Project Title : Sentiment Analysis : Predicting sentiment of COVID-19 tweets

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.

Data Description –

You are given the following information:

• Location – gives the location from which tweet has been done in the form of country name.

• Tweet At – tweetted date

• Original Tweet – Tweet

• Sentiment – Gives Sentiment type :

Positive

Negative

Neutral

Extremely Positive

Extremely Negative

Data Analysis –

• Sentiment column has total 41157 entries.

• In that, Positive: 11422

Negative: 9917

Neutral: 7713

Extremely Positive: 6624

Extremely Negative: 5481.

• In data most of the tweets are positive as compared to other sentiments.

• Graph shows top 10 locations which has highest count of tweets.

In that London is on the Top.

Data Processing-

Raw text consist of noise means punctuations, special characters,numbers,symbols and stop words which don’t carry much weightage in context to the next.

In this process we remove this noise from text and make it proccesible.

• Removing @user

• Removing Hashtags

• Removing special characters

• Removing punctuations

• Removing stopwords

• Stemming

Stemming is a technique used to extract the base form of the words by removing affixes from them.

• Lemmatization

Lemmatization is a linguistic term that means grouping together words with the same root or lemma but with different inflections or derivatives of meaning so they can be analyzed as one item.

Tokenization

● Tokenization basically refers to splitting up a larger body of text

into smaller lines, words or even creating words for a non-English language.

● Tokenization can be done in python by using NLTK library’s

word\_tokenize() function.

Vectorization

● Vectorization is a technique to implement arrays without the use of

loops. Using a function instead can help in minimizing the running time

and execution time of code efficiently.

● We choose count vectorizer as our vectorizer with minimum document

Frequency = 10.

● It will create a sparse matrix of all words and the number of times they

are present in a document.

Models Used –

1. Naïve Bayes

2. Logistic Regression

3. Random Forest

4.XGBoost

5.Support Vector Machines

6.CatBoost

7. Stochastic Gradient Descent

Models Result –

For multi-class classification:

CatBoost Model gives highest accuracy score - 62%

For Binary Classification:

Logistic Regression Gives highest accuracy score- 87%