Capstone Project-3

Coronavirus Tweet Sentiment Analysis

Project by- Potdar Vinayak

Contents

Abstract

Covid -19 was a great disaster that shook the whole world. The world stopped and was forced to quarantine, which caused some discomfort and setback. We saw a surge of people tweeting online for asking help or providing help. Some were criticizing the government's work and approach, a few were angry about the situation.

In this experiment we built a predictive model, which can understand and extract the sentiment of a person through his/her tweet.

Problem Statement

This challenge asks you 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.

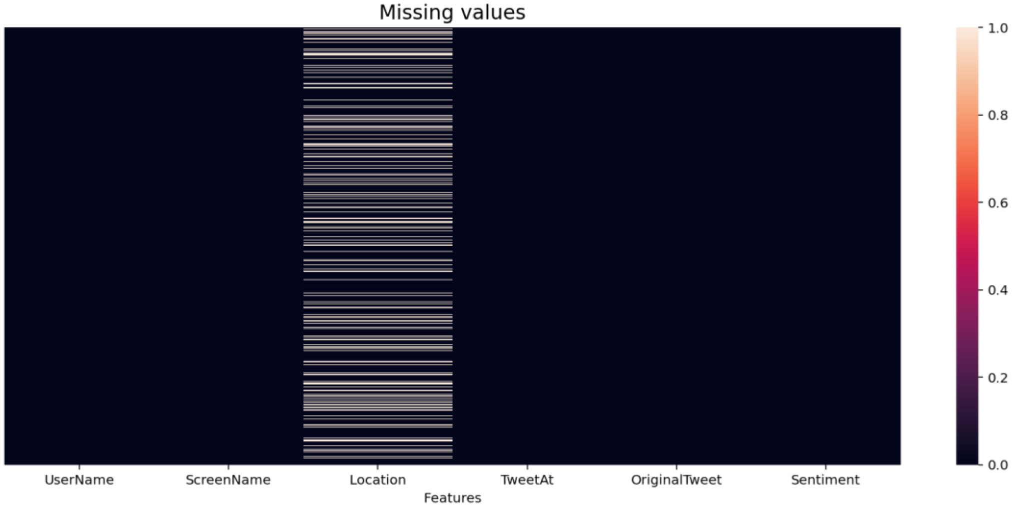
Understanding the Dataset

The provided dataset has 41157 tweets from various users. We have 6 listed features, one of them is the dependent variable i.e. sentiment. The rest of the features can be considered the meta data for the tweets, as they represent the location, username(id), tweet date and the screen name.

Null Value Analysis and Treatment

Out of the whole dataset, there is just one feature which consists of missing values. Location is the feature with missing values. Almost 20% of the records of location are missing.

Since this data is not going to be of any help in the sentiment analysis process, we can leave this feature untouched. This feature can be of help in the EDA.



EDA

1. When did the tweets start?

- **The tweets in this dataset have been captured from march and April of 2020.**
- **The tweets regarding covid-19 started from the mid of march.**

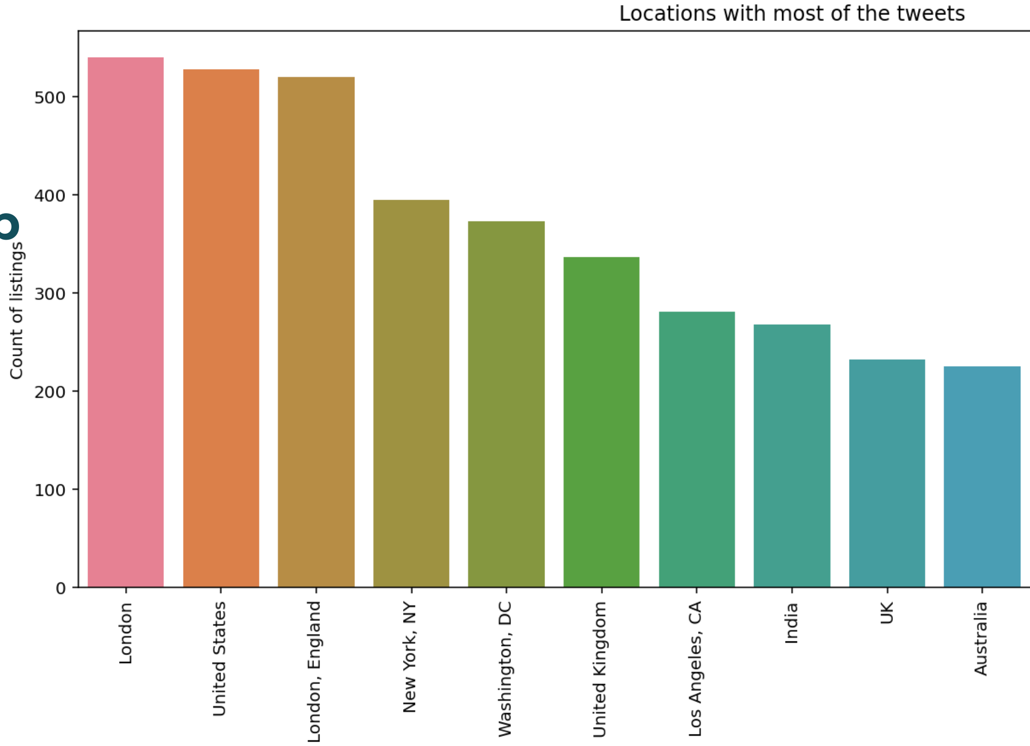
```
# Understanding when the tweets started to appear regarding Covid-19.
dataset.TweetAt.value_counts(sort= True)
```

```
20-03-2020    3448
19-03-2020    3215
25-03-2020    2979
18-03-2020    2742
21-03-2020    2653
22-03-2020    2114
23-03-2020    2062
17-03-2020    1977
08-04-2020    1881
07-04-2020    1843
06-04-2020    1742
24-03-2020    1480
09-04-2020    1471
13-04-2020    1428
26-03-2020    1277
05-04-2020    1131
10-04-2020    1005
02-04-2020     954
11-04-2020     909
03-04-2020     810
12-04-2020     803
04-04-2020     767
16-03-2020     656
01-04-2020     630
27-03-2020     345
31-03-2020     316
14-04-2020     284
29-03-2020     125
30-03-2020      87
28-03-2020      23
Name: TweetAt, dtype: int64
```

EDA

2. Origin of Tweets and Top Tweeters

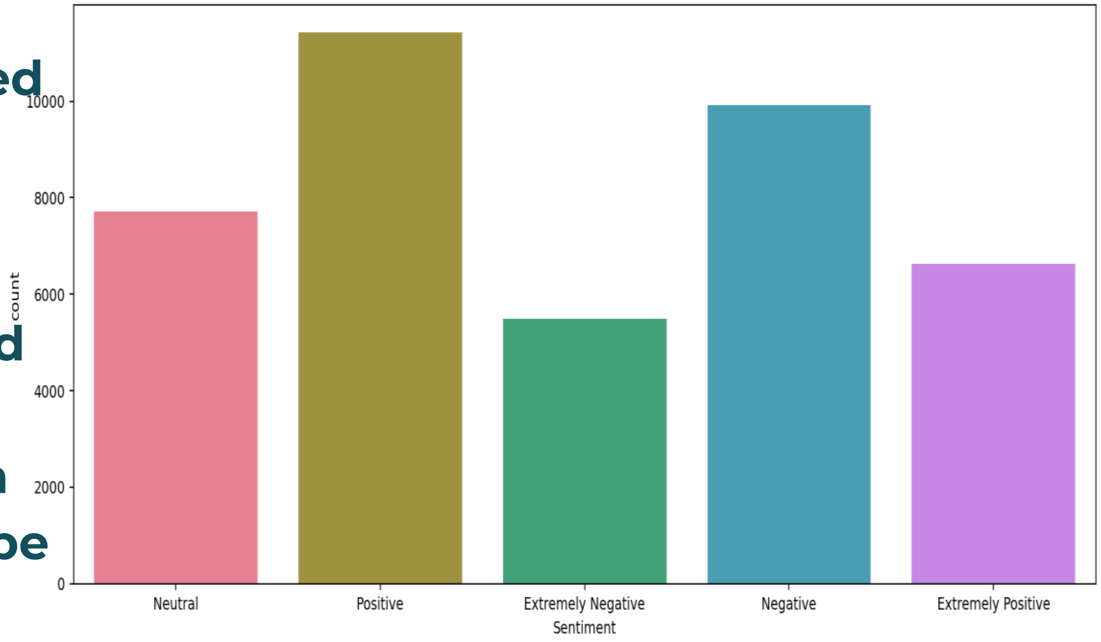
- Here we can see that the top tweeters are from UK, US, India, Australia.
- We can see that unique locations are 12221 but a lot of redundant locations have been mentioned e.g. ‘London’ and ‘London, England’



EDA

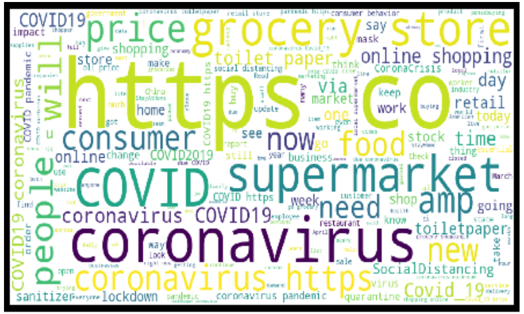
3. Sentiment Distribution

- In order to evaluate the modeling process we need to understand which evaluator to give more importance.
- For dataset with balanced distribution of classes(sentiments) would mean that Accuracy score can be given more importance.

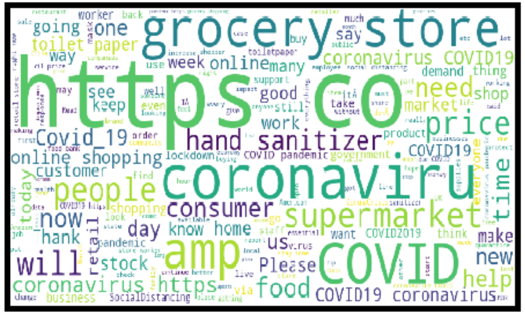


EDA

4. Major Keywords in each sentiment tweet

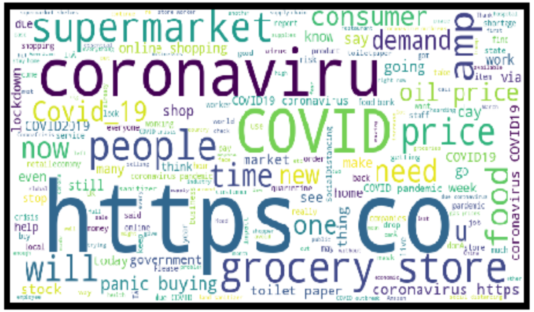


Neutral



Positive

Extremely Positive



Negative

Extremely Negative

EDA

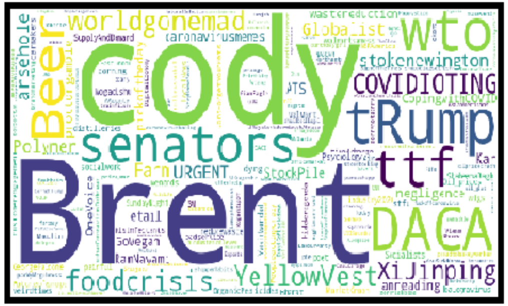
5. Getting Word Clouds for unique hashtags for each 'Sentiment'

Neutral

Positive

Extremely Positive

Negative



Extremely Negative

Data Preprocessing

In this phase we need to start processing the data, in order to feed it to the classification models. These are the actions we took on the data-

- Remove URL
- Remove Tagged Usernames
- Remove Punctuations and Special Characters
- Remove Stop Words
- Stemming

Utility Functions

We created some utility functions for the data preprocessing phase. Each of them handled the following process.

- `remove_urls (ot)` – **Remove URL**
- `remove_user(ot)` – **Remove Tagged Usernames**
- `remove_punctuation(ot)` – **Remove Punctuations and Special Characters**
- `stopword(ot)` – **Remove Stop Words**
- `stemming(ot)` – **Stemming**

Vectorization

The Machine Learning models cannot understand text data for training. In order to make it compatible with the training models, we need to convert them into integer format.

This is where the vectorization comes in the picture. We used CountVectorizer() as our vectorization function. It emphasizes on the number of occurrence of the word as well.

Model Evaluation and Selection

We performed two experiments where, we had a set with 5 classes and other with 3 where extremely positive and extremely negative were combined with positive and negative respectively.

In both the scenarios, CatBoost turned out to be the best fitting model with 0.8069.

	Classification Models	Testing score
3	CatBoost	0.806973
2	Random Forest	0.762026
0	Support Vector Machines	0.748907
1	Naive Bayes Model	0.681487

5 Classes

	Classification Models	Testing score
3	CatBoost	0.806973
0	Support Vector Machines	0.775267
2	Random Forest	0.767736
1	Naive Bayes Model	0.681487

3 Classes

Conclusion

Here, we conducted two experiments with 5 classes and 3 classes. The first was conducted with hyperparameter tuning and Catboost turned out to be the best performing model with 0.8069 testing accuracy. In the second scenario we got Catboost as the best performing model again. Here SVM improved its performance drastically as compared to the first scenario.