## Abstract:

As the Covid-19 outbreaks rapidly all over the world day by day and also affects the lives of million, a number of countries declared complete lockdown to check its intensity. During this lockdown period, social media platforms have played an important role to spread information about this pandemic across the world, as people used to express their feelings through the social networks. Considering this catastrophic situation, we developed an experimental approach to analyse the reactions of people on Twitter taking into account the popular words either directly or indirectly based On this pandemic.


## Introduction:

Sentiment Analysis is the process of computationally identifying and categorizing opinions expressed in a piece of text, especially in order to determine whether people attitude towards a particular topic is positive, negative, or neutral.

On 31st December, 2019 the Covid-19 outbreak was first reported in the Wuhan, Hubei Province, China and it started spreading rapidly all over the world. Finally, WHO announced Covid-19 outbreak as pandemic on 11th March, 2020, when the virus continues to spread.

We are analyzing data during pandemic time to gather correct information for making policy for further use.

## Problem Statement

The challenge is 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.

We are given the following information:

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

## Objective of our Project:

● For multiclass classification, the best model for this dataset.

● For binary classification, the best model for this dataset would be Stochastic Gradient Descent

## Steps involved:

### Exploratory Data Analysis

- Most of the peoples are having positive sentiments and Negative about various issues shows us their optimism during pandemic times.
- Very few people are having extremely negatives thoughts about Covid-19.

### Null values Treatment

Our dataset doesn't contain null values.

### Encoding of categorical columns

We have classified the tweets on the basis of the compound sentiments into two different classes, i.e. Positive is (1), Negative (-1). Then we have assigned the sentiment polarity rating for each tweet based on the algorithm.

### Feature Selection

After pre-processing we have developed the Bag-of-Words (BOW) model using the frequently occurred words from the word lexicon and we obtained a list of most frequent Covid-19 exclusive words. We have represented a dense word-cloud in some of the mostly used words within the corpus.

### Standardization of features

Our main goal like we want to develop a polarity-popularity model based on the features extracted during this experiment so that we can assign the refined sentiment ratings to the tweet based on the polarity of mostly recurred words [3]. With that data we will train the deep learning model to enhance the validation accuracy of our system.

### Removed HTTP And URLS from Tweets

HTTP and URLs were also irrelevant to the analysis since they are also like usernames are unique therefore they were also removed

## Removing Punctuation

Punctuation, numbers and special characters will only add to the dimensionality of our final dataset without being of any value thus they were also removed.

## Removing Short Words

We also determined that very short words of the length of two or lower are also not of much importance.

## Tokenization

Tokenization is the process of tokenising or splitting a string or text into a list of tokens. One can think of tokens as parts like a word is a token in a sentence, and a sentence is a token in a paragraph.

## Stemming

Stemming is the reduction of a word to its word stem that affixes to suffixes and prefixes or to the roots of words known as a lemma. This basically means that here we transformed each word into its base form.

## Hashtags Analysis

Understanding the impact of hashtags on the sentiment of the tweets.

## Removing Stopwords

A stop word is a commonly used word (such as “the”, “a”, “an”, “in”) that a search engine has been programmed to ignore, both when indexing entries for searching and when retrieving them as the result of a search query. We would not want these words to take up space in our database, or take up the valuable processing time. Therefore we have removed them

## Countvectorizer

We chose 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 Countvectorizerto transform a given text into a vector based on the frequency (count) of each word that occurs in the entire text.

## Building Classification Models

There are five types of sentiments so we have to train our models so that they can give us the correct label for the test dataset. I am going to built different models like Naive Bayes, Logistic Regression, Random Forest, XGBoost, Support Vector Machines, and Stochastic Gradient Descent.

## Fitting Different Models

- Support Vector Machine
- Naive Bayes Classifier
- Stochastic Gradient Descent
- Random Forest Classifier
- Extreme Gradient Boosting
- Logistic Regression

## Model Evaluation Matrices

Accuracy is the quintessential classification metric. Accuracy is the proportion of true results among the total number of cases examined. It is easily suited for binary as well as multiclass classification problems Accuracy is a valid choice of evaluation for classification problems which are well balanced and not skewed or no/less class imbalance. Accuracy = (TP + TN) / (TP + TN + FP + FN) Precision (Macro): Precision answers what proportion of predicted positives is truly positives. Precision is a valid choice of evaluation metric when we want to be very sure of our prediction. Precision Macro = (Sum of Precision for each class) / (No. of Classes) Recall (Macro): Recall answers what proportion of actual Positives is correctly classified. The recall is a valid choice of evaluation metric when we want to capture as many positives as possible. Recall Macro = (Sum of Recall for each class) / (No. of Classes) F1 Score: The F1 score is a number between 0 and 1 and is the harmonic mean of precision and recall. F1 score sort of maintains a balance between the precision and recall for the classifier. If precision is low, the F1 is low and if the recall is low again F1 score is low. The F1 score manages the tradeoff. F1 Score = 2 * (Precision * Recall) / (Precision + Recall) AUC: AUC is the area under the ROC curve. AUC ROC indicates how well the probabilities from the positive classes are separated from the negative classes. AUC is scale-invariant. It measures how wel lpredictions are ranked, rather than their absolute values.

## Support Vector Machine

Support vector machines are a set of supervised learning methods used for classification, regression, and outliers' detection. All of these are common tasks in machine learning. A simple linear SVM classifier works by making a straight line between two classes. That means all of the data points on one side of the line will represent a category and the data points on the other side of the line will be put into a different category. This means there can be an infinite number of lines to choose from. The linear SVM algorithm is better than some of the other algorithms, like k-nearest neighbours, in that it chooses the best line to classify your data points. It chooses the line that separates the data and is the furthest away from the closest data points as possible. SVM is Effective on datasets with multiple features or in cases where the number of features is greater than the number of data points. It uses a subset of training points in the decision function called support vectors which makes it memory efficient. It has different kernel functions that can be specified for the decision function. But SVMs don't directly provide probability estimates. Those are calculated using an expensive five-fold cross-validation. Linear Kernel is commonly recommended for text classification because most of these types of classification problems are linearly separable. The linear kernel works really well when there are a lot of features, and text classification problems have a lot of features. Linear kernel functions are faster than most of the others and you have fewer parameters to optimise. The LinearSVC module from the sklearn library provides support for linear support vector machines. The function for linear kernel is as below, f(X) = wT * X + b

## Naive Bayes Classifier

Naive Bayes algorithm is a supervised learning algorithm, which is based on the Bayes theorem and used for solving classification problems. It is mainly used in text classification that includes a high-dimensional training dataset. Naïve Bayes Classifier is one of the simplest and most effective Classification algorithms which helps in building fast machine learning models that can make quick predictions. It is a probabilistic classifier, which means it predicts on the basis of the probability of an object. Some popular examples of the Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

## Stochastic Gradient Descent

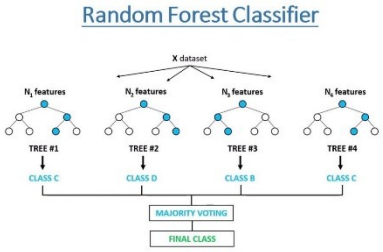
The word 'stochastic' means a system or process linked with a random probability. Hence, in Stochastic Gradient Descent, a few samples are selected randomly instead of the whole data set for each iteration. In Gradient Descent, there is a term called "batch" which denotes the total number of samples from a dataset that is used for calculating the gradient for each iteration. In typical Gradient Descent optimization, like Batch Gradient Descent, the batch is taken to be the whole dataset. Although using the whole dataset is really useful for getting to the minima in a less noisy and less random manner, the problem arises when our dataset gets big. Suppose, you have a million samples in your dataset, so if you use a typical Gradient Descent optimization technique, you will have to use all of the one million samples for completing one iteration while performing the Gradient Descent, and it has to be done for every iteration until the minima are reached. Hence, it becomes computationally very expensive to perform. This problem is solved by Stochastic Gradient Descent. In SGD, it uses only a single sample, i.e., a batch size of one, to perform each iteration. The sample is randomly shuffled and selected for performing the iteration. So, in SGD, we find out the gradient of the cost function of a single example at each iteration instead of the sum of the gradient of the cost function of all the examples. In SGD, since only one sample from the dataset is chosen at random for each iteration, the path taken by the algorithm to reach the minima is usually noisier than your typical Gradient Descent algorithm. But that doesn't matter all that much because the path taken by the algorithm does not matter, as long as we reach the minimum and with a significantly shorter training time.

## Random Forest Classifier

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and improve the performance of the model. As the name suggests, "Random Forest is a classifier

that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, predicts the final output.

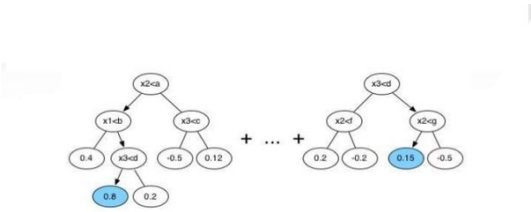
● The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting. The below diagram explains the working of the Random Forest algorithm:




## Extreme Gradient Boosting

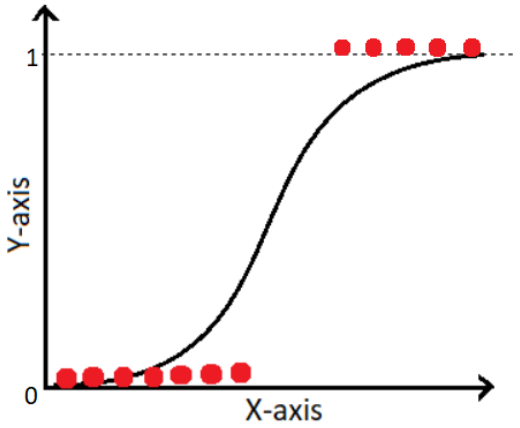
XGBoost is an implementation of Gradient Boosted decision trees. This library was written in C++. It is a type of Software library that was designed basically to improve speed and model performance. It has recently been dominating in applied machine learning. XGBoost models majorly dominate in many Kaggle Competitions. In this algorithm, decision trees are created in sequential form. Weights play an important role in XGBoost. Weights are assigned to all the independent variables which are then fed into the decision tree which predicts results. The weight of variables predicted wrong by the tree is increased and the variables are then fed to the second decision tree. These individual classifiers/predictors then ensemble to give a strong and more precise model. It can work on regression, classification, ranking, and user-defined prediction problems XGBoost Feature The library is laser-focused on computational speed and model performance, as such, there are few frills. Model Features Three main forms of gradient boosting are supported:

● Gradient Boosting
● Stochastic Gradient Boosting
● Regularized Gradient Boosting



## Logistic Regression

Logistic Regression is a statistical method that is used for building machine learning models where the dependent variable is dichotomous i.e. binary. Logistic regression is used to describe data and the relationship between one dependent variable and one or more independent variables. The independent variables can be nominal, ordinal, or interval type. The name “logistic regression” is derived from the concept of the logistic function that it uses. The logistic function is also known as the sigmoid function. The value of this logistic function lies between zero and one. The following is an example of a logistic function we can use to find the probability of a vehicle breaking down, depending on how many years it has been since it was serviced last Here is how you can interpret the results from the graph to decide whether the vehicle will break down or not.



## Conclusion

We focused on sentiment analysis for sentence labelling. We described the preprocessing steps, and pipeline steps within which text normalization and model cross-validation is included, and performance has been measured using balanced accuracy, f1 score etc. We used “Stemming” instead of Lemmatization to reduce dimensions, for the same reason we haven’t tried tf-idf or term frequency vectorizer. We concentrated on feeding our model with word count information. We assume, in the case of binary classification, we can further improve this score.