Empirical Study and Comparison of Models via Multiclass Classification of COVID-19 Tweets using Natural Language Processing

Simran Anand[1*], Deepasikha Mishra[2]
[1]*Bachelor of Technology, Department of Computer Science Engineering, Vellore Institute of Technology, Amaravati, India*
[2]*Assistant Professor, Department of Computer Science Engineering, Vellore Institute of Technology, Amaravati, India*

***Abstract*: This paper represents the empirical study of sentiment analysis of the Covid-19 tweets during the pandemic period. Various industries have been psychologically affected throughout the country during this period. Through the Covid-19 tweets, a study has been established to determine whether the people's attitude is positive, negative or neutral during the pandemic. In this work, Natural Language Processing, Exploratory Data Analysis and Machine Learning are used to analyze textual data consisting of Covid- 19 tweets. Different Machine Learning and Deep Learning techniques like Naive Bayes, Logistic Regression, Extreme Gradient Boost (XGBoost), Stochastic Gradient Descent, Random Forest, SVM, Bidirectional LSTMs (BiLSTM) and Backpropagation neural networks have been incorporated to analyze and predict efficiently. Finally, a comparison of each model's performance based on evaluation metrics like accuracy, precision, recall and F1-score has been done.**

***Keywords*: Natural Language Processing, Sentiment Analysis, COVID-19, Machine Learning, Deep Learning, Data Analysis.**

1. Introduction

During the COVID-19 pandemic, numerous people have been affected dramatically and it is important to understand their feelings and sentiments during the pandemic. To understand the feelings and sentiments of people all over the world, a process called Sentiment Analysis can be performed on the tweets collected from various individuals. The technique of computationally recognizing and categorizing opinions stated in a piece of text, particularly to establish whether the writer's attitude toward a given issue is Positive, Negative, or Neutral, is known as sentiment analysis. It is a vital application of Natural Language Processing and belongs to the core of Artificial Intelligence [Sam+14; KSN]. The task at hand is to create a classification model that can predict Covid-19 tweet sentiment. A high volume of Twitter data has been leveraged to understand the public sentiment for the covid-19 outbreak. The tweets were manually tagged after being retrieved from Twitter. Information like Location, usernames of tweet sources, Original Tweets, and labels indicating retweets are incorporated for the research. Classifiers are built using a variety of ML and DL models, and their accuracy is tested across multiple feature sets. Such classifiers will assist businesses, corporations, political parties, and analysts, among others, in assessing public perceptions of them and developing policies to meet their concerns.

This study is based on Sentiment analysis and it leverages Natural Language Processing (NLP) to analyze Textual Data consisting of Covid19 Tweets. Exploratory Data Analysis has been done to work with the dataset and its various features and attributes. It incorporates various Machine Learning and Deep Learning models and algorithms to perform an empirical analysis and comparison of the different algorithms based on their computed accuracy, precision, recall, and F1 Scores. It includes Long Short Term Memory (LSTM) of two types of models, one regularized LSTM model which is a Recurrent Neural Network (RNN) using softmax activation and another is LSTM model that is an RNN using tanh and sigmoid activation function. Back Propagation Neural Network, Random Forests Classifier, Support Vector Machine(SVM), Stochastic Gradient Descent(SGD), Extreme Gradient Boosting (XGBoost), Logistic Regression, Naive Bayes Classifier and Count vectorizer models are few other models used to achieve the objective of the study. NLP's python libraries like TextBlob and vader sentiment are leveraged to perform sentiment analysis of the collected preprocessed tweets data. Keras and TensorFlow are examples of neural network libraries that were used for Deep learning techniques to evaluate the model's performance for apt classification of texts and tweets [Rus+19; KSN].

2. Dataset

Data were extracted from a Twitter dataset consisting of tweets of people from different backgrounds, locations, and various social media platforms. External data sources like Covid19 tweets from Twitter are collected for sentiment analysis. We used data from Indian user tweets from the Twitter website during the COVID- 19 shutdown period in the country for the investigation [CBA21]. The data collection, which contains the cleaned tweets on themes like COVID-19, coronavirus, and lockdown, was retrieved from github.com (https://github.com/gabrielpreda/CoViD- 19-tweets (accessed on 12 January 2021)). For the analysis, a dataset of extracted tweets from the Indian Twitter network was used. COVID-19,

*Corresponding author: simranand112@gmail.com

coronavirus, and lockdown were among the subjects discussed in the tweets.

3. Proposed Scheme

The study aims to build a multiclass classification model to predict the sentiments of COVID-19 tweets. It focuses on classifying tweets into three categories of sentiments: positive, negative, and neutral. The tweets were manually tagged after they were extracted from Twitter. Leveraging Natural Language Processing, sentiment analysis is to be performed on the data. In general, data can be textual as well as audio consisting of speech like in AI voice assistants. We have worked on Textual data for our study. Additionally, machine learning algorithms are to be incorporated to evaluate accuracy score and classification prediction by the trained model [PS18]. It thus aims to analyze datasets from a social point of view to help understand people's perceptions during the pandemic. The mentioned machine learning models are to be used to classify the tweets and opinions into a segregated class by assigning polarity scores. The results have been plotted in the form of graphs by making use of Python libraries and frameworks.

This study proposes a scheme that comprises processes and techniques like Data preprocessing via text mining (NLP techniques like Tokenization, Lemmatization, Parsing, Stemming, and Vectorization), Exploratory Data Analysis (EDA), Sentiment Analysis, Data Visualization, Multiclass Classification using Machine Learning algorithms, building a Multi-Layer Perceptron (MLP), Neural networks with Regularization techniques, Forward Pass, Backpropagation, LSTM and finally comparing each model's performance based on evaluation metrics like accuracy, precision, recall and F1-score obtained from classification reports.

A. COVID-19 Outbreak Analysis

This section of the research involves three parts: Text mining via NLP techniques, Exploratory Data Analysis(EDA) performed on the tweets with Sentiment Analysis and then plotting the results via Data Visualization process.

1) Text mining via NLP: Most text and documents in Natural Language Processing (NLP) contain many terms that are redundant for text classification, such as stopwords, misspellings, slang, and so on. As a result, removing these characteristics is critical. Additionally, the NLTK library for pre-processing of fetched tweets and the Textblob dataset to evaluate the tweets has been used, after which the exciting results indicate positive, negative, neutral feelings throughout various visualizations [KS20]. Word embedding is a Natural Language Processing approach (NLP) used to convert words into vector arrangements, to capture the semantic and syntactic relationship between words, thus simulating human learning of language vocabulary. The problem with encoding is one of great interest in this area of knowledge, which is why we should not just consider superficial forms of a text to represent words (e.g., using symbols, characters, word chains, sentences, and documents), but also look for significant similarities (e.g., semantic or syntactic ones) between text fragments. The fundamental idea is to use linear algebra to represent each word inside a large body of text as a feature vector so that the similarity between vectors (e.g., words) may be measured (e.g., using cosine similarity). TF-IDF, which stands for Term Frequency-Inverse Document Frequency, is a numerical metric that measures how important a word is to a document in a corpus or collection. The TF–IDF value rises in proportion to the number of times a word appears in the document and is offset by the number of documents in the corpus that contain the term, which helps to compensate for the fact that some words appear more frequently than others in general [KS20].

$$W(d,t) = TF(d,t) * log(N/df(t)) \quad (1)$$

N is the number of documents and df(t) is the number of documents containing the term t in the corpus. The first portion would increase recall, while the second would improve word embedding precision. Although tf-idf attempts to solve the problem of frequent terms in documents, it still has significant descriptive limitations. Because each word is supplied as an index, tf-idf cannot account for the similarity between terms in the document. All the data cleaning techniques have been incorporated in the preprocessing part for textual data preprocessing.

2) Exploratory Data Analysis(EDA): In this process, the information about the shape and size of the dataset is obtained, following which null or missing values have been looking for. The percentage of null values obtained against each column has been plotted graphically. TextBlob package in Python has been used to calculate and categorize each tweet as Negative, Neutral, and Positive by building a function that determines subjectivity and polarity from the textblob package [Sha+20]. Another Natural Language Processing package in python, which is, vader sentiment has been used to analyze the intensity of the sentiments associated with the tweets after the multi class classification had been done [Rus+21]. Vader (Valence Aware Dictionary and Sentiment Reasoner) is a vocabulary and rule-based approach that uses social media sites to determine sentiment scores. It's also effective on text from other websites. It assigns a level of intensity to each phrase in the tweets, then adds together all of the levels to get the emotion score. Vader was chosen as a comparison character because several research studies have demonstrated that Vader performs better on social media typewriting. Compound score, a metric that estimates the sum of all lexical ratings that have been balanced between -1 (most extreme negative) and +1 (most extreme positive), has been used to detect tweet sentiment (most extreme positive). The tweets are segregated as positive: (compound score more than or equal to 0.05), negative: (compound score less than or equal to -0.05) and neutral otherwise.

3) Data Visualization: This section of the analysis encompasses the process of data visualization [BJ19]. It involves story generation from the analysis of tweets. Python libraries like matplotlib and seaborn have been incorporated for data visualization. The words with high frequencies are larger, whereas the words with low frequencies are smaller. This is based on the common occurrences of specific words in the tweets. The common words indicate the high presence of a particular sentiment in the tweets collected. The Wordcloud

addresses questions regarding the most common words in the entire dataset, the most common words in the dataset for negative and positive tweets, the number of hashtags there in a tweet, the trends associated with the dataset, trends associated with either of the sentiments and whether they are compatible with the sentiments.

B. COVID-19 Outbreak Prediction

This step of the research consists of predictions done using Machine Learning and Deep Learning techniques [DCH13]. For applying Machine Learning algorithms to the data obtained after the stages of Data Preprocessing and Sorting (Natural Language Processing was done using NLTK, gensim, vader sentiment, and TextBlob libraries), the dataset was split into training, test, and validation datasets. 80 percent of data was used for training and the remaining 20 percent for validation and testing. The words are converted to vectors (Bag of Words technique) using CountVectorizer so that models can be trained after feature engineering is performed. After train test split and feature extraction, various ML algorithms and classifiers have been applied to classify the data into multiple classes as required. In the training step, data is fed to the ML model so that it learns from the train set (models used are based on Supervised learning) and then it predicts the outcome after few instances of expected outputs are provided to it. The difference between the predicted and actual value gives the residual error and is an important factor to calculate the accuracy score [DCH13]. Accuracy of the model increases if the predicted value and the expected (actual) value for output are near. Along with accuracy, a classification report with 4 metrics: precision, recall, f1-score, and support is obtained from the prediction made and the actual value of the labels. The ML models that have been incorporated in this analysis are: Random Forest Classifier, SVM (Support Vector Machine), Logistic Regression, Stochastic Gradient Descent (SGD), Extreme Gradient Boosting (XGBoost)and Naive Bayes Classifier for Multinomial classification.

Random Forests: Random Forest is a classifier that combines several decision trees on different subsets of a dataset and averages the results to increase the dataset's predicted accuracy. Instead of relying on a single decision tree, the random forest. collects the forecasts from each tree and predicts the final output based on the majority votes of predictions. The goal of utilizing this model was to save time during training compared to previous techniques. It predicts output with good accuracy, and it runs quickly even with a huge dataset. When a considerable amount of the data is missing, it can still maintain accuracy. The bigger the number of trees in the forest, the more accurate it is and the problem of overfitting could be avoided [Tix18]. Every tree in the Random Forest tries to come up with rules that ensure that each consequent node is as pure as feasible. The more the purity, the more likely you are to make specific decisions. Thus, parameters like Entropy, Information gain and Gini Index are essential in this model of prediction. Entropy is calculated to assess the impurity or randomness (unpredictability) of a dataset.

$$Entropy(x) = -(P(x = k) * \log_2(P(x = k)) \quad (2)$$

Where P(x=k) is the probability that a target feature takes a specific value, k. Logarithm of fractions gives a negative value and hence a '-' sign is used in entropy formula to negate these negative values. The maximum value for entropy depends on the number of classes. To determine the best feature which serves as a root node in terms of information gain, we first use each descriptive feature and split the dataset along the values of these descriptive features and then compute the entropy of the dataset. This offers us the leftover entropy once we have split the dataset along the feature values. Then we subtract this number from the dataset's original entropy to see how much this feature splitting reduces the original entropy, which gives the feature's information gain and is calculated as:

$$Information\ Gain\ (feature) = Entropy\ (Dataset) - Entropy\ (feature) \quad (3)$$

The feature with the largest information gain should be used as the root node to start building the decision tree.

$$GiniIndex = 1 - (P(x = k))^2 \quad (4)$$

Gini index is calculated by subtracting the sum of squared probabilities of each class from one. It favors larger partitions and easy to implement whereas information gain favors smaller partitions with distinct values. A feature with a lower Gini index is chosen for a split.

1) Support Vector Machine (SVM): Each data item is plotted as a point in n-dimensional space (where n is the number of features), with the value of each feature being the value of a certain position in the SVM algorithm. The extreme points/vectors that assist create the hyperplane are chosen via SVM. Support vectors represent extreme examples, which is why the technique is called Support Vector Machine. Then we accomplish classification by locating the hyper-plane that clearly distinguishes the two classes. A major reason we have used SVMs is that they can find complex relationships between the data without needing to do a lot of transformations. When working with smaller datasets with tens to hundreds of thousands of features, it's a wonderful The equation now becomes (describing that every point is at least 1/‖w‖2 distance apart from hyperplane) as

2) Stochastic Gradient Descent (SGD): Some users may be led to believe that SGD is a classifier by the moniker Stochastic

Gradient Descent - Classifier (SGD-Classifier. Logistic Regression, also known as linear Support Vector Machine, is a machine learning methodology, whereas SGD is an optimization method. SGD-Classifier is a Linear classifier with SGD training. A loss function is defined by a machine learning model, and the optimization approach minimizes or maximizes it. In contexts with large-scale data sets, when standard methods are no longer viable, methods for parameter estimation are offered. Here, algorithms use stochastic approximations, which are computationally efficient since they keep one iterate as a parameter estimate and update iterate by iterate based on a single data point. The stochastic approximation is known as standard when the update is based on a noisy gradient. In recent

applications with enormous datasets, stochastic gradient descent has become critical [TTA15]. In each iteration that takes place for number of training samples, the following equation represents the algorithm.

3) Logistic Regression: Under the Supervised Learning most prominent Machine Learning algorithms is logistic regression. It's a method for predicting a categorical dependent variable from a set of independent variables. A categorical dependent variable's output is predicted using logistic regression. As a result, the result must be a discrete or categorical value [AG19]. There should not be any multi-collinearity in the model, which means the independent variables must be independent of each other. The sigmoid function is another name for the logistic function. This logistic function has a value between zero and one. The main reason behind incorporating it to train our model is that it is very fast at classifying unknown records and efficiently trains the model making no assumptions about distributions of classes in feature space.

The above function is the Logistic function, also called Sigmoid function. Here, x is the input, a real number and e is Euler's function.

4) Naive Bayes: Na¨ıve Bayes is a Bayesian probabilistic machine learning technique that is employed in a wide range of classification applications. It is mostly utilized in text classification tasks that require a large training dataset. The Naive Bayes Classifier is a simple and effective classification method that aids in the development of fast machine learning models capable of making quick predictions. It's a probabilistic classifier, which means it makes predictions based on an object's probability. They are quick and simple to implement, but the necessity that predictors be independent is their major drawback. Multinomial distribution can be utilized with categorical data like counts or labels. A binomial distribution can be employed if the variables are binary, such as yes/no or true/false. A Gaussian distribution is frequently employed when a variable is numerical, such as a measurement. The words in a text can be encoded using binary (word presence), count (word occurrence), or frequency (tf/idf) input vectors, as well as binary, multinomial, or Gaussian probability distributions [Tix18]. The following equations are important mathematical theorems on which Naive Bayes classifier is based.

$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c) \tag{5}$$

The above equation represents Bayes theorem. It provides a way of calculating posterior probability P(c—x) from P(c), P(x), and P(x—c). Here, P(c—x) is the posterior probability of class(c,target) given predictor(x,attributes). This represents the probability of c being true, provided x is true. P(c) is the prior probability of class. This is the observed probability of class out of all the observations. P(x—c) is the likelihood which is the probability of predictor-given class. This represents the probability of x being true, provided x is true. P(x) is the prior probability of predictor. This is the observed probability of predictor out of all the observations.

5) Extreme Gradient Boosting (XGBoost): Gradient boosting is a machine learning technique for regression, classification, and other problems that generates a prediction model from an ensemble of weak prediction models, most commonly decision trees. The resulting technique is called gradient boosted trees when a decision tree is the weak learner, and it usually outperforms random forest. The gradient boosting decision tree algorithm is implemented in the XGBoost package. Gradient boosting, multiple additive regression trees, stochastic gradient boosting, and gradient boosting machines are all terms used to describe this approach. Boosting is an ensemble strategy that involves adding new models to old models to remedy faults. Models are added one by one until there are no further improvements to make. It constructs the model in the same stage-by-stage manner as other boosting approaches, but it broadens the scope by allowing optimization of any differentiable loss function. Tree boosting is a popular and successful machine learning technique. It is a scalable end-to-end tree boosting system extensively utilized by data scientists to obtain state-of-the-art outcomes on a variety of machine learning issues [Che+16]. To avoid overfitting, XGBoost includes a regularization component in the objective function.

The challenge of optimizing an objective function is reduced to identifying the minimum of a quadratic function. XGBoost also has a stronger ability to avoid overfitting due to the addition of the regularization term. The Deep Learning models that have been incorporated in the analysis are: The back Propagation model and LSTM (Long Short Term Memory) model to test the performance. One Hot Encoding is done on the textual data to convert strings to numbers, after the data normalization and scaling process. Packages like TensorFlow and Keras are imported to build Sequential models in the Neural Network. Multi-Layer Perceptron Classifier is used and the dense layers of the network are activated by passing activation functions like Rectified Linear Unit (relu) and softmax. A Recurrent Neural Network (RNN) has been built on word sequences after steps like Tokenization, Padding, Embedding, and Standardization were performed [CBA21].

6) Long Short Term Memory (LSTM): LSTM (Long-Short Term Memory) is a type of Recurrent Neural Network and it is used to learn sequence data in deep learning. A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. The three gates control the flow of information into and out of the cell, and the cell remembers values across arbitrary time intervals. Because there might be lags of undetermined duration between critical occurrences in a time series, LSTM networks are well-suited to categorizing, processing, and making predictions based on time series data. In this research, two kinds of RNN have been constructed as LSTM. One RNN without any regularization, which used activation functions like tanh and sigmoid. The other RNN is a regularized one and it used a softmax activation function. The steps performed were: Preparing data (Tokens/words are embedded to vectors/numbers), Building, training, and applying the LSTM Neural Network model, and Predicting test data [SK19]. In LSTM as we have 3 gates: Input Gate, Forget Gate and Output Gate. Gates in LSTM are the sigmoid

Table 1
Model and accuracy

Model	Accuracy	Class	Precision	Recall	F1 score
Random Forest	0.91	Negative	0.72	0.92	0.81
		Neutral	0.99	0.90	0.94
		Positive	0.87	0.94	0.90
		macro avg.	0.86	0.92	0.89
		weighted avg.	0.92	0.91	0.92
SVM	0.88	Negative	0.62	0.90	0.73
		Neutral	1.00	0.84	0.91
		Positive	0.81	0.95	0.88
		macro avg.	0.81	0.90	0.84
		weighted avg.	0.90	0.88	0.88
Logistic Regression	0.90	Negative	0.72	0.89	0.80
		Neutral	0.98	0.89	0.93
		Positive	0.86	0.93	0.89
		macro avg.	0.86	0.90	0.88
		weighted avg.	0.91	0.90	0.90
Naïve Bayes	0.82	Negative	0.54	0.76	0.63
		Neutral	0.87	0.88	0.88
		Positive	0.86	0.76	0.81
		macro avg.	0.76	0.80	0.77
		weighted avg.	0.83	0.82	0.82
LSTM (RNN with tanh and sigmoid activation functions)	0.83	Negative	0.73	0.69	0.71
		Neutral	0.88	0.87	0.88
		Positive	0.81	0.84	0.82
		macro avg.	0.81	0.80	0.80
		weighted avg.	0.83	0.83	0.83
LSTM (Regularized RNN with softmax activation function)	0.84	Negative	0.72	0.68	0.70
		Neutral	0.88	0.88	0.88
		Positive	0.82	0.84	0.83
		macro avg.	0.81	0.80	0.80
		weighted avg.	0.84	0.84	0.84
BPNN	0.84	Negative	0.68	0.72	0.70
		Neutral	0.87	0.88	0.88
		Positive	0.84	0.81	0.83
		macro avg.	0.80	0.81	0.80
		weighted avg.	0.84	0.84	0.84
Stochastic Gradient Descent (SGD)	0.92	Negative	0.80	0.90	0.85
		Neutral	0.99	0.91	0.95
		Positive	0.87	0.96	0.91
		macro avg.	0.89	0.92	0.90
		weighted avg.	0.93	0.92	0.92
Extreme Gradient Boost (XGBOOST)	0.77	Negative	0.27	0.89	0.42
		Neutral	1.00	0.71	0.83
		Positive	0.64	0.94	0.76
		macro avg.	0.64	0.85	0.67
		weighted avg.	0.88	0.77	0.79

activation functions i.e. they output a value between 0 or 1 and in most of the cases it is either 0 or 1.

The Covid-19 sentiment analysis results of the machine learning models Random Forest, SVM, Stochastic Gradient Descent Classifier, XGBoost, Logistic Regression, Naive Bayes, and deep learning models using LSTM and Backpropagation are presented in this section. The models that yielded good accuracies for the humongous data are Stochastic Gradient Descent, Random Forest Classifier and Logistic Regression. SVM, BPNN and Regularized LSTM also gave fairly good accuracies and generalized well on high-dimensional data for classification. There is scope of improving XGBoost and LSTM model's accuracy. By analyzing the observation table that depicts the classification report for each model's performance, the following can be inferred about each model.

Logistic regression: The model yielded an accuracy of 0.90 which indicates its efficiency to be trained. Thus, it gives good accuracy for many simple data sets and performs well when the dataset is linearly separable. As can be seen from the above findings, the Logistic Regression algorithm outperformed the Naive Bayes algorithm very well. This could be because the Logistic Regression algorithm does not make as many assumptions as the Naive Bayes algorithm does. This model is also less inclined to overfitting. If the number of observations are lesser than the number of features, Logistic Regression should not be used, otherwise it may lead to overfit. Incase of high dimensional data where there is a possibility of overfitting, Regularization techniques can be incorporated to avoid overfitting. Logistic regression is easier to implement, interpret and very efficient to train. On text data, it can be concluded that Logistic Regression is highly productive, and the underlying process is also fairly simple to grasp. More importantly, in the domain of Natural Language Processing, Logistic Regression is widely regarded as a good first approach for text classification.

Random Forest: The model yielded an accuracy of 0.91 which indicates its efficiency to be trained. Random forest is a versatile, easy-to-use machine learning technique that, in most

cases, gives excellent results even without hyper-parameter adjustment. Because of its simplicity and versatility, it is also one of the most widely used algorithms. They are more likely to outperform SVMs in terms of performance. Random forests are usually much faster than (nonlinear) SVMs due to the way methods are implemented (and for theoretical reasons). Overfitting is a problem with the Random Forest method. When more trees are added to the Random Forest process, the generalisation error variance decreases to zero. To avoid overfitting in Random Forest, the algorithm's hyper-parameters should be fine-tuned. The number of samples in the leaf, for example. Ensemble-based learning methods include random forest classifiers, which are a subset of ensemble- based learning methods. They're easy to set up, operate quickly, and have a long track record of performance in a variety of fields. The random forest strategy is based on the building of several "simple" decision trees in the training stage, followed by a majority vote (mode) across them in the classification step. This voting technique, among other things, corrects for the unfavourable feature of decision trees to overfit training data. Random forests use the broad approach known as bagging to individual trees in the ensemble during the training stage.

XGBoost: XGBoost model yielded an accuracy of 0.77 which indicates its efficiency to be trained. It improves the performance of weak decision trees by boosting their performance. Although XGBoost is really fast when compared to other implementations of gradient boosting, it overfits if the model is not stopped early. The model takes quite a while to fail, that's another drawback when compared to more naive approaches. XGBoost is a library for developing fast and high performance gradient boosting tree models. The most significant component in XGBoost's success is its scalability across all scenarios. On a single machine, the system is more than ten times faster than existing popular methods, and it scales to billions of samples in distributed or memory-limited environments. XGBoost's scalability is due to a number of major systems and algorithmic enhancements. These innovations include: a novel tree learning algorithm for handling sparse data; a theoretically justified weighted quantile sketch procedure for handling instance weights in approximate tree learning; and a theoretically justified weighted quantile sketch procedure for handling instance weights in approximate tree learning. Learning is accelerated by parallel and distributed computing, which allows for faster model exploration. XGBoost incorporates a regularized model to prevent overfitting [Che+16].

Stochastic Gradient Descent model: SGD-Classifier yielded an accuracy of 0.92 which indicates the high efficiency of the model's performance. Minibatch (online/out-of-core) learning is possible with SGD. As a result, SGD makes sense for large-scale situations where it is highly efficient. Because the minimum of the Logistic Regression cost function cannot be calculated directly, Stochastic Gradient Descent, also known as Online Gradient Descent is used, to try to minimize the loss function. Another reason to use SGD Classifier is that if you can't maintain the record in RAM, SVM or logistic regression won't function. SGD Classifier, on the other hand, continues to function. SGD classifier works faster compared to the linear SVM in terms of the training time. SGD Classifier is an SGD-optimized linear classifier (SVM, logistic regression, etc.). These are two distinct ideas. SGD is an optimization method, whereas Logistic Regression, also known as linear Support Vector Machine, is a machine learning algorithm. The gradient is computed using stochastic gradient descent (SGD) with a single sample. SGD-Classifier has many advantages over simple linear models and Naive Bayes classifier.

Naive Bayes Classifier: The model yielded an accuracy of 0.82. On testing the model on the test data set we get an accuracy of 0.82 which is slightly less may be due to the assumptions that the Naive Bayes algorithm makes. In fact, it called "Naive" due to its assumptions. The Naive Bayesian algorithm is a simple classification system that relies on the likelihood of events. It is based on the Bayes Theorem, which presupposes that the variables are not interdependent. The Naive Bayes classifier is much faster with its probability calculations. The main reason for this algorithm's text classification friendliness is that it calculates probabilities effectively.

Support Vector Machine: The model yielded an accuracy of 0.88. It has two key advantages over newer algorithms like neural networks: greater speed and better performance with a limited number of samples (in the thousands). This makes the approach particularly well suited to text classification issues, where it's common to only have access to a few thousand tagged samples. You can use a support vector machine to classify data that is linearly separable. You can use the kernel method to make it work if it isn't linearly separable. However, when it comes to text categorization, it's best to remain with a linear kernel. SVM Classifiers outperform the Naive Bayes and LSTM models in terms of accuracy and prediction speed. They also utilize less memory in the decision phase because they only use a subset of training points. With a large dimensional space and a clear separation margin, SVM performs well. Support Vector Machine is preferred because it produces great accuracy while using less computing power. When the data set contains more noise, such as overlapping target classes, SVM does not perform well. The SVM will underperform if the number of features for each data point exceeds the number of training data samples. Support vector machines (SVMs) cannot be trained on large data sets with millions of data points because to their time complexity. We may speed things up by using hyper parameter search libraries, which are far more efficient than grid search. SVM models outperform trees like the XGBoost model in general when dealing with sparse data.

Backpropagation: The model yielded an accuracy of 0.84 which indicates its efficiency to be trained. In data mining and machine learning, backpropagation (backward propagation) is a useful mathematical tool for boosting prediction accuracy. It is a technique that is used to swiftly calculate derivatives and is a learning algorithm used by artificial neural networks to compute a gradient descent with respect to weights. Desired outputs are compared to actual system outputs, and the systems are fine-tuned by altering connection weights to close the gap as much as possible. The weights are updated backwards, from

output to input. It is a fast, basic, and straightforward programming method. Apart from the number of inputs, there are no parameters to tweak. It is a flexible method because it does not require prior knowledge of the network. It is a typical method that works well in most cases.

Long Short Term Memory Transformers: The performance of LSTM was also compared to that of machine learning models. The LSTM models show an accuracy score of 0.83 and 0.84. The embedding layer between the input layer and the LSTM layer provides a vector of input text features for the LSTM layer, which is used by the LSTM. The activation function 'relu' is used to generate a dense layer of 256 neurons (Rectified Linear Unit). The neurons are then dropped during training using a dropout layer with a 0.5 dropout rate, lowering the likelihood of over-fitting the model. When the dataset is too little for a deep learning model to work, LSTM models tend to underperform, as compared to other neural network architectures [Rus+21]. This is in line with a study that found deep learning models to perform poorly on short datasets. Regularized LSTM and unregularized LSTM have been used in this study and they yielded an accuracy score of 0.84 and 0.83 respectively. Regularized LSTM using softmax activation function takes more training time as compared to LSTM without regularization that uses tanh and sigmoid functions but Regularized model gives slightly higher accuracy. Without further exploration of LSTM on other datasets or with a more optimized architecture, the results cannot be definitive [Rus+21].

After the evaluation, two distinct sets of labeled tweet sets, one train set, and one refined test set were obtained. These corpora can now be merged into one. Now, this data is a valuable data insight into our predictive analysis. For a timeline-based prediction of the number of case reports and rise in the volume of the Coronavirus research data, the concept of Bayesian regression has been applied here. Bayes regression is a special categorization of linear regression, in which the prior predictions take place based on the parameters, and even the posterior distribution coverage determined from the parameters is also considered when making predictions [Cha+20].

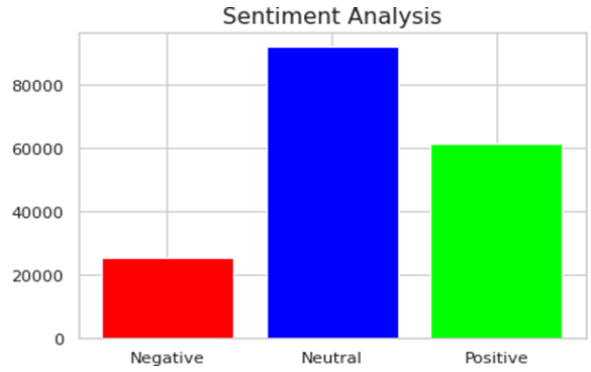

Fig. 1. Bar graph depicting the distribution of sentiments in tweets as observed

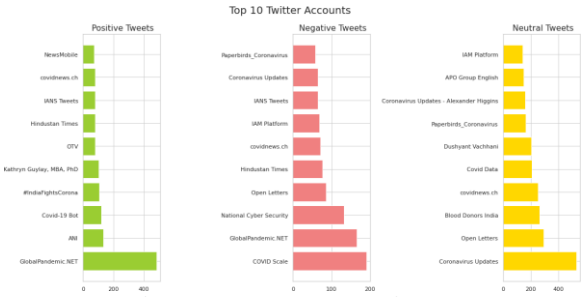

Fig. 2. A plot of the top 10 Twitter accounts from where maximum number of tweets were generated

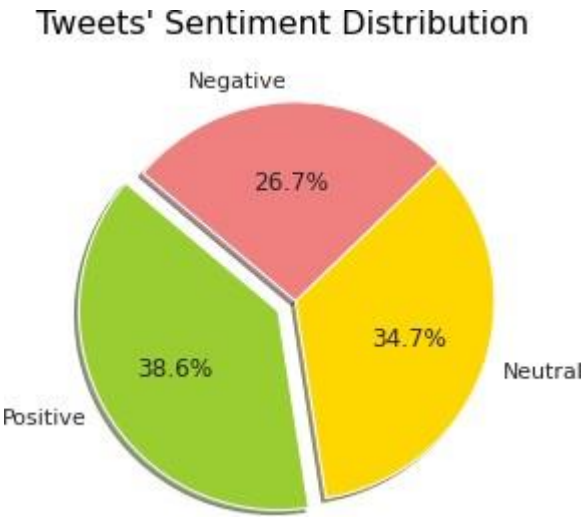

Fig. 3. Pie chart depicting the percentage proportion of each class of tweets

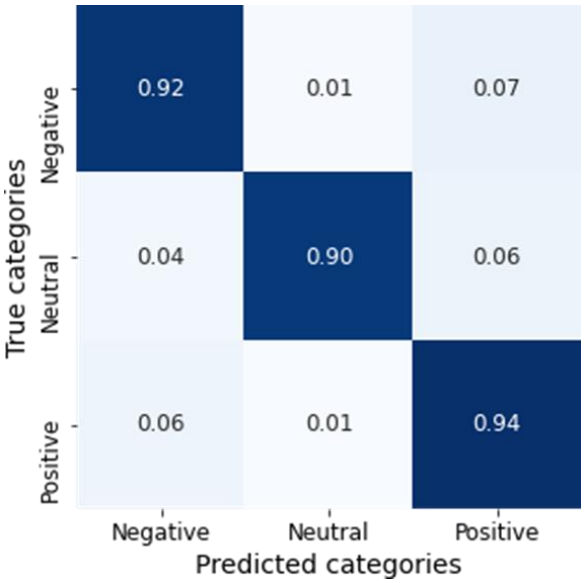

Fig. 4. Confusion matrix for Random Forests Classifier model

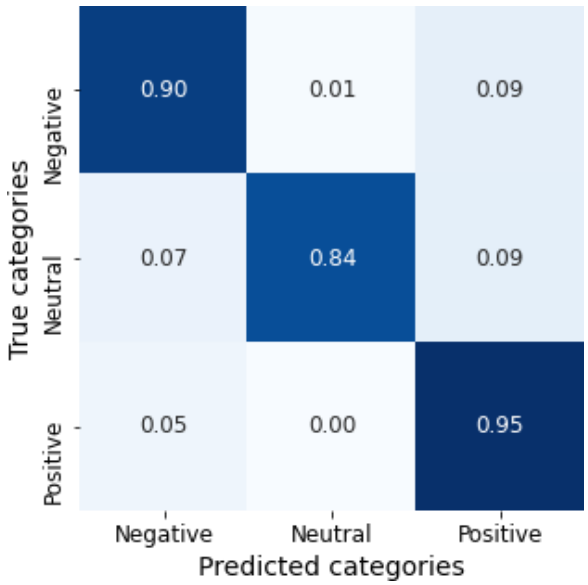

Fig. 5. Confusion matrix for Support Vector Machine (SVM) model

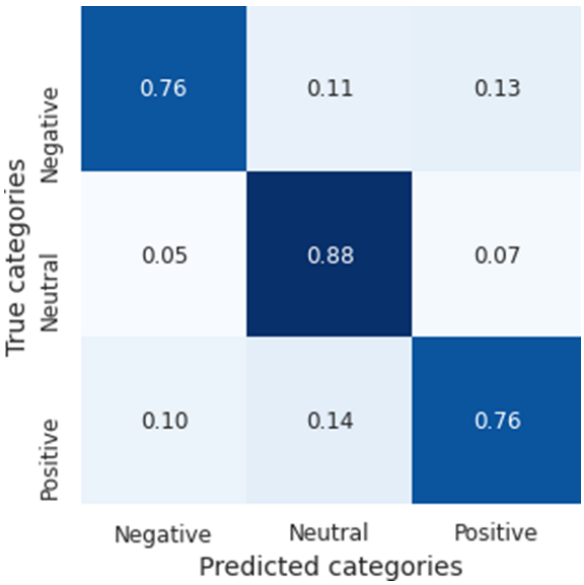

Fig. 8. Confusion matrix for Naive Bayes Classifier model

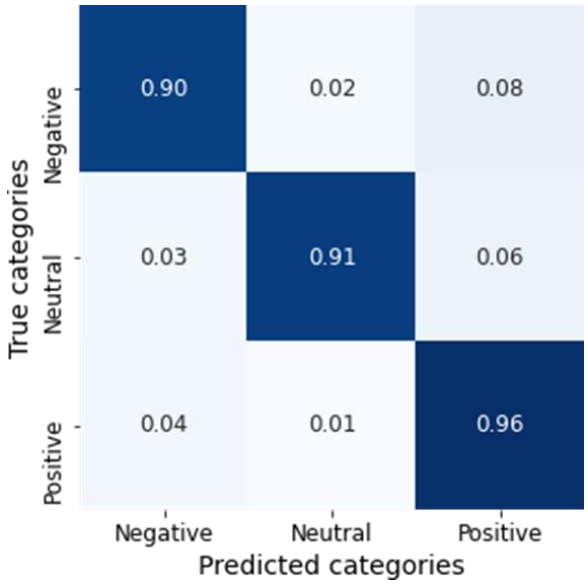

Fig. 6. Confusion matrix for Stochastic Gradient Descent (SGD) model

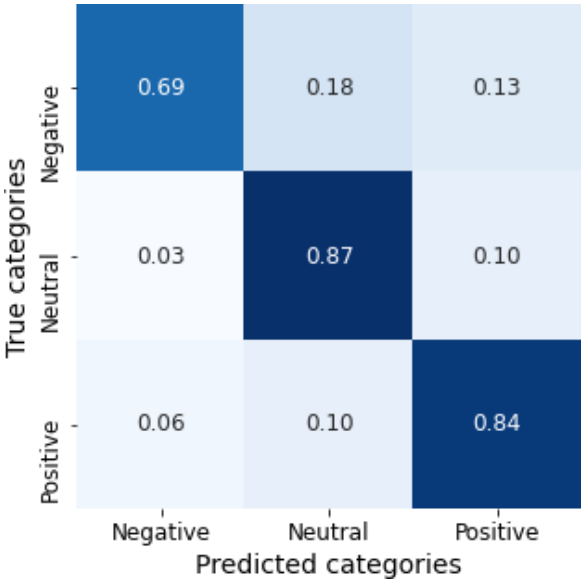

Fig. 9. Confusion matrix for Extreme Gradient Boosting(XGBoost) model

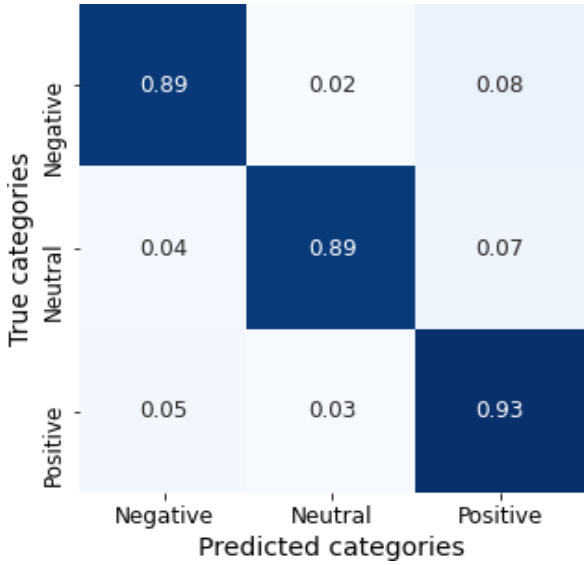

Fig. 7. Confusion matrix for Logistic Regression model

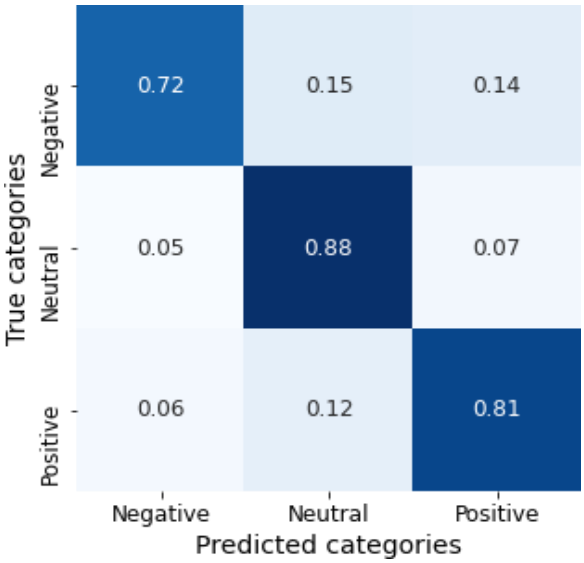

Fig. 10. Confusion matrix for Backpropagation(BPNN) model

4. Conclusion and Future Work

This paper, which aimed to analyze people's attitudes and emotions during the COVID-19 pandemic, was completed successfully. The Twitter platform was chosen for the study because of the data's trustworthiness and the simplicity with which users' tweets could be extracted [KSN]. Almost all countries were tweeting about COVID19 with good sentiments during the study, indicating that people have become accustomed to COVID-19 and that the recovery rate has improved over time. Similarly, it was discovered that individuals worldwide are tweeting phrases like Pandemic, COVID, Virus, Hospitals, Health, Fight, Stay, Safe, Help, Emergency, Death, and Masks with various emotions when analyzing word clouds from various nations. This research gave a thorough examination of people's attitudes and mindsets on Covid-19, allowing us to see that people thought at almost the same level. This research can be used in future studies to look at the changing emotions and feelings of people from different countries and see if there are any notable changes over time. Without further ado, all governments should deploy Fact-checkers in social media to prevent further sharing of unnecessary information for cases that are of such serious concern. Laws can be designed to impose restrictions on sharing false and useless news during emergencies. This work does not have the features to attend multilingual tweets, which could be considered as a probable future work in this direction. Apart from English, tweets from other languages can be collected for analyzing the sentiments of people worldwide [NK21]. As an extension to this work, researchers can think of incorporating emotional intelligence into the tweets so that the sentiments of the people can be further explored in a fruitful approach. Emotional intelligence applied to the tweets will also be an advantageous source of taking measures to put appropriate filters on these tweets, so that the old aged or sensitive people (living alone) do not get targeted to diseases like depression and anxiety [KSN]. The future scope of this project is to try enhancing the accuracy of the LSTM model and XGBoost which was used during the empirical study, to enhance the model evaluation metrics like f1-score and precision. Fuzzy rule-based approaches should be explored to yield better results in identifying sentiments. A very immediate and necessary work could be done by collecting all the available resources and creating an all-in-one repository relating to this pandemic so that it could be easy for statisticians, researchers, doctors, and people worldwide to have a one-stop solution to diseases like the dreaded COVID-19 that has kept the world devastated in the year 2020. Hence, by performing an empirical study on covid 19 data and analyzing sentiments behind various tweets, the psychology of people as well as the repercussions of the pandemic on their lives can be well interpreted. There exists a lot of scope and credibility in this domain of research. The study is for a global cause to mitigate the consequences of the pandemic on people's mental health. By deriving insights from the results obtained, the government can make data-oriented.

References

[1] Dustin Tran, Panos Toulis, and Edoardo M Airoldi. "Stochastic gradient descent methods for estimation with large data sets".

[2] Tianqi Chen et al. "Guestrin, C.: XGBoost: A Scalable Tree Boosting System". In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'16). 2016, pp. 785–794.

[3] Chae Won Park and Dae Ryong Seo. "Sentiment analysis of Twitter corpus related to artificial intelligence assistants". In: 2018 5th International Conference on Industrial Engineering and Applications (ICIEA). IEEE. 2018, pp. 495–498.

[4] Yuanhang Su, Yuzhong Huang, and C-C Jay Kuo. "Efficient text classification using tree-structured multi-linear principal component analysis". In: 2018 24th International Conference on Pattern Recognition (ICPR). IEEE. 2018, pp. 585–590.

[5] Antoine J. P Tixier. "Notes on deep learning for NLP". 2018.

[6] Sattam Almatarneh and Pablo Gamallo. "Comparing supervised machine learning strategies and linguistic features to search for very negative opinions". In: Information 10.1 (2019), p. 16.

[7] Venkateswarlu Bonta and Nandhini Kumaresh and N. Janardhan. "A comprehensive study on lexicon based ap- proaches for sentiment analysis". In: Asian Journal of Computer Science and Technology, 8. S2 (2019), pp. 1–6.

[8] Furqan Rustam et al. "Tweets classification on the base of sentiments for US airline companies". In: Entropy 21.11 (2019), p. 1078.

[9] Yuanhang Su and C-C Jay Kuo. "On extended long short-term memory and dependent bidirectional recurrent neural network". In: Neurocomputing 356 (2019), pp. 151–161.

[10] Yuanhang Su, Ruiyuan Lin, and C-C Jay Kuo. "Tree-structured multi-stage principal component analysis (TMPCA): theory and applications". In: Expert Systems with Applications 118 (2019), pp. 355–364.

[11] Koyel Chakraborty et al. "Sentiment Analysis of COVID-19 tweets by Deep Learning Classifiers—A study to show how popularity is affecting accuracy in social media". In: Applied Soft Computing 97 (2020), p. 106754.

[12] Chhinder Kaur and Anand Sharma. Twitter sentiment analysis on coronavirus using textblob. Tech. rep. EasyChair, 2020.

[13] Zeeshan Shaukat et al. "Sentiment analysis on IMDB using lexicon and neural networks". In: SN Applied Sciences 2.2 (2020), pp. 1–10.

[14] Salvador Manuel Malagon Soldara, Juan Paulo Maldonado Rodriguez, and Jose Vladimir Maldonado Espino.

[15] Miner 'I, "A of Data in a Local Server to Classify Words by the M E´ All of Cosine (Datamining in a Local Server to Classify Words Based On Cosine Method)". In: Educational Tracks 42.137 (2020).

[16] Mario Jojoa Acosta et al. "Sentiment Analysis Techniques Applied to Raw-Text Data from a Csq-8 Question- naire about Mindfulness in Times of COVID-19 to Improve Strategy Generation". In: International Journal of Environmental Research and Public Health 18.12 (2021), p. 6408.

[17] Nalini Chintalapudi, Gopi Battineni, and Francesco Amenta. "Sentimental Analysis of COVID-19 Tweets Using Deep Learning Models". In: Infectious Disease Reports 13.2 (2021), pp. 329–339.

[18] Sourav Das and Anup Kumar Kolya. "Predicting the pandemic: sentiment evaluation and predictive analysis from large-scale tweets on Covid-19 by deep convolutional neural network". In: Evolutionary Intelligence (2021), pp. 1– 22.

[19] Subasish Das and Anandi Dutta. "Characterizing public emotions and sentiments in COVID-19 environment: A case study of India". In: Journal of Human Behavior in the Social Environment 31.1-4 (2021), pp. 154–167.

[20] La´szlo´ Nemes and Attila Kiss. "Social media sentiment analysis based on COVID-19". In: Journal of Information and Telecommunication 5.1 (2021), pp. 1–15.

[21] Furqan Rustam et al. "A performance comparison of supervised machine learning models for Covid-19 tweets sentiment analysis". In: Plos one 16.2 (2021), e0245909.

[22] Mohammad Abu Kausar, Arockiasamy Soosaimanickam, and Mohammad Nasar. "Public Sentiment Analysis on Twitter Data during COVID-19 Outbreak".