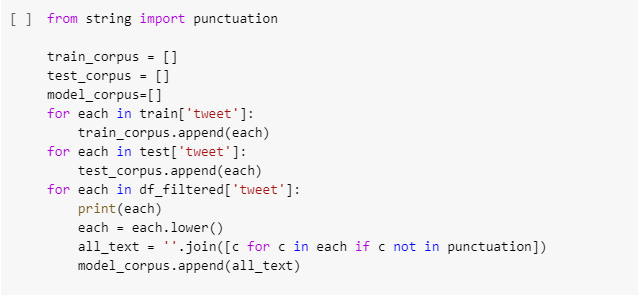
Img15

Find the accuracy from the textblob with sentimental dataset and accuracy of our trained model with sentimental dataset. We can say that our trained model gives more accuracy.

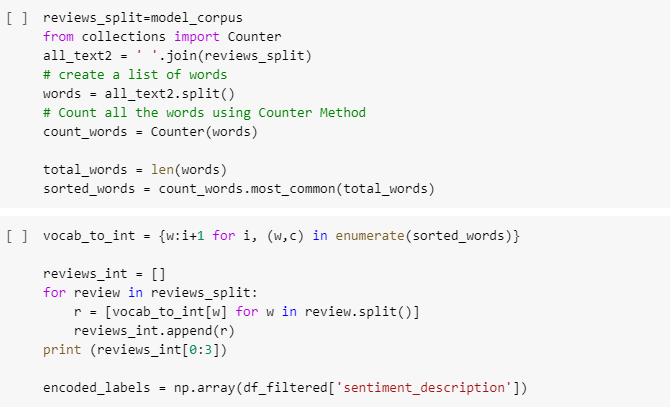


Img16

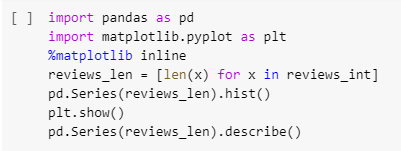
Deep Learning:



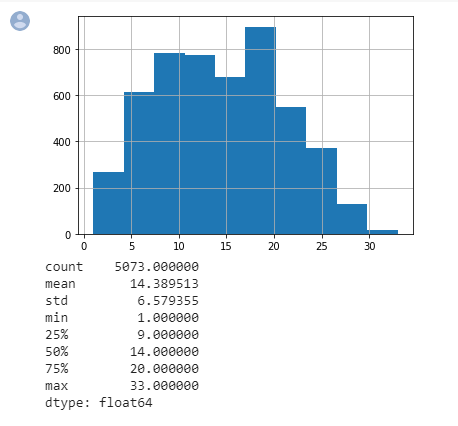
Here we have used punctuation library, where this include all the punctuations but we have taken c not in punctuation and joined it to **all\_text**. Here only alphabets are focused for retrieving of data.

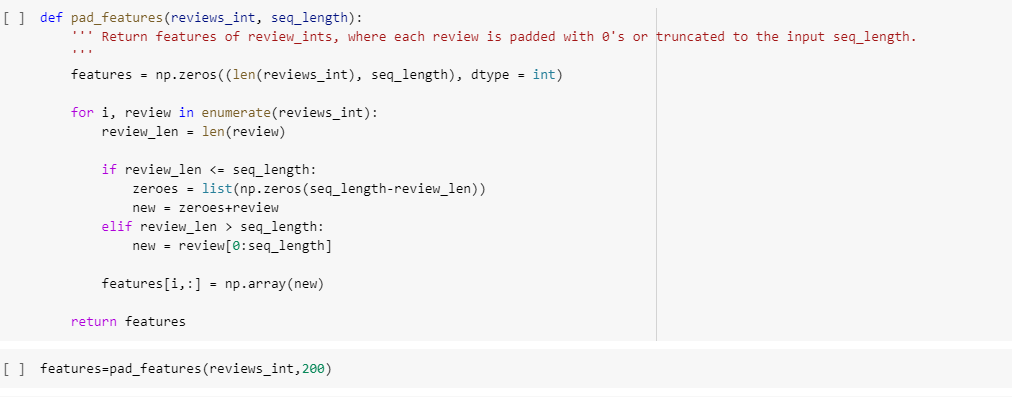


After that we have joined all those words into a complete text and used **vocab\_to\_int** which maps all the words to a number, and we have counted aall the words in the text and assigned that value to **count\_words**.

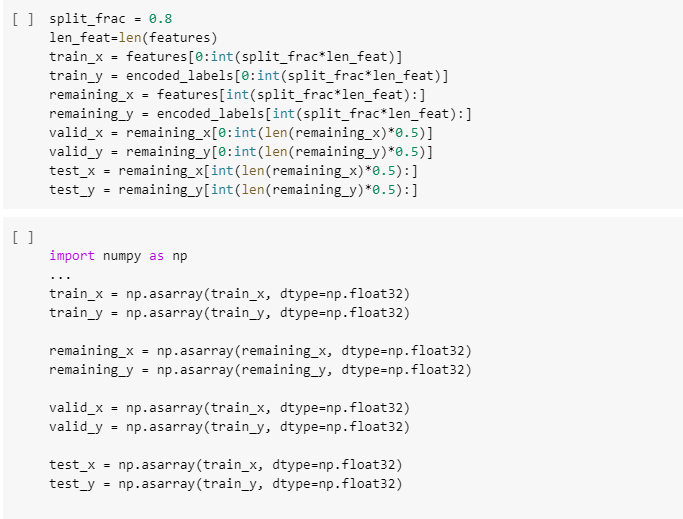


Further we have taken matplot library to plot number mean, standard deviation and others for all those count words





Here we have added padding feature to our data that we are collecting from reviews. So, this padding feature make sure that all the reviews would be of same size if not it adds up zeros at the end of the sentence. And we have given padding size as 200 that is the maximum number of words a review in a twitter would contains. If a review is less than 200 all the other space would be filled with zeros.



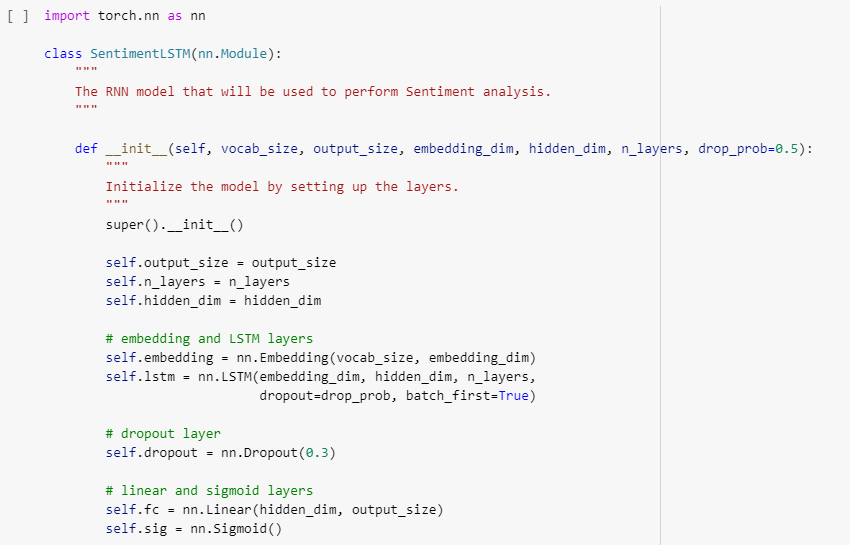
Here we are splitting the data for training purpose. We have taken 80% of the data for training and the 20% data is titled as remaining which is further equally divided as validated and test.

We have further used the numpy library to convert the data into numpy array format as DataLoaders demand for that format.



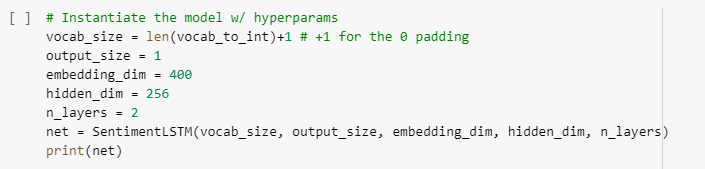
**DataLoader** gives data based on batch size. We have taken batch size as 50 and number of epochs are 4. As we have observed from 5th epoch the loss have been increasing so in order to avoid that we have considered 4.

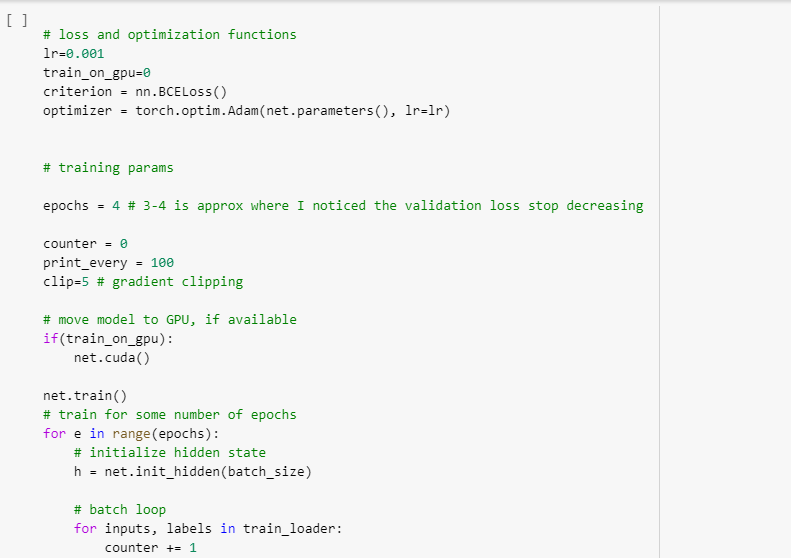
Also we have used **drop\_last** which make sure that the needed data of batch size is given to DataLoader and others is discarded. In order to avoid the discarding of same data all the time we are using **shuffle=true** which makes sure that for every iteration the data gets shuffled. And finally we have printed all the data that we have loaded.



As a part of DeepLearning concept we have used **LSTM (Long Short Term Memory)** as it have advantages of removing the useless data by tan functioning. Also, we have added an extra embedded layer in which we have given embedded dimension, hidden dimension and number of layers a LSTM should include for processing of the entire data.

Additionally we have used a **Dropout**, dropout is used for dropping data for not being biased, and the activation layer we have used here is sigmoid.

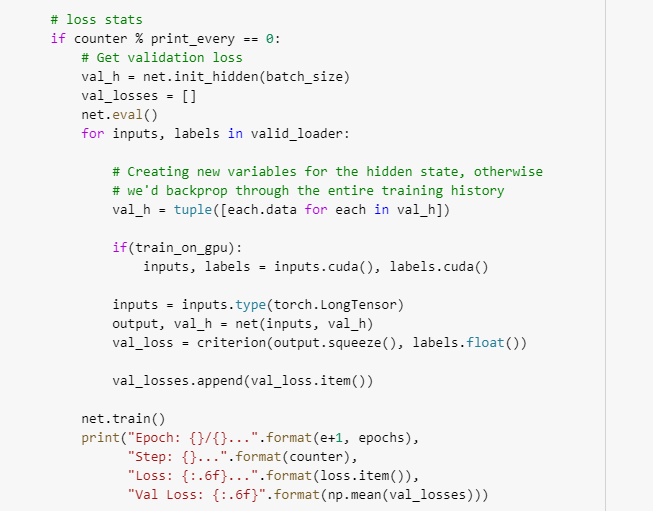


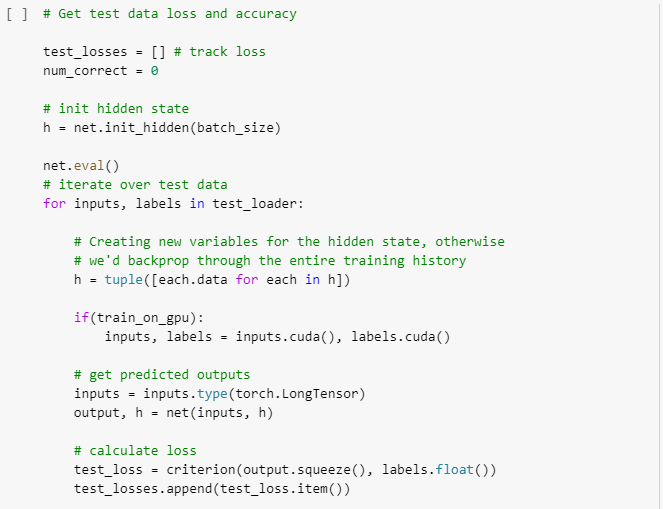


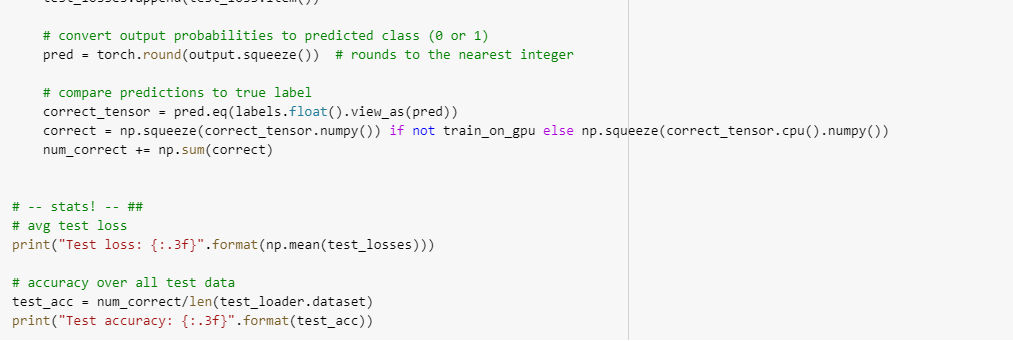
Here we are considering loss and optimization function we have taken **LAN rate (lr)** as 0.01%. Which means there should be atleast 0.01% change in the data from the previous LAN, we do this until we get a saturation stage where we cannot further get the added data.

As an optimization function we have used Adam optimizer. Adam is an adaptive learning rate optimization algorithm that’s been designed specifically for training deep neural networks. Optimizers make sure to make the data better than the previous LAN data.

Also, we are keeping the track of the data loss by **BCEloss function** (Binary Cross Entropy).







Finally after keeping the track of loss data and training the model we have got the loss of 0.5% and accuracy of around 98%. Which means there is 10% increase in the accuracy compared to the previous model that we have trained with linear regression and the downloaded twitter dataset.

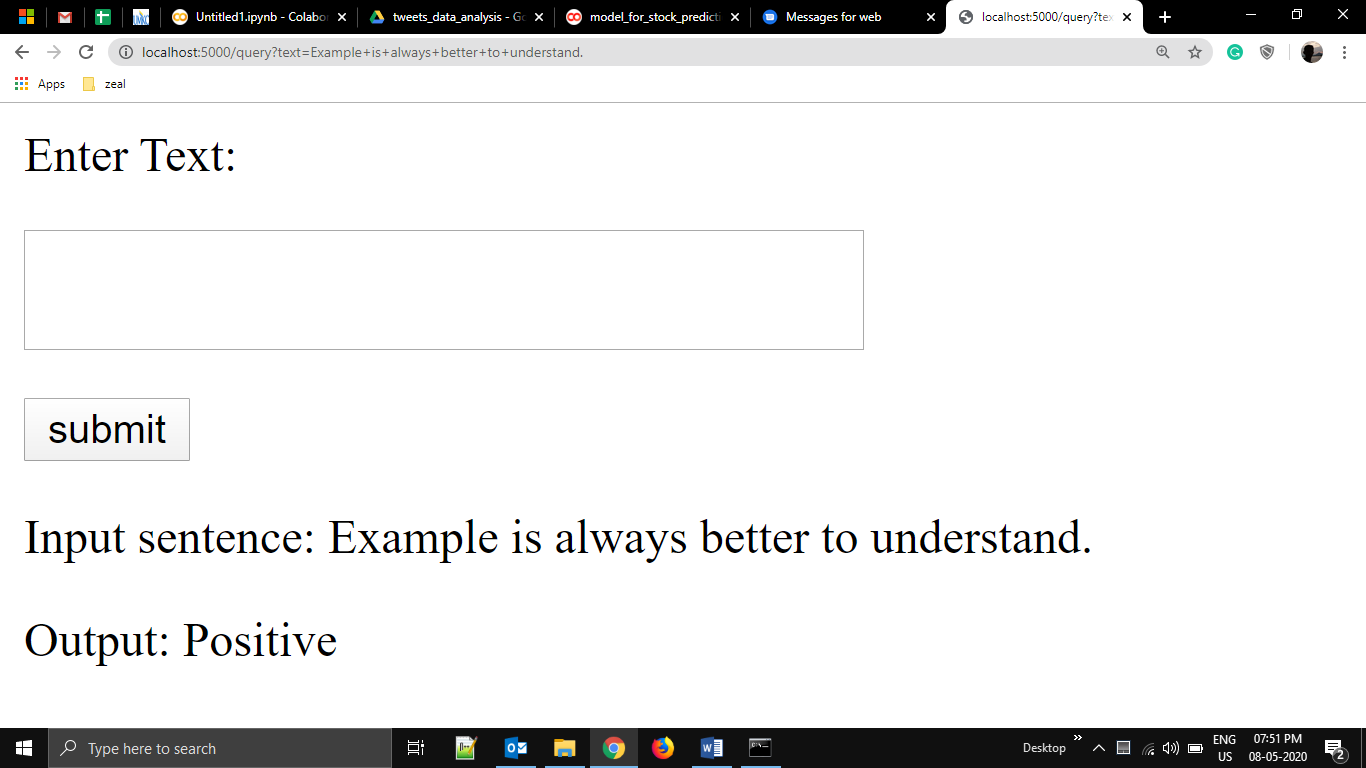
**Output:**



User interface for sentimental analysis

In order to make user interface we have used flask in order to run our model prediction in python backend.

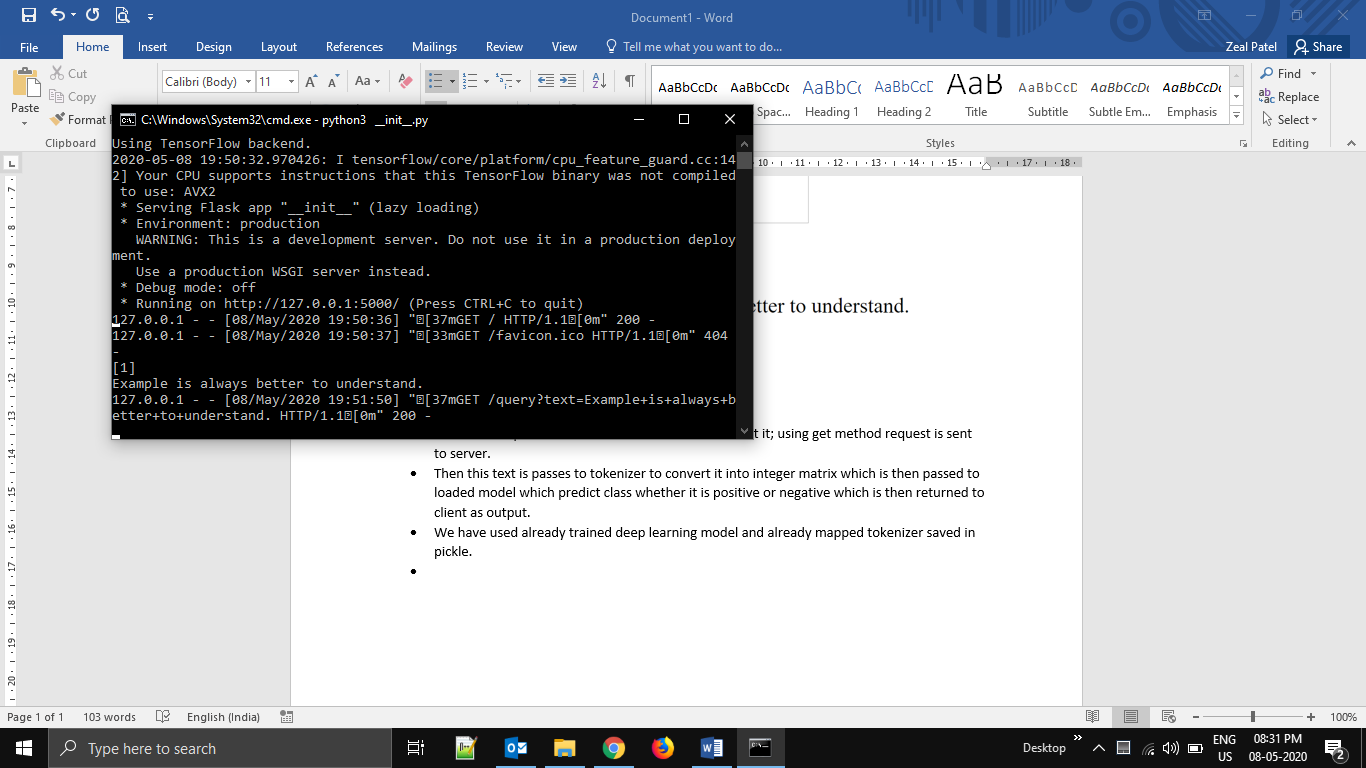
Here is user interface of on browser:



When user input sentence in text box and then submit it; using get method request is sent to server.

Then this text is passes to tokenizer to convert it into integer matrix which is then passed to loaded model which predict class whether it is positive or negative which is then returned to client as output.

We have used already trained deep learning model and already mapped tokenizer saved in pickle.



Here is flask running screen.

Contribution:

Zeal: Working on Data collection from different sources and collect more tweets for product based companies and build the website using flask.

Sree Valli: trained the model using LSTM deep learning .

Sahaja: trained model training using machine learning logistic regression and compare the model with already existing textblob model .