Capstone Project – 3

Machine Learning – Classification

Team **Champions** : Coronavirus Tweet Sentiment Analysis

Team Members

Varsha Rani

Vivek Chandrakant Pawar

Rabista Parween

Tushar Gaikwad

POINTS FOR DISCUSSION

- Problem Description
- Data Summary
- Data Inspection
- Feature Engineering
- Exploratory Data Analysis
- Data Modeling
- Performance Metrics and Accuracy
- Confusion Metrics
- Conclusion

Problem Description

This challenge asks us to build a classification model to predict the sentiment of COVID-19 tweets. The tweets have been pulled from Twitter and manual tagging has been done then.

The names and usernames have been given codes to avoid any privacy concerns.

We are given the following information:

1. Location
2. Tweet At
3. Original Tweet
4. Label

Data Summary

We are given data set : **Coronavirus Tweets.csv**

We are given the dataset having 6 columns –

- UserName
- ScreenName
- Location
- TweetAt
- OriginalTweet
- Sentiment

We have removed punctuation and stop words from ‘OriginalTweet’ column and added a new column ‘clean_tweets’ to the dataset.

We have done classification using five models namely –

- Logistic Regression Model
- Decision Tree Classifier
- Random Forest Classifier
- Gradient Boosting Classifier
- Passive Aggressive Classifier.

Importing Libraries & Data Inspection

- Pandas – Manipulation of tabular data in Dataframes
- Numpy – Mathematical operations on arrays
- Matplotlib – Visualization
- Seaborn – Visualization
- Sklearn – Data Modeling
- Nltk – Pre Processing / Feature Engineering
- WordCloud – Visualization

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment
0	3799	48751	London	16-03-2020	@MeNyrbie @Phil_Gahan @Chrisitv https://t.co/i...	Neutral
1	3800	48752	UK	16-03-2020	advice Talk to your neighbours family to excha...	Positive
2	3801	48753	Vagabonds	16-03-2020	Coronavirus Australia: Woolworths to give elde...	Positive
3	3802	48754	NaN	16-03-2020	My food stock is not the only one which is emp...	Positive
4	3803	48755	NaN	16-03-2020	Me, ready to go at supermarket during the #COV...	Extremely Negative

Original Dataset contains 6 columns and 41157 rows.

Feature Engineering

Step 1 : Converted all characters to lowercase.

Step 2 : Removed Punctuation.

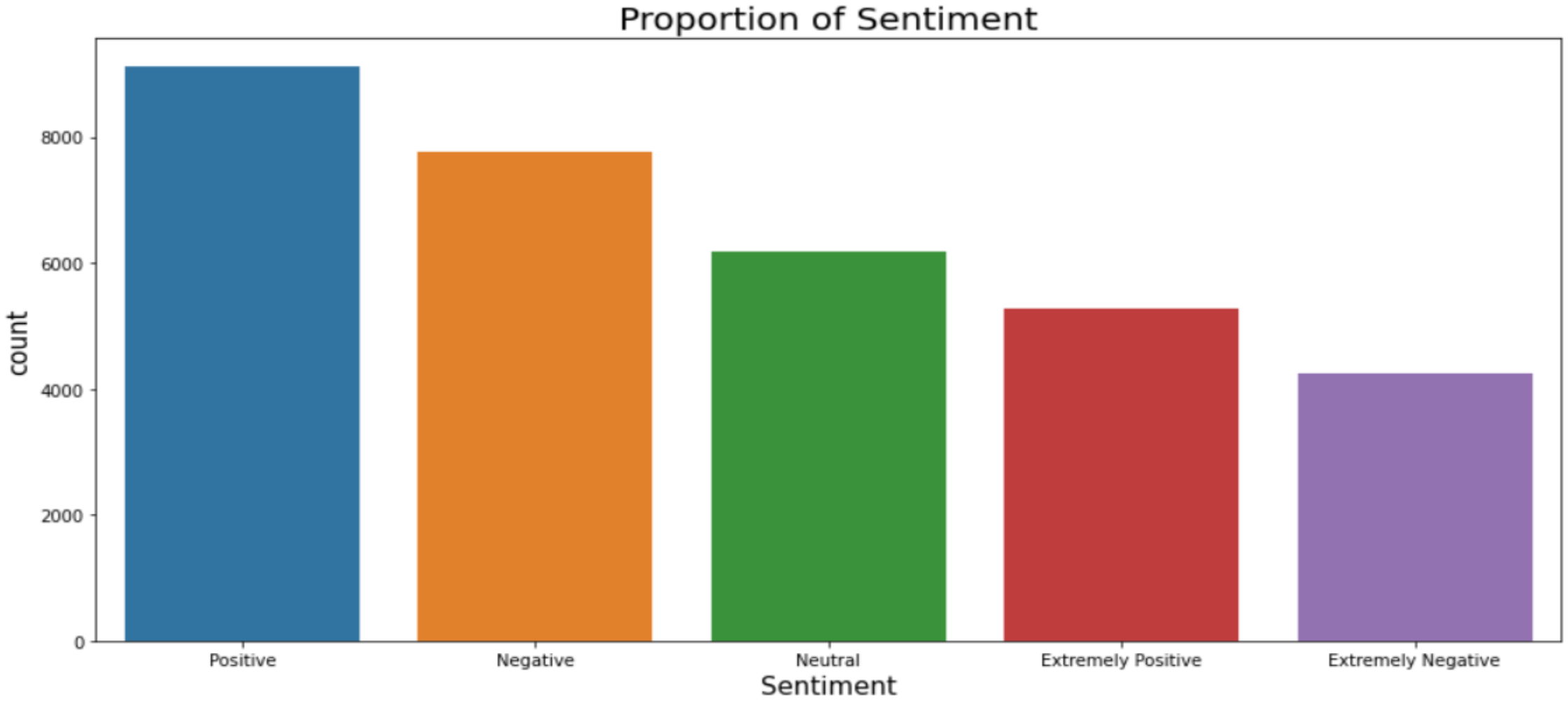
Step 3 : Removed stop words.

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment	clean_tweets
0	3799	48751	London	16-03-2020	@menyrbie @phil_gahan @chrisitv and and	Neutral	menyrbie philgahan chrisitv
1	3800	48752	UK	16-03-2020	advice talk to your neighbours family to excha...	Positive	advice talk neighbours family exchange phone n...
2	3801	48753	Vagabonds	16-03-2020	coronavirus australia: woolworths to give elde...	Positive	coronavirus australia woolworths give elderly ...
3	3802	48754	NaN	16-03-2020	my food stock is not the only one which is emp...	Positive	food stock one empty please dont panic enough ...
4	3803	48755	NaN	16-03-2020	me, ready to go at supermarket during the #cov...	Extremely Negative	ready go supermarket covid outbreak im paranoi...

After dropping the null values and adding a new column ‘clean_tweets’, now we have 7 columns and 32567 rows.

Exploratory Data Analysis

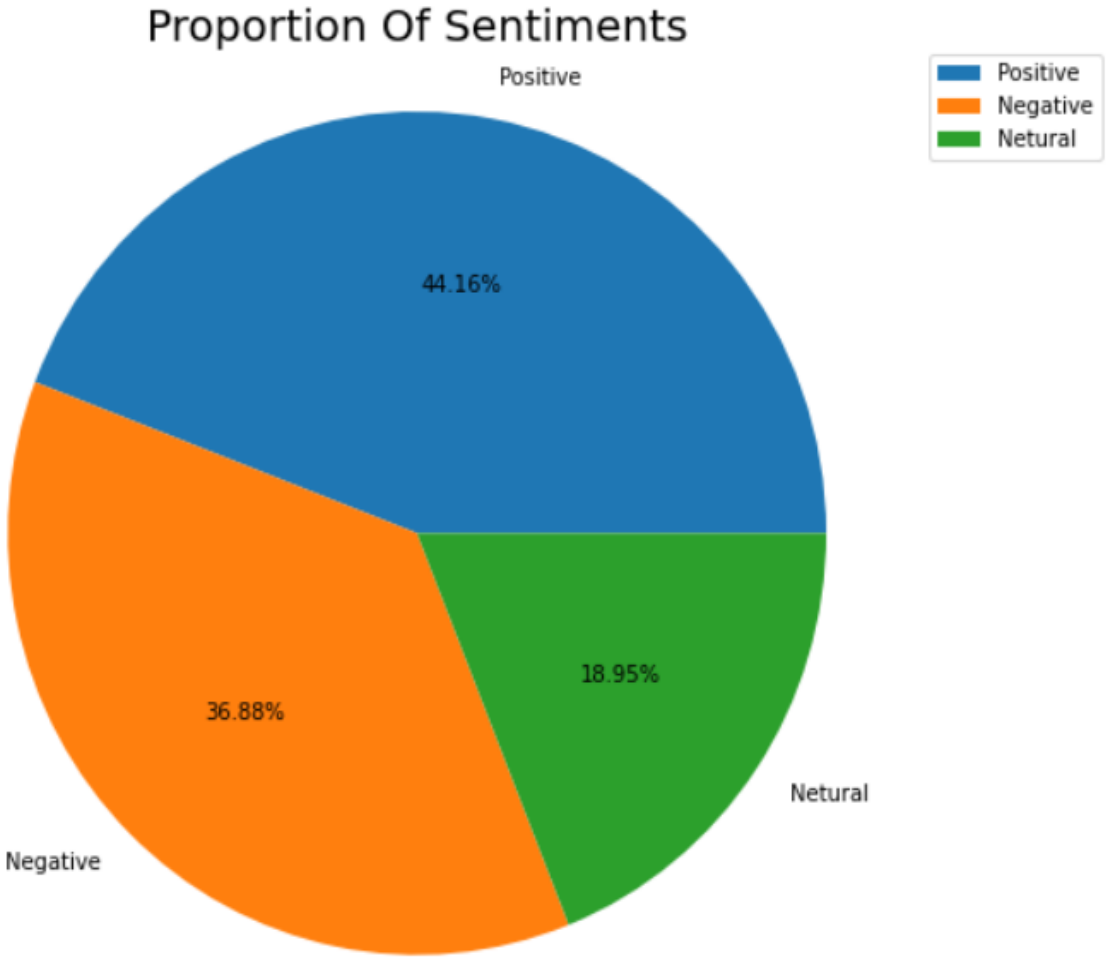
Sentiments



There are five types of sentiments. The above graph shows count of each sentiment.

EDA Continued…

New Sentiments



As there was five types of sentiments – Positive Sentiment, Extremely Positive Sentiment, Negative Sentiment, Extremely Negative Sentiment and Neutral Sentiment. So, we have replaced Extremely Positive Sentiment by Positive Sentiment and Extremely Negative Sentiment by Negative Sentiment. Now we have three types of sentiments – Positive Sentiment, Negative Sentiment and Neutral Sentiment.

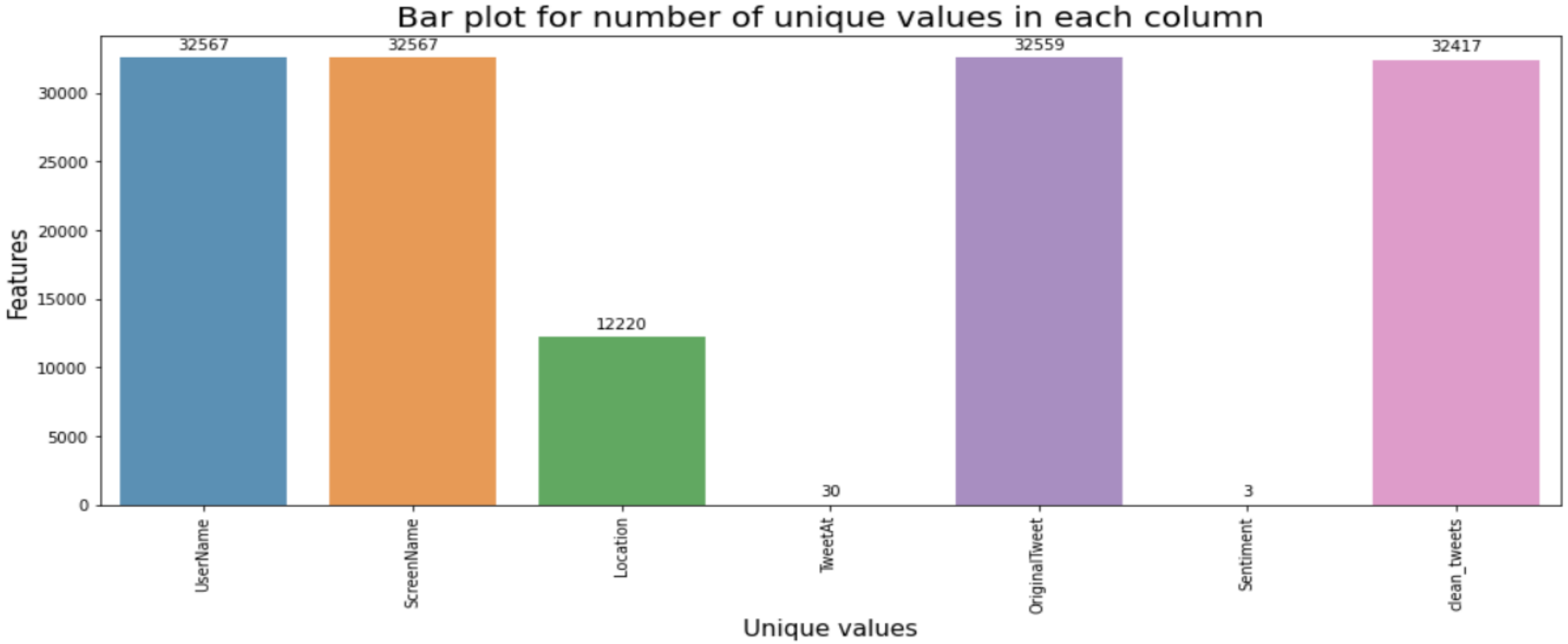
The Pi Chart shows the proportion of each sentiment.

There are 44.16% Positive Sentiments, 36.88% Negative Sentiments and 18.95% Neutral Sentiments.

Positive Sentiments are having higher proportion among all.

EDA Continued…

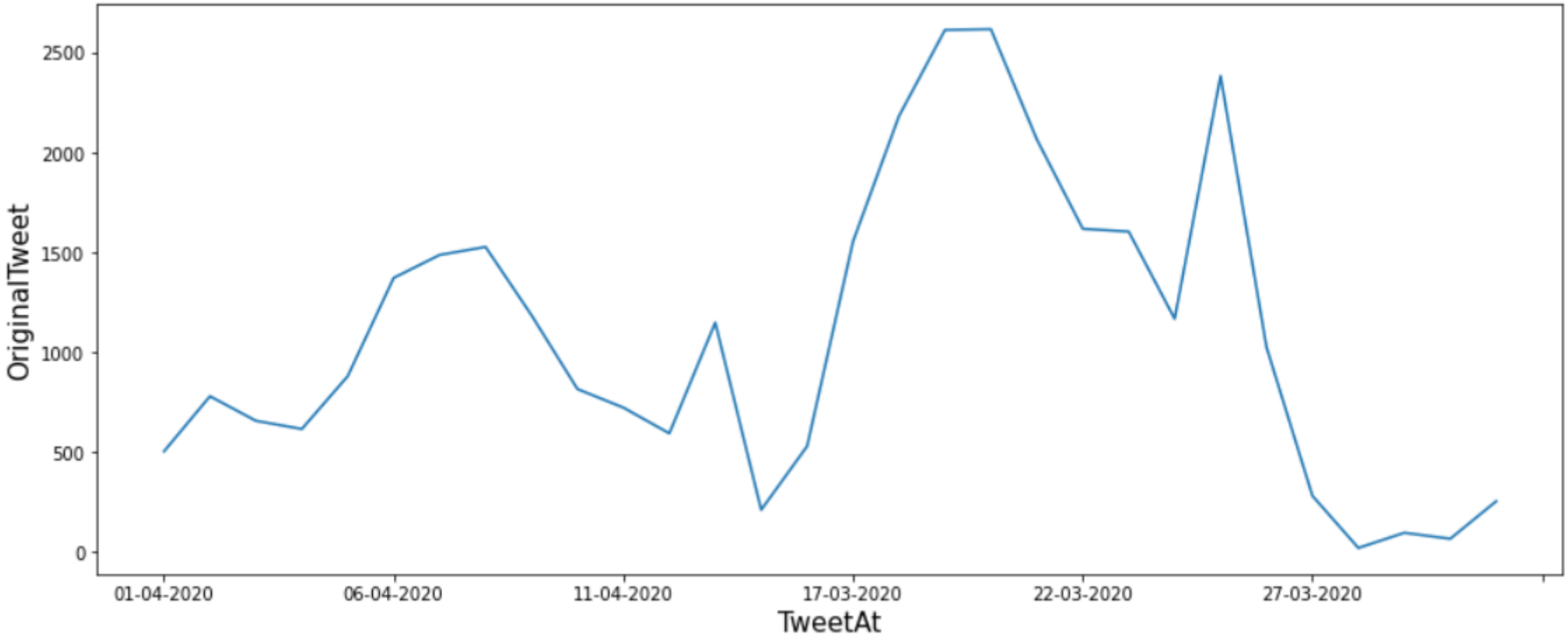
Unique values in each column



The above graph shows unique values in each column.

EDA Continued…

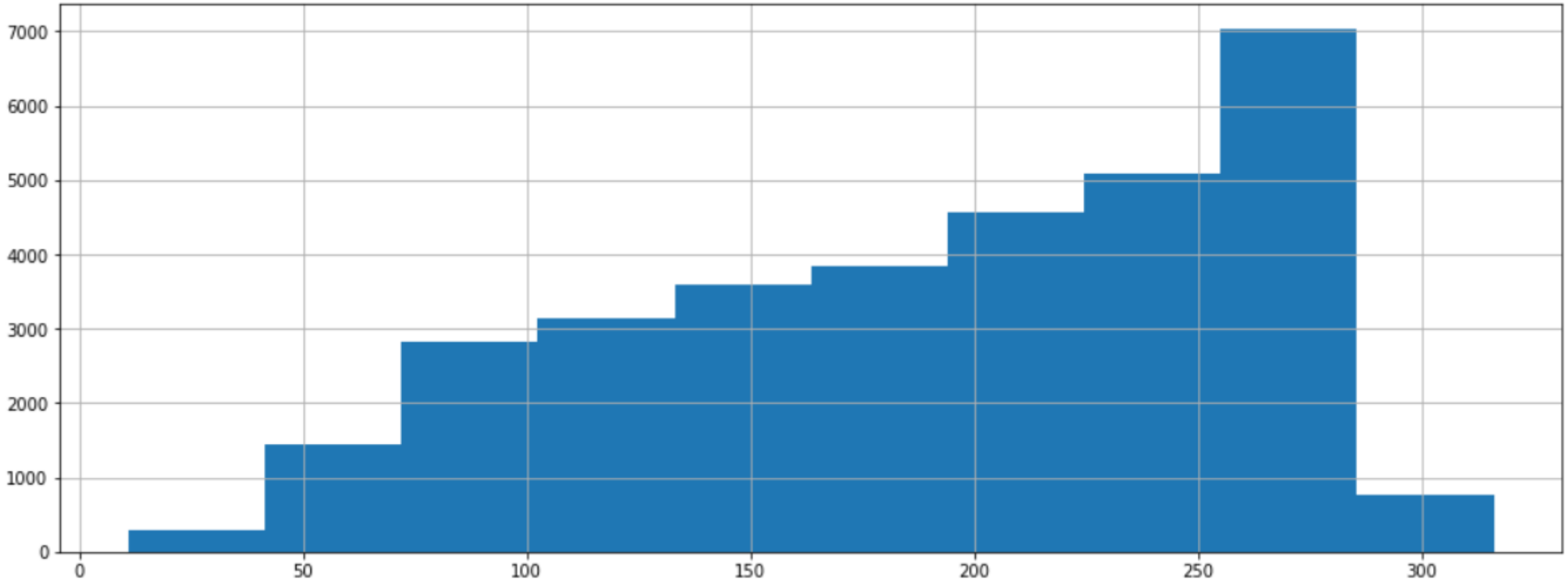
Original Tweet according to ‘TweetAt’ column



The above graph shows OriginalTweet according to TweetAt column.

EDA Continued…

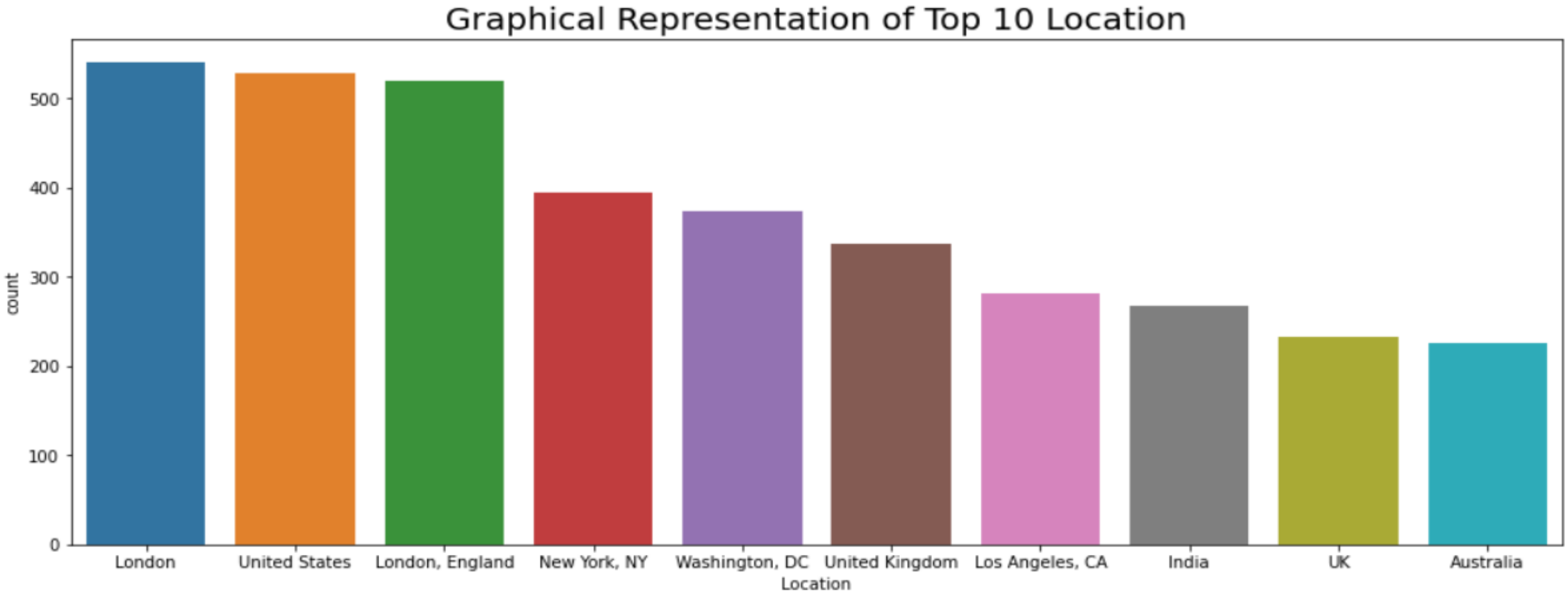
Histogram plot of Original Tweet



The above histogram plot shows original tweets.

EDA Continued…

Top 10 Locations



The above graph shows top 10 locations from where tweets are received. Most tweets received from London followed by United States.

EDA Continued…

Most common words in tweets

EDA Continued…

Word Cloud

Word Clouds are **visual displays of text data – simple text analysis**. Word Clouds display the most prominent or frequent words in a body of text.

Model Preprocessing

Extracting Features From Text

- **Count Vectorizer -** CountVectorizer is a great tool provided by the scikit-learn library in Python. It is used to transform a given text into a vector on the basis of the frequency (count) of each word that occurs in the entire text.

```
Document-1: He is a smart boy. She is also smart.
Document-2: Chirag is a smart person.
```

The dictionary created contains the list of unique tokens(words) present in the corpus

```
Unique Words: ['He', 'She', 'smart', 'boy', 'Chirag', 'person']
```

Here, D=2, N=6

So, the count matrix M of size 2 X 6 will be represented as –

	He	She	smart	boy	Chirag	person
D1	1	1	2	1	0	0
D2	0	0	1	0	1	1

Now, a column can also be understood as a word vector for the corresponding word in the matrix M.

Model Preprocessing

Extracting Features From Text

- **NGram Vectorizer -** Similar to the count vectorization technique, in the N-Gram method, a document term matrix is generated, and each cell represents the count. Count vectorization is a special case of N-Gram where n=1. N-grams consider the sequence of n words in the text; where n is (1,2,3..) like 1-gram, 2-gram. for token pair. Unlike BOW, it maintains word order.

```
“I am studying NLP” has four words and n=4.
if n=2, i.e bigram, then the columns would be – [“I am”, “am reading”, ‘studying NLP”]
if n=3, i.e trigram, then the columns would be – [“I am studying”, ”am studying NLP”]
if n=4,i.e four-gram, then the column would be -[‘“I am studying NLP”]
```

If we choose N as a smaller number, then it may not be sufficient enough to provide the most useful information. But on the contrary, if we choose N as a high value, then it will yield a huge matrix with lots of features. Therefore, N-gram may be a powerful technique, but it needs a little more care.

Model Preprocessing

Extracting Features From Text

- **TF-IDF -** Count Vectorizer method is simple and works well, but the problem with that is that it treats all words equally. As a result, it cannot distinguish very common words or rare words. So, to solve this problem, TF-IDF comes into the picture! Term frequency-inverse document frequency (TF-IDF) gives a measure that takes the importance of a word into consideration depending on how frequently it occurs in a document and a corpus.

TF – Term Frequency

Term frequency denotes the frequency of a word in a document.
It is the percentage of the number of times a word (x) occurs in a particular document (y) divided by the total number of words in that document

$$TF(\boldsymbol{term}) = \frac{Number\ of\ times\ \boldsymbol{term}\ appears\ in\ a\ document}{Total\ number\ of\ items\ in\ the\ document}$$

The formula for finding Term Frequency is given as:

```
tf ('word') = Frequency of a 'word' appears in document d / total number of words in the documen
```

For Example, Consider the following document

```
Document: Cat loves to play with a ball
```

Model Preprocessing

Extracting Features From Text

Inverse Document Frequency - It measures the importance of the word in the corpus. It measures how common a particular word is across all the documents in the corpus.
It is the logarithmic ratio of no. of total documents to no. of a document with a particular word.

The difference in the TF-IDF method is that each cell doesn't indicate the term frequency, but contains a weight value that signifies how important a word is for an individual text message or document

$\mathtt{tf}(t, d)$

	blue	bright	can	see	shining	sky	sun	today
1	1/2	0	0	0	0	1/2	0	0
2	0	1/3	0	0	0	0	1/3	1/3
3	0	1/3	0	0	0	1/3	1/3	0
4	0	1/6	1/6	1/6	1/6	0	1/3	0

- TF-IDF: Multiply TF and IDF scores, use to rank importance of words within documents
  - Most important word for each document is highlighted

x

$\mathtt{idf}(t, D)$

	blue	bright	can	see	shining	sky	sun	today
	0.602	0.125	0.602	0.602	0.602	0.301	0.125	0.602

$\mathtt{tfidf}(t, d, D) = \mathtt{tf}(t, d) \cdot \mathtt{idf}(t, D)$

	blue	bright	can	see	shining	sky	sun	today
1	**0.301**	0	0	0	0	0.151	0	0
2	0	0.0417	0	0	0	0	0.0417	**0.201**
3	0	0.0417	0	0	0	**0.100**	0.0417	0
4	0	0.0209	**0.100**	**0.100**	**0.100**	0	0.0417	0

TF-IDF score computation. [Image Source]

Model Training

Algorithm Used

- Logistic Regression
- Decision Tree
- Random Forest
- Gradient Boosting

Performance Metrics and Accuracy

Performance of Logistic Regression Model

Accuracy : 0.771063620732006 8

Precision : 0.785789615433756 9

Recall : 0.771063620732006 8

Performance of Decision Tree Classifier

Accuracy : 0.612994350282485 8

Precision : 0.610901348236181 1

Recall : 0.612994350282485 8

Performance of Random Forest Classifier

Accuracy : 0.724883321051338 8

Precision : 0.728233195715117 8

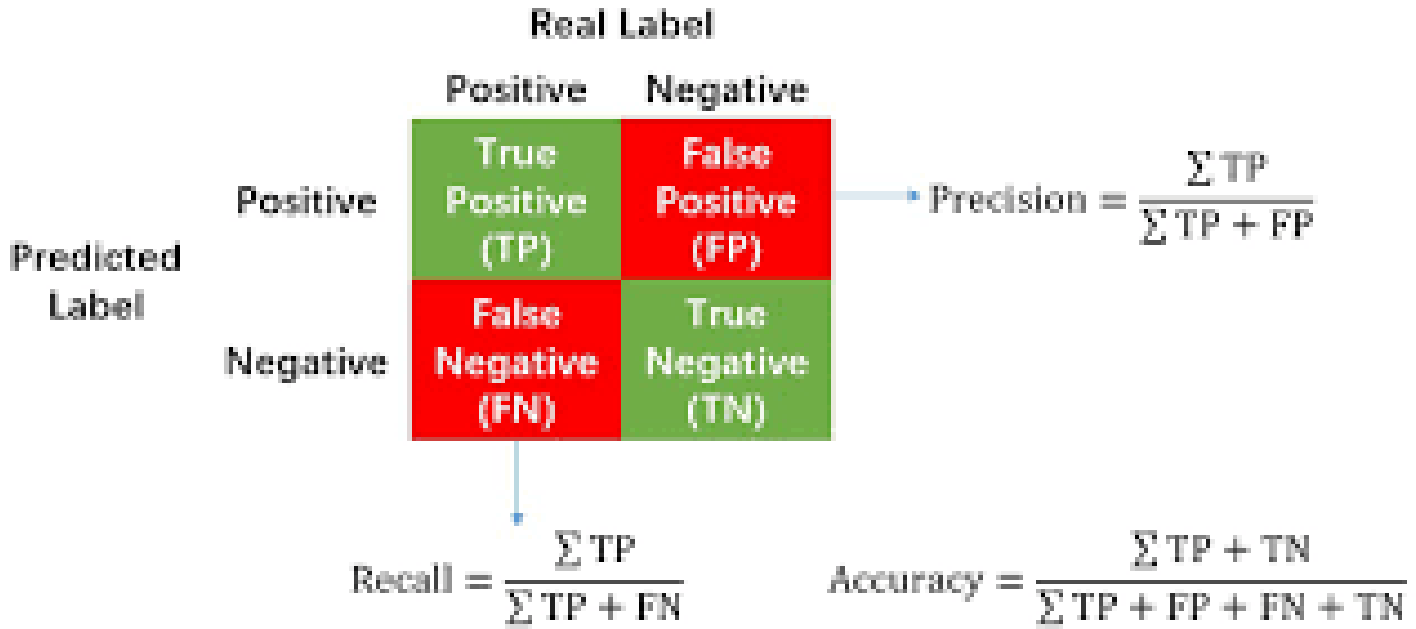
Recall : 0.724883321051338 8

Performance Metrics and Accuracy

Performance of Gradient Boosting Classifier

Accuracy : 0.6545074920167036
Precision : 0.6962747491301503
Recall : 0.6545074920167036

Precision Accuracy And Recall



Precision Accuracy And Recall

AI

Precision

Precision is the proportion of correct predictions among all predictions of a certain class. In other words, it is the proportion of true positives among all positive predictions.

$$Precision = \frac{TP}{FP + TP}$$

Accuracy

Accuracy is the proportion of examples that were correctly classified. More precisely, it is sum of the number of true positives and true negatives, divided by the number of examples in the dataset.

$$Accuracy = \frac{TP + TN}{FP + FN + TP + TN}$$

Precision Accuracy And Recall

AI

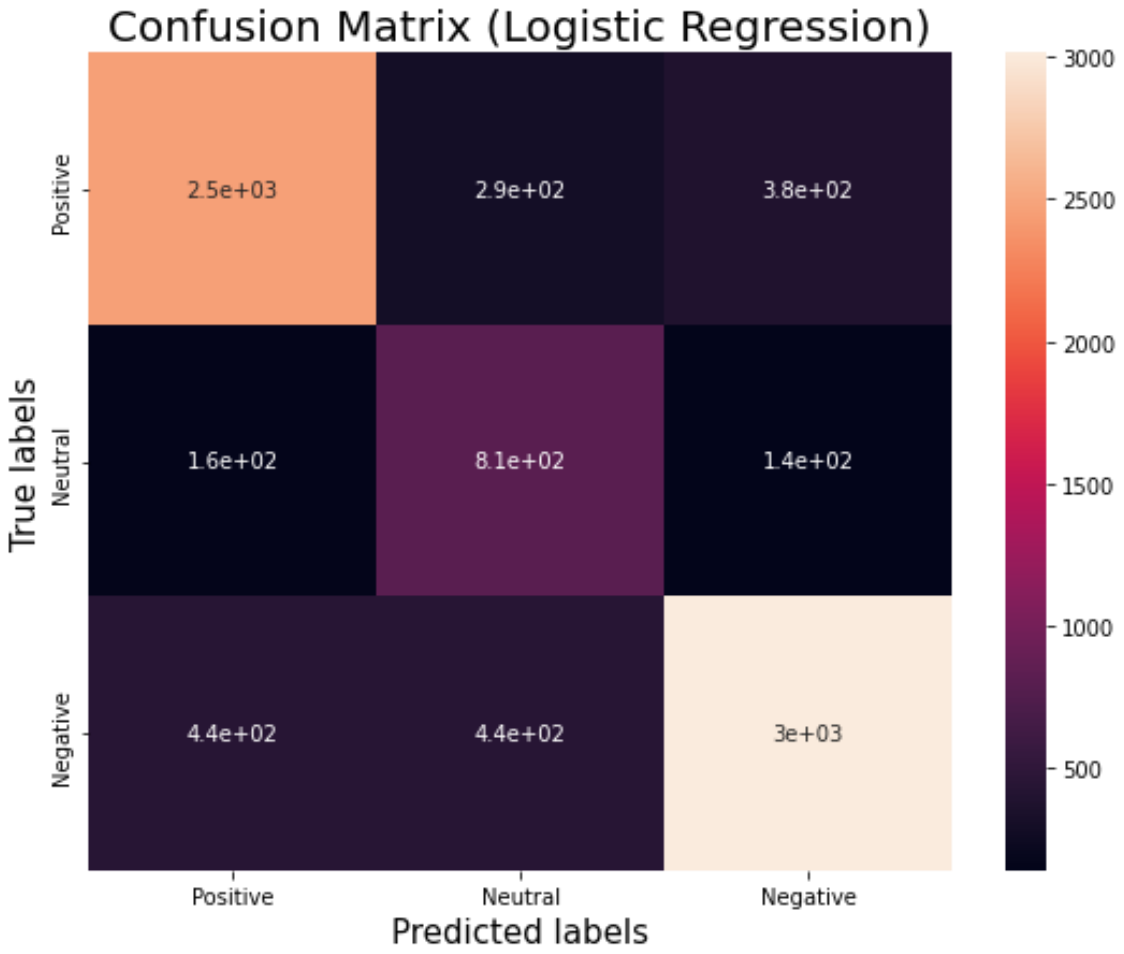
Recall

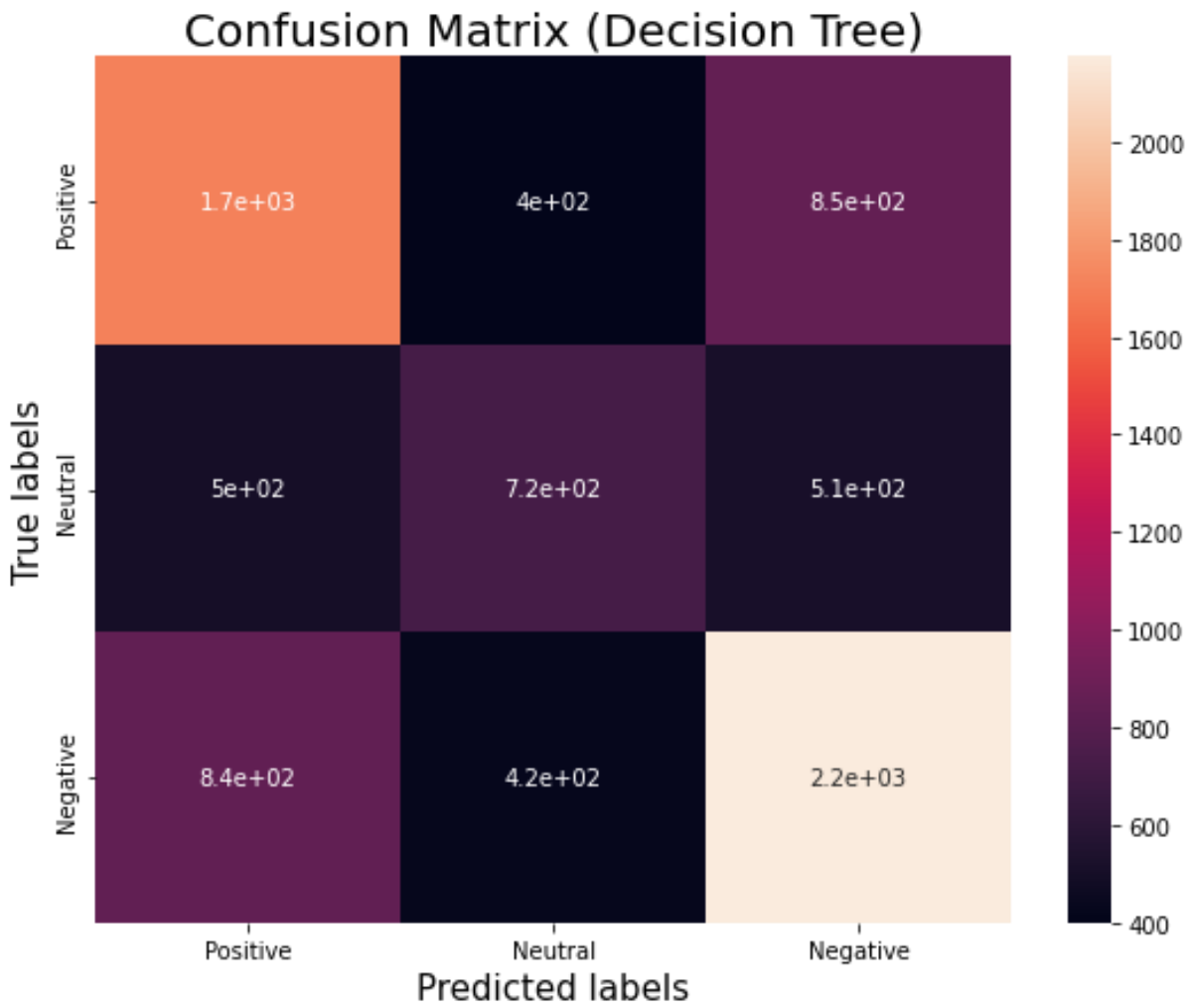
Recall is the proportion of examples of a certain class that have been predicted by the model as belonging to that class. In other words, it is the proportion of true positives among all true examples.

$$Recall = \frac{TP}{FN + TP}$$

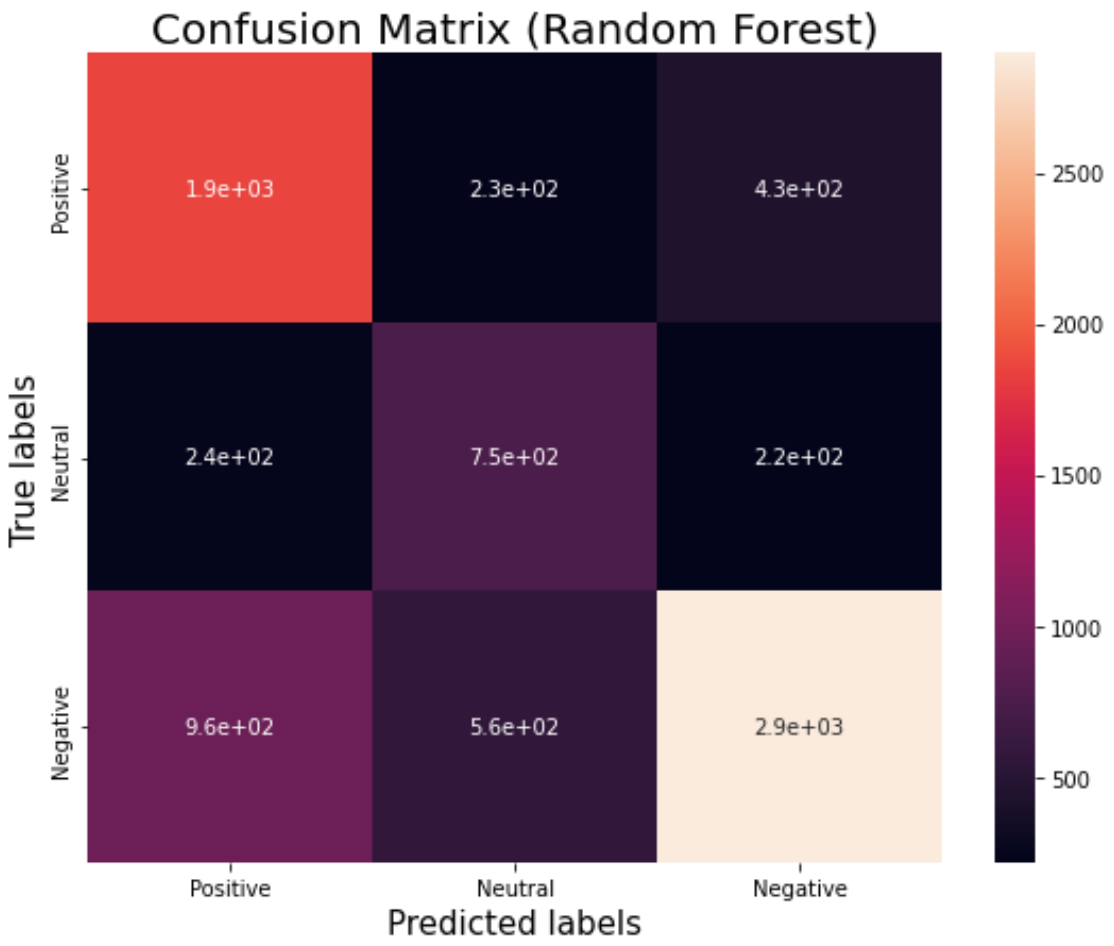
Confusion Matrix

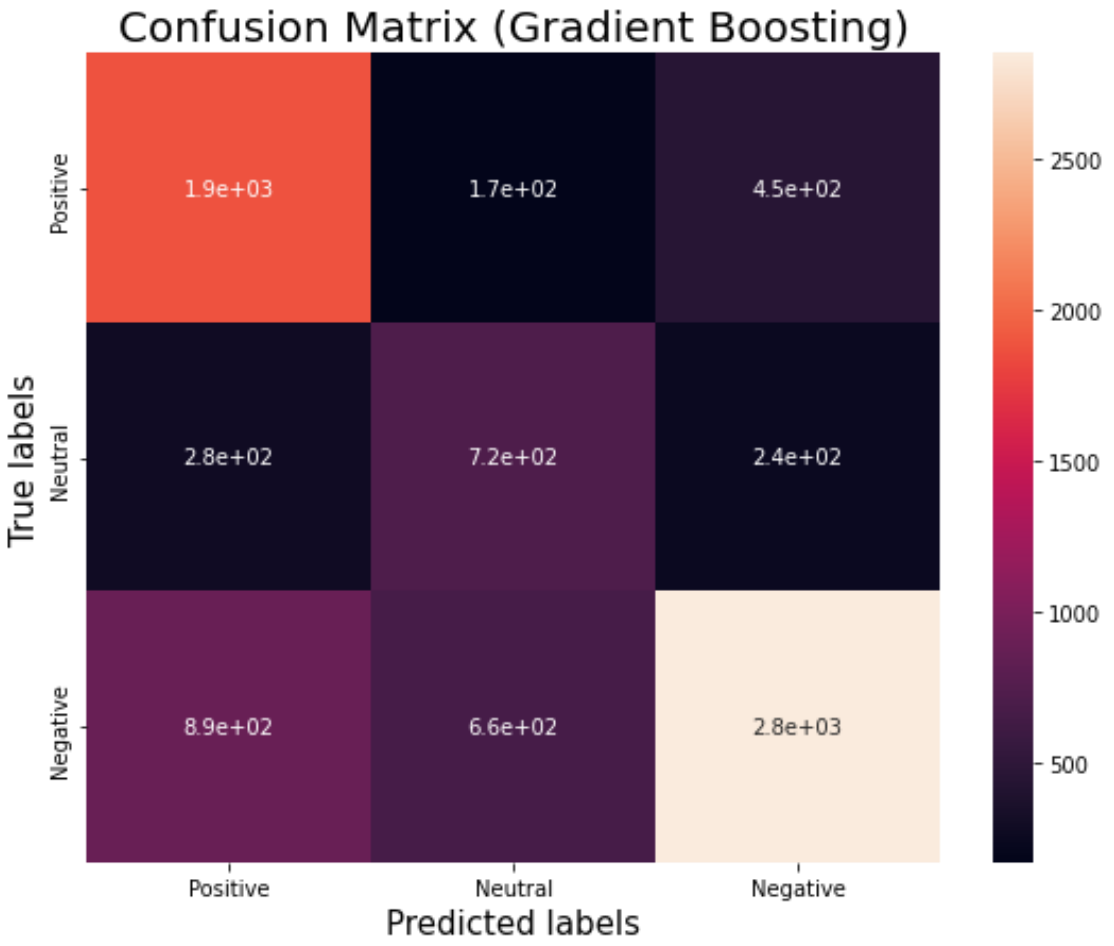
A **Confusion matrix** is an N x N matrix **used for** evaluating the performance of a classification model, where N is the number of target classes.





Confusion Matrix





Conclusion

Conclusion on EDA:

- Original Dataset contains 6 columns and 41157 rows.
- Location column contains null values. So, we have dropped the null values.
- And we added a new column "clean_tweets" after cleaning the tweets.
- After dropping and adding a new column, now we have 7 columns and 32567 rows.
- In order to analyze the data we required only two columns "OriginalTweet" and "Sentiment".
- The columns such as "UserName" and "ScreenName" does not give any meaningful insights for our analysis.
- There are five types of sentiments - Extremely Positive, Positive, Extremely Negative, Negative and Neutral.
- We have renamed the Extremely Positive and Extremely Negative sentiments to Positive and Negative respectively. And we are left with three types of sentiments - Positive, Negative and Neutral.
- The pie chart shows the proportion of sentiments.
- Bar plot for unique values shows us the number of unique values in each column.
- The graphical representation of top 10 locations shows us that most of the tweets came from London followed by United States.

Conclusion on Model Training:

- At the end we conclude our classification project with five models namely - Logistic Regression Model, Decision Tree Classifier, Random Forest Classifier and Gradient Boosting Classifier.
- We are getting the highest accuracy of about 77% with Logistic Regression.

Thank You!!