Sentiment Analysis: Predicting sentiment of COVID-19 tweets

**Abstract:**

Public sentiments or customer feedbacks are one of the most important aspects in terms of business discussion and government policy making. Sentiment analysis is one of the most effective ways to understand the public/customers emotion, requirement and concern.

COVID-19 one the deadliest pandemic the world has ever witnessed. During this period people faced various concern like oxygen storage, economic instability, lack of medical resources, food scarcity etc. People from different parts of the world shared their conditions through twitter and other social media platform.

Our goal is to analyse the tweet collected from twitter and build a classification model to classify the tweets based on its sentiments. The sentiments may be of various types such as positive, negative, neutral etc.

**Problem Statement**

In this challenge we have 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. This is a supervised ML classification problem.

**Introduction**

The study of sentiment analysis of various tweets during COVID-19 can be helpful for different stakeholders.

For example, government can make use of this information for making new policies by understanding people’s condition and all the challenges they have faced.

Various profit organisation can understand the public requirement and can offer those product and services which have more demand but less supply. For instance, one of the various tweets is talking about scarcity of masks and toilet paper.

NGO’s can also make strategy of how to manage the medical and economical requirement with limited resources.

**Overview of data**

The Features of The Dataset:

1. UserName- Encoded Username of the user
2. ScreenName- Encoded version of the name displayed on the user screen
3. Location- The location of the tweet
4. Tweet At- Date of the tweet
5. Original Tweet- The text/content of the tweet
6. Sentiment-the sentiment of the tweet. It has five labels-Positive, Negative, Neutral, Extremely Positive, Extremely Negative

**Steps Involved**

**1.EDA (Exploratory data analysis)**

1. Analyse the basic information of the dataset like number of observations and features, Data Type of different features, null values of each feature.
2. Take a deep dive into “original\_tweet” column to understand a brief overview of tweets with different sentiments. Also find the tweets with highest and fewest number of characters.
3. Finding top 20 location with highest number of tweets

## Top 20 Date with Highest Number of Tweets

## Finding percentage distribution of tweets in terms of sentiments

## Finding Top 50 hashtags of all category

## 2. Data Processing

1. Remove noisy data such as URLs, user names, Punctuations, Numbers, and Special Characters from tweets.
2. Tokenize the tweets using nltk library and removing the stop words such as is, are, and etc which does not contribute any valuable information to the model.
3. c) Apply nltk WordNetLemmatizer function to convert the words to its base word.

**3. Conclusions from the data**

1. Plotting graphs which gives information about our data
2. Pie chart for percentage distribution of sentiment of tweets. Positive sentiment tweets have the largest share.
3. Graph for top 20 locations with highest of tweets
4. Graph for top 20 dates with highest of tweets.
5. Graph for top 50 hashtags. coronavirus is the top hashtags with a count of 12954. Total number of hashtags – 81151, number of unique hashtags - 18367
6. Number of positive hashtags – 22055, number of unique positive hashtags – 6772
7. Number of negative hashtags – 18172, number of unique negative hashtags – 5601
8. Number of neutral hashtags – 19678, number of unique neutral hashtags – 6502
9. Number of extremely positive hashtags – 12536, number of unique extremely positive hashtags – 4425
10. Number of extremely negative hashtags – 8709, number of unique extremely negative hashtags – 2732

**4. Training the model**

1. We cannot pass the textual data directly to the ML algorithm. These words need to then be encoded as integers, or floating-point values. We can do it using following methods
2. Count Vectorizer Method

* Count vectorizer converts a collection of text documents to matrix of integers. Where each integer represents the frequency of the word token in that document.
* This can be implemented in python through the CountVectorizer Method. This implementation produces a sparse representation of the counts using scipy.sparse.csr\_matrix.

1. TF-IDF Method

* TF-IDF method represents not only the count of the word token in the document it also reflects how important a word is to a document in the collection of corpuses. This can be implemented in python through TfidfVectorizer.
* TF = (Number of times term t appears in a document) / (Number of terms in the document)
* IDF = log(N/n), where, N is the total number of documents and n is the number of documents the term t has appeared in.
* TF-IDF = TF\*IDF

1. Selecting the independent and dependent variables from the dataset
2. Splitting the model into train and test sets.
3. Fitting different models on train set for multi class classification. The targeted classes are positive, negative, neutral, extremely positive, extremely negative.
4. Converting the original multi class targeted variable into binary targeted variable. The targeted classes are positive, negative. Fit different model on this converted binary classification.
5. Finding the predicted values from the trained model for train as well as test data.

**5. Evaluating our models**

1. Finding accuracy score for train as well as test data.
2. Displaying classification report for each model which contains precision, recall and f1-score.
3. Getting a confusion matrix for every model.
4. Displaying ROC-AUC curve for each model.

**Algorithms Used:**

1. **Naive Bayes Classifier**

Naive Bayes classifiers are a group of class algorithms primarily based on Bayes’ Theorem. It is a generative model which uses probability estimates and likelihood to model the data.

The essential Naive Bayes assumption is that every feature makes an independent and equal contribution to the outcome.

Bayes’ Theorem reveals the chance of an event given the probability of any other event that has already occurred. Bayes’ theorem is said mathematically as the subsequent equation:

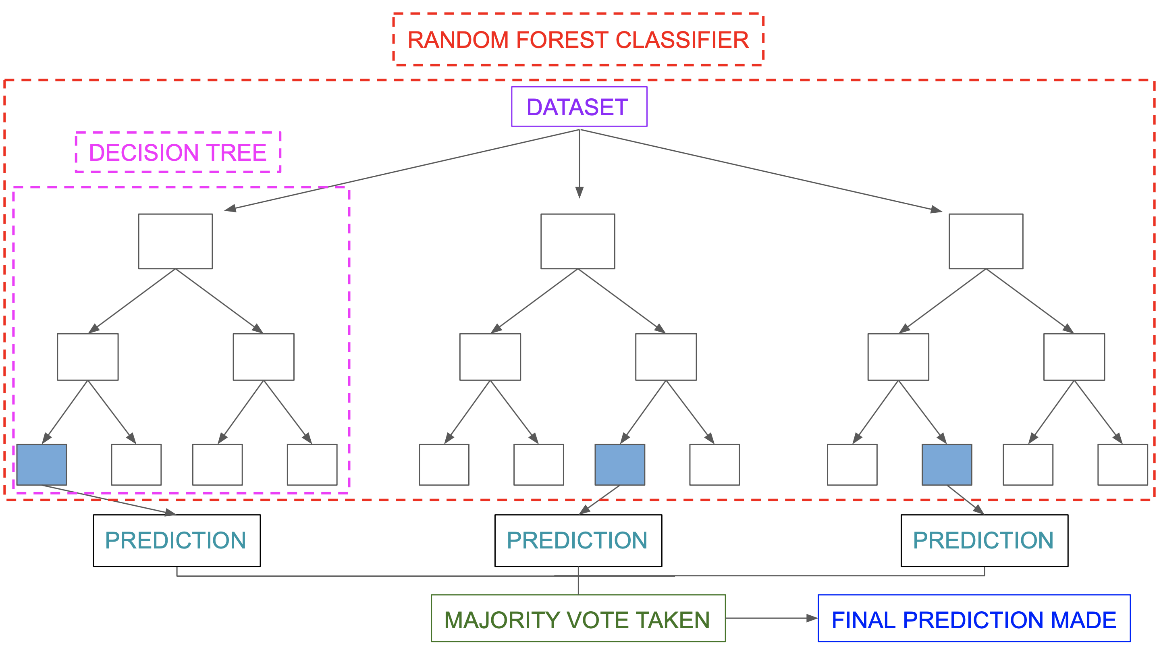
P (Y | X) =P (X | Y) P(Y) / P(X)

posterior=prior ∗ likelihood / evidence

This model can be used for both multi class and binary classification.

1. **Random Forest Classifier**

Every decision tree has an excessive variance, however, while we integrate them all collectively in parallel then the ensuing variance is low as every selection tree receives perfectly trained on that sample data, and as a result, the output doesn’t rely upon one decision tree however on more than one decision tree. In the case of a classification problem, the final output is taken through the use of majority voting. In the case of a regression problem, the final output is the average of output of different trees.



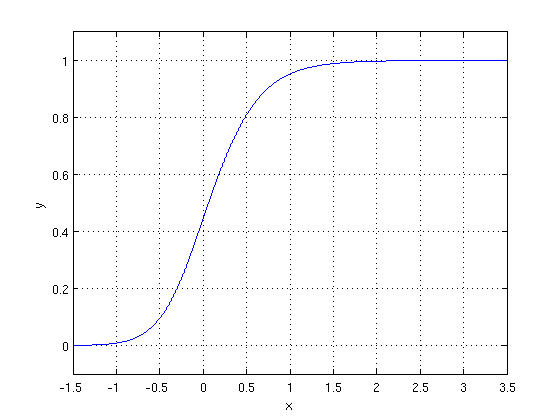
Random Forest is an ensemble approach able to act on each regression and classification with the usage of multiple discussion trees and a method referred to as Bootstrap and Aggregation, generally referred to as bagging.

1. **Logistic Regression**

Logistic regression is essentially a supervised classification algorithm. In a classification problem, the target variable (or output), can take discrete values for a given set of features (or inputs).

Just like Linear regression assumes that the data follows a linear function, Logistic regression models the statistics with the use of the sigmoid function. Sigmoid function also called as logistic function given by:

f(x)= 1/1+e ^(-x)



1. **XGBOOST**

XGBOOST is an ensemble modelling approach that tries to construct a robust classifier from the range of weak classifiers.

Gradient Boosting is a famous boosting algorithm. In gradient boosting, every predictor corrects its previous error. XGBOOST is an extension of gradient boosting.

Optimization and Improvement

Regularization:

XGBOOST uses both Lasso and Ridge Regression regularization

Tree Pruning: XGBOOST uses max\_depth parameter to restrict overfitting

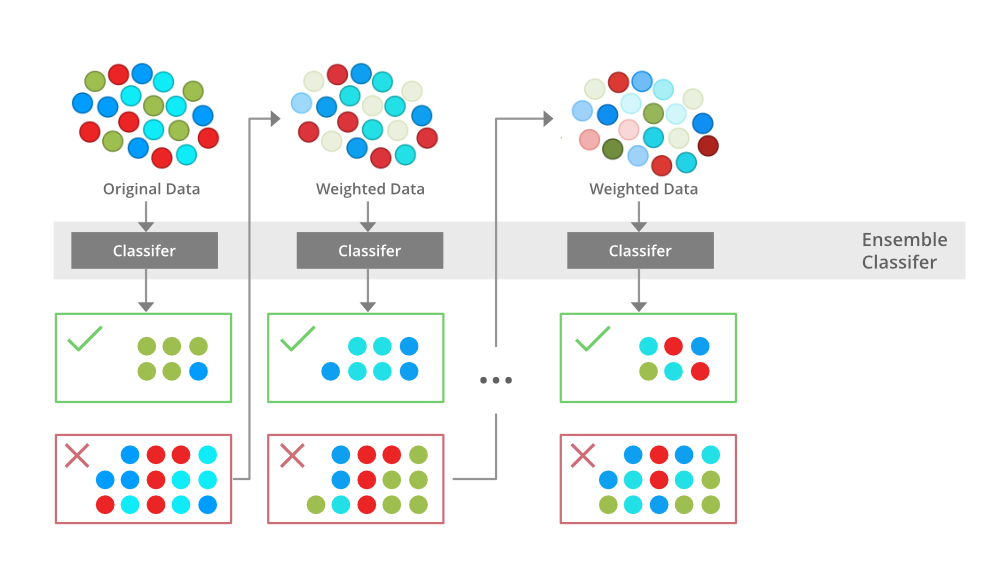
Cross-validation: XGBOOST implementation comes with a integrated cross-

validation method.

It also supports parallelization (it can generate the different nodes of tree parallel), can

use the hardware resources efficiently through Cache-Awareness, it can also handle

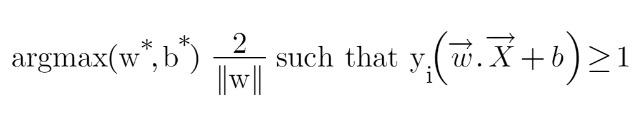
sparse data efficiently.



1. **Support Vector Machine Classifier**

Support Vector Machine (SVM) is a Supervised Machine Learning Algorithm used for both classification and regression.

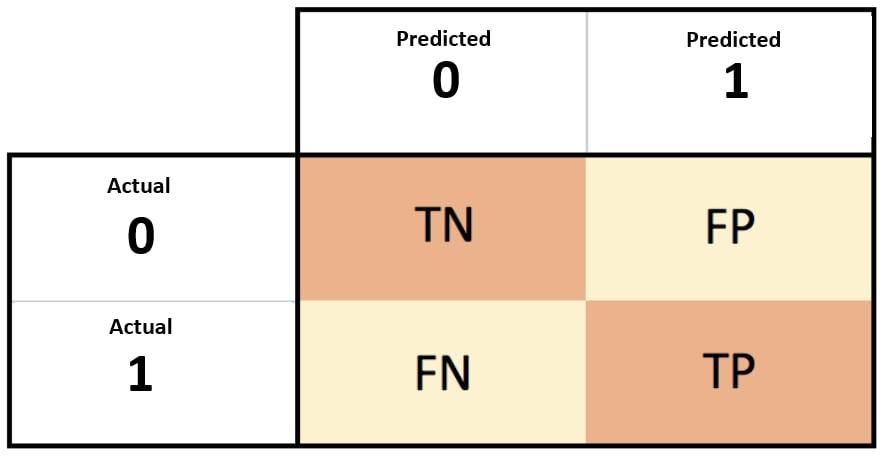
Basically, SVM reveals a hyper-plane that creates a boundary among the different classes of data. In 2-dimensional space, this hyper-plane is not anything but a line. SVM works well without any changes for linearly separable data. Kernel functions can be used to map data to higher dimensions when there is non linearity.

In SVM we try to increase the margin between two hyperplanes passing through the support vectors by optimising the following equation 

**Model Performance:**

1. **Confusion Matrix**

It is a table which is used in classification problems to assess where errors are made in the model. The rows represent the actual classes. While the columns represent the predictions we have made. Using this table, it is easy to see which predictions are wrong.



1. **Accuracy**

Accuracy is the fraction of predictions our model made correct.

Accuracy = Number of correct predictions / Total number of predictions. In terms of the confusion matrix, it is given by: TP+TN/TP+TN+FP+FN

1. **Precision**

Precision tells the quality of positive predictions made by the model. Precision refers to the number of true positives divided by the total number of positive predictions.

In terms of the confusion matrix, it is given by: TP/FP+TP

1. **Recall**

Recall is defined as the number of true positives divided by the total number of elements that actually belong to the positive class. In terms of the confusion matrix, it is given by: TP/FN+TP

1. **ROC AUC**

AUC – ROC curve is a performance measurement for classification problems at different threshold values. ROC is a probability curve, and AUC represents the degree or measure of separability. It tells how significantly model is able of distinguishing between classes. More the AUC, better the model is at forecasting 0s as 0s and 1s as 1s.

**Hyper Parameter Tuning:**

A machine learning model is defined as a mathematical model with a number of parameters to be learned from data. By training the model with existing data, we are able to adjust the model parameters.

However, there is another kind of parameters, known as hyperparameters, which cannot be directly learned from the normal training process. They are usually fixed before starting the actual training process.

Models can have many hyperparameters and finding the best combination of parameters is important. The two best strategies for tuning hyperparameters are:

1. **Grid Search CV**

In GridSearchCV approach, the machine learning model is evaluated for a range of hyperparameter values. This approach is called GridSearchCV, because it searches for the best set of hyperparameters from a grid of hyperparameters values. The major drawback of GridSearchCV is that it will go through all the combinations of hyperparameters which makes grid search computationally very expensive.

1. **Randomized Search CV**

RandomizedSearchCV0 solves the disadvantage of GridSearchCV, as it goes through only a fixed number of hyperparameters. It moves within the grid in a randomly to find the best set of hyperparameters. This approach reduces unnecessary computation. The disadvantage of this method is the combinations the hyperparameter choose is beyond our control.

**Conclusion:**

We started the project by loading the dataset collected from twitter. Then to understand the deeper insight of the data we performed EDA on the data. EDA gives us very important insights which help us not only to select the right features but also to choose the right ML model for the dataset. The distributions of sentiments, location with highest number of tweets are some of some important insights. After EDA, we extracted and cleaned the important features and pre-process it to a matrix of numbers so that it can be passed to the ML algorithms. Since the target variable have multiple classes, we followed two different approaches for building the models, firstly we pass these pre-processed data directly to the multiclass classifier and get the output as multiple classes and secondly, we manipulated the target variable to binary variable and performed binary classifier on it. We applied different ML algorithms such as Naive bayes classifier, support vector classifier, Random Forest Classifier, XGBOOST classifier, Logistic Regression etc for both multiclass and binary target variable and evaluated it with different metrics like accuracy score, precision, recall, f1 score etc. We also performed hypermeter tuning to enhance the performance and reduce overfitting of the models. Finally, we got SVC model as the best multiclass classifier model with 61.1% test accuracy and logistic regression model as the best binary classifier model with 86.5% test accuracy.