Building A Combination of Lexicon Based and Machine Learning Based Model for Twitter Sentiment Analysis

Bharath Raj S

Final Thesis Report

May 2023

ABSTRACT

Sentiment analysis is a booming research field where many research works are being done. Analysing the given data and finding the overall sentiment or emotion of it and classifying it has positive, negative or neutral is sentiment analysis also known as opinion mining. Twitter, a renowned microblogging platform allows the users to express their thoughts and emotions on various issues. If we can collectively predict the overall opinion or sentiment of people in a specific topic, it can help in many different ways like, to know the target audience for brands/businesses, take necessary precautions for a pandemic or trading shares for investors etc., So, the aim of this study is to build a combination of lexicon-based and machine learning-based model which can classify the sentiment of the tweets correctly. A publicly available 'Twitter Sentiment Analysis' dataset from Kaggle is taken for this study. Since the tweets will contain some irregular data, it needs to be cleaned and pre-processed. The process includes tokenization, lemmatization, stemming, removal of stop words, non-English words, numerical values, hashtags, URLs, user mentions etc., Then the sentiments of the pre-processed tweets were predicted using the lexicon-based methods viz. TextBlob, VADER, and SentiWordNet. Sadly, the lexicon-based techniques used didn't perform well on our dataset as all the three techniques got less than 50% accuracy. Hence, for ML classifiers, the annotations of lexicon-based methods were didn't used and just the original labels given was considered as our target variable. This study used nine different ML algorithms with a combination of different data balancing approach, feature extraction techniques, and train-test split ratios to predict the sentiment of the pre-processed tweets. For data balancing, SMOTE, ROS (Random Over Sampling), and RUS (Random Under Sampling) were implemented. And the features of the balanced dataset was extracted using BoW (Bag of Words) and TF-IDF (Term Frequency - Inverse Document Frequency) along with N-grams. The extracted feature matrices were split into train and test datasets in two different ratios i.e., 70:30 and 80:20. For the sentiment classification, nine different ML algorithms were used viz. Logistic Regression (LR), Bernoulli Naïve Bayes (BNB), Multinomial Naïve Bayes (MNB), Decision Trees (DT), Random Forest (RF), Support Vector Classifier (SVC), K-Nearest Neighbour (KNN), eXtreme Gradient Boosting (XGBoost), and AdaBoost. Then the results of the ML models were evaluated based on accuracy, precision, recall, and F1-Score. The results had shown that, KNN and SVC with 80:20 split ratio outperformed all the other combinations with the best accuracy of 94.37% and 94.16%. Followed by them, LR and MNB with 80:20 split ratio on random oversampled dataset attained the best accuracy of 93.08% and 92.02%. Hence, the top models KNN and SVC can be used to predict the sentiment of the people's opinion shared on social media platforms like twitter with the accuracy of 94%. In future, our work can be extended to test the framework's performance with other datasets. And to improve the performance of lexicon-based techniques, we can experiment with advanced pre-processing techniques or use other lexicon-based techniques like SentiStrength and RapidMiner for a more robust data labelling process. Finally, since people express their thoughts and opinions in their regional languages, we can extend this study to work on other regional languages as well.

DEDICATION

I dedicate my thesis work to my family and friends. I completed this thesis only because of my parents. They were my big moral support. In first place, My father encouraged me to pursue master's degree. And my brother guided me to choose Masters in Machine Learning and AI. At first, I thought I won't pull it off, but my family and friends had trust in me and supported me to successfully complete this tough journey.

I also dedicate this dissertation to the memory of my grandfather. We get lot of inspiration from him to lead a happy and healthy life. This is for him.

ACKNOWLEDGEMENTS

I would like to thank my supervisor Shamik Tiwari, for his guidance and support throughout this journey. And thanks to my fellow learners who supported and clarified my doubts. Most importantly, I am grateful for my family's unconditional and loving support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AB..... AdaBoost
ACM..... Association for Computing Machinery
AI..... Artificial Intelligence
AMD..... Advanced Micro Devices
ANN..... Artificial Neural Network
API..... Application Programming Interface
AUROC..... Area Under the Receiver Operating Characteristics
AWS..... Amazon Web Services
B2B..... Business to business
BBC..... British Broadcasting Corporation
BERT..... Bidirectional Encoder Representations from Transformers
BNB..... Bernoulli Naïve Bayes
BoW..... Bag of Words
BTC..... Bitcoin
CBOW..... Continuous Bag of Words
CE-B-MHA..... Comparison Enhanced Bi-LSTM with Multi-Head Attention
CNN..... Convolutional Neural Network
CPU..... Central processing unit
CSV..... Comma-Separated Values
CV..... Cross Validation
DL..... Deep Learning
DT..... Decision Trees
EMO..... Emoticon
ETC..... Extra Trees Classifier
FN..... False Negative
FP..... False Positive
GPOMS..... Google-Profile Of Mood States
GPU..... Graphics Processing Unit
GRU..... Gated Recurrent Unit
IMDB..... Internet Movie Database
IQR..... Interquartile Range
KNN..... K-Nearest Neighbour
LM..... Loughran and McDonald's
LR..... Logistic Regression
LSTM..... Long Short-Term Memory
ML..... Machine Learning
MLP..... Multi-Layer Perceptron

MNB ..... Multinomial Naive Mayes
MOOC ..... Massive Open Online Courses
MR ..... Movie Review
MTL-MSCNN-LSTM ..... Multi-task learning model based on multi-scale CNN and LSTM
NB ..... Naïve Bayes
NEG ..... Negative
NLP ..... Natural Language Processing
NLTK ..... Natural Language Tool-Kit
NN ..... Neural Networks
OL ..... Opinion Lexicon
POS ..... Positive
RAM ..... Random-Access Memory
RF ..... Random Forest
RNN ..... Recurrent Neural Network
ROBERTa ..... Robustly Optimized BERT Pre-training Approach
ROS ..... Random Oversampling
RT ..... Retweets
RUS ..... Random Under Sampling
SMO-DT ..... Sequential Minimal based Optimization Decision Tree
SMOTE ..... Synthetic Minority Oversampling Technique
SSD ..... Solid-State Drive
SST ..... Stanford Sentiment Treebank
SVC ..... Support Vector Classifier
SVM ..... Support Vector Machine
TF-IDF ..... Term Frequency-Inverse Document Frequency
TN ..... True Negative
TP ..... True Positive
TSA ..... Twitter Sentiment Analysis
TWINT ..... Twitter Intelligent Tools
UK ..... United Kingdom
URL ..... Uniform Resource Locator
US ..... United States
VADER ..... Valence Aware Dictionary for Sentiment Reasoning
XGBoost ..... eXtreme Gradient Boosting

CHAPTER 1

INTRODUCTION

1.1 Background of the Study

The online social media platforms such as Facebook, Twitter and Instagram make the people to share their perspective, thoughts or opinions about online products, latest movies, government elections, social issues etc.,(El Rahman et al., 2019). In Twitter for every day 550M tweets were posted. With this huge amount of data, we can easily find out, what is the publics opinion about something whether it is positive, negative or neutral (Mandloi and Patel, 2020). To mention a few,

- Film makers can know how their movie is received by the sentiment of the public reviews.
- A mobile brand can find how their new mobile model launched is performing with the sentiment of the product reviews.
- Youtubers can find out how well their new video is received by the viewers with the comments section's sentiment.
- An investor or a trader can find wheatear to invest/buy in a particular stock or not with the sentiment of the stock in the twitter.

Huge companies/ brands/ political parties will definitely have an account in twitter, so that they can find out the sentiment of their products or services with their own tweet's comments and retweets and plan their next steps (El Rahman et al., 2019).

The reason for choosing twitter for the sentiment analysis are as follows,

- As mentioned earlier, the amount of data we get every day in twitter.
- And the restriction of 280 characters per tweet allows us to capture the context of the message better (Mandloi and Patel, 2020).

In recent years, the sentiment analysis research area is growing rapidly (Naresh and Venkata Krishna, 2021). To perform the sentiment analysis on peoples' opinions shared on social media, different kinds of techniques have been implemented which are lexicon-based techniques or supervised techniques like Machine Learning or Deep Learning algorithms. But in this paper, we are going to focus only on the machine learning models because of the size of the data. If the dataset we have in hand is not so big, there is no scope for Deep learning algorithms to train well (Rustam et al., 2021).

We are getting 'Twitter Sentiment Analysis' dataset from Kaggle.com which has tweets with their corresponding sentiments. But we are going to find the sentiment of the tweets again using lexicon-based approach. So that we can have some comparison between the two results with two different target variables for the ML models we are going to train, which may help us to improve the performance of the model (Rustam et al., 2021).

For the lexicon-based approach, we are going to implement three most widely used techniques i.e., TextBlob, VADER, and SentiWordNet. And for the machine learning part, nine different classifiers are going to be experimented with viz., Logistic Regression, Bernoulli Naïve Bayes, Multinomial Naïve Bayes, SVM, Decision Trees, Random Forest, K-Nearest Neighbour, AdaBoost, and XGBoost. For feature extraction, most widely used two techniques BoW and

TF-IDF were utilized along with N-grams (Mujahid et al., 2021). The performance of both lexicon-based and ML based models are going to be evaluated with four evaluation metrics viz., accuracy, precision, recall and f1-score (Rustam et al., 2021). To carry out this study of Twitter Sentiment Analysis, the following are the necessary software and hardware requirements.

Table 1.1. The software and hardware requirements to carry out this study.

Hardware Requirements	Software Requirements
CPU - AMD Ryzen 4th Gen (4000 series) or an equivalent Intel Core processor will also be optimal. **GPU** - GPU with a size of 8GB minimum. **RAM** - 8 GB Minimum. **Operation System** - Microsoft Windows 10 or higher. **Storage** - SSD of 256 GB is minimum.	A python environment - Anaconda or Jupyter Notebook. Microsoft Word, Excel, Google doc, sheets, and draw.io. Mendeley. Turnitin. Important python libraries required to execute our work: Scikit Learn, Pandas, NumPy, SciPy, SpaCy, TensorFlow, Keras, Matplotlib, NLTK, TextBlob, and vaderSentiment.

1.2 Problem Statement

The unused or lack of knowledge about the people/targeted audience's opinion towards something was mainly addressed in this study. As mentioned in section 1.1, 550M tweets were posted on twitter every day. So, the amount of knowledge about the peoples' opinions/thoughts/reviews towards an event, which is available for free on social media platform like twitter will be missed out by someone like business analysts, investors, governments etc., Hence, this study will help them to get those valuable insights.

1.3 Aim and Objectives

The main aim of this research is to propose a combination of lexicon-based and ML based algorithm which can automatically predict the sentiment of a given tweet as positive, negative or neutral. The identification of the correct sentiment of the tweets allow us to better understand the society/target audience or to make necessary precautionary measures for any surprises.

The research objectives are formulated based on the aim of this study which are as follows:

- To go through the State-of-the-art papers, and techniques used among them.
- To find the best performing lexicon-based technique for this study.
- To find the best combination of pre-processing and feature engineering techniques, which will lead us to the best results.
- To develop and evaluate multiple ML models which can classify the sentiment of the tweets correctly.
- Compare our best model with the SoA.

1.4 Research Questions

In this study about sentiment analysis of twitter data, the questions going to get answered at the end of this study is given below.

1. Will the emoticons in the tweets help us to find the sentiment of the tweets better?
2. What are all the benefits from this study for the business community?
3. What are the pre-processing steps required for this analysis?
4. Which feature engineering technique will give us better results?
5. Which lexicon-based technique performed better with our dataset?
6. Which one of the Machine Learning Models will win the battle of best performing model?

1.5 Scope of the Study

In this study, there are some boundaries/limitations for many reasons, which has been shared in detail below.

1.5.1 In Scope

- Sentiment analysis on Twitter.
- Lexicon-based and Machine Learning-based Models.
- Textual Data and just for lexicon-based techniques emojis/emoticons were included.
- English words.

1.5.2 Out of Scope

- Any other social media platforms except Twitter like Facebook, YouTube etc., Since the tweets are restricted to 280 characters, it is easier to get the context out of it (Mandloi and Patel, 2020).
- Deep learning Models. Because the dataset we have in hand is not so big (Rustam et al., 2021).
- Numerical data, images and videos. The study only involves textual data and emojis /emoticons, as numbers won't add much to predict the sentiment of the tweets.
- Non-English words. Even though twitter allows different languages, in this study we are focusing only on English words to identify the sentiment of the tweets.

1.6 Significance of the Study

As mentioned earlier, in twitter at least 500M tweets were sent per day. So, the data of twitter can tell us all about the society's thoughts, emotions and sentiments (Mandloi and Patel, 2020) which can help us to,

- get better reviews - E.g., Opinion about an election results, brands, products or movies.
- make necessary precautions - E.g., In 2019-2020, It would have helped to eliminate the mass fear from incomplete or wrong information about coronavirus (Samuel et al., 2020).
- know the target audience - E.g., A upcoming OTT platform can target people who prefers web series.
- Make investment plan - E.g., Know the recent trends of share market and which stocks to invest in.

To achieve all this, a good sentiment analysis algorithm/methodology of twitter data would help. And at the end of this study, a combination of lexicon-based and ML based algorithm is going to be built and it will do the job for us.

1.7 Structure of the Study

The structure of the study is as follows. Chapter 1 starts with the background of the study in twitter sentiment analysis. In section 1.2 the problem statement considered for this study is discussed. In section 1.3, aim and objectives of the study is presented. In section 1.4, the research questions considered were presented and then in section 1.5 & 1.6 scope and significance of the study were discussed.

In chapter 2, the previous works done in this area were discussed in detail. In section 2.1, how the previous studies looked into twitter sentiment analysis were discussed. In section 2.2, recent studies on covid related TSA were explored. In section 2.3, 2.4 & 2.5 the recent studies in TSA based on lexicon-based techniques, machine learning, and deep learning-based algorithms were discussed. In section 2.6, the overall summary of this chapter was given.

In chapter 3, the methodology proposed for TSA in this study was discussed in detail. In section 3.1, a small introduction of this chapter was given. In section 3.2, the description of the dataset used for our analysis was discussed. In section 3.3, techniques used to handle the unbalance in the dataset was given. In section 3.4 & 3.5, data cleaning and data pre-processing steps implemented in this study was explained. In section 3.6, the techniques used for feature extraction were explained. In section 3.7, the lexicon and ML based algorithms implemented for sentiment analysis in this study was discussed. In section 3.8, the metrics used to evaluate the algorithms developed were explored. Finally, in section 3.9, the overall summary of this chapter was given.

In chapter 4, the analysis and implementation of the proposed methodology was discussed in detail. In section 4.1, an outline of the implementation process was given. In section 4.2, the basic understanding of the dataset was given. In section 4.3 & 4.4, the data cleaning and pre-processing steps implementation was given. In section 4.5, the data was visually explored. In section 4.6, the process of train test split of the dataset used for this study was given. In section 4.7, the implementation of the different feature extraction techniques was given. In section 4.8, the techniques implemented for balancing the dataset was discussed. In section 4.9, the implementation process of lexicon-based techniques used in this study was given. In section 4.10, the hyperparameter tunning of the ML algorithms used in this study was discussed. Finally, in section 4.11, the overall summary of this chapter was given.

In chapter 5, the results and outcomes of the implemented methodology was discussed. In section 5.1, a small introduction of this chapter was given. In section 5.2, the results of the lexicon-based techniques implemented were explored. In section 5.3, the results based on the feature extraction combinations were explored. In section 5.4, the results based on the data balancing techniques were discussed. In section 5.5, the results based on the train-test split ratios were given. In section 5.6, the best performing ML algorithms were explored. In section 5.7, the overall summary of this chapter was given.

In chapter 6, the conclusion and recommendations based on the results of the twitter sentiment analysis performed was given. In section 6.1, an outline of the final conclusion was given. In section 6.2, the overall outcome and conclusion of the analysis done in this study was discussed. In section 6.3, the contribution to the existing knowledge in this field of research was discussed. Finally, in section 6.4, the future works and recommendations for the development of twitter sentiment analysis was given.

CHAPTER 2

LITERATURE REVIEW

In this section, let's discuss the previous works done in the field of twitter sentiment analysis and what are all the techniques/algorithms used for preprocessing, feature extraction, model building and evaluation. In model building, lexicon-based techniques and supervised techniques like Machine Learning and Deep Learning based algorithms were discussed. To begin with, a broad introduction to twitter sentiment analysis is given in the introduction part. Then studies related to sentiment analysis on Covid-19 related tweets were discussed. Finally, the recent studies on TSA using ML, DL and lexicon-based techniques were explored.

2.1 Introduction

In this digital environment, millions of data are created every second. People tents to share their thoughts, ideas and emotions about products, movies, politics etc., in social media platforms (Mishra et al., 2022). A stat show that 70% of the world's population aware of social media platforms at a global level (Singh et al., 2022). Among the different social media platforms Facebook, Instagram, WhatsApp, Twitter etc., companies, researchers and who are interested in opinion mining mostly prefer twitter, because of its unique properties i.e.,

- Openness - In twitter, we don't need to follow anyone to see their posts, opinions, reviews etc., which makes it easier to collect data.
- The length limitation - In twitter, a post can be only 140 characters long which makes the content crisp, precise and reliable.
- Hashtags in twitter helps the companies and brands to collect and analyze the data specific to their products and services which help them to, know what people think about their services, take precautionary steps etc.,
- Certain level of fidelity (Bouazizi and Ohtsuki, 2019).

As of 2023, Twitter has 450 million monthly active users and 500 million tweets per day which gives us vast amount of information and from that we can make useful insights. Twitter is used by different kinds of people viz common man, sports players, diplomats, celebrities, companies, and popular brands across the world (Singh et al., 2022). This had led a boom in sentiment analysis on twitter data to better understand the opinions shared by the people (Pandya et al., 2021).

The field of sentiment analysis has become the most significant techniques of natural language processing (Chandralekha et al., 2022). Sentiment analysis (also known as contextual mining or opinion mining) is a method that extracts the sentiment, opinion, or emotion of a text (Pandya et al., 2021). This helps the business, organizations, and brands to analyze the reviews of their customers, and understand people's feelings towards their products or services so that they can make necessary steps or develop a new business strategy etc. (Singh et al., 2022). The fundamental principle of sentiment analysis is extracting the polarity of a word or document (i.e., whether the word/document is positive, negative, or neutral) and then building an algorithm to predict the sentiment automatically (Khan and Yadav, 2021).

Twitter Sentiment Analysis (TSA) is becoming a crucial field of research. It is one of the best ways to find the people's opinion about a product, movies, politics, or brands etc., "R and Python are two widely used languages for sentiment analysis on twitter data" (Khan and Yadav, 2021). To find the sentiment of a text or tweet in this case, we use NLP (Natural Language

Processing) and Machine Learning algorithms. First, the tweet we have in hand need to be preprocessed and then need to be labelled according to the polarities of the tweet which can be done with the help of NLP techniques. Now, we can apply ML or deep learning algorithms on top of it to automatically classify the tweets as positive, negative, or neutral respectively. However, there are some challenges which have not been touched mostly viz. opinion object identification, maintaining opinion time, and hidden sentiments identification where we find the actual hidden sentiment of the tweet like anger, joy, happiness and sad (Bouazizi and Ohtsuki, 2019).

The researchers have different motivations or purposes to dive into TSA. For example, the main purpose of this study (Bengesi et al., 2023) was the monkeypox outbreak, to better understand the people's perception about the disease and to take precautionary steps. Likewise, the table 2.1 shows us the different motivations of researchers to begin their study in twitter sentiment analysis.

Table 2.1. Researchers' motivation/purpose for the study on Twitter Sentiment Analysis.

Citation	Motivation/Purpose
(Rustam et al., 2021)	To handle the pandemic situation better and make informed decisions.
(Samuel et al., 2020)	To handle the incomplete and inaccurate information about Covid-19.
(Mujahid et al., 2021)	To study the effectiveness of E-learning during Covid-19 crisis.
(Onan, 2021)	To analyze the MOOCs reviews and build an effective sentiment classification algorithm.
(Neogi et al., 2021)	To understand the public sentiment about the farmers'' protest in digital world.
(Bengesi et al., 2023)	To better understand the people's perception about the disease and to take precautionary steps.
(Mishev et al., 2020)	To bring out actionable signals from the finance related news.
(Jin et al., 2020)	Performance of classification models decreases because of the multi-tasking problem.
(Aslam et al., 2022)	To understand the peoples' perception about the cryptocurrency and its use.
(Abdullah et al., 2019)	To overcome the underlying challenges of multi-source and multi-domain sentiment analysis.
(Xiao et al., 2019)	To reduce the effect of the imbalanced data class on the performance of the model.
(Kumar and Zymbler, 2019)	To improve the customers, experience on their airlines.

2.2 Sentiment analysis on Covid-19 related tweets

At present, the sentiment analysis on Twitter won't be complete if we didn't speak about the pandemic of corona virus. The outbreak of corona virus caused so many issues across the world. It was not only about the quarantines, vaccinations, or health concerns, along with it there were other serious issues viz. share markets collapsed, people lost or unable to continue their jobs, students unable to continue their studies, entire food systems were spoiled, and it was difficult to even get the basic necessities to live. But on top of all these, the main concern was the misleading or incorrect information people got through social media caused mass fear and panic phenomena (Samuel et al., 2020). So, to understand and analyze people's thoughts and opinions about the pandemic, sentiment analysis on tweets related to covid had helped (Rustam et al., 2021).

To begin this sentiment analysis on twitter, we need the covid related tweets. And in most of the previous studies the authors had used twitter API or Twippy library to extract the tweets from twitter. And some of the common keywords used while extracting was "corona", "coronavirus", "covid", "pandemic", "covid-19", "ncov2019", "2019ncov" etc., (Rustam et al., 2021). In this study (Mujahid et al., 2021) the authors discussed about the online education during corona virus. And they have used keywords like "coronaeducation", "covidneducation", "distancelearning", "Onlineclasses", and "onlinelearning" while extracting the tweets using twitter API. Since the data were scrapped directly from twitter, they were not labelled. The table 2.2. shows the labelling technique used in the previous studies. Mostly there were only three techniques used repeatedly for labelling i.e., TextBlob, VADER and SentiWordNet.

TextBlob: TextBlob, a python library uses subjectivity and polarity of the text to label the data as positive, negative, or neutral. (Chandralekha et al., 2022).

VADER: which stands for Valence Aware Dictionary and Sentiment Reasoner. VADER is a rule-based technique which considers the intensity of the emotion (positive/negative) in the text. It works well on social media text like product reviews, tweets etc., (Mujahid et al., 2021)

SentiWordNet: SentiWordNet is an opinion lexicon which score the text as positive, negative, or neutral based on the average of the word synsets score (Remali et al., 2022).

Table 2.2. The dataset and labelling technique used in the previous studies.

Citation	Dataset	Labelling Technique Used
(Rustam et al., 2021)	Using Twippy Library extracted 7528 tweets.	TextBlob
(Samuel et al., 2020)	Using Twitter API extracted over 900,000 tweets from February to March 2020.	Combination of Syuzhet and sentimentr
(Mujahid et al., 2021)	Using Twitter API, extracted over 17,155 tweets.	TextBlob, VADER, and SentiWordNet. TextBlob out formed the others.
(Khan and Yadav, 2021)	Used Twippy Library to extract tweets.	SentiWordNet
(Chandralekha et al., 2022)	Using Twippy Library extracted 1,24,000 tweets.	TextBlob
(Remali et al., 2022)	Using Twitter Intelligent Tools (TWINT) scraped 38,602 tweets posted between 23rd July and 14th August 2020.	VADER performed better than SentiWordNet
(Gupta et al., 2021)	Using Twippy library extracted 12,741 tweets from April 5, 2020, to April 17, 2020.	Intersection of TextBlob and VADER lexicon.
(Ghasiya and Okamura, 2021)	A total of 1,02,278 news articles and headlines from eight different newspapers and four different countries between January 1st, 2020, and December 1st, 2020.	Combination of VADER, Textblob, and SentiWordNet.

In this study (Gupta et al., 2021) since they had the data in hand is straightly extracted from twitter, it had to be cleaned and preprocessed before applying any ML algorithms. So, the authors had applied the preprocessing techniques mentioned in the below figure 2.1. Firstly, they removed the noisy data which are insignificant for the sentiment analysis like symbols, numbers, hashtags, URLs etc., Then the removal of punctuation, tokenization, stop words removal, stemming and lemmatization was implemented. In another study (Samuel et al., 2020), the authors had used parts of speech tagging and parsing as one of the preprocessing techniques. So, we had created a table 2.3. with the most common preprocessing techniques used in recent years and whether they were used as one of the pre-processing techniques in the recent studies of covid related sentiment analysis.

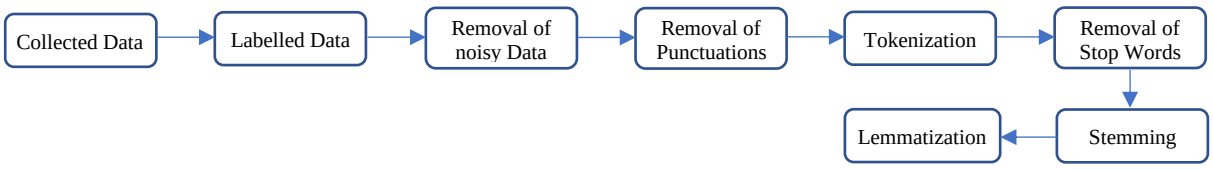


Figure 2.1. Pre-processing techniques used in (Gupta et al., 2021).

Table 2.3. Pre-processing techniques used in the recent studies of covid related sentiment analysis.

Citation	Removal Of						Lower Case	Stemming	Lemmatization	Tokenization
	Stop Words	User Mentions	Symbols	URLs	Numeric Values	Hashtags				
(Rustam et al., 2021)	✓	✓	✓	✓	✓	×	✓	✓	×	×
(Samuel et al., 2020)	×	×	×	×	×	×	×	✓	✓	✓
(Mujahid et al., 2021)	✓	×	✓	✓	✓	✓	✓	✓	✓	×
(Khan and Yadav, 2021)	✓	✓	✓	✓	×	✓	×	✓	✓	×
(Chandralekha et al., 2022)	✓	×	✓	✓	×	✓	×	×	✓	✓
(Remali et al., 2022)	✓	✓	✓	✓	×	✓	×	×	✓	✓
(Gupta et al., 2021)	✓	✓	✓	✓	✓	✓	×	✓	✓	✓
(Ghasiya and Okamura, 2021)	✓	×	✓	×	×	×	✓	×	✓	✓

In the study (Rustam et al., 2021), the authors had used a unique technique to extract the features from the preprocessed data i.e., the concatenation of Bag of Words (BoW) approach and TF-IDF vectorizer. Then they implemented ETC which uses a random number of decision trees (week learners) to train with the distinct dataset samples, and it outperformed the other five classifiers (RF, XGBoost, SVC, DT and LSTM) the authors had experimented with. The table 2.4. indicates that BoW, TF-IDF or CountVectorizer, one of which was used as the feature extraction technique for the studies that experiments with ML algorithms.

In this study (Remali et al., 2022) the authors tried to find the sentiment of the people about the online education during covid-19 pandemic. And with the help of the ML technique SVM, the authors found that majority of the people (i.e., 49.53%) feels positive about it, 34.42% and 16.05% of the people feels neutral and negative about the same. In this study (Mujahid et al., 2021) the authors had tried to find the effectiveness of the e-learning during covid-19 but the data they had after cleaning and preprocessing was not balanced, so they used SMOTE (Synthetic Minority Oversampling Technique) to balance the dataset by oversampling the minority class. With that they had experimented with both ML and deep learning algorithms, but ML algorithms performed better because of the smaller size of the dataset. And some of the concerns for the online learners turned out to be network issues, unable to grasp the concepts through e-learning, uncertainty towards the opening date of schools/colleges.

One of the important precautionary decisions taken by the government during covid-19 pandemic is lockdowns and quarantines. In this study (Gupta et al., 2021) the authors have analyzed the people's opinion towards the lockdowns in India. For that, covid related tweets were taken, once the cleaning, preprocessing, and labelling were done. Eight different ML classifiers were tried and LinearSVC with unigrams got the highest accuracy of 84.4%. And the results were turned out to be, 48.69% people were positive/supportive about the lockdowns. 21.50% people felt negative and 29.81% people were neutral.

In this study (Ghasiya and Okamura, 2021) the authors analyzed four different countries' eight major newspapers to better understand the people's sentiments on covid-19 crisis. They had tried nine different algorithms for sentiment classification and ROBERTa attained the highest accuracy of 90%. And the results shown that UK was the worst affected country when compared to India, Japan, and South Korea. Also, UK had the most negative headlines (i.e., 73%) when

compared to other countries. On the other hand, South Koreas seems to be handled the pandemic very well. It had the most positive headlines (i.e., 54%) and had only 25 deaths/million.

This study (Samuel et al., 2020) has given an interesting analysis, the accuracy of the ML models' predictions was decreased when the length of the tweets increased. The accuracy of the Naive bayes model for shorter tweets is 91% and for longer tweets 57%. And with logistic regression the accuracy decreases from 74% to 52% as the length of the tweets increases. In this study (Khan and Yadav, 2021) the authors have tried to find the best ML classifier for sentiment analysis, so they had experimented with nine different classifiers and Logistic Regression attained the highest test score i.e., 0.785. In another study (Chandralekha et al., 2022) the authors had tried eleven different classifiers and SVM with the unigram got the highest accuracy of 93%. Table 2.4. shows the evaluation metrics used in all the covid related sentiment analysis studies. And one of the main evaluation metrics used in all the studies is accuracy. Then precision, recall and f1-score were used in most of the studies.

Table 2.4. Techniques/Algorithms used by recent studies on covid related sentiment analysis.

Year	Citation	Feature Extraction	Models/Algorithms	Evaluation Metrics
2021	(Rustam et al., 2021)	Concatenation of BoW and TF-IDF	ETC outperformed the other five models (RF, XGBoost, SVC, DT and LSTM).	Accuracy, precision, recall, and F1 score
2020	(Samuel et al., 2020)	N-grams	For short tweets, Naïve Bayes got 91% accuracy. For longer tweets, the performance rapidly decreased i.e., 57%.	Accuracy, Sensitivity, Specificity
2021	(Mujahid et al., 2021)	BoW and TF-IDF	DT, SVM and RF achieved an accuracy of 95% with BoW approach. And RF achieved the highest accuracy of 95% with TF-IDF vectorizer.	Accuracy, Precision, Sensitivity, and F1 score
2021	(Khan and Yadav, 2021)	N-grams with Count vectorizer and TF-IDF	Logistic Regression with TF-IDF vectorizer attained the highest test score i.e., 0.785.	Accuracy
2022	(Chandralekha et al., 2022)	N-grams with Count vectorizer and TF-IDF	SVM using unigram shown the best performance i.e., 93%.	Accuracy, precision, recall, and F1 score
2022	(Remali et al., 2022)	TF-IDF vectorizer	SVM with 80:20 split got 90.41% accuracy when VADER lexicon was used.	Accuracy, precision, recall, and F1 score
2021	(Gupta et al., 2021)	N-grams with CountVectorizer	LinearSVC with unigram got the highest accuracy of 84.4%.	Accuracy, precision, recall, F1 score, and AUROC curves.
2021	(Ghasiya and Okamura, 2021)	-	ROBERTa attained the best validation accuracy of 90%.	Accuracy

2.3 Recent works in TSA based on Lexicon-based Techniques

For sentiment analysis, one of the most reliable approaches is lexicon-based approach. In recent days, the lexicon-based techniques were used by many authors in multiple domains. (Wunderlich and Memmert, 2020) had used lexicon-based techniques for sports related twitter sentiment analysis. (Taj et al., 2019) had presented a lexicon-based model for sentiment analysis of BBC news articles. (Hota et al., 2021) had compared the most affected six countries sentiments towards covid-19 including India using lexicon-based and VADER based approaches. (Neshan and Akbari, 2020) focused on finding the effect of combining multiple ML classifiers into one meta-classifier which uses the results of lexicon-based techniques as inputs. (Juanita et al., 2022) proposed a model for emotion detection in E-Marketplace users' opinions using lexicon-Based and Naïve Bayes Model. To summarize the recent studies' motivation and objectives for the lexicon-based sentiment analysis, table 2.5 shows the objectives and their future works/challenges of the recent studies on lexicon-based sentiment analysis.

One of the important steps for a lexicon-based approach after gathering the data is preprocessing and cleaning the data. Preprocessing steps in (Mishev et al., 2020) include: Tokenization, stop-word removal, stemming, named entity conversion, and left padding.

In this study (Neshan and Akbari, 2020) the authors included POS tagging, Verbs, Adverbs, and Adjectives Extraction, Lemmatization, and Negotiation Rules. (Hota et al., 2021) included removing noise: hashtag, profile picture, retweet, emoji, URLs, usernames, numbers, punctuations, white space, and @ sign. Then lowercase conversion, Tokenization, Stop words removal, and Lemmatization. Preprocessing steps used in recent studies on lexicon-based sentiment analysis is given in table 2.6.

The most commonly used lexicon-based techniques for sentiment analysis were VADER, SentiWordNet, and TextBlob. In this study (Kumaresh et al., 2019) the authors had used these three lexicons and found that VADER had outperformed both TextBlob and SentiWordNet with the accuracy of 77%. Also, they mentioned that if the sentiment was the only thing need to be extracted from blog text with speed, then VADER was the best option. One more advantage of VADER was it supports emoji sentiments. Hence VADER was the best lexicon-based technique for twitter sentiment analysis. In this study (Al-Shabi and Al-Shabi, 2020) the authors had experimented with five different lexicons and the results had shown that VADER was the best performing lexicon with the accuracy of 72% and 65% in Stanford and Sandars dataset respectively. In this study (Juanita et al., 2022) TextBlob and VADER had been tried along with I Bayes models. The best model was turned out to be TextBlob with Multinominal Naïve Bayes model with an accuracy of 90% on the Lazada dataset, 85% on the Tokopedia dataset, and 85% on the Shopee dataset.

In some cases, a combination of lexicon and ML based models were used. In this study (Moussa et al., 2018) the authors had built an ensemble of five different lexicons (MaxDiffTwitter, HuAndLiu, VADER, SenticNet, and SentiWordNet) and experimented with a collection of four different datasets (52,039 reviews). The results had shown that the ensemble model if performing better than the five lexicons when they were used individually.

In this study (Neshan and Akbari, 2020) a hybrid model had been built using three lexicons and four machine learning based models i.e., the results of the three lexicons (SentiWordNet, SentiStrength, and Liu's lexicon) were given as inputs for the machine learning models (NB, SVM, MLP, and DT) and then the results of the ML models were given to a meta-classifier. This hybrid model outperformed all the four models (when they were used individually) with an accuracy of 77.94% in amazon reviews dataset and 76.3% in movies reviews dataset. The popular lexicon-based methods used in the recent studies on sentiment analysis was shown in table 2.6.

In this study (Nugroho, 2021), to forecast the results of US presidential elections, a VADER based sentiment analysis model was built and experimented on the twitter data of one week before the elections. The results were not exactly the same as the real data provided by BBC, but it did predict the Joe Biden's victory over Donald Trump. In future, the authors had an idea of adding more news data from various online media so that it can be used as comparative data for sentiment analysis. In this study (Hota et al., 2021) the authors had used a lexicon and VADER based approaches to compare the most affected six countries sentiments towards covid-19 including India. And the results had shown that majority of the people expressed positive sentiment towards covid-19. In future, the authors suggested, top 6 to 10 most covid affected countries can be analyzed using DL and hybrid methods. And work with multiple emotions such as joy, sadness, hate etc.,

In this study (Taj et al., 2019) the authors had proposed a lexicon-based framework for sentiment analysis on BBC news articles. And it was observed that business and sports related articles had more positive sentiments whereas entertainment and tech related articles had more negative sentiments. In future, the authors planned for the sentiment analysis on news data with various ML algorithms need to be built to develop an online application from where users can read topics of their interest. Future works of this study (He, 2022) were, to improve the accuracy of the sentiment analysis, how to define the scope of the context and analyze the influence of the context on sentiment polarity. Then explore what are the factors that affects the accuracy of ML and DL algorithms.

The below table 2.5 summaries the future works and challenges of the recent studies on lexicon-based sentiment analysis. But the results of the lexicon-based methods were not so good when compared to ML and DL methods (Neshan and Akbari, 2020).

Table 2.5. Objectives, best performing models and the future challenges of the recent studies on lexicon-based sentiment analysis.

Citation	Objectives	Best performing model	Future works/challenges
(Moussa et al., 2018)	To propose a generic lexicon-based framework for sentiment analysis.	The combination of five lexicons (MaxDiffTwitter, HuAndLiu, VADER, SenticNet, SentiWordNet).	1. To test the framework with other datasets that is specific to a domain. 2. To test another sentiment engine, synonyms features and word sense disambiguation. 3. To improve the accuracy of neutral bias module
(Al-Shabi and Al-Shabi, 2020)	To Evaluate the performance of the most important Lexicons used for Sentiment analysis.	VADER	1. To work with other lexicons such as Opinion Finder and GPOMS. 2. To work with different datasets and study the effect of data pre-processing in the accuracy of the models.
(Hota et al., 2021)	To compare the most affected six countries sentiments towards covid-19 including India using lexicon-based and VADER based approaches.	Lexicon based model and VADER.	1. To analyse the sentiments of top 6 to 10 most covid-19 affected countries using DL methods and hybrid methods. 2. To work with multiple emotions such as joy, sadness, hate etc., 3. To work with advanced feature selection techniques.
(Taj et al., 2019)	To propose a lexicon-based framework for sentiment analysis on BBC news articles between 2004 and 2005.	SentiWordNet	Sentiment analysis on news data with various ML algorithms to develop an online application from where users can read topics of their interest.
(Juanita et al., 2022)	To propose a model for emotion detection in E-Marketplace users' opinions using lexicon-Based and Naïve Bayes Model.	TextBlob with Multinomial Naïve Bayes model	To translate the dataset which contains Indonesian user opinions about the E-Marketplace into English.
(He, 2022)	"To assist the future investigators comprehending the influence of online reviews with heterogeneous polarity and the possible improvement of exiting words".	Deep Learning Methods	1. To explore what are the factors that affects the accuracy of ML and DL algorithms. 2. To improve the accuracy of the sentiment analysis, how to define the scope of the context and analyse the influence of the context on sentiment polarity.
(Nugroho, 2021)	To forecast the results of the US presidential election and compare it with the actual results.	VADER	To add more news data from various online media so that it can be used as comparative data for sentiment analysis.

Table 2.6. Lexicons and pre-processing steps used by recent studies on lexicon-based sentiment analysis.

Citation	Preprocessing Steps	Lexicons Used	Results
(Mishev et al., 2020)	Tokenization, stop-word removal, stemming, named entity conversion, and left padding.	Harvard IV-4, Loughran and McDonald's (LM).	ML and DL based algorithms are twice as good as lexicon based models. NLP transformers Distilled-BERT and Distilled-ROBERTa were the best performing models.
(Neshan and Akbari, 2020)	POS tagging, Verbs, Adverbs, and Adjectives Extraction, Lemmatization, and Negotiation Rules.	SentiWordNet, SentiStrength, and Liu's lexicon.	In amazon reviews dataset, meta-classifier (combination of NB, SVM, MLP, and DT algorithms) got the highest accuracy i.e., 77.94%. In movies reviews dataset also, meta-classifier got the highest accuracy i.e., 76.3%. MLP was the second best in both the datasets. The performance of lexicon based models were very minimum when compared to ML models.
(Kumaresh et al., 2019)	Tokenization, Stop words removal, Stemming, and POS tagging.	NLTK, VADER, and TextBlob.	VADER performed better than TextBlob and NLTK with an accuracy of 77% and F1-Score of 81.60%.
(Moussa et al., 2018)	Stop words removal, POS tagging, stemming or lemmatization, and exaggerated word shortening.	MaxDiffTwitter, HuAndLiu, VADER, SenticNet, and SentiWordNet	In a collection of four different datasets (52,039 reviews), the combination of five lexicons had performed better than the five lexicons when they were used individually.
(Al-Shabi and Al-Shabi, 2020)	-	VADER, SentiWordNet, SentiStrength, Liu and Hu opinion lexicon, and AFINN-111	VADER was the best performing lexicon with the accuracy of 72% and 65% in Stanford and Sandars dataset respectively.
(Hota et al., 2021)	1. Removing noise: hashtag, profile picture, retweet, emoji, URLs, user names, numbers, punctuations, white space, and @ sign 2. Lowercase conversion 3. Tokenization 4. Stop words removal 5. Lemmatization.	Lexicon based model and VADER.	In lexicon based approach, negative sentiment was 20.39%. In comparison, in VADER based approach the negative sentiment was 33.68% and similar higher results were found for positive and neutral sentiments also. In lexicon based approach, UK had the highest negativity (23.03%) followed by France (22.71%). In VADER based approach, France had the highest negativity (35.92%) followed by UK (35.68%).
(Juanita et al., 2022)	Removing duplicates, lowercase conversion, stop words removal, stemming, punctuations, numbers, and tags removal.	VADER and TextBlob	TextBlob with Multinomial Naïve Bayes model was the best performing model with an accuracy of 90% on the Lazada dataset, 85% on the Tokopedia dataset, and 85% on the Shopee dataset.
(Nugroho, 2021)	Stop words, punctuations, numbers, and white space removal, lowercase conversion, and Stemming	VADER	The results obtained by the sentiment analysis was exactly not the same as the real data but the results represented Joe Biden's victory over Donald Trump correctly.

2.4 Recent works in TSA based on Machine Learning based Techniques

There are various machine learning techniques to analyze the sentiment of twitter data (Mandloi and Patel, 2020). In this study (Dagar et al., 2021) the authors had tried three different ML techniques viz Logistic Regression, Multinomial Naïve Bayes and Linear SVM. For this study, they used a publicly available Kaggle dataset "Sentiment140" which had 16 lakhs labelled data. At the end, Linear SVM attained the best accuracy of 83.71%. In another study (Pandya et al., 2021) the authors had used the same "Sentiment140" dataset and tried both ML and Deep Learning algorithms. But the best performing Bi-LSTM model of this study was unable to outperform the Linear SVM model of the previous study (Dagar et al., 2021). This shows us the efficiency of the SVM algorithm in sentiment analysis of textual data. To reassure the point made, in this study (Bengesi et al., 2023), a similar kind of dataset was used, and six different ML algorithms were tried. The results shown us, again the SVM model outperformed the other five algorithms and got an accuracy of 93.48%. So, we can clearly state that SVM is one of the best models for sentiment analysis, at least among the ML algorithms. So, in this study (Satrya et al., 2022) the authors decided to go with SVM and try different balancing techniques and

different train-test split ratios. After experimenting with all the combinations, the under sampling with 80:20 split ratio got the best results.

The table 2.7. shows us the datasets used in the recent studies on twitter sentiment analysis using ML techniques, along with the different labelling techniques used for unlabeled datasets, observation period, and the data count for positive, negative, and neutral sentiments are mentioned. In one of the studies (Naresh and Venkata Krishna, 2021), the authors had used just 1000 tweets and proposed a unique model called SMO-DT (Sequential Minimal based Optimization Decision Tree) which can be used when time is taken as concern. Also, this model attained the highest accuracy i.e., 89.47% when compared with other ML algorithms namely KNN, SVM, DT and KNN+SVM. As we discussed earlier, in this study (Bengesi et al., 2023) the authors got the best performing model as SVM, but they also tried different labelling techniques, pre-processing techniques and feature extracting techniques. And ended up with the best performing combination as TextBlob annotation + Lemmatization + CountVectorizer + SVM. To summarize the techniques used in the recent studies on twitter sentiment analysis, table 2.8. shows us the different feature extraction techniques, ML algorithms and evaluation metrics used in the recent studies on TSA.

Table 2.7. The datasets and labelling techniques used in the recent studies on TSA using ML algorithms.

Citation	Dataset	Labelling Technique	Number of Tweets				Observation Period
			Positive	Negative	Neutral	Total	
(Dagar et al., 2021)	Sentiment140 - A labelled dataset publicly available on Kaggle.	Already Labelled	8L	8L	-	16L	Not mentioned
(Naresh and Venkata Krishna, 2021)	The airline-based tweets were extracted using an API, which consists of id, sentiment, date, time etc.,	Not mentioned	466	277	257	1000	Not mentioned
(El Rahman et al., 2019)	Using twitter API, 7000 tweets each for McDonalds and KFC were extracted.	Not mentioned	4260	2900	6840	14000	Not mentioned
(Neogi et al., 2021)	A python library Twippy with keywords like "farmers", "protest" and "farmers protest" was used to extract the tweets.	TextBlob	5000+	3000+	8000+	18000	5th November 2020 to 5th Match 2021
(Ho et al., 2019)	The dataset "Twitter US Airline Sentiment" taken from Kaggle has tweets about 6 major US airlines.	Already Labelled	2361	9179	3100	14640	Feb 2015
(Bengesi et al., 2023)	A python library Twippy with keyword "monkey-pox" was used to extract the multilingual tweets.	VADER	29295	27628	50473	107396	July 2022 to September 2022
		TextBlob	39594	14788	52954		
(Kumar and Zymbler, 2019)	Major airlines-based tweets were extracted using twitter API.	Not mentioned	-	-	-	120766	1 March 2019 to 11 March 2019
(Satrya et al., 2022)	A python library twint was used to extract bitcoin related tweets.	TextBlob	12283	4904	13012	30199	June 2022
(Pandya et al., 2021)	Sentiment140 - A labelled dataset publicly available on Kaggle.	Already Labelled	8L	8L	-	16L	Not mentioned
(Gupta and Joshi, 2021)	The benchmark Twitter data set SemEval-2013 Task 2 was taken.	Already Labelled	4284	1646	5286	11216	January 2012 to January 2013

Table 2.8. The feature extraction techniques, ML algorithms and evaluation metrics used in the recent studies on TSA.

S No	Citation	Feature Extraction	Algorithms / Models	Evaluation Metrics
1	(Dagar et al., 2021)	Unigrams and Bigrams	Linear SVM had the highest accuracy when compared with LR and MNB i.e., 83.71	Accuracy, Precision, Recall, F1-Score, AUROC
2	(Naresh and Venkata Krishna, 2021)	SMO	Sequential Minimal based Optimization Decision Tree (SMODT)	Accuracy, Precision, Recall, F1-Score
3	(El Rahman et al., 2019)	-	Based on cross validation, Maximum Entropy had the highest accuracy.	Accuracy, Precision, Recall, F1-Score
4	(Neogi et al., 2021)	BoW and TF-IDF	Random Forest model with the BoW (Bag of Words) features.	Confusion Matrix, Accuracy, Precision, Recall, F1-Score
5	(Mandloi and Patel, 2020)	Unigram, bigram, and n-gram features.	Naïve Bayes Classifier had the highest accuracy when compared with SVM and Maximum Entropy i.e., 86%	Accuracy, Precision

6	(Ho et al., 2019)	Simple n-gram CountVectorizer and TF-IDF Vectorizer.	LR, MNB, Random Forest, SVM and Perceptron	Precision, Recall, F1-Score
7	(Bengesi et al., 2023)	Count vectorization and TF-IDF vectorization.	SVM + TextBlob annotation + Lemmatization + CountVectorizer had the highest accuracy of 93.48%	Accuracy, Precision, Recall, F1-Score
8	(Singh et al., 2022)	TF-IDF	SVM had the highest accuracy when compared with RF, GB and XGBoost i.e., 83.74	Accuracy, Precision, Recall, F1-Score
9	(Satrya et al., 2022)	TF-IDF	SVM with Under sampling and a ratio of 80:20 train-test split was the best model.	Confusion Matrix, Accuracy, Precision, Recall, F1-Score
10	(Mishra et al., 2022)	-	Naïve Bayes Classifier	Accuracy, Precision, Recall, F1-Score

In this study (Ho et al., 2019) the authors had shown us that combining diverse scoring systems can increase the performance of the ML algorithms. And in this study (Gupta and Joshi, 2021) the authors particularly focused on the negation handling so that the polarity detection of the given text can be improved. Hence, they proposed a feature-based twitter sentiment analysis system with the improved negation handling. In this study (El Rahman et al., 2019) the authors did the sentiment analysis on two restaurants twitter data i.e., KFC and McDonalds to find out which one is more popular. They had experimented with six different ML algorithms and based on cross validation; the best performing model is maximum entropy with the accuracy of 78%. The results shown us, McDonalds is more popular when compared with KFC.

In this study (Khanam and Sharma, 2021) the best performing model was decided based on the precision, recall and F1-Score likewise, in this study (Neogi et al., 2021) confusion matrix is considered as one of the important metrics to determine the best performing model. In this study (Kumar and Zymbler, 2019) the best performing model is determined based on the user-defined metrics viz. Support, Confidence and Lift. Table 2.9. shows us the most common evaluation metrics used in the recent studies on TSA using ML algorithms.

Table 2.9. Most common evaluation metrics values of best performing models in recent TSA related studies using ML algorithms.

S No	Citation	Best performing Algorithms/Models	Evaluation Metrics			
			Accuracy	**Precision**	**Recall**	**F1-Score**
1	(Dagar et al., 2021)	Linear SVM	83.71	82.28	88.41	81.17
2	(Naresh and Venkata Krishna, 2021)	Sequential Minimal based Optimization Decision Tree (SMODT)	89.47	91.60	89.50	96.30
3	(El Rahman et al., 2019)	Maximum Entropy	78.00	58.00	78.00	67.00
4	(Neogi et al., 2021)	Random Forest	96.60	97.33	95.33	96.00
5	(Mandloi and Patel, 2020)	Naïve Bayes Classifier	86.00	88.69	-	-
6	(Bengesi et al., 2023)	SVM + TextBlob annotation + Lemmatization + CountVectorizer	93.48	93.43	93.48	93.42
7	(Singh et al., 2022)	Support Vector Machine (SVM)	83.74	84.00	92.08	87.85
8	(Satrya et al., 2022)	SVM with Under sampling and a ratio of 80:20 train-test split	94.64	94.59	93.78	95.43
9	(Mishra et al., 2022)	Naïve Bayes Classifier	94.00	76.00	75.00	74.00

In this study (Neogi et al., 2021) the authors tried to find the people's opinion on farmers' protest against the three farm acts passed, so they implemented the sentiment analysis on tweets related to this protest. They experimented with four different ML algorithms and the best performing model is turned out to be Random Forest with the accuracy of 96.6%. And one of the main future works of this study was, how the protest caused the covid spread since the protest was held when there is a rise of covid-19 cases in India. In this study (Singh et al., 2022) the authors proposed a ML based method (SVM) to analyze the social media data for sentiment analysis on text data. But they are fixed with the TF-IDF feature extraction technique. So, in future they had decided to work on other feature extraction techniques. Same as that, in this

study (Bengesi et al., 2023) they had TextBlob + Lemmatization + CountVectorizer + SVM as their best performing model, and in future they are planning to work on other different techniques like doc2Vec for word embedding, Azure for labelling the sentiments and to work on deep learning and transformers. One of the common future works of recent studies is to work on other regional languages' tweets. Table 2.10. shows us the summary of important future works of the recent studies on TSA using ML algorithms.

Table 2.10. Objectives, model used and the future challenges of the recent studies in TSA using ML models.

S No	Citation	Objective	Algorithms / Models	Future Works/Challenges
1	(Dagar et al., 2021)	To help business organizations, entrepreneurship, etc., to get a deeper insight about their products and services.	Linear SVM	1. To work on other regional languages. 2. To build a hybrid classifier which can analyze the complex emotions like sarcasm.
2	(Naresh and Venkata Krishna, 2021)	To build an optimization-based machine learning algorithm to classify the twitter data.	Sequential Minimal based Optimization Decision Tree (SMODT)	Focus on improving the performance by applying some optimization techniques.
3	(El Rahman et al., 2019)	We did sentiment analysis on two restaurants KFC and McDonalds to find out which one is more popular	Maximum Entropy	To build an algorithm which automatically classify tweets.
4	(Neogi et al., 2021)	To understand the publics' sentiments shared on twitter about the farmers' protest.	Random Forest	1. Use various methods like unsupervised learning. 2. How did the protest affected the covid-19 spread, snice covid cases were increasing during the protest.
5	(Mandloi and Patel, 2020)	To analyze various ML algorithms and find out which one performs better for sentiment analysis	Naïve Bayes Classifier	Need to improve the performance measure.
6	(Bengesi et al., 2023)	To analyze public sentiments on the recent monkeypox outbreak, to get a better understanding of the public perceptions of the disease.	SVM + TextBlob annotation + Lemmatization + CountVectorizer	To explore more algorithms and techniques: 1. doc2Vec. 2. Azure Machine Learning for labelling. 3. Deep learning and transformer algorithms.
7	(Singh et al., 2022)	To propose a machine learning (ML) based method to analyze social media data for sentiment analysis on text data.	Support Vector Machine (SVM)	Experiment with other feature extraction techniques.
8	(Satrya et al., 2022)	To generate positive and negative trends based on cryptocurrency related twitter data.	SVM with Under sampling and a ratio of 80:20 train-test split	-
9	(Mishra et al., 2022)	To discover tweet analysis more efficiently.	Naïve Bayes Classifier	1. Need to work on other regional languages besides English. 2. Analyze complicated emotions like sarcasm and create a hybrid classifier.
10	(Gupta and Joshi, 2021)	To build a feature-based twitter sentiment analysis system with improved negation accounting.	Support Vector Machine (SVM)	1. Explore more about feature engineering and morphological negation handling. 2. Focus on different types of polarity shifters such as intensifiers, conditionals, and diminishers.

2.5 Recent works in TSA based on Deep Learning based Techniques

In various domains, deep learning algorithms provided state of the Art results. So, in recent times, various kinds of deep learning architectures were implemented for sentiment classification problem and provided better performances when compared to traditional ML techniques (Seo et al., 2020). So, this subsection talks about, in recent studies what kind of datasets were considered, what kinds of preprocessing techniques, feature extraction techniques were involved, what kind of deep learning architecture were implemented and finally what are all the future works and research gaps in this area.

2.5.1 Datasets

To begin the deep learning-based sentiment analysis, we need proper data. In most of the studies, publicly available datasets were considered like IMDB dataset, SemEval dataset, Kaggle datasets etc., These datasets were mostly labelled with the sentiment of the text whether it is positive, negative, or neutral. When it comes to twitter sentiment analysis, most of the times the data will be extracted using twitter API and these datasets need to be labelled with their respective sentiments for further analysis. In this study (Aslam et al., 2022) the authors had extracted the twitter data using Twippy library and for labelling the data, a famous python library TextBlob was used, which is a lexicon-based labelling technique. In this study (Vyas et al., 2021) the authors had used a publicly available Kaggle dataset which comprised of covid related tweets. But it was not labelled. So, another lexicon-based technique VADER was used to label the data. In another study (Jayasurya et al., 2022), the intersection of both TextBlob and VADER was considered to label the data, which means, a tweet will be further considered for the sentiment analysis only if it was labelled as same sentiment by both of them. For example, if a tweet was labelled as negative by TextBlob and positive by VADER, it won't be considered for further analysis. So, they had taken four different datasets for this study which had 1,61,241, 2,01,029, 69,716 and 4,12,721 tweets respectively. And after the labelling they left with 1,06,129, 1,18,909, 47,939 and 2,59,329 tweets respectively.

In this study (Aslam et al., 2022) the authors had extracted around 40,000 tweets in total around the period of July and August 2021. Then experimented with different ML models and deep learning models. At the end of the study, they found that, when the training data is decreased the performance of the deep learning models also decreased. So, rather than experimenting with different combinations of deep learning architectures, spending more time on creating good training data is giving better results (Kamiş and Goularas, 2019). The below table 2.11 shows the different datasets considered for twitter sentiment analysis using deep learning algorithms.

Table 2.11. The datasets used in the recent studies on TSA using DL algorithms.

Citation	Dataset	Number of Tweets				Observation Period
		Positive	Negative	Neutral	Total	
(Kumar and Zymbler, 2019)	Major airlines-based tweets were extracted using twitter API	-	-	-	1,20,766	1 March 2019 to 11 March 2019
(Pandya et al., 2021)	Sentiment140 - A labelled dataset publicly available on Kaggle.	8,00,000	8,00,000	-	16,00,000	Not mentioned
(Roy and Ojha, 2020)	A csv file with the tweet id, sentiment, and the tweet. Training dataset - 8,00,000 tweets; Test dataset - 2,00,000 tweets.	4,00,312	3,99,688	-	8,00,000	Not mentioned
(Onan, 2021)	From coursetalk.com, the MOOC (Massive open online courses) reviews were taken.	33,000	33,000	-	66,000	Not mentioned
(Jin et al., 2020)	Dataset 1 - Reviews about Apparel, camera, electronics, housewares, magazines, and sports commodity.	5,999	5,965	-	11,964	Not mentioned
	Dataset 2 - Reviews about Books, daily necessities, entertainment, and media.	11,783	11,959	-	23,742	
(Aslam et al., 2022)	A python library Twippy was used to extract the tweets with keywords like "#cryptocurrency", "#cryptomarket", and "#BTC".	14,541	3,712	21,747	40,000	July 2021 to August 2021
(Hameed and Garcia-Zapirain, 2020)	Movie Review (MR) dataset	5,331	5,331	-	10,662	2005
	ACL Internet Movie Database (IMDB)	25,000	25,000	-	50,000	2011
	Stanford Sentiment Treebank (SST2)	Not mentioned	Not mentioned	-	9,613	2013
(Venkatesh et al., 2021)	Tweets related to FIFA world-cup 2018 was extracted using twitter API.	5000+	1000+	3000+	10,007	Not mentioned
(Lin et al., 2020)	Large Movie Review Dataset (IMDB)	25,000	25,000	-	50,000	2011
	Semeval2017-task4-A English	7000+	3000+	-	10,000+	2017
	Stanford Sentiment Treebank (SST)	5,000	4,500	-	9,500	2013

(Jayasurya et al., 2022)	Tweets related to covid with the hashtag #CovidVaccine were considered in this dataset from Kaggle.	41,048	10,549	39,739	91,336	August 18, 2020, to March 11, 2021
	Tweets with the hashtags #covid vaccine were extracted using twitter API.	44,016	20,611	43,700	1,08,327	February 12, 2020, to October 22, 2020
	Tweets related to covid-19 vaccines were collected using Tweepy library.	14,784	4,514	20,600	39,898	December 12, 2020, to April 14, 2021

2.5.2 Pre-processing Techniques

Once we have the labelled data in hand, the next step is to clean and preprocess the data. To implement the deep learning algorithm in twitter data, the most performed pre-processing techniques were lower case conversion, removal of stop words, special characters, punctuations, URLs, and numeric values. In this study (Kumar and Zymbler, 2019) the authors used a python library regular expression for cleaning and preprocessing the data. In this study (Venkatesh et al., 2021) other than the common preprocessing steps, the authors also worked with the slang words, spelling mistakes, white spaces, and abbreviations.

One of the recent preprocessing steps used by most of the studies is working with the emoticons. It can be either removed or tagged as positive emoticons or negative emoticons based on what it conveys. In this study (Roy and Ojha, 2020) the authors had matched the possible emoticons with either EMO_POS or EMO_NEG based on what the emoticon conveys. They also removed the retweets using regular expressions and performed the common preprocessing techniques mentioned above. In these studies, (Kamiş and Goularas, 2019; Pandya et al., 2021) the authors simply removed the emoticons from the twitter data.

In some of the papers, stemming and lemmatization were used to convert each word of the twitter data into their base form or root form, as it won't affect the sentiment of the word and makes the sentiment classification process easier as it removes the prefixes and suffixes of the word (Aslam et al., 2022). To summarize, what are all the preprocessing techniques used in the deep learning-based recent studies, the table 2.12 shows us the common preprocessing techniques in column header and the deep learning-based recent studies in row header. If a study had performed that technique, a tick mark is given else a cross/wrong mark is given.

Table 2.12. Pre-processing techniques used in recent TSA related studies using DL algorithms.

Citation	Removal Of							Lower Case	Stemming	Lemmatization	Tokenization
	Stop Words	User Mention	Special Characters/ Punctuations	URL	Numeric Values	Emoticons	Hashtags				
(Kumar and Zymbler, 2019)	✓	✓	✓	✓	✓	×	×	✓	✓	×	×
(Pandya et al., 2021)	✓	✓	✓	✓	✓	✓	×	✓	✓	×	×
(Roy and Ojha, 2020)	×	✓	✓	✓	×	✓	✓	✓	×	×	×
(Gandhi et al., 2021)	✓	×	✓	✓	×	×	✓	×	×	×	✓
(Onan, 2021)	✓	×	✓	✓	×	×	×	✓	✓	×	✓
(Aslam et al., 2022)	✓	×	✓	✓	✓	×	✓	✓	✓	✓	✓
(Hameed and Garcia-Zapirain, 2020)	✓	×	✓	×	×	×	×	✓	×	×	×
(Venkatesh et al., 2021)	×	×	✓	✓	✓	×	×	✓	×	×	×
(Kamiş and Goularas, 2019)	×	×	✓	✓	×	✓	×	✓	×	×	×
(Vyas et al., 2021)	✓	×	×	×	×	×	×	✓	×	✓	✓

2.5.3 Deep Learning Algorithms and Techniques

In recent days, as the communication and information technology developed, online/distance education had become a huge part of the education domain and these massive open online courses (MOOCs) were the recent innovation in distance education. So, the reviews of MOOCs can help us to get some real answers for the research questions about sentiment analysis on educational data. Hence, the author had experimented with ML, DL, and ensemble techniques on a corpus of 60,000+ MOOC reviews. And the results had shown that the deep learning models' performances were better than the ML and ensemble techniques. For feature representation, word2vec, fastText, and GloVe word-embedding schemes were used. Finally, LSTM (long short-term memory) network with GloVe representation got the highest accuracy i.e., 95.8% (Onan, 2021).

In this study (Jin et al., 2020) the authors had proposed a new algorithm MTL-MSCNN-LSTM (Multi-task learning model based on multi-scale CNN and LSTM) to address the problem of multi-tasking sentiment classification. The proposed model with the average accuracy of 86.85% outperformed most of the existing state-of-the-art models, this is because "the adversarial multi-task learning approach improves the encoding quality of the encoder". In this study (Lin et al., 2020) the authors had proposed a new algorithm CE-B-MHA (Comparison Enhanced Bi-LSTM with Multi-Head Attention) to improve the performance of the sentiment analysis of textual data. Word2vec was used for the feature representation. The proposed model had performed better than the existing models on the three sentiment analysis datasets the authors had worked with.

In this study (Roy and Ojha, 2020), to deal with the problem of sentiment analysis on twitter data, the authors had experimented with multiple machine learning and deep learning algorithms and then they built an ensemble model with the top five best performing algorithms. The summary of the recent deep learning algorithms used for the sentiment analysis on textual data along with the evaluation metrics and the feature extraction/representation techniques was shown in the table 2.13.

When it comes to deep learning algorithms, CNN and LSTM network were most popular and better performing algorithms. In this study (Venkatesh et al., 2021) the authors had experimented with multiple ML and deep learning algorithms, and it was observed that the hybrid CNN and LSTM network with GloVe features was robust and comprehensive for sentiment analysis. In this study (Kamiş and Goularas, 2019) the Multiple CNNs and Bi-LSTM model with the GloVe features was the best performing model when compared with seven other ML and DL techniques. In another study (Aslam et al., 2022) the authors had observed that the combination of LSTM and GRU had outperformed the state-of-the-art models with the accuracy of 99%. They also observed that the performance of the DL algorithms was decreased when the training data was decreased. Table 2.14 shows the best performing models of the recent studies on sentiment analysis along with the performance measures.

Table 2.13. The feature extraction/representation techniques, DL algorithms and evaluation metrics used in the recent studies on TSA.

S No	Citation	Feature Extraction/ Representation	Algorithms / Models	Evaluation Metrics
1	(Pandya et al., 2021)	Word2Vec	Bi-LSTM had the highest accuracy when compared with Naïve Bayes, SVM, RNN and LSTM i.e., 81.33%	Accuracy, Precision, Recall, F1-Score
2	(Roy and Ojha, 2020)	Unigrams and Bigrams	Ensemble of their best 5 models, 1. LSTM-NN 2. 4-Conv-NN 3. 4-Conv-NN features + SVM 4. 4-Conv-NN with max_length = 20 5. 3-Conv-NN.	Accuracy
3	(Gandhi et al., 2021)	Word2Vec	CNN and LSTM	Accuracy
4	(Onan, 2021)	word2vec with skip-gram and CBOW, fastText, and GloVe.	LSTM with the GloVe features outperformed all the other DL, ML and Ensemble models tried and attained the accuracy of 95.8%	Accuracy, F1-Score
5	(Jin et al., 2020)	Shared Encoders	Proposed a new algorithm MTL-MSCNN-LSTM (Multi-task learning model based on multi-scale CNN and LSTM).	Accuracy, F1-Score
6	(Aslam et al., 2022)	-	Combination of LSTM and GRU.	Accuracy, Precision, Recall, F1-Score
7	(Hameed and Garcia-Zapirain, 2020)	-	Single-layered Bi-LSTM	Accuracy, Precision, Recall, F1-Score
8	(Venkatesh et al., 2021)	GloVe and Sentiment lexicon	Hybrid CNN and LSTM	Accuracy, Precision, Recall, F1-Score
9	(Kamiş and Goularas, 2019)	Word2Vec with CBOW and GloVe	Out of the seven different DL models tried, Multiple CNNs and Bi-LSTM model with the GloVe features had the best performance.	Accuracy, Precision, Recall, F1-Score
10	(Vyas et al., 2021)	n-grams	LSTM attained the best accuracy when compared with the other algorithms i.e., 83%	Accuracy, Precision, Recall, F1-Score
11	(Lin et al., 2020)	Word2Vec	Proposed a new algorithm CE-B-MHA (Comparison Enhanced Bi-LSTM with Multi-Head Attention).	Accuracy, Precision, Recall, F1-Score, AUROC

Table 2.14. Most common evaluation metrics values of best performing models in recent studies on sentiment analysis using DL algorithms.

S No	Citation	Best performing Algorithms/Models		Evaluation Metrics			
				Accuracy	Precision	Recall	F1-Score
1	(Pandya et al., 2021)	Bi-LSTM		81.33	82.13	80.99	81.54
2	(Onan, 2021)	LSTM		95.80	-	-	96.00
3	(Jin et al., 2020)	MTL-MSCNN-LSTM (Multi-task learning model based on multi-scale CNN and LSTM)		86.85	-	-	86.85
4	(Aslam et al., 2022)	Ensemble of LSTM and GRU		99.00	99.00	98.00	98.00
5	(Hameed and Garcia-Zapirain, 2020)	Single-layered Bi-LSTM	MR dataset	80.50	80.51	80.50	80.49
			IMDB dataset	90.58	90.58	90.58	90.58
			SST2 dataset	85.78	85.85	85.78	85.77
6	(Venkatesh et al., 2021)	Hybrid CNN and LSTM		85.00	82.00	78.30	80.00
7	(Kamiş and Goularas, 2019)	Multiple CNN's and Bi-LSTM		59.00	60.00	55.00	56.00
8	(Vyas et al., 2021)	LSTM		83.00	83.00	82.00	82.00
9	(Lin et al., 2020)	CE-B-MHA (Comparison Enhanced Bi-LSTM with Multi-Head Attention)	MR dataset	89.50	92.90	-	-
			SemEval2017	80.40	70.30	-	-
			SST dataset	73.50	73.20	-	-

2.5.4 Future Works and Challenges

In this study (Pandya et al., 2021) the best performing model was Bi-LSTM with the accuracy of 81.33%, so the performance of the model can be improved by including more comprehensive labelling like highly negative, moderately negative, or positive, etc. In this study (Gandhi et al., 2021) an IMDB dataset on Kaggle was taken and the authors had experimented with CNN and LSTM networks which attained an accuracy of 87% and 88% respectively. The performance can be further improved by using huge corpora of twitter data to better understand the implicit and explicit aspect level of DL methods.

In this study (Kumar and Zymbler, 2019) the authors presented a machine learning approach to analyse the tweets about the passengers' opinions of their flight travels and try to improve the customers' experience. The authors experimented with SVM, ANNs, and CNN models and found that CNN provided a better performance when compared to other techniques. But the experiment was made only on the English language tweets. So, to work with different languages and get more useful feedbacks was the major future work.

To better understand the peoples' thoughts about the cryptocurrency market, the authors tried to build an ensemble model of LSTM and GRU to analyse the sentiment of the tweets posted on twitter about the cryptocurrency. The results shows that a larger number of people feel happy about cryptocurrency followed by fear and surprise emotions. For future work, with the help of analysed information in hand, price prediction on cryptocurrency can be put to test (Aslam et al., 2022).

In this study (Roy and Ojha, 2020) some of the major future works were handling emotion ranges, including symbols like commas, exclamation marks etc., which may help to predict the sentiment better, to explore richer linguistic analyses like parsing, semantic analysis, or subject modelling. In this study (Jin et al., 2020) the authors had addressed the problem of multi-tasking sentiment classification. And for future work, doing the multi-class sentiment analysis for other NLP related tasks was considered. To summarize the deep learning related studies future works, table 2.15 shows the objectives, the deep learning algorithm used and the future works of the recent studies on sentiment analysis.

Table 2.15. Objectives, model used and the future challenges of the recent studies on TSA using DL models.

S No	Citation	Objective	Algorithms / Models	Future Works/Challenges
1	(Kumar and Zymbler, 2019)	To improve the customer's experience, presenting a machine learning approach and analyze the tweets.	CNN	To work on different language tweets other than English, so that the airline industry can get better and more insights from their customers' opinions.
2	(Pandya et al., 2021)	To help the organizations to get better insights from their customers' opinion and take effective decisions, presenting a sentiment analysis technique on tweeter data.	Bi-LSTM	The performance of the models can be improved. 1. By using a larger dataset. 2. By including more comprehensive labelling like highly negative, moderately negative, positive, etc.
3	(Roy and Ojha, 2020)	To deal with the problem of sentiment analysis on twitter data.	Ensemble of their best 5 models, 1. LSTM-NN 2. 4-Conv-NN 3. 4-Conv-NN features + SVM 4. 4-Conv-NN with max_length = 20 5. 3-Conv-NN.	1. Handling emotion ranges. 2. including symbols. 3. parsing, 4. semantic analysis, and 5. subject modelling.
4	(Gandhi et al., 2021)	To correctly interpret the context of the tweet words whether it is truly positive or negative.	CNN and LSTM	1. To work on CNN model with different techniques and more complex features. 2. Then use huge corpora of tweeter data to better understand the implicit and explicit aspect level of DL methods.

5	(Jin et al., 2020)	To address the problem of multi-tasking sentiment classification.	MTL-MSCNN-LSTM (Multi-task learning model based on multi-scale CNN and LSTM)	1. To improve the encoder on the existing basis, so that we can further improve the performance of sentiment classification for multi-task learning. 2. To perform multi-class sentiment analysis for other NLP related tasks.
6	(Aslam et al., 2022)	To get a holistic view of peoples' sentiments on cryptocurrency.	Ensemble of LSTM and GRU	To perform price prediction on cryptocurrency with the analyzed insights of peoples' sentiments.
7	(Hameed and Garcia-Zapirain, 2020)	To find the polarity of peoples' opinions, emotions and reviews presenting a computationally efficient deep learning model.	Single-layered Bi-LSTM	1. multi-class sentiment classification along with the multilingual approach 2. Bi-GRU can be applied. 3. To apply similar simpler approaches in other NLP related tasks such as speech recognition and machine translation.
8	(Vyas et al., 2021)	To develop an automated framework to analyze and classify the tweets as positive, negative, and neutral sentiment using ML techniques.	LSTM	"To uncover deliberate efforts to steer societal sentiments into prescribed directions".

2.6 Summary

In this section, previous works done in sentiment analysis were explored. We began with an introduction of twitter sentiment analysis and how the previous studies have approached this problem were discussed. Then the major study on twitter sentiment analysis in recent days i.e., covid-related tweets sentiments were explored. And what are all the different pre-processing steps, feature extraction techniques, different algorithms used for model building and evaluation measures used were discussed. Finally, the recent works on sentiment analysis of textual data using lexicon-based methods, machine learning and deep learning-based methods were explored. But in our study, only lexicon-based methods and machine learning based algorithms were analysed and experimented with. Because the dataset we have in hand was not large enough to be experimented with deep learning models. The figure 2.2 summarizes the major techniques used for all kinds of processes included in twitter sentiment analysis according to the previous studies on TSA.

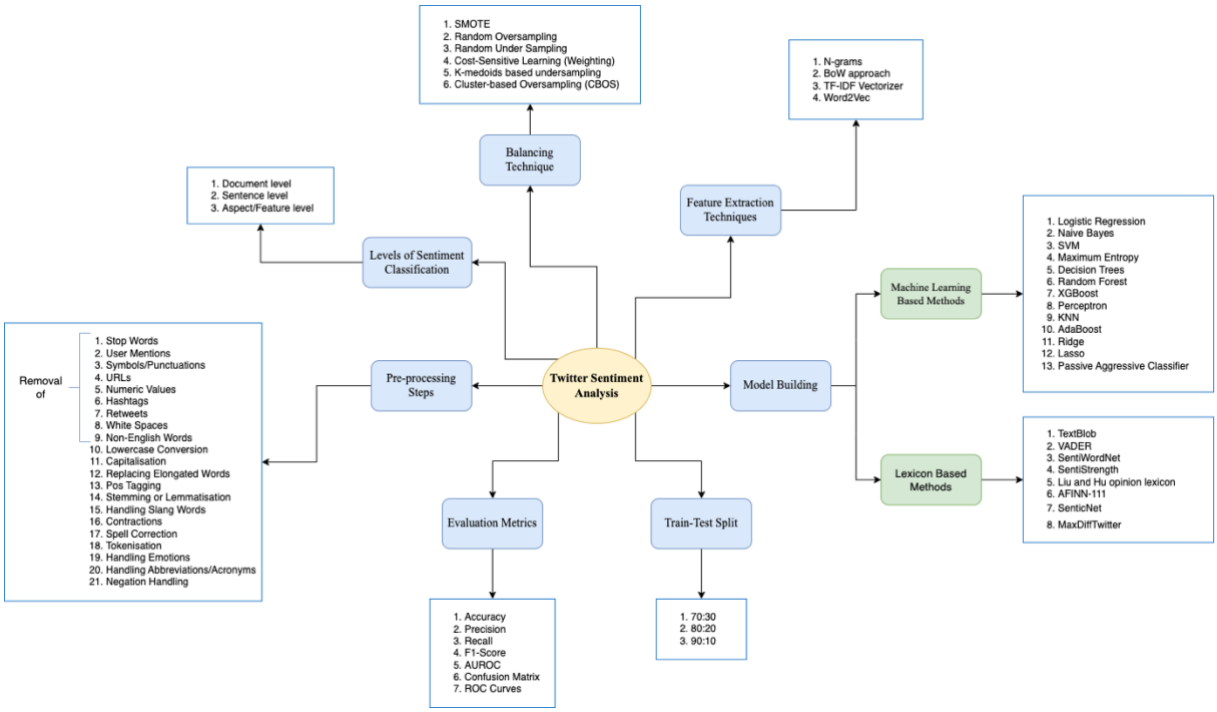


Figure 2.2. Major techniques used in twitter sentiment analysis using ML and lexicon-based approach.

CHAPTER 3

PROPOSED METHODOLOGY

In this section, we are going to discuss about the process we used to find the sentient of the tweets in detail. Firstly, about the data we have in hand which is gathered from Twitter. Then the techniques used to clean and pre-process the data and after that how it is feature engineered to get in the understandable shape of machine language. Further the lexicon-based and machine learning based techniques used to classify the data into positive, negative, and neutral was discussed. At last, the evaluation metrics used to evaluate the performance of the algorithms used was given.

3.1 Introduction

In this section, the entire framework for the sentiment analysis of twitter data using ML and lexicon-based techniques was discussed. To experiment with ML and Lexicon-based techniques, an open-source high-level programming language Python was used, which provides multiple rich libraries viz. NumPy, Pandas, Sci-kit-learn, Seaborn, Matplotlib, NLTK, RegEx etc., (Hota et al., 2021). We have three levels of sentiment analysis i.e., document level, sentence level, and aspect level analysis. In this study, we are going to use sentence level analysis since the main objective of this study is to find the sentiment of the raw tweets. The various phases of the proposed framework as shown in figure 3.1 are discussed in detail as follows.

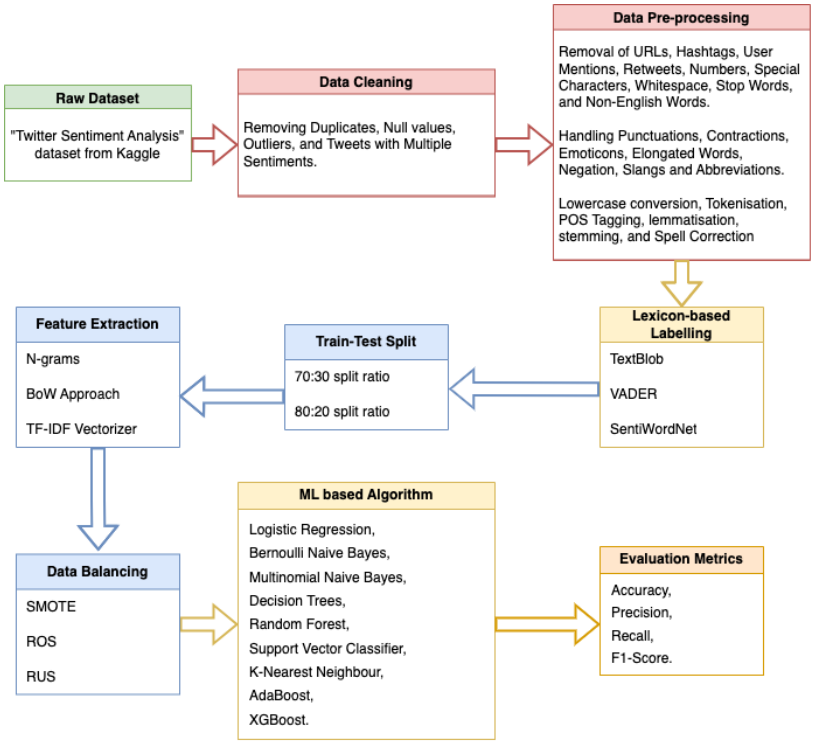


Figure 3.1. Architecture of the proposed methodology

3.2 Data Description

There are many social media platforms like Facebook, Instagram, Twitter etc., which are used to express peoples' opinions, emotions, and feedbacks about a product/issue/service. But when compared to others, twitter data can be easily extracted through twitter API and tweets were restricted to 280 characters which helps us to get straightforward and precise information about any specific events, latest news, or trending topics. The dataset used in this study is taken from

Kaggle, an online community for data scientists and machine learning practitioners. The "Twitter Sentiment Analysis" dataset considered for this study was an entity-level sentiment analysis dataset of twitter. The data was extracted using twitter API and stored in a comma-separated value file (CSV) with their respective sentiments. Link to the source file of the dataset is given as reference (Twitter Sentiment Analysis | Kaggle, 2023).

The labels used in this study were positive, negative, and neutral. But the dataset had one more label called Irrelevant i.e., the tweets which were not relevant to the given entity. Since we are focusing on sentiment analysis in general, the data labelled with Irrelevant was removed. The dataset consists of two different csv files, "twitter_training.csv" file to train the model and "twitter_validation.csv" file to validate the performance of the model built. Both the csv files had four attributes viz. tweet id, entity, sentiment, and tweet content. Random samples from the training dataset were shown in the table 3.1. Tweet content is the tweet posted by a user, which has textual data with Hashtags, URLs, User mentions, Emoticons, etc., Also, the textual data will have misspelled words or text slang. So before applying any algorithms the data need to be cleaned and pre-processed.

Before getting a glace of the dataset statistics, we had gone through some simple data cleaning steps i.e., dropped unnecessary columns (tweet id, entity) for this study, removal of duplicate values and null values. As we can see in the table 3.2, the training dataset consists of 19,138 positive sentiments, 21,237 negative sentiments, and 17,110 neutral sentiments with a total of 57.5k tweets approximately. And the test dataset consists of 276 positive sentiments, 266 negative sentiments, and 285 neutral sentiments with a total of 1k tweets approximately. We sought to deal with the issue of slight data imbalance, so we had used the balancing techniques mentioned in the following section 3.3.

Table 3.1. Random instances of the training dataset.

Tweet Content	Sentiment
that was the first borderlands session in a long time where i actually had a really satisfying combat experience. i got some really good kills	Positive
Nice job @HomeDepot pic.twitter.com/sZUumDI571	Positive
The Google video conferencing platform "meet" is by far the worst	Negative
What an idiot lmaooo	Negative
@ ShopifySupport only a few sales. Every couple of months I make sales even with google and facebook ada Sorry for the late response.	Neutral
@ EAMaddenNFL @ EA I'm glad we're getting some fixes for the game, especially the franchise.	Neutral

Table 3.2. Class distribution of the training and test datasets after removing duplicates and null values.

Set	Positive Class	Negative Class	Neutral Class	Total Count
Training Set	19138	21237	17110	57485
Test Set	276	266	285	827
Total	19414	21503	17395	58312

3.3 Techniques to handle unbalanced dataset

This section briefly explains the commonly used resampling techniques to handle the unbalanced dataset i.e., Synthetic Minority Oversampling Technique (SMOTE), Random Under-Sampling (RUS), and Random Over-Sampling (ROS). The respective techniques will be presented in the following subsections. Figure 3.2 shows an unbalanced data used as a base for figure 3.3, the class 0 elements are represented as a circle and the class 1 elements are represented as 'x' (Pereira and Saraiva, 2020).

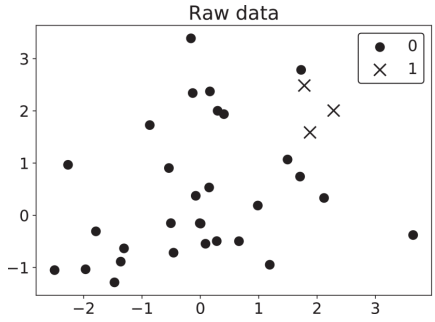


Figure 3.2. Unbalanced data (Pereira and Saraiva, 2020).

3.3.1 Synthetic Minority Oversampling Technique (SMOTE)

SMOTE is used to solve the imbalanced dataset problem by oversamples the minority class so that the number of minority class samples becomes equal or almost equal to the majority class samples (Mujahid et al., 2021). In order to eliminate the overfitting issues created by the traditional oversampling methods, the SMOTE method uses a unique way to select the instances that requires sampling (Mrozek et al., 2020).

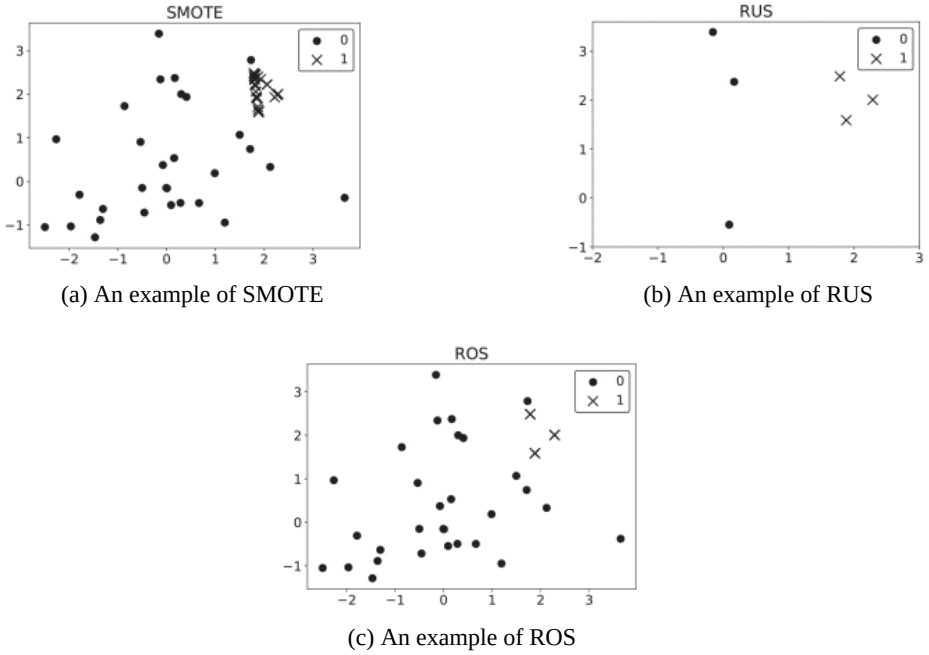
Artificial minority samples were generated from the interpolation using the existing minority training samples and the k nearest minority neighbours. On the contrary to ROS, SMOTE reduces the duplicate instances count by using interpolation (Pereira and Saraiva, 2020). This technique is more efficient when we have a lower dimensional space because, the converters will be closer to each other and hence the accuracy of the resampling will be even better (Mrozek et al., 2020). Figure 3.3 - (a) shows the SMOTE method applied in the dataset presented in the figure 3.2.

3.3.2 Random Under-Sampling (RUS)

In some studies, undersampling is preferred over oversampling because there is a chance for data leakage when using oversampling (Aslam et al., 2022). Random Under-Sampling is a method that randomly selects the majority class samples and removes it, so as to equalize the number of samples of majority class and minority class (Pereira and Saraiva, 2021). This method is proven to be efficient in solving overfitting issue, as it increases the recognition rate of minority class. Undersampling is more appropriate when there is a sufficient number of samples in the minority class. But the major drawback of this method is some important or crucial information can be removed from the majority class in the process of undersampling (Mrozek et al., 2020). From this point of the paper, Random Under-Sampling will be referred as just 'RUS'. Figure 3.3 - (b) shows the application of RUS in the dataset presented in the figure 3.2.

3.3.3 Random Over-Sampling (ROS)

In ROS, the minority class samples are replicated randomly so that the number of samples in minority and majority class are equal or almost equal (Pereira and Saraiva, 2021). Random Over-Sampling works on the contrary to RUS, as it oversamples the minority class samples instead of removing the majority class samples. However, oversampling can cause overfitting because of the repetition of the same elements in the training dataset (Pereira and Saraiva, 2020). From this point of the paper, Random Over-Sampling will be referred as just 'ROS'. Figure 3.3 - (c) shows the application of ROS in the dataset presented in the figure 3.2.



(a) An example of SMOTE

(b) An example of RUS

(c) An example of ROS

Figure 3.3. Unbalanced data handling techniques (Pereira and Saraiva, 2020).

3.4 Data Cleaning

The dataset used for this study was a bunch of raw tweets hence there is a high chance that the dataset might consists of noisy or irrelevant data. So, to ensure that the information is accurate and usable, we need to clean it up. Data cleaning ensures removing such irrelevant or inappropriate data and creating more trustworthy and stable dataset for further analysis (Dagar et al., 2021). Some of the common data cleaning steps were,

Removal of duplicate values: The unnecessary duplicate values need to be removed from the dataset using the python library pandas.

Handling null values: Due to technical errors or unavoidable circumstances, we might get some null values in our dataset. These data add no values for further analysis; hence they were removed using the same python library pandas.

Handling outliers: Most of the time, the dataset contain multiple noisy or undesirable data points called outliers, which may lead to an unintended outcome (Mishra et al., 2022). So, these values need to be identified and removed using data visualization techniques like matplotlib or seaborn library from python.

3.5 Data Pre-processing

In sentiment analysis, pre-processing is considered as a mandatory and vital task which had to be performed before using any classifiers (Hameed and Garcia-Zapirain, 2020). Raw tweets extracted from twitter will definitely be noisy and inconsistent because of the casual nature of peoples' usage of social media (Roy and Ojha, 2020). And generally, tweets have a special characteristic such as hashtags, user mentions, re-tweets, emoticons etc., (Naresh and Venkata Krishna, 2021). This unstructured nature of the twitter data makes the pre-processing task challenging (Hota et al., 2021). Hence to handle all these redundancies and anomalies, we had used the following pre-processing techniques which was implemented using the Natural Language Toolkit (NLTK) and regular expressions (RegEx) libraries of python (Hameed and Garcia-Zapirain, 2020). We briefly explained each of the techniques with an example and also in which order it has to be applied.

Lowercasing: The primary goal of converting all the words in the corpus to lowercase is because the word such as "Good" and "good" are distinct for a machine (Pradha et al., 2019). Hence, it is beneficial to convert entire corpus into lowercase or uppercase. By doing so, the dimensionality will be reduced since the same words are merged (Symeonidis et al., 2018). The lowercasing was done using the built-in method "lower()" of python.

Removing URLs: Users often share hyperlinks of other web resources in their tweets, but these links play no role in sentiment determination of that tweet. Although, keeping it till the classification may cause some serious false predictions. For example, a tweet "I logged in to www.happyworld.com today" should be defined as neutral statement but if we didn't handle the URL properly, it may be considered as a positive statement (Arpita et al., 2020). So, we removed all the URLs from the corpus using the python's regular expression package.

Handling Hashtags, User mentions, Retweets and HTML tags: In twitter, majority of tweets will contain hashtags (#), User mentions (@), and/or Retweets (RT). Hashtags are unspaced phrases begins with a hash symbol (#) which are helpful to search particular posts and mention a trending topic in their tweets. Every user of twitter will have a handle associated with them. So, a user can mention other users in his/her tweets using @handle (Roy and Ojha, 2020). Retweets are tweets which are reshared again by someone else in twitter. These elements are very useful and powerful tool in twitter but for sentiment analysis using machine learning models, they are unnecessary and just increases the dimensionality which reduces the performance of the classifiers (Arpita et al., 2020; Bengesi et al., 2023). So, these elements were removed from our corpus.

Handling Contractions: Contractions are a made-up word which are shortened and combined using two different words. For example, "don't", "shouldn't", and "I've" are contractions of "do not", "should not", and "I have" respectively. Hence, these contractions need to be replaced with their source words else the meaning of the words might be interpreted wrongly by the classifiers. For example, the word "don't" will become "don" and "t" after tokenization, and the word "t" probably considered as a noise and "don" might be considered as a root form of word "done". But this interpretation was totally incorrect. Hence, the contractions need to be handled before tokenization (Symeonidis et al., 2018).

Handling Punctuations: Punctuations are a set of symbols used in a sentence to increase the text's clarity and understandability. Some of the commonly used punctuations are full stop/period (.), comma (,), exclamation mark (!), question mark (?), semi colon (;) and so on (Aslam et al., 2022). All the punctuation marks can be removed using the regular expressions since it will help in dimensionality reduction and improve the learning process (Bengesi et al., 2023).

Removing Special Characters: Special characters like (~$#@%) add no values for the classification process. Hence, they were removed using the regular expression.

Handling Emoticons: Facial expressions of a person explains best about a person's feeling likewise, emojis in text explains best about the people's opinions shared on social media (Arpita et al., 2020). However, sometimes people use these emojis to express a sarcastic comment. For example: A tweet says "The first day of gym was soo awesome ;-)" was very hard to classify as positive or negative sentiment because of the sarcastic nature of the text and the emoji used. So, the process of emoji replacement becomes even more difficult. Hence for this study, we are going to remove emoticons from our dataset (Alam and Yao, 2019). In future, we will work on the emoticons and the sarcastic comments.

Handling Whitespace: Firstly, the white space at the end and the beginning of the sentence were removed. Then in between, two or more spaces were replaced with single space (Roy and Ojha, 2020). This work was achieved using regular expressions.

Replacing Elongated Words: Elongated words are words which contains a character that is repeated more than twice wrongly. These words need to be replaced with their respective correct word. Otherwise, the classifier will treat that word has a separate unique word and may be removed in the process of non-English words removal (Symeonidis et al., 2018). For example, a tweet "That dish was sooooo awesomeeee!" will be converted into "That dish was so awesome!".

Negation handling: As a first step, we search for a "not" word in a sentence, then check if the very next word has an antonym. If it exists, the opposite word will be replaced with it (Symeonidis et al., 2018). For example: "not right" will be replaced with "wrong".

Replacing Slang Words and Abbreviations: Social media texts are naturally written in an informal way which includes slang words and abbreviations. Slang words are words that are constructed in an informal way that just matches the sounding of the source word. And abbreviations are a shortened form of the source words (Symeonidis et al., 2018). For example, the phrases "B2B", "4ever", "gr8", and "omg" really means "back-to-back", "forever", "great", and "oh my god". Hence, these slang words and abbreviations are replaced to their source words using the word list created.

Removing Numbers: Numerical data also doesn't add useful information for models' training. Hence removing the numbers from our dataset using regular expressions, which also helps in feature space reduction (Aslam et al., 2022). This step must be implemented only after handling slang words, because some slang words contain numbers in it viz. 2day, 4ever, r8, gr8 etc., (Symeonidis et al., 2018).

Stop Words Removal: Stop words are more frequently used words in the English language which doesn't contribute much for the sentiment detection. All articles, propositions, pronouns, and conjunctions are examples of stop words. Some of them are "me", "my", "an", "the", "are", "is", "he", "at", and so on (Ahuja et al., 2019; Hota et al., 2021). In our study, the stop words provided by NLTK were removed.

Tokenization: Tokenization is the process of NLP that breaks down a given tweet sentence into words (tokens) (Vyas et al., 2021). A token can be a word, a symbol, a number, or a phrase. For example, a given tweet sentence "I graduated with distinction!" can be tokenized as {'I', 'graduated', 'with', 'distinction', '!'}. Also, tokenization will simplify the feature extraction process of sentiment analysis.

POS tagging: Each word of the corpus was assigned with a parts-of-speech tag such as noun, adjective, preposition and so on. The main purpose of POS tagging is to retain the words which are significant for the classification prediction. Hence the words with noun, verb, adverb, and adjective POS tags were only retained because they are good indicator of subjectivity and sentiment analysis (Mandloi and Patel, 2020). POS tagging of each word of the corpus was implemented using NLTK pos_tags.

Stemming: Stemming is an important pre-processing step which converts a given text into its root form. By doing so, all the different forms of a root word will be eliminated which results in dimensionality reduction (Symeonidis et al., 2018). For example: "run", "running", "runner" to "run"; "fishing", "fisher", "fish" to "fish". But the main disadvantage of stemming is it just removes the affixes so there is a possibility that, the root word can be an incomplete or incorrect word (Aslam et al., 2022). In this study, Porter stemming was used for stemming implementation.

Lemmatization: Since there is no standard format for tweets, a word can be written in multiple forms. For example: "catching" and "caught" are different forms of the word "catch". By using lemmatization, we convert all those different words into its base word (i.e., lemma) from the vocabulary (Bengesi et al., 2023). Lemmatization uses parts-of-speech tag to determine the lemma of the given word (Hota et al., 2021).

Lemmatization evaluates the context of the given word first while stemming just removes the affixes of the given word. For example, the word "studies" lemmatized as "study" but stemmed as "studi" (Aslam et al., 2022). Both these techniques are mutually exclusive so only one of them can be used. We are going to inspect the performance of both the techniques separately and carry on with the best efficient technique.

Handling Non-English Words: Even though twitter allows multiple languages, this study is only focused on English language. Hence the Non-English words were removed using the 'word' corpus of NLTK (Mandloi and Patel, 2020). This step must be implemented only after handling slang words, contractions, punctuations, and special characters because English words with these elements may be considered as a non-English word by the machine. For example: "t-shirt", "don't", and "w8" are English words expressed with the special characters or slang words, but the machines will understand these words only if it was given as "tshirt", "do not", and "weight" else they will be considered as a non-English word and probably removed. Hence before removing non-English words, we need to implement the pre-processing steps mentioned above.

Spelling Correction: Again, it is very common to make spelling mistakes in social media texts which need to be replaced with the correct spelling. The python package TextBlob was used for spelling correction in our study.

The pre-processing steps mentioned above will help us to reconstruct the raw twitter data into a meaningful form. Most of the pre-processing steps used in this study were referred from (Symeonidis et al., 2018). Table 3.3 shows the pre-processing techniques used in this study along with the order it has to be applied.

Table 3.3. Pre-processing techniques used with the order of execution.

Number	Pre-processing technique
1	Lowercasing
2	Replacing URLs
3	Handling Hashtags, User mentions, Retweets and HTML tags
4	Handling Contractions
5	Handling Punctuations
6	Removing Special Characters
7	Handling Emoticons
8	Handling Whitespace
9	Replacing Elongated Words
10	Negation handling
11	Replacing Slang Words and Abbreviations
12	Removing Numbers
13	Stop Words Removal
14	Tokenization
15	POS tagging
16	Stemming
17	Lemmatization
18	Handling Non-English Words
19	Spelling Correction

3.6 Feature Engineering

In this phase, the pre-processed textual data were converted into word vectors since the machine learning classifiers take only numerical data for classification. So, the textual data were converted into feature vectors before it was inserted into the classifiers (Jayasurya et al., 2022). In text mining and sentiment analysis tasks, most commonly used feature extraction techniques were bag-of-words approach (BoW), term frequency - inverse document frequency (TF-IDF) and N-grams. In this study, all these techniques were implemented for the feature extraction process which are explained in detail below:

3.6.1 N-grams

In text mining tasks, N-grams plays an important role as a representation scheme. A sequence of ‘n’ consecutive words from the given text is called n-grams. The common N-gram models used for text mining are Unigram (n=1), Bigram (n=2), and Trigram (n=3) (Kumar and Rajini, 2019; Onan, 2021).

Unigrams: Unigram is the simplest and commonly used feature of classification problems, which is the occurrence of the single words in the texts (Roy and Ojha, 2020).

Bigrams: Bigrams are the pairs of two consecutive words in the given sentence. For example, a sentence “The movie was not great” will produce the following bigrams “The movie”, “movie was”, “was not”, and “not great” (Ahuja et al., 2019).

Trigrams: Pairs of three consecutive words in the given text are called trigrams. For example, let’s take a same sentence “The movie was not great”, which will produce three trigrams “The movie was”, “movie was not”, and “was not great”.

3.6.2 Bag of Words Approach

The Bag of Words approach popularly known as BoW, is a simple and most commonly used feature extraction technique in NLP and information retrieval (Rustam et al., 2021). For text classification, BoW approach involves two steps. The first step is to create a vocabulary of all the distinct words in the entire dataset. The second step is to create a matrix of all the distinct words (column) and count their occurrences in each of the tweets (row) in the dataset (Neogi et al., 2021). For example: two sample tweets were given below, and their respective BoW feature vectors are shown in the table 3.4 (Mujahid et al., 2021).

Sample tweet 1, T1: Barcelona won against Arsenal.
Sample tweet 2, T2: Messi congratulated Barcelona for winning against Arsenal.

Table 3.4. BoW feature vectors for sample tweets given.

T	Barcelona	won	against	Arsenal	Messi	Congratulated	for	winning	Length
1	1	1	1	1	0	0	0	0	**4**
2	1	0	1	1	1	1	1	1	**7**

3.6.3 TF-IDF (Term Frequency-Inverse Document Frequency)

TF-IDF is a feature extraction technique regularly used for natural language processing tasks due to its clarity and efficiency (Singh et al., 2022). Some studies had mentioned that TF-IDF features outperform BoW approach in the sentiment analysis task because it returns the weighted features from the text data which increases the performance of the classifiers (Neogi et al., 2021; Rustam et al., 2021). TF-IDF finds the weight of each word in a corpus by multiplying the Term Frequency (TF) with the Inverse Document Frequency (IDF).

As shown in formula 3.1, TF is number of times a term t occurred in the document d divided by the total length of the document.

$$TF_{t,d} = \text{count}_{t,d} / \text{total count}_{d} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.1)$$

As shown in formula 3.2, IDF is number of documents N divided by number of documents which has the term t.

$$IDF_t = \log (N / D_t) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots..(3.2)$$

So, highly recurrent words which lacks the informational value like stop words, will get a low IDF value (Rustam et al., 2021). Finally, TF-IDF is defined as $TF_{t,d} * IDF_t$. For example, Taken the same sample tweets used in the above section 'BoW' and checked how TF-IDF vectorizer handled them. The results were shown in the table 3.5.

Table 3.5. TF-IDF feature on the sample tweets given.

Terms	TF of T1	TF of T2	IDF	TF-IDF of T1	TF-IDF of T2
Barcelona	1/4	1/6	0	0	0
won	1/4	0	0.3010	0.075	0
against	1/4	1/6	0	0	0
Arsenal	1/4	1/6	0	0	0
Messi	0	1/6	0.3010	0	0.050
Congratulated	0	1/6	0.3010	0	0.050
for	0	1/6	0.3010	0	0.050
winning	0	1/6	0.3010	0	0.050

3.7 Proposed Methodology

This section discusses about the hybrid approach of ML and Lexicon-based techniques used in our study for sentiment analysis on twitter data. Basically, we try to find the sentiment/polarity of the given text (in our case, tweet) whether it's positive, negative, or neutral. Lexicon-based techniques finds the polarity of the given document based on the polarity of the words present in the document. On the other hand, ML techniques find the polarity of the given text based on the feature values. The efficiency of the ML models increases based on the quality of the extracted features' ability to classify the text (Braig et al., 2023). The techniques used in both the approaches were explained in detail below.

3.7.1 Lexicon based approach

Generally, lexicon analysis is based on some existing dictionaries which aims to predict the polarity of the given document with the help of semantic orientation of the words/phrases present in the document (He, 2022). Based on the captured semantic orientation, the sentences (in our case, tweets) can be labelled as positive, negative, or neutral sentiment. The main advantage of using lexicon-based methods are labelled training data is not required, it is much simpler to understand and can be adapted to different languages easily (Neogi et al., 2021; Braig et al., 2023). In our study, we had used three popular lexicon-based approaches i.e., TextBlob, VADER, and SentiWordNet.

3.7.1.1 TextBlob

Textblob is a famous python package which is used in multiple NLP tasks such as text mining, sentiment analysis, part-of-speech tagging, translation etc., (Kumaresh et al., 2019; Rustam et al., 2021). It is extremely popular because of its simple approach and easy methods. This study uses TextBlob for sentiment classification of twitter data. First, it takes the pre-processed data as input and split it into individual sentences. Then based on the count of positive and negative sentences, the entire document's polarity is determined (Sharma and Rohini, 2022).

TextBlob will return subjectivity and polarity for the given input text. Subjectivity will check whether the given text is objective or subjective. On the other hand, polarity will check whether the given text is positive, negative, or neutral. The subjectivity scores will range from 0 to 1 where 0 is most objective and 1 is most subjective. On the other hand, polarity scores will range from -1 to 1 which is used to label the data (Aslam et al., 2022; Bengesi et al., 2023). The given text will be labelled as

- Positive if, 0 < score <= 1
- Negative if , -1 <= score < 0
- Neutral if, score == 0.

3.7.1.2 VADER

VADER (Valence Aware Dictionary and Sentiment Reasoner) is a lexicon and rule-based tool which is specialized in sentiment analysis of social media texts (Al-Shabi and Al-Shabi, 2020). This is because VADER not only predicts the given text as positive or negative but also shows the depth of the polarity. It is fast enough to be used online and it doesn't get affected by the speed-performance trade-off (Kumaresh et al., 2019).

VADER finds the polarity of the given text using the polarity_score() function which will return four metrics viz. positive, negative, neutral and compound scores for the given text. Compound score is the one used for labelling the data (in our case, tweets) which calculates the normalized sum of positive, negative, and neutral words. The compound score falls within the range of -1

and 1 where -1 is extreme negative and 1 is extreme positive (Bengesi et al., 2023). The given text will be labelled as

- Positive if, score >= 0.05
- Negative if , score <= -0.05
- Neutral if, -0.05 < score < 0.05.

3.7.1.3 SentiWordNet

SentiWordNet is a lexicon resource available in NLTK (Natural Language ToolKit) which provides multiple text processing libraries viz. tokenization, stop words, stemming, POS tagging, semantic reasoning etc., (Sharma and Rohini, 2022). In SentiWordNet, the sentiment score was calculated using the polarity of the each WordNet synset. Synset is a set of one or more synonyms which share a common meaning. Each pre-processed words given as input for SentiWordNet, have multiple synsets. And each synset includes three scores i.e., positive, negative, and neutral. The total average of the word's synsets scores was considered as the final score of the word. The average is nothing but, the total positive and total negative word synsets scores were calculated and then the total negative score was subtracted from the total positive score (Al-Shabi and Al-Shabi, 2020; Remali et al., 2022).

3.7.2 Machine Learning based Approach

In recent days, ML models gained more popularity for sentiment classification. Sentiment classification is a process of classifying a text/opinion based on the polarities of opinions whether it is positive, negative, or neutral. There are various supervised machine learning algorithms which supports classification of sentiments (Alam and Yao, 2019; Kumar and Rajini, 2019). After some analysis of previous works done in this area, a total of nine different ML algorithms were implemented in our study i.e., Logistic Regression, Bernoulli Naïve Bayes, Multinomial Naïve Bayes, Support Vector Machine, Decision Trees, Random Forest, K-Nearest Neighbour, AdaBoost and XGBoost. Below is the discussion of the nine different ML algorithms implemented.

3.7.2.1 Logistic Regression (LR)

Logistic Regression is a supervised machine learning model which performs well on binary outcomes but also good in multi-class classification (Mujahid et al., 2021). LR classifier is more suitable for categorical outcomes such as spam detection, sentiment prediction, banking fraud detection etc., (Khanam and Sharma, 2021). In this model, the probabilities describing the outcomes of the input data are modelled using a logistic function which transform the outcome into binary values (Samuel et al., 2020). The output produces a S-shaped curve also known as sigmoid (Dagar et al., 2021). LR assumes that each datapoints are independent from each other. Logistic Regression belongs to a class of generalized linear models and it is also known as Maximum Entropy (Symeonidis et al., 2018).

3. 7.2.2 Naïve Bayes

Naïve Bayes is a simple and effective machine learning technique for text classification which includes high-dimensional training data (Mandloi and Patel, 2020). It works well on both binary and multiclass classification tasks such as document classification, spam detection, sentiment classification etc., (Mishra et al., 2022). This algorithm gives great results on a dataset with millions of data or drastically on a smaller datasets as well (Ahuja et al., 2019).

It is a probabilistic classifier which uses conditional probability to find the likelihood class of the given input and also assumes that the features are independent from each other (Bengesi et al., 2023). It uses the Bayes Theorem mentioned in the formula 3.3 for the classification problems.

$$P(A/B) = (P(B/A) * P(A)) / P(B) \quad \text{…….(3.3)}$$

Here, P(A/B) is posterior probability, P(B/A) is likelihood probability and P(A) & P(B) are prior probabilities (Neogi et al., 2021). In this study, we implemented the two types of naïve bayes models viz. Multinomial Naïve bayes and Bernoulli Naïve Bayes. We used sci-kit-learn library to implement these algorithms.

- Multinomial Naïve bayes: Binary representation of the features.
- Bernoulli Naïve Bayes: Different from Boolean Naïve Bayes based on the consideration of the terms, it takes account of the terms that did not appeared in the sentence (Symeonidis et al., 2018).

3.7.2.3 Support Vector Machine (SVM)

SVM, a supervised machine learning method is considered as one of the most popular ML methods for classification problem (Symeonidis et al., 2018). It is a linear model used for both classification and regression problems such as text classification, object recognition, handwritten digits identification etc., (Alam and Yao, 2019). SVM tries to find the best hyperplane that can differentiate between the classes. The best separation is achieved when the difference between the classes is maximized (Braig et al., 2023).

However, it won't work for non-linear data. Hence kernels are introduced, which will transform the data into the needed form i.e., Linear kernel, Polynomial kernel, Gaussian kernel, and Radial Basis Function kernel (Pandya et al., 2021). The effect of SVM increases as the dimensional space increases. But the main drawback of SVM is it doesn't perform well on huge datasets and also takes long training time (Ahuja et al., 2019).

3.7.2.4 Decision Trees (DT)

Decision Trees are a Tree-structured classifier model which splits the record according to the series of feature values until it reaches the root node. The DT's performance mainly depends on how well the tree's built upon the training data (Rustam et al., 2021). Decision trees consists of root node, leaf node, internal node, and branches. Root node is the extracted features we give into the model. Internal nodes represent the attributes of the dataset. Branches are the rules used to make the split. Finally, the leaf node is the result i.e., the sentiment of the tweet (positive, negative, or neutral) (Neogi et al., 2021). This technique can be used for both classification and regression.

3.7.2.5 Random Forest (RF)

Random Forest is an ensemble technique used for both regression and classification. A RF is a collection of decision trees, if the count of decision trees increase, then the prediction power of the RF will increase (Ahuja et al., 2019). The problem of overfitting is eliminated by using the bootstrap sampling technique (Rustam et al., 2021). We say 'Random' forest because we select the number of trees randomly with the random number of data samples and include them in our forest (Mujahid et al., 2021). For the final prediction, we take voting to get the aggregated results from the trees and the class is finalized based on the majority voting (Braig et al., 2023). RF needs three basic hyper-parameters to be in place i.e., number of trees, number of features sampled, and node size (Bengesi et al., 2023).

3.7.2.6 K-Nearest Neighbour (KNN)

KNN is one of the most popular non-parametric classifiers used not only for text classification but also in regression problems, pattern identification and information retrieval (Bengesi et al., 2023). KNN is also called as lazy learner as it classifies text based on the similarity between the nearest neighbours using the distance between them. And the majority voting of the nearest neighbours will decide the class (Samuel et al., 2020; Mujahid et al., 2021). Euclidean distance is the metric used to calculate the distance between the data points (Ahuja et al., 2019).

The strengths of KNN are easy to use, easy to implement, and works well on multi-class classification (Samuel et al., 2020). The main drawback of KNN is it doesn't perform well on large data (Mujahid et al., 2021).

3.7.2.7 AdaBoost (AB)

Adaptive Boosting also popularly known as AdaBoost is a boosting-based ensemble technique which builds a stronger classifier by combining multiple weaker classifiers. AdaBoost uses decision trees as the base learner by default. The main objective of AdaBoost is to train classifiers with the falsely classified data points from the previous classifiers (Kaya et al., 2019; Mujahid et al., 2021).

3.7.2.8 eXtreme Gradient Boosting (XGBoost)

XGBoost was developed upon supervised learning algorithms, ensemble learning, decision trees, and Gradient Boosting Classifier (Bengesi et al., 2023). It is a tree-based model works in a similar way as Gradient Boosting Classifier with an additional feature of assigning weights to each sample. Unlike AdaBoost and Gradient Boosting, XGBoost uses regularization techniques (L1 and L2) to handle overfitting and also fits multiple weak learners (decision trees) parallelly. It uses a log loss function which considers false classifications, helps to reduce loss and increases accuracy (Rustam et al., 2021).

It as many advantages over other ML techniques i.e., support regularization, parallel processing, handling missing data, running cross-validation, and it is best suited for small and medium datasets but also works well on larger datasets as well (Bengesi et al., 2023).

3.8 Evaluation Metrics

The performance of the discussed algorithms were evaluated using four evaluation metrics: Accuracy, Precision, Recall and F1-Score. To measure the values of these metrics, TP, TN, FP, and FN were used (Mujahid et al., 2021). True positive (TP) is the number of correctly predicted positive class samples and True Negative (TN) is the number of correctly predicted negative class samples. On the other hand, False Positive (FP) is the number of falsely predicted positive class samples and False Negative (FN) is the number of falsely predicted negative class samples (Rustam et al., 2021). All the four metrics were described in detail below.

Accuracy: Accuracy is the most widely used evaluation metric which is the ratio of correctly classified prediction to the overall prediction (Naresh and Venkata Krishna, 2021). Maximum accuracy score is 1 and minimum accuracy score is 0 (Rustam et al., 2021). It is written as shown in formula 3.4.

$$\text{Accuracy} = (TP + TN) / (TP + FP + TN + FN) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3.4)$$

Precision: Precision shows us the exactness of the algorithm used and lies in [0,1] (Rustam et al., 2021). It is the ratio of correctly classified samples to the total number of positive samples (Naresh and Venkata Krishna, 2021). It is written as shown in formula 3.5.

Precision = TP / (TP + FP) ………………………………………(3.5)

Recall: Recall shows us the completeness of the algorithm used and lies in [0,1] (Rustam et al., 2021). It is the ratio of correctly predicted positive samples to all the observations of the actual positive class (Ahuja et al., 2019). It is written as shown in formula 3.6.

Recall = TP / (TP + FN) …………………………………………(3.6)

F1-Score: F1-Score is another widely used evaluation metric for machine learning models (Onan, 2021). It is a harmonic mean of Precision and Recall which lies on [0,1] (Rustam et al., 2021). The weighted average of Precision and Recall is known as F1-Score i.e., more useful than accuracy when we have an unbalanced data in hand (Ahuja et al., 2019). It is written as shown in formula 3.7.

F1-Score = (2 * Precision * Recall) / (Precision + Recall) …………(3.7)

We also used k-fold cross validation to evaluate the model's performance. The training data will be validated more efficiently by this way i.e., the training data will be split into k equal parts. Then one of the parts is treated as testing dataset and remaining k-1 parts are treated as training dataset. This process will be repeated for k number of times. And the average of those is considered as the final measure (accuracy, precision, recall etc.,) of the model (Jayasurya et al., 2022).

3.9 Summary

In this section, we had discussed the entire framework proposed for sentiment analysis on twitter data. Started with the data considered for this study. The dataset had a total of 58,312 tweets and its class distribution after removing duplicates and null values was 19,414 positive tweets, 21,503 negative tweets, and 17,395 neutral tweets. To handle this unbalanced dataset, we used three balancing techniques i.e., Random Undersampling, Random Oversampling, and SMOTE. Then the dataset was cleaned and pre-processed. Most of the pre-processing steps were referred from (Symeonidis et al., 2018). From the cleaned data, the features were extracted using N-grams, BoW, and TF-IDF vectorizer. With the extracted features, lexicon-based and ML based algorithms (eight different algorithms) were used to classify the tweets. For lexicon-based approach, we considered three popular lexicons i.e., TextBlob, VADER, and SentiWordNet. Finally, to evaluate the performance of the algorithms, we used accuracy, precision, recall, and F1-Score.

CHAPTER 4

ANALYSIS AND IMPLEMENTATION

In this chapter, the implementation of the methods and techniques mentioned in chapter 3 were discussed. Along with that, the basic analysis of the dataset in hand was done. The implementation was performed in python and libraries used for each of the tasks mentioned in chapter 3 was given as follows. The data interpretation and manipulation were performed using NumPy and Pandas libraries. For data pre-processing, RegEx and NLTK libraries were used. Matplotlib and seaborn libraries were used for data visualization. For feature extraction and train-test split "sklearn" library was used. Data balancing techniques used to handle the imbalance in the dataset was implemented using "imblearn" library. For lexicon-based analysis, textblob, vaderSentiment, and NLTK libraries were used. To build the ML algorithms and for data evaluation "sklearn" library was used.

4.1 Introduction

The implementation process was begun with understanding of the dataset in hand and then the data cleaning steps were implemented i.e., handling the data redundancies, removing null values, outliers, and tweets with multiple sentiments. Then the data was further pre-processed so that the unwanted elements in the tweets were removed and some of the elements were highlighted or transformed into meaningful format. The pre-processed data was analysed using the data visualization. Then the imbalance in the "sentiment" class was handled before the feature extraction. Finally, lexicon-based and ML based techniques mentioned in chapter 3 were implemented using the extracted features. At the very end of the chapter, the hyperparameter tunning of ML algorithms were given.

4.2 Data Understanding

As we mentioned earlier in section 3.2, The dataset used for twitter sentiment analysis was taken from Kaggle which had two csv files. One of them was to train the data and the other one was to validate the performance of the model built. The "twitter_training.csv" file had a total of 69,491 raw tweets along with their sentiments. And the "twitter_validation.csv" file had a total of 999 raw tweets along with their sentiments. Both the files had the same four columns i.e., id, entity, tweet content, and sentiment. To read and understand the data, we used NumPy and Pandas libraries of python.

In the dataset, there was only one numerical field i.e., "tweet id". Since, it didn't add any value for the sentiment analysis, it was dropped. The field "tweet content" contains raw tweets which are naturally noisy and irrelevant. Also, based on the analysis, we found that the dataset had some duplicate values and null values. Hence, these inconsistency in the data were cleaned up using the data cleaning steps mentioned in the following section 4.3.

4.3 Data Cleaning

As we discussed in section 3.2, this study is not focusing on entity-based sentiment analysis, it is aiming to build a model to predict the sentiment of the general text. Hence, the entity field is removed from both the datasets. Also, Other than positive, negative, and neutral, the "sentiment" field has one more value "Irrelevant" which is associated for the tweets which are irrelevant to the given entity. Since this study is not focusing on entity-level, tweets with "Irrelevant" sentiment was also dropped. After this, the data cleaning steps mentioned in the section 3.4 were executed one by one.

Handling null values: In training dataset, the "tweet content" field had 571 null values and the test dataset didn't have any null values. Hence all the 571 rows were dropped.

Removing duplicate values: The training dataset had 3.5k duplicate rows approximately. And the test dataset had only one duplicate row.

Removing tweets with multiple sentiments: In some cases, a same tweet was labelled as different sentiments. For example, the tweet "Thank God" was labelled as positive and negative. Same as that, The training dataset had 46 tweets with two different labels and 71 tweets with three different labels. Hence, these tweets with multiple sentiments were dropped. On the other hand, the test dataset didn't have any inconsistent labelling.

Table 4.1 shows, how the data cleaning process had changed the distribution of class sentiments in our training and test datasets. The data cleaning steps performed in this study were removing unwanted columns, dropping "irrelevant" sentiment tweets, handling null values, and removing duplicate values.

Table 4.1. Class distribution of the training and test dataset before and after data cleaning.

Sentiment Class		**Positive**		**Negative**		**Neutral**		**Total**	
		Training	Test	Training	Test	Training	Test	Training	Test
No. of tweets	Before Cleaning	20832	277	22542	266	18318	285	57485	828
	After Cleaning	19032	276	21138	266	17010	285	57180	827

4.4 Data Pre-processing

The cleaned data was pre-processed, so that it will become neat and meaningful which can be used for visualization and feature engineering. The reason behind the pre-processing steps performed and the order in which they were applied was mentioned in the section 3.5. The implementation process and the techniques, tools or libraries used for each pre-processing step were given below.

Lowercasing: As a first step, the "tweet content" field was lowercased using the lower() method of python.

Removing URLs: Most of the tweets had a link which doesn't add any value for that respective tweet's sentiment. So, we removed all the URLs using the regular expression "(www\.[\S]+|https?://[\S]+|pic\.[\S]+)" which removed all the links had "www", "http" or "pic". But some of the tweets had a whitespace between the forward slash(/) of the URLs. So handled this with the regular expression "('(/ | /|/)','/',tweet)" and then applied the above-mentioned regular expression. Some of the samples, before and after removal of URLs was given in table 4.2.

Table 4.2. Before and after removal of URLs.

Before removing URLs	**After removing URLs**
I love the new DLC!!!. @ Borderlands @ DuvalMagic @ GearboxOfficial pic.twitter.com / PKZJxlGXLj	i love the new dlc!!!. @ borderlands @ duvalmagic @ gearboxofficial
Cisco and NVIDIA are ready to provide superiority on any device, anywhere oal.lu / FQNeh https: / / www.co / eeQA2fZePs	cisco and nvidia are ready to provide superiority on any device, anywhere oal.lu/fqneh
Google News For Just One Effortless Robotic Dog Trot more at www.com/news/google-re... https://t.co/X3mem3ktor]	google news for just one effortless robotic dog trot more at

Removing Hashtags, User mentions, Retweets and HTML tags: As mentioned in the section 3.5 these elements are very useful and powerful tool in twitter but for sentiment analysis using machine learning models, they are unnecessary and just increases the dimensionality. So, these elements were removed from our corpus using the following regular expressions:

- For Hashtags - #[\S]*
- For User mentions - @[\s]?[\S]+
- For Retweets - r'\brt'
- For HTML tags - <[\S]+>

How the tweet content changed after removal of hashtags, user mentions, retweets, and HTML tags was shown in table 4.6.

Handling Contractions: A contractions dictionary with and without the punctuations was created because, our dataset had some unwanted spaces between the punctuations. For example, "haven't" was spelled as "haven ' t". So, the contractions replacement was applied before and after removing punctuations. Table 4.3. shows some samples of the contractions dictionary used for contraction replacement. And how it affected the tweet content after replacement was shown in table 4.6.

Table 4.3. Samples of the contractions dictionary created.

Contraction	Replacement Word
he'll, he'd, I'd've	he will, he would, I would have
might've, couldn't	might have, could not
ain't, isn't, I'm	is not, is not, I am
don't, y'all'd, ma'am	do not, you all would, madam

Removing Punctuations, Special characters and Emoticons: These elements were removed from the tweet content using a simple regular expression '[^a-z0-9]' which retains only numbers, alphabets and white spaces. The tweet content before and after removal of these elements were shown in table 4.6.

Handling Whitespace: The space at the beginning and the end of the tweet were removed using the regular expression ('^ | $', '',tweet). And two or more spaces in between the tweets were removed using the regular expression (' +',' ',tweet). Some samples before and after whitespace removal from our dataset is shown in table 4.6.

Elongated Words: The elongated words were removed using the regular expression ("(.)\\1{2,}","\\1\\1",tweet) which checks for two or more occurrence of a character and replace it with just two occurrences of the character. For example, "sooooo keeeeeeen" was replaced as "soo keen". Some samples before and after replacement from our dataset was shown in table 4.6.

Negation Handling: Firstly, a "not_words" function was created which will take the words immediately after the 'not' word in the tweet and store in a list. The study also included words immediately after 'not so' since it also serves the same purpose. Then using the NLTK wordnet's synsets, the first antonyms found for the words in the "not_words" function were once again stored in a separate list. Then a dictionary was created with both the lists. Some samples of the dictionary were shown in table 4.4. Finally, a "not_words_replacement" function was created which gone through each tweet one by one and replaced the "not + word" with the replacement word (antonym of the word). For example, "not good" was replaced as bad, "not so complex" was replaced with simple. From our dataset, some sample tweets before and after negation handling were shown in table 4.6.

Table 4.4. Samples of the not_words_replacement dictionary created.

not + word	Replacement Word (Antonym)
not above	below
not spending	income
not credited	debit
not so hot	cold

Slang words and Abbreviations: A slang words dictionary was created with the help of (Chat / Internet Slang | Abbreviations | Acronyms | Kaggle, 2023). Additionally, some of the common slang words used in social media was added to the dictionary. A total of 170 slang words or abbreviations were found in our dataset which were replaced with the proper English words. Some of the samples were shown in the table 4.5.

Table 4.5. Samples of the slang words dictionary created.

Slang word	Replacement Word
2day	today
lol	laughing out loud
yt	youtube
wth	what the hell
sry	sorry

Numbers Removal: Firstly, all the words containing numbers were removed using the regular expression "\S*\d\S*" because those words were mostly a left-over part of the links because of the whitespaces between the links or just numbers. Then, just to make sure the dataset consists of only alphabets and whitespaces, the regular expression '[^a-z]' was used.

Stop Words Removal: Using the NLTK's stop words, and the language as English, we had created a "stop_words_removal" function which search word by word in each tweet for these stop words and removes it. Some samples before and after stop words removal was shown in table 4.6.

Table 4.6. Tweets before and after each pre-processing step implementation.

Pre-processing step used	Before implementation	After implementation
Hashtags removal	It's like the universe doesn't want me to get my shit together and wants me to just sit on my ass playing video games all day #	it's like the universe doesn't want me to get my shit together and wants me to just sit on my ass playing video games all day
User mentions removal	Platinum is the best booty @ Borderlands	platinum is the best booty
Retweet removal	RT report: "Walking around India is the worst thing!"	report: "walking around india is the worst thing!"
HTML tag removal	Check<unk> this epic streamer!.	check this epic streamer!.
Contractions	cheers didn't want the legendary one anyway	cheers did not want the legendary one anyway
Contractions	going to finish up borderlands 2 today. i've got some new commands…	going to finish up borderlands 2 today. i have got some new commands…
Punctuations / Special Characters Removal	damn, that looks stupid - _ -	damn that looks stupid
Emoticons removal	cod black ops cold war. sorry but the mechanics on the game are really shite. it feels like i'm playing a really old game.. 🤣. 🤦. .	cod black ops cold war sorry but the mechanics on the game are really shite it feels like i m playing a really old game
Handling whitespace	…being 6 years behind on nvidia drivers and cars i have no idea how i ever didn t notice	…being 6 years behind on nvidia drivers and cars i have no idea how i ever didn t notice
Handling Elongated Words	me omg i miss rhys soooo much i would do anything to see him borderlands 3 me	me omg i miss rhys soo much i would do anything to see him borderlands 3 me
Negation Handling	he base weapons of black ops cold war not so good	the base weapons of black ops cold war badness
Negation Handling	amazon does not function in my hood	amazon does malfunction in my hood

Slang words and Abbreviations	this was adorable i almost got ran from lol still unsure	this was adorable i almost got ran from laughing out loud still unsure
Slang words and Abbreviations	what the hell is goin on and been with trying this 4 2day to do the live	what the hell is goin on and been with trying this 4 today to do the live
Slang words and Abbreviations	trend deleted wth govt of pakistan	rend deleted what the hell govt of pakistan
Numbers removal	i am getting on borderlands 2 and i will murder you me all	i am getting on borderlands and i will murder you me all
Numbers removal	today s best game movies fifa 13 18 borderlands 8 15 more 9to5toys…	today s best game movies fifa borderlands more totoys…
Stop words removal	i am coming to the borders and i will kill you all	coming borders kill
Stop words removal	just realized that the windows partition of my mac is like years behind nvidia drivers and i have no idea how i did ignore	realized windows partition mac like years behind nvidia drivers idea ignore

Lemmatization: Firstly, the tweets were tokenized using the NLTK tokenize package. Then for each word, POS tags were found using NLTK pos_tags. With the help of these tags, the lemma (root form) of the words were found. For lemmatization, WordNet lemmatizer was used. Some of the lemmatized tweets were shown along with its stemmed form in the table 4.7.

Stemming: Again, the tweets were tokenized first. Then each word of the tweets were stemmed (root form) using Porter Stemmer.

Handling Non-English Words: Some of the words in our corpus was incorrectly spelled or simply incorrect word which might be left over part of the link because of the whitespaces between the links. So, firstly all the proper English words were kept aside using the "words" corpus of NLTK. If a word is present in the NLTK "word" corpus, it is considered as a proper English word and all the remaining words were saved in a separate list and called as non-English words. Then the less occurred words (<= 10) of this list was dropped with the help of pandas library since those words where probably incorrect words which were less occurred. Hence, dropping these words won't highly affect the sentiment of their respective tweets also helps the performance of the models because of the dimensionality reduction. Some of the removed words with their number of occurrences shown in table 4.7.

Additionally, one letter and non-English two letter words were removed from the corpus as those words had a total of 21,500 occurrences which didn't add any value to the sentiment prediction. Some of these samples were shown in table 4.8 with their number of occurrences.

Table 4.7. Dropped less occurred non-English words with their number of occurrences (count).

Words	Count
abbott, nemo, nydailynews, geez, yogg	10
abc, ns, pos, youtubers, fanart, fech	9
acct, qitmeer, yusho, fortnites, rubick	8
rem, redeemment, erm, ellison, eb	7
abella, provi, pupcol, gordonke, gorsky	6
aages, paytm, gameriv, gamehag, zysola	5
aat, presstv, probablys, gawwd, zofia	4
aatmanirbhar, otk, pindick, zika, pbe	3
phelps, pepsi, pcg, patts, freeders, pfbly	2
funnyaks, zurich, aall, saab, julsow, dem	1

Table 4.8. Removed one letter and non-English two letter words with their number of occurrences (count).

Word Type	Words	Occurrence Range
One letter	u, x, v	>1000
	c, e, w, p, r, b	300 to 1000
	g, n, j, f, l, h	100 to 300
	k, q, m, z, o, t…	<100
non-English two letter	tv, uk, tt, co, ok	>200
	ii, et, yu, rn, tf…	100 to 200
	dr, vc, dp, nr, wn…	50 to 100
	cs, gd, rp, xp, cc…	<50

Spelling Correction: Firstly, the non-English words list saved in the previous step was taken. Then, with the help of spellchecker and the TextBlob's spelling module, the misspelled words in the list were identified and replaced with the correct spelling. In the process, we found that some of the words were correctly identified and replaced with the correct spelling but some of them were wrongly identified and replaced with different words. Some samples of both the cases were shown in the table 4.9. Since this approach unable to differentiate the correctly replaced and the wrongly replaced words. This spelling correction step was not implemented in this study. In future, some of the advanced techniques will be implemented to identify the misspelled words correctly and replace it with correct spelling.

Table 4.9. non-English words (replaced with correct spelling vs replaced with incorrect words).

	Original words	Replaced words
Correctly replaced	activiti	activity
	achievs	achieve
	academie	academic
	fina	find
	whaat	what
Wrongly replaced	playstation	plantation
	fifa	if
	google	goose
	xbox	box
	soo	so
	retweet	between

The lemmatized tweets were used for both spelling correction and non-English words removal. The stemmed tweets were not used because how it processes the words. Since it just removes the affixes of the words, the stemmed words won't be present in the NLTK's "words" corpus and just considered as a non-English word. Hence, only lemmatized tweets were considered for the further study. Table 4.10. shows the comparison of tweets before and after pre-processing of training dataset. And the figure 4.1 shows the entire pre-processing structure along with the total words count and the number of unique words in each pre-processing steps. All the pre-processing steps implemented on the training dataset were also applied on the test dataset and some of the samples of test dataset before and after applying pre-processing were shown in table 4.11.

Table 4.10. Sample tweets of training dataset before and after pre-processing.

Original Tweet Content	Pre-processed Tweet	Sentiment
Rock-Hard La Varlope, RARE & POWERFUL, HANDSOME JACKPOT, Borderlands 3 (Xbox) dfr.it / RMTrgF	rock hard la rare powerful handsome borderlands xbox	Neutral
Mini Stream Schedule: . Wed: July 22nd at 3pm Borderlands 2. . Have a great week..	mini stream schedule wed july borderland great week	Neutral
Yooooo, that could be so bad!	yoo could bad	Positive
If employees understand the reasons behind the rules and regulations, there is a good chance that they will respect them. - Johnson & Johnson Co.	employee understand reason behind rule regulation good chance respect johnson johnson	Positive

If a video game inside a games system crashes but you would you call it just videogamecrashception? I mean to call that!.. youtu.be/H9jUp8kt9wU	video game inside game system crash would call mean call youtu	Negative
@JeffBezos Must be feeling the pain of his divorce. If you cannot deliver what you promise, do not promise to deliver.. . Do you back your word, or not? I expected a package that I needed. Amazon ...	must feel pain divorce can not deliver promise promise deliver back word expect package need amazon admit fault seller pay subscription	Negative

Table 4.11. Sample tweets of test dataset before and after pre-processing.

Original Tweet Content	Pre-processed Tweet	Sentiment
@Microsoft Why do I pay for WORD when it functions so poorly on my @SamsungUS Chromebook? 🙄	pay word function poorly chromebook	Negative
surprise i still love #borderlands ' jack pic.twitter.com/hAuwN4M4pS	surprise still love jack	Positive
@6th__man playing red dead redemption-\n\n“Oh shit a bear!” *starts running* \n\nMe- you can’t out run a bear they run like 40mph \n\nHim- what?\n*immediately mauled by bear*	play red dead redemption oh shit bear start run can not run bear run like immediately maul bear	Neutral

Figure 4.1. Pre-processing steps implemented along with the total words count and the number of unique words in each pre-processing steps.

4.5 Data Visualization

This subsection helps us to get more insights and understand the dataset even better with the help of data visualization. From the pre-processed dataset, the sentiment field and the final pre-processed tweet field were considered for data analyses and visualization. Firstly, both the fields were analysed separately i.e., called as univariate analysis. Then the most frequent words of each sentiment were visualized using word clouds and N-grams.

4.5.1 Univariate Analysis

The only independent variable “pre-processed tweet” was explored first. After pre-processing it had almost 500 new empty tweets, which were removed. And so, it ended up with a total of 56,634 tweets which consists of 18,860 positive tweets, 20,938 negative tweets and 16,836 neutral tweets. Table 4.12 shows some of the samples of new empty tweets after pre-processing. The boxplot and histogram plot showed that the distribution of the “pre-processed tweet” field was positively skewed. And it had almost 1000 tweets with a length of 30 to 120 words which were considered as outliers and removed with the help of traditional outliers removal formula. Anything above high fence and below low fence were considered as outliers as mentioned in the formulae 4.1 and 4.2.

High fence = Q3 + 1.5 * IQR ……………………………………(4.1)
Low fence = Q1 - 1.5 * IQR ……………………………………(4.2)

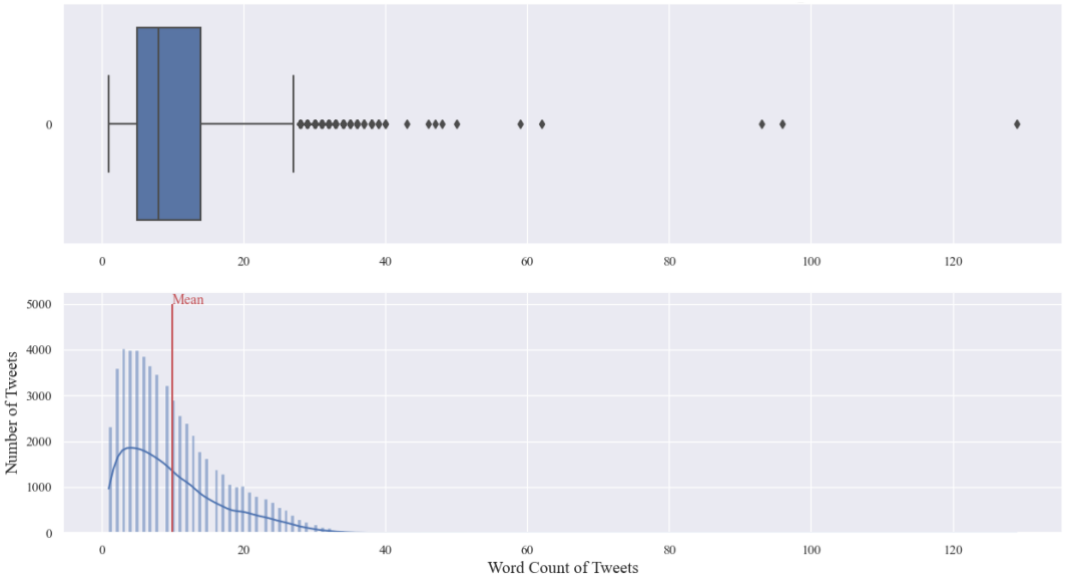
where Q1 is 25^{th} percentile, Q3 is 75^{th} percentile and IQR (Interquartile Range) is Q3 - Q1. In our case, the mean and median were around nine words. The 25^{th} and 75^{th} percentile were 5 and 14 words respectively. So, the high fence and low fence were turned out to be 27.5 and -8.5 words. Hence, the final retained tweets length was 1 to 27 words. The distribution of number of words of tweets before and after removing outliers were shown in figure 4.2. And some of the outliers of our dataset i.e., longer tweets were shown in table 4.13.

Table 4.12. Sample empty tweets after pre-processing with their sentiments.

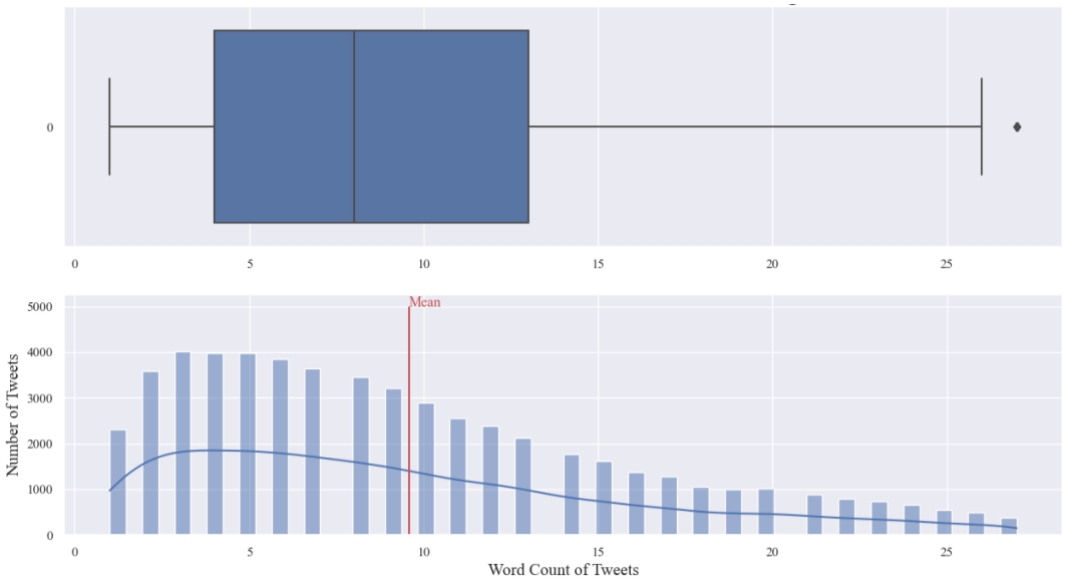
Original Tweet Content	Pre-processed Tweet	Sentiment
2014 . . pic.twitter.com/P1A5IeWDOa		Neutral
"Why shouldn't I?" "If I can do it, then you can too."		Neutral
Who's down with some @Borderlands on		Positive
@ NVIDIAGeForce @ nvidia		Positive
??????????????????????????????????????		Negative
Not my gunblade pic.wikipedia.org / Gfk75BoW		Negative

Table 4.13. Samples of outliers i.e., longer tweets with their sentiments.

Tweet	Sentiment
event dedicate victory day great patriotic war hold part celebration anniversary victory great patriotic war attend veteran great patriotic war home front worker home	Neutral
know pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa pa	Positive
time many low income country world today include united state united kingdom italy	Negative



(a) Before removing outliers



(b) After removing outliers

Figure 4.2. The distribution of number of words of tweets before and after removing outliers.

After the outliers were removed, the sentiment distribution was slightly unbalanced. The neutral class was low in number when compared to negative and positive classes. As the figure 4.3 shows, the neutral class was 29.7%, the positive class was 33.3%, and the negative class was 37.0%. Hence the class imbalance was addressed in section 4.8.

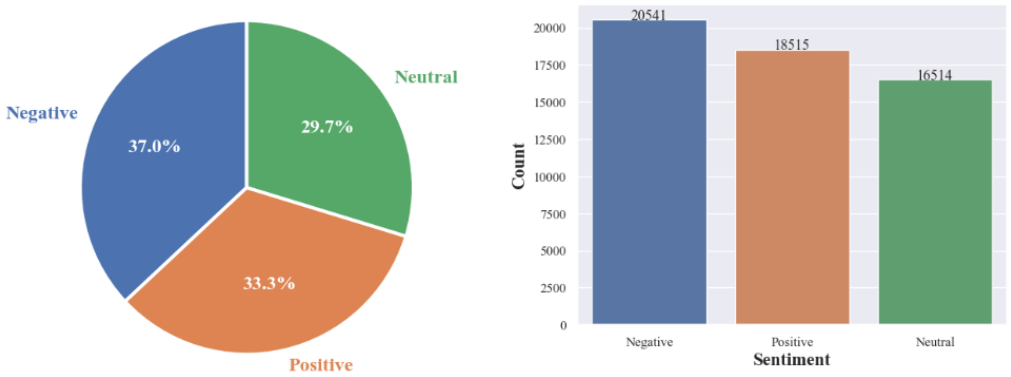


Figure 4.3. Sentiment field's (a) class distribution and (b) the number of samples of each class.

4.5.2 Word Cloud

Word cloud is a commonly used visualization technique which shows how frequent the words were appeared in the dataset. The size of the words in the cloud is directly proportional to the frequency of the words in the dataset. And the word clouds can be of any user-defined irregular shape. So, since our study was related to twitter, the word clouds were generated in the form of twitter logo with the help of python libraries such as matplotlib, WordCloud, Image, and ImageColorGenerator. As shown in the figure 4.4, after removing stop words we generated four different word clouds: one for the entire dataset and the other three for each of the sentiment polarities (positive, negative, and neutral) (Bengesi et al., 2023). "game" was the most commonly appeared word on all the three sentiments. And "love", "go", "good", and "play" were the common words of positive class. "fuck", "go", "shit", and "bad" were the common words of negative class. But the neutral class didn't have any frequently appearing words other than "game" and "new".

Figure 4.4. Word clouds. Most common words of the entire dataset and all the three sentiment polarities (Positive, Negative, and Neutral).

4.5.3 N-grams

Other than the frequent/common words identification, finding the word pairs also known as word chains was the important aspect of textual analysis. And it is known as N-grams identification (Samuel et al., 2020). In this study, we transformed the "pre-processed tweet" field into a word corpus and generated Unigrams, Bigrams, and Trigrams with the help of python libraries sklearn's CountVectorizer, Seaborn, and matplotlib.

Unigrams: Unigram is the occurrence of the single words in the given sentence. In our dataset, the most frequent words were "game", "play", "fuck", "like" and so on. The figure 4.5 shows the top 10 unigrams of entire dataset and also for each sentiment polarities. It was observed that the most frequent words popped out on the word clouds were showed up on the unigrams figure 4.5 as well. And the word "game" was the most frequent word in all the classes. Other than that, the words "play" and "like" were also appeared in all the three classes' top 10 words.

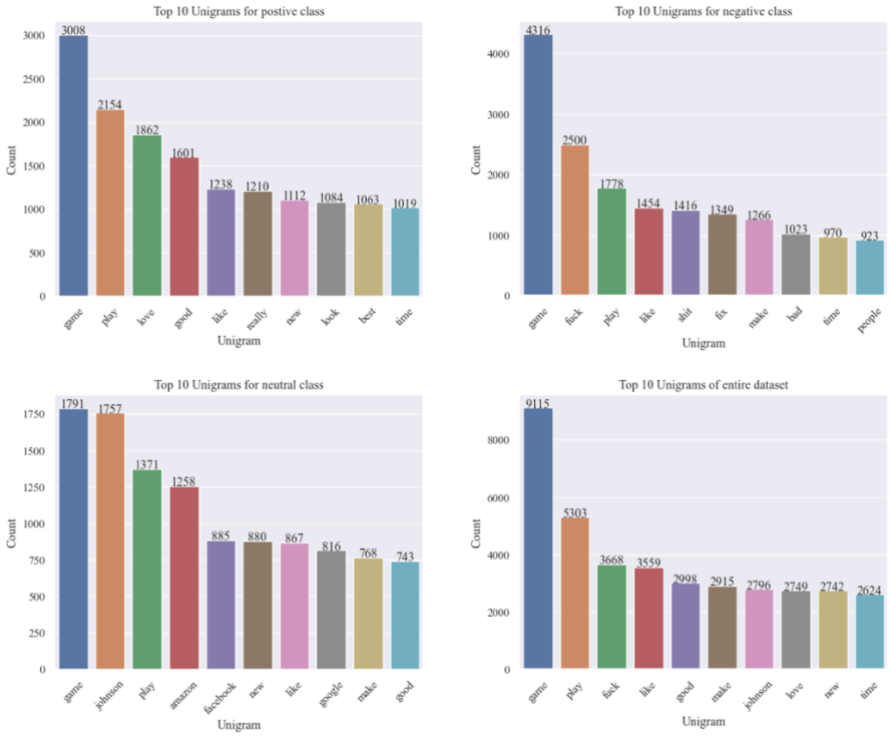


Figure 4.5. Top 10 unigrams of entire dataset and for each sentiment polarities (Positive, Negative, and Neutral).

Bigrams: Bigrams are the pairs of two consecutive words in the given sentence. And some of the most frequent bigrams of our dataset were "red dead", "johnson johnson", "dead redemption", "home depot" and so on. But "red dead" and "dead redemption" were appeared in all the three classes' top 10 words. The figure 4.6. shows the top 10 bigrams of all the three sentiment polarities and the entire dataset.

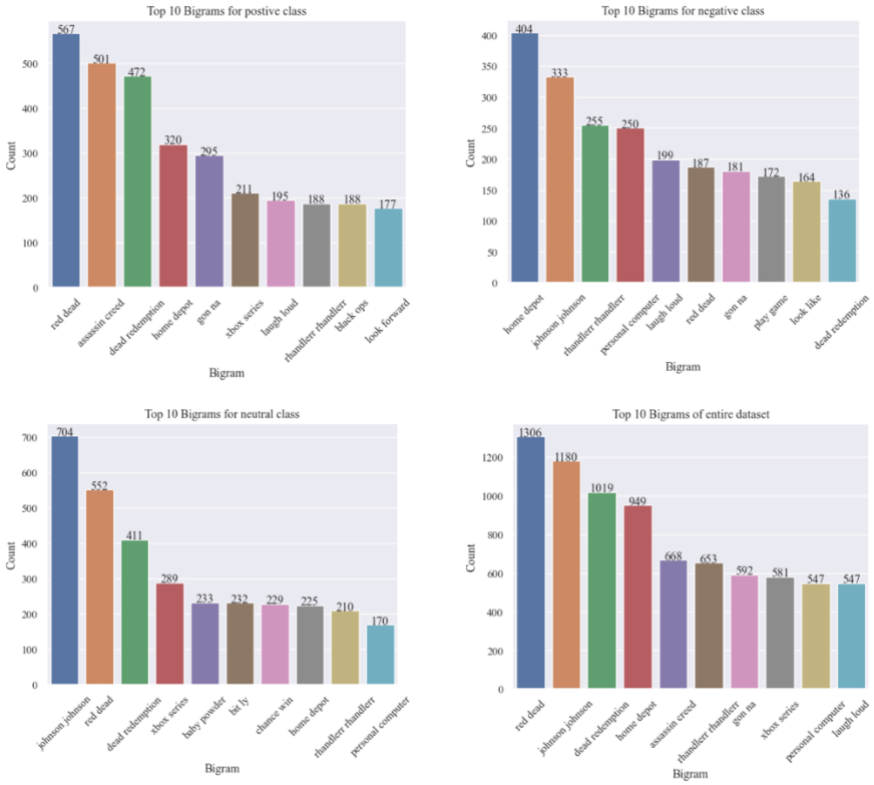


Figure 4.6. Top 10 bigrams of entire dataset and for each sentiment polarities (Positive, Negative, and Neutral).

Trigrams: Trigrams are the pairs of three consecutive words in the given sentence. In our dataset, the trigram "red dead redemption" was the most appeared one with the frequency of 949. And the second most frequent trigram was "rhandler rhandler rhandler" with the frequency of just 358. And both of them were appeared in all the three sentiment polarities. "black ops cold" and "ops cold war" were the next frequently appeared trigrams which showed up on both positive and negative classes. The top 10 trigrams of all the three sentiment polarities and for the entire dataset was shown in figure 4.7.

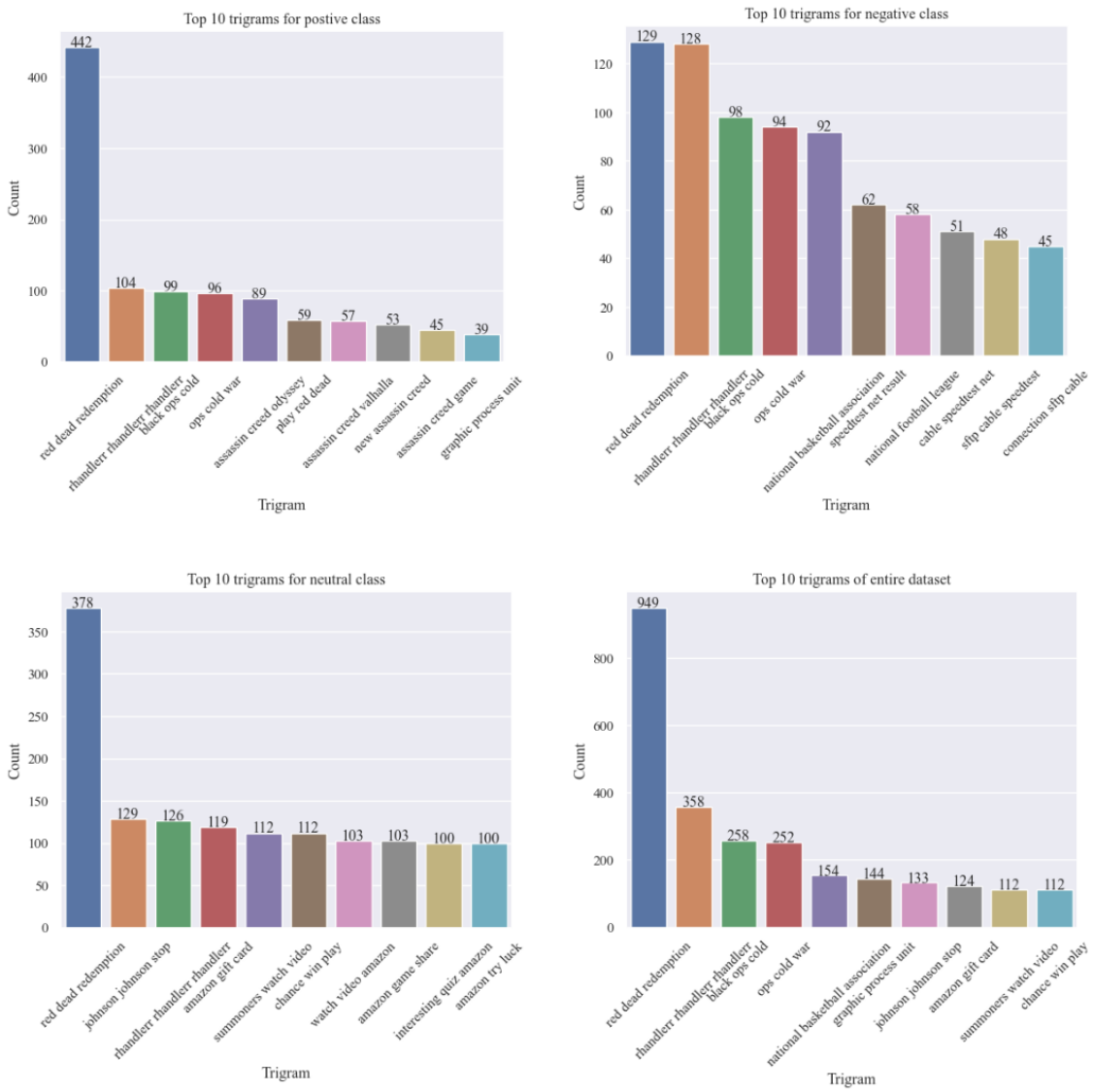


Figure 4.7. Top 10 trigrams of entire dataset and for each sentiment polarities (Positive, Negative, and Neutral).

The unigrams, bigrams, and trigrams were used in the feature extraction process which is further explained in section 4.7.

4.6 Train-Test Split

The pre-processed data was split into parts. Before that, the given test dataset was very small when compared to the training dataset. As mentioned in table 4.1, the test dataset was less than 2% when compared to the training dataset. Hence, both the given datasets were pre-processed and combined together. Then, it was split into two parts. One is used to train the models while the other one is used to test the performance of the trained models. The common idea for the train-test split ratio was, if the data in hand was huge, the best split ratio would be 90:10 or 80:20. If the data was small, then the best ratio would be 70:30 or 60:40.

As we see in table 4.14, the data in hand was neither huge nor very small. Hence, we had experimented with 80:20 and 70:30 split ratios. The data count after both the split ratios were shown in table 4.15. With the ratio of 70:30, the combined total of 56,394 tweets were split into 39,475 and 16,919 for training and testing the models respectively. And for 80:20 ratio, the total data was split into 45,115 data for training and 11,279 data for testing the models.

Table 4.14. The given train and test data sets' value counts after pre-processing.

	Given training dataset	Given test dataset	Combined dataset
Positive	18,515	276	18,791
Negative	20,541	265	20,806
Neutral	16,514	283	16,797
Total	55,570	824	56,394

Table 4.15. The data count of train-test split with 70:30 and 80:20 split ratios.

	Combined dataset	Train-Test Split	
		70 : 30	80 : 20
Positive	18,791	13,160 : 5,631	14,994 : 3,797
Negative	20,806	14,548 : 6,258	16,673 : 4,133
Neutral	16,797	11,767 : 5,030	13,448 : 3,349
Total	56,394	39,475 : 16,919	45,115 : 11,279

4.7 Feature Extraction

The pre-processed tweets were converted into feature vectors using two of the popular methods of feature extraction i.e., BoW approach and TF-IDF vectorizer. The feature extraction process was implemented using sklearn.feature_extraction module's CountVectorizer and TfidfVectorizer. The only independent variable "pre-processed tweet" was converted into feature vectors using the above-mentioned methods. The values of dependent variable "sentiment " was converted into numeric data as follows: "positive" : 0, "negative" : 1, and "neutral" : 2.

The table 4.16 shows the static and dynamic parameters of both the feature extraction techniques used to convert the "pre-processed tweet" variable into feature vectors. Along with that, it also shows the shape of the feature matrix generated for train and test datasets of 70:30 and 80:20 split ratio. The hyper-parameters used for both count vectorizer and TF-IDF vectorizer were same. The dynamic (experimented with multiple values) parameters were "min_df" and "ngram_range". The parameter "min_df = k" allows words/tokens occurred at least in "k" number of documents/sentences. And the parameter "ngram_range" takes the N-grams (which we discussed in sub-section 4.5.3) as input and works accordingly. The "min_df" values used were 2 and 5. And the "ngram_range" values used were (1,1) - unigrams, (2,2) - bigrams, (3,3) - trigrams, and (1,3) - all three n-grams. The shape of the feature matrix will be same for both the approaches as the parameters used were same. Only the values of the feature matrix will differ according to their respective approach which was already discussed in section 3.6.

Table 4.16. The static and dynamic parameters of BoW and TF-IDF vectorizer along with the shape of feature matrix generated for train and test datasets of 70:30 and 80:20 split ratio.

Static Parameters	Dynamic Parameters	Shape of feature matrix of 70:30 ratio		Shape of feature matrix of 80:20 ratio	
		Train dataset	Test dataset	Train dataset	Test dataset
stop_words='english', tokenizer=word_tokenize, max_df=0.95	min_df=2, ngram_range=(1,3)	(39475, 107818)	(16919, 107818)	(45115, 118025)	(11279, 118025)
stop_words='english', tokenizer=word_tokenize, max_df=0.95	min_df=2, ngram_range=(1,1)	(39475, 8002)	(16919, 8002)	(45115, 8201)	(11279, 8201)
stop_words='english', tokenizer=word_tokenize, max_df=0.95	min_df=2, ngram_range=(2,2)	(39475, 52655)	(16919, 52655)	(45115, 57057)	(11279, 57057)
stop_words='english', tokenizer=word_tokenize, max_df=0.95	min_df=2, ngram_range=(3,3)	(39475, 47161)	(16919, 47161)	(45115, 52767)	(11279, 52767)
stop_words='english', tokenizer=word_tokenize, max_df=0.95	min_df=5, ngram_range=(1,3)	(39475, 19507)	(16919, 19507)	(45115, 25753)	(11279, 25753)
stop_words='english', tokenizer=word_tokenize, max_df=0.95	min_df=5, ngram_range=(1,1)	(39475, 5605)	(16919, 5605)	(45115, 6016)	(11279, 6016)
stop_words='english', tokenizer=word_tokenize, max_df=0.95	min_df=5, ngram_range=(2,2)	(39475, 10465)	(16919, 10465)	(45115, 14189)	(11279, 14189)
stop_words='english', tokenizer=word_tokenize, max_df=0.95	min_df=5, ngram_range=(3,3)	(39475, 3437)	(16919, 3437)	(45115, 5548)	(11279, 5548)

4.8 Data Balancing

To handle the slight imbalance in the dataset, three different balancing techniques had been deployed as mentioned in the section 3.3. The balancing is applied only in training dataset, so that the models can train well on all the classes and avoid the distribution bias in predictive ability. And not applied on the test dataset, so that it remains unbiased and reflects the true evaluation of the trained models. The balancing techniques were deployed using SMOTE and RandomOverSampler from imblearn.over_sampling module and RandomUnderSampler from imblearn.under_sampling module.

As shown in table 4.17, after handling data imbalance using SMOTE and RandomOverSampler (ROS) for the training dataset with a train-test split ratio of 70:30 had a total of 43,644 data, while the training dataset with a train-test split ratio of 80:20 had a total of 50,019 data to be trained into the model. On the other hand, when the data imbalance was balanced using RandomUnderSampler (RUS), the training dataset with a train-test split ratio of 70:30 had a total of 35,301 data and for the training dataset with a train-test split ratio of 80:20 had a total of 40,344 data to be trained into the model.

Table 4.17. The class distribution of training dataset before and after applying balancing techniques: SMOTE, ROS and RUS for 70:30 and 80:20 train-test split ratios.

	70:30 Split Ratio				80:20 Split Ratio			
	Positive	Negative	Neutral	Total	Positive	Negative	Neutral	Total
Unbalanced	13,160	14,548	11,767	39,475	14,994	16,673	13,448	45,115
SMOTE	14,548	14,548	14,548	43,644	16,673	16,673	16,673	50,019
ROS	14,548	14,548	14,548	43,644	16,673	16,673	16,673	50,019
RUS	11,767	11,767	11,767	35,301	13,448	13,448	13,448	40,344

As shown in figure 4.8, the training dataset of 70:30 train-test split ratio had 36.9%, 33.3%, and 29.8% of negative, positive, and neutral polarity of sentiments which was slightly imbalanced. With SMOTE and ROS, the positive and neutral classes were oversampled so that they match the negative class value counts. And with RUS, the positive and negative classes were under sampled so that they match the neutral class value counts.

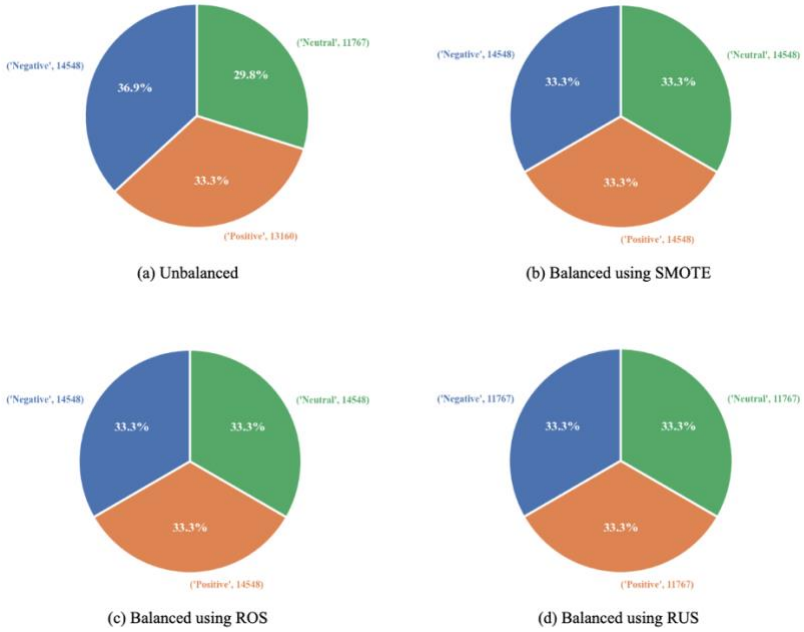


Figure 4.8. The class distribution after applying balancing techniques on training dataset of 70:30 split ratio.

On the other hand, as shown in figure 4.9, the training dataset of 80:20 train-test split ratio had 37.0%, 33.2%, and 29.8% of negative, positive, and neutral polarity of sentiments which was balanced using the same process mentioned for 70:30 split ratio. Negative class had the highest distribution and neutral class had the lowest distribution in both the cases.

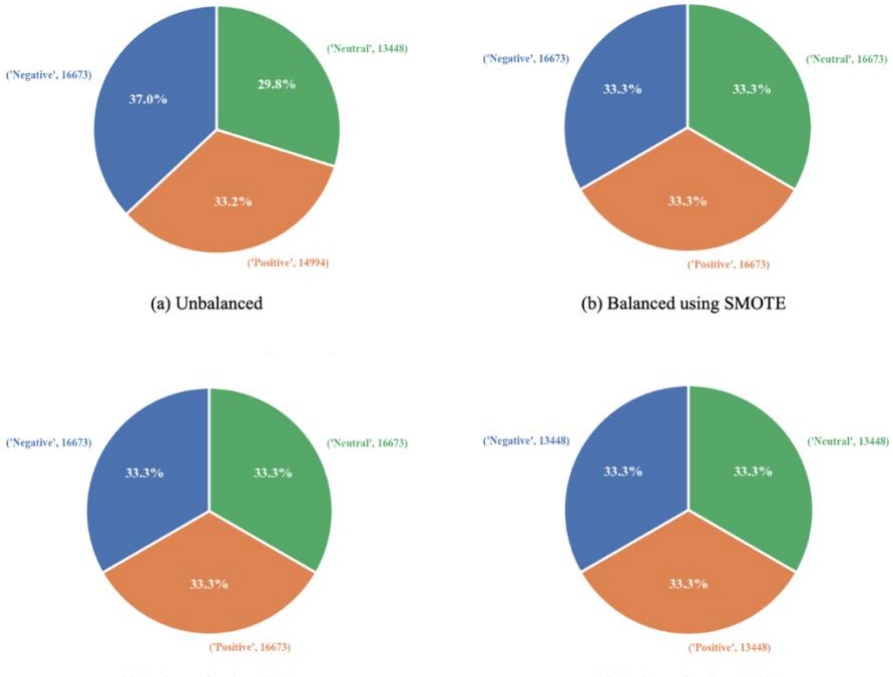


Figure 4.9. The class distribution after applying balancing techniques on training dataset of 80:20 split ratio.

4.9 Lexicon-based sentiment analysis

As discussed in section 3.7.1, this study used three popular lexicon-based techniques for the sentiment analysis of twitter data i.e., TextBlob, VADER, and SentiWordNet. Table 4.18 shows the different scoring system of all the three lexicon-based methods used to classify the tweets as positive, negative, or neutral.

TextBlob: Using the textblob library of python, we implemented this technique on our pre-processed tweets. As shown in table 4.18, the polarity score was used to classify the tweets. Also, as another approach, instead of directly finding the polarity of the tweets, the "correct()" method of textblob was applied on the pre-processed tweets to correct some misspelled words in the tweets and then tried to find the polarity of the tweets. The sentiment prediction of some sample texts using textblob was shown in table 4.19.

VADER: For VADER, the already pre-processed tweets was not used because VADER supports some of the pre-processing steps used to label the tweets. Some samples of those pre-processing steps was shown in table 4.20. Hence, to implement VADER in our dataset, the pre-processing steps mentioned in figure 4.10 was used. Then using the python library vaderSentiment's SentimentIntensityAnalyzer class, the pre-processed tweets were classified into positive, negative, and neutral. As mentioned in table 4.18, the compound score was used to classify the tweets.

SentiWordNet: Using the lexical resource "SentiWordNet" of NLTK library, we implemented this technique on our pre-processed tweets. As mentioned in table 4.18, the synsets of all the words in the tweets were extracted and then using the "SentiWordNet" the positive scores and negative scores of those synsets were taken. Finally, the total negative scores of all the words was subtracted by the total positive scores of all the words in the tweets. This final score was used to classify the tweets as positive, negative, or neutral according to the scoring range mentioned in the table 4.18. The sentiment prediction of some sample texts using SentiWordNet was shown in table 4.21.

Table 4.18. The different scoring system of TextBlob, VADER, and SentiWordNet for labelling the given texts.

Lexicon technique	Scoring system	Scoring range	Sentiment
TextBlob	Polarity Score	score > 0	Positive
		score == 0	Neutral
		score < 0	Negative
VADER	Compound Score	score >= 0.05	Positive
		0.05 > score > -0.05	Neutral
		score <= -0.05	Negative
SentiWordNet	(Total positive score - Total negative score) of all words' synsets	score >= 0.05	Positive
		0.05 > score > -0.05	Neutral
		score <= -0.05	Negative

Table 4.19. The sentiment prediction of sample texts using textblob.

Sample text	Polarity score	Sentiment
Youtube shorts was the best stress buster but also very addictive	0.5	Positive
The girl was gorgeous in that dress	0.7	Positive
I missed the train by 5 minutes	0.0	Neutral
The least favourite movie of marvel was Ironman	-0.3	Negative
It was the worst day ever in my life	-1.0	Negative

Table 4.20. The pre-processing steps of textual analysis supported by VADER with examples.

Pre-processing step	Example	Compound Score
Capitalization	I love the way kohli celebrated this century	0.8360
	I LOVE the way kohli celebrated this century	0.8636
Punctuation	That movie was awesome	0.6249
	That movie was awesome!	0.6588
	That movie was awesome!!	0.6892
Conjunctions	loved watching the interesting matches but the commentary was horrible	-0.3506
	the commentary was horrible but loved watching the interesting matches	0.8248
Degree Modifiers	Friends sitcom on Netflix was good.	0.7184
	Friends sitcom on Netflix was extremely good.	0.7425
	Friends sitcom on Netflix was somewhat good.	0.6915
Emoticons	The show was great	0.6249
	The show was great :-)	0.7506
	The show was great :-) :-)	0.8271
Emojis (utf-8 encoded)	The show was great 💘	0.8519
	The show was great 😘	0.7845
Slangs	That was a great joke	0.7430
	That was a great joke lmao	0.8807
	That was a great joke lol	0.8442
Negations	Last night wasn't that fun	-0.4023
	Last night was not that fun	-0.4023

Table 4.21. The sentiment prediction of sample texts using SentiWordNet.

Sample text	Senti_score	Sentiment
Youtube shorts was the best stress buster but also very addictive	0.25	Positive
The girl was gorgeous in that dress	0.75	Positive
I missed the train by 5 minutes	0.00	Neutral
The least favourite movie of marvel was Ironman	1.00	Positive
It was the worst day ever in my life	-0.50	Negative

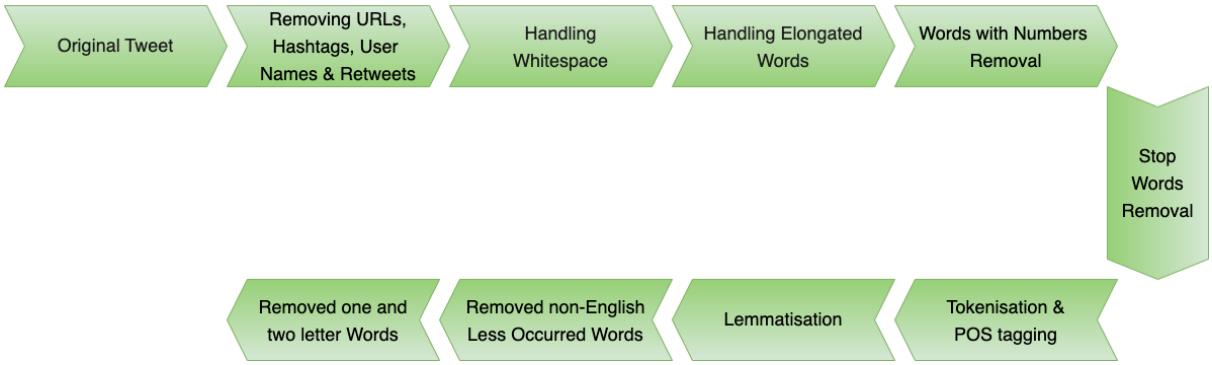


Figure 4.10. Pre-processing steps used for VADER implementation.

How the lexicon-based techniques TextBlob, VADER, and SentiWordNet performs was shown with some sample texts and their implementation in our dataset was discussed here. The results and the further discussions of these lexicon-based techniques were continued on chapter 5.

4.10 Hyperparameter tunning for ML Algorithms

After data balancing step, a total of 128 different combinations of datasets were available as shown in figure 4.11. Those datasets were fed into the 9 different ML algorithms which discussed in section 3.7. For the implementation of these ML algorithms, sklearn's ML models were used as shown in table 4.22. But each of these models had multiple parameters which need to be tunned so that we get best results out of it. So, GridSearchCV and RandomizedSearchCV were used to finetune these parameters.

GridSearchCV validates the given model's performance across all possible combinations of the given parameters grid. Let's say, the "cv" (cross validation split strategy) was given as five. Then the given dataset will split into 4:1 ratio. And the model will be trained upon the four parts of the dataset and validated on the hold out part. The same procedure will be repeated for five times and for each time, the hold out part will change accordingly as it covers the whole dataset.

RandomizedSearchCV performs almost same as GridSearchCV except it won't consider all possible combinations as GridSearchCV. Instead, random number of combinations of the given parameters distribution will be considered and used to train the model. The number will be given in "n_iter" parameter.

All the eight ML algorithms except XGBoost used GridSearchCV to finetune the parameters. Since XGBoost had multiple parameters with multiple values which need to be tuned, we ended up with almost 18,000 different combinations. So RandomizedSearchCV was implemented, and 200 random combinations of the given parameters distribution were chosen and tested. The parameters settings of all the nine different ML algorithms was given in table 4.22.

Table 4.22. Fixed and tunned parameters for nine different ML Algorithms.

ML technique	Model Used	Fixed Parameter	Tunned Parameter
Logistic Regression	LogisticRegression	random_state = 42	C: [100, 10, 5, 4, 3, 2, 1, 1.0, 0.1, 0.01]
Naïve Bayes	BernoulliNB	-	alpha: [0.5, 1.0, 1.5, 2.0, 5], fit_prior: [True, False]
	MultinomialNB	-	alpha: [0.5, 1.0, 1.5, 2.0, 5], fit_prior: [True, False]
Decision Trees	DecisionTreeClassifier	random_state = 42	max_depth: [2, 3, 5, 10, 20], min_samples_leaf: [5, 10, 20, 50, 100], criterion: ["gini", "entropy"], min_samples_split: [2, 3, 4]
Random Forest	RandomForestClassifier	oob_score = True, random_state = 42	max_depth: [2, 3, 5, 10, 20], min_samples_leaf: [5, 10, 20, 50, 100], criterion: ["gini", "entropy"], n_estimators: [10, 25, 50, 100]
Support Vector Machine	SVC	random_state = 42	C: [0.1,1,3,5,10,100], gamma: [0.0001, 0.001, 0.01, 0.1, 1]
K-Nearest Neighbour	KNeighborsClassifier	-	n_neighbors: 1 to 20
AdaBoost	xgboost	random_state = 42	n_estimators: [100, 200, 350, 500], learning_rate: [0.5, 0.8, 1.0]
XGBoost	AdaBoostClassifier	n_jobs = -1, random_state = 42	n_estimators: [100, 200, 500, 750], learning_rate: [0.01, 0.02, 0.05, 0.1, 0.25], min_child_weight: [1, 5, 7, 10], gamma: [0.1, 0.5, 1, 1.5, 5], subsample: [0.6, 0.8, 1.0], colsample_bytree: [0.6, 0.8, 1.0], max_depth: [3, 4, 5, 10, 12]

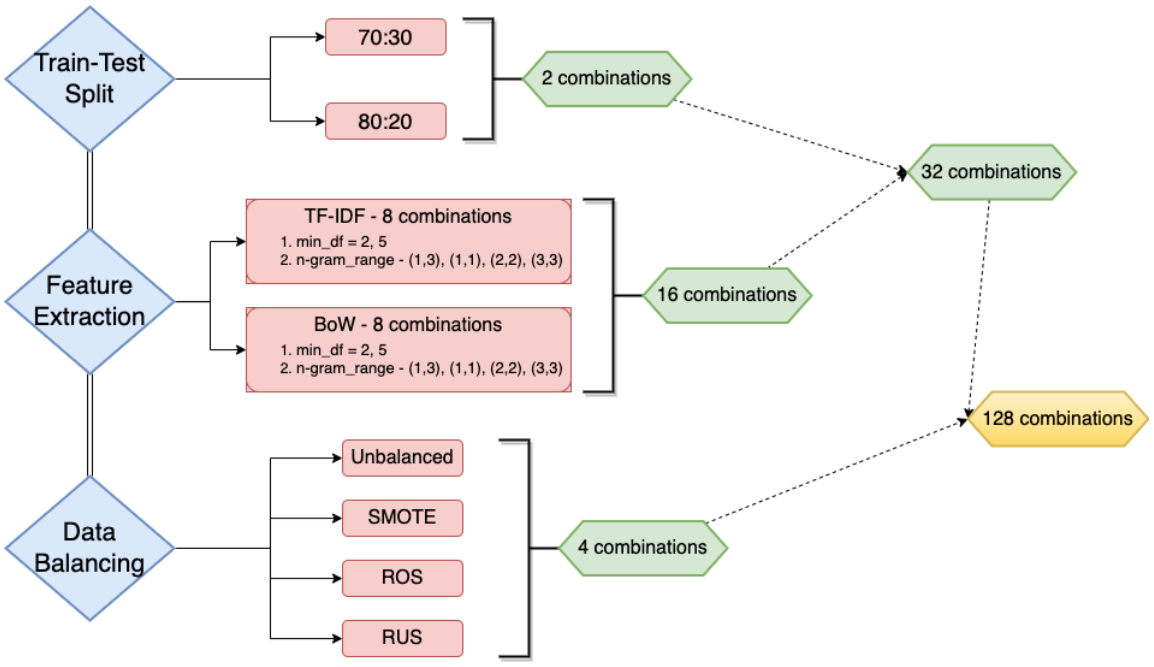


Figure 4.11. Total combinations of datasets after each step.

4.11 Summary

The implementation of the sentiment analysis was performed on python. The dataset used for our analysis had some unwanted columns (i.e., tweet id and entity) and rows (i.e., null values, duplicate values, outliers, and tweets with multiple sentiments). As shown in figure 4.1, the pre-processing steps were implemented, which ended up with 55,570 tweets in training dataset and 824 tweets in test dataset. The pre-processed tweets were analysed using N-grams and word clouds of each polarity of the "sentiment" field through data visualization. Since the size of the test dataset was very small in size when compared to training dataset, both the datasets were combined together. And then the combined dataset was split into train and test datasets again using sklearn's "train_test_split" method. The train-test split ratios used in this analysis were 70:30 and 80:20. The feature extraction was performed using two approaches i.e., TF-IDF and BoW approach. The imbalance in the dataset was handled using three different approaches i.e., SMOTE, ROS (Random Over Sampling), and RUS (Random Under Sampling). At the end of this step, we had 128 different combinations of datasets as shown in figure 4.11. These 128 combinations of datasets were used upon 9 different ML models as shown in table 4.22. On the other hand, the combined dataset was used for lexicon-based techniques i.e., TextBlob, VADER, and SentiWordNet. The results of these nine different ML models and three lexicon-based techniques were discussed in next chapter.

CHAPTER 5

RESULTS AND DISCUSSIONS

In this chapter, the results and outcomes of the techniques implemented for feature extraction, data balancing, and train-test split were discussed. And the best performing lexicon-based, and machine learning based models for the dataset of twitter sentiment analysis in hand was presented. In results tables and confusion matrix, 0, 1, 2 represent positive, negative, and neutral sentiments respectively.

5.1 Introduction

In this study, we developed and evaluated around 1152 models based on feature extraction, data balancing, and train-test split ratios. As shown in figure 4.11, we had 128 different combinations of datasets where we trained and evaluated using nine different ML algorithms viz. Logistic Regression (LR), Bernoulli Naïve Bayes (BNB), Multinomial Naïve Bayes (MNB), Decision Trees (DT), Random Forest (RF), Support Vector Classifier (SVC), K-Nearest Neighbour (KNN), eXtreme Gradient Boosting (XGBoost), and AdaBoost. On the other hand, the pre-processed tweets were used to train and evaluate the lexicon-based techniques viz. TextBlob, VADER, and SentiWordNet. The performance of the lexicon-based and ML based models were evaluated using accuracy, precision, recall, and F1-Score.

5.2 Results of Lexicon-based methods

For lexicon-based approach, we implemented three different techniques viz. TextBlob, VADER, and SentiWordNet. The pre-processed tweets used for ML techniques were used for TextBlob and SentiWordNet as well. But since VADER supports emojis, acronyms, punctuations etc., different pre-processing steps were used just for VADER implementation as shown in figure 4.10. But based on the results delivered, none of the three techniques performed well on our dataset when compared to the results of ML models. The results/outcomes of all three lexicon-based techniques were discussed in the upcoming subsections.

5.2.1 TextBlob Classification

As we discussed in section 4.9, TextBlob was implemented in two ways. In first way, TextBlob was directly implemented on the pre-processed tweets. In second way, spelling correction using "correct()" method of TextBlob was applied on the pre-processed tweets first and then the spell corrected tweets were classified using the TextBlob's polarity score. Going forward, the first way of implementation was mentioned as "TextBlob", and the second way was mentioned as "spell corrected TextBlob". The TextBlob's classification of some sample tweets from our dataset was shown in table 5.1.

Table 5.1. TextBlob classification of sample tweets from our dataset.

Lexicon Used	Pre-processed Tweet / Spell Corrected Tweet	TextBlob Score	TextBlob Sentiment	Original Label
TextBlob	can not wait play mercy moira junk overwatch week	0.00	Neutral	Positive
	beg viking theme assassin creed year excite fuckk	0.00	Neutral	Positive
	true hahaha useless gameplay	0.02	Positive	Negative
	aww yeah baby birthday	0.30	Positive	Neutral
	love gta san andreas red death redemption also still old playstation slim would love get please	0.28	Positive	Positive
	fifa must joke see horrible thing ift	-1.00	Negative	Negative
Spell Corrected TextBlob	can not wait play mercy more junk overwatch week	0.50	Positive	Positive
	beg viking theme assassin creed year excite fuck	-0.40	Negative	Positive
	true havana useless gameplay	-0.08	Negative	Negative
	www yeah baby birthday	0.00	Neutral	Neutral
	love ta san andrews red death redemption also still old plantation slim would love get please	0.28	Positive	Positive
	if must joke see horrible thing it	-1.00	Negative	Negative

In some cases, the spelling of the tweets were correctly changed but wrongly classified and in some cases, unnecessary words' spelling were changed incorrectly but classified correctly and vice versa. For example, the word "playstation" was replaced with "plantation" but classified correctly, since this word didn't contribute much for that tweet's sentiment. In another case, the word "fuckk" was replaced with "fuck" but wrongly classified. Since the original label was positive which itself is argumentative, but the corrected word is a strong negative word.

As we see in figure 5.2(b), TextBlob classified 42.8% of tweets as positive, 32.1% of tweets as negative, and 25% of tweets as neutral. When we compare this TextBlob classification with original label, the TextBlob's positive polarity was almost 6% higher, the negative polarity was 4% lesser, and the neutral polarity was just 1.2% lesser than the original label. On the other hand, as we see in figure 5.2(c), spell corrected TextBlob classified 43.4% of tweets as positive, 32% of tweets as negative, and 24.6% of tweets as neutral. Hence, when the classification of TextBlob and spell corrected TextBlob were compared, both of them had almost similar classification which shows that spelling correction didn't work well in our dataset.

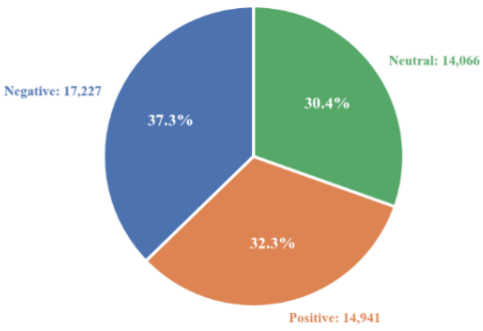
5.2.2 VADER Classification

As we discussed in section 4.9, two VADER variations were implemented i.e., VADER with emojis and VADER without emojis. But both the VADER variations gave similar classification. Out of 46k tweets, just only 1% i.e., 450 tweets was differently classified, which conveys that emojis didn't played major role in our dataset for the sentiment classification. Some of the samples from the dataset with their respective VADER sentiment and compound score with and without the emojis were shown in table 5.2.

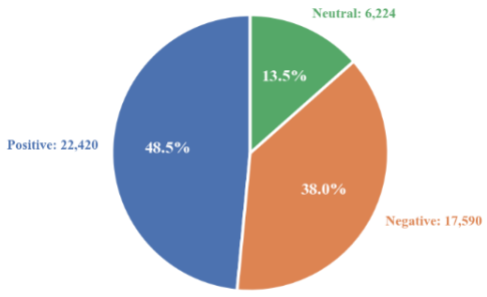
Table 5.2. VADER classification with and without the emojis of sample tweets from our dataset.

Lexicon Used	Tweet	Pos Score	Neg Score	Neu Score	Compound Score	VADER Sentiment
VADER without Emojis	This one bad madden year shit hilarious	0.196	0.514	0.290	-0.6597	Negative
	I get follow also please I buy almost worth much I love game	0.448	0.000	0.552	0.7968	Positive
	Assassins creed odyssey probably favorite series	0.375	0.000	0.625	0.4588	Positive
	I really make Amaru rage quit	0.000	0.436	0.564	-0.5945	Negative
	youtu GLITCH CALL OF DUTY MOBILE PLAY THE GAME LOST NOOBSTERS Full vid On YouTube link provide	0.148	0.143	0.709	0.0258	Neutral
VADER with Emojis	This one bad madden year shit hilarious 🤣🤣	0.323	0.252	0.426	0.2500	Positive
	I get follow also please 😍😍 I buy almost worth much I love game	0.587	0.000	0.413	0.9709	Positive
	Assassins creed odyssey 🔥🔥 probably favorite series	0.234	0.375	0.391	-0.2023	Negative
	I really make Amaru rage quit 🤣	0.199	0.241	0.560	-0.1689	Negative
	youtu GLITCH CALL OF DUTY MOBILE PLAY 🏴‍☠️ THE GAME LOST NOOBSTERS Full vid On YouTube link provide	0.135	0.131	0.734	0.0258	Neutral

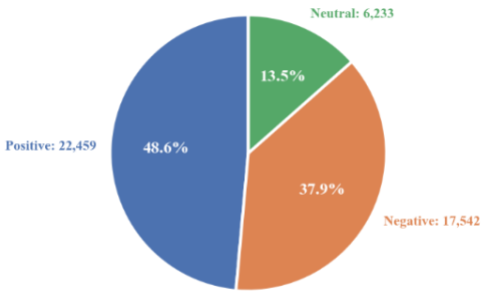
As we see in figure 5.1(b), VADER with emojis classified 48.5% of tweets as positive, 38% of tweets as negative, and 13.5% of tweets as neutral. When we compare this classification with the original label, VADER with emojis' positive polarity was almost 11% higher, the negative polarity was 17% lesser, and the neutral polarity was almost 6% higher than the original label. On the other hand, VADER without emojis' neutral polarity was exactly same as the VADER with emojis' neutral polarity. And the positive and negative polarities of VADER with and without emojis had just 0.1% difference which reconfirms that both the VADER variations had similar classification. And both of them, not performed well on our dataset as it highly differs from the original classification.



(a) Original Label



(b) VADER with Emojis



(c) VADER without Emojis

Figure 5.1. VADER classification with and without the Emojis.

5.2.3 SentiWordNet Classification

Similarly, SentiWordNet was applied on the same pre-processed tweets. As we discussed in section 4.9, the total positive scores and total negative scores of the words' synsets were used to classify the tweets as positive, negative, or neutral. The SentiWordNet score and classification of some of the samples from our dataset with its corresponding original label was shown in table 5.3. As a result of SentiWordNet classification, out of 56,394 pre-processed tweets, 41.6% of tweets were classified as positive, 34.1% of tweets as negative, and 24.2% of tweets as neutral. The figure 5.2(d) shows the SentiWordNet classification and the frequency of each polarity. And when we compare this with the original label shown in figure 5.2(a), the SentiWordNet positive polarity was almost 4.5% higher, the negative polarity was 5.6% lesser, and the neutral polarity was just 0.8% higher than the original label.

Table 5.3. SentiWordNet classification of sample tweets from our dataset.

Tweet	SentiWordNet Score	SentiWordNet Sentiment	Original Label
maybe great musical moment video game ever	0.250	Positive	Positive
holy shit price much low expect faster	-0.125	Negative	Positive
never understand people enjoy overwatch	-0.625	Negative	Negative
india ban chinese apps include pubg	0.000	Neutral	Negative
johnson johnson stop sell talc base baby powder canada johnson johnso	0.000	Neutral	Neutral
microsoft surface headphone review perfect work home life lnkd	0.625	Positive	Neutral

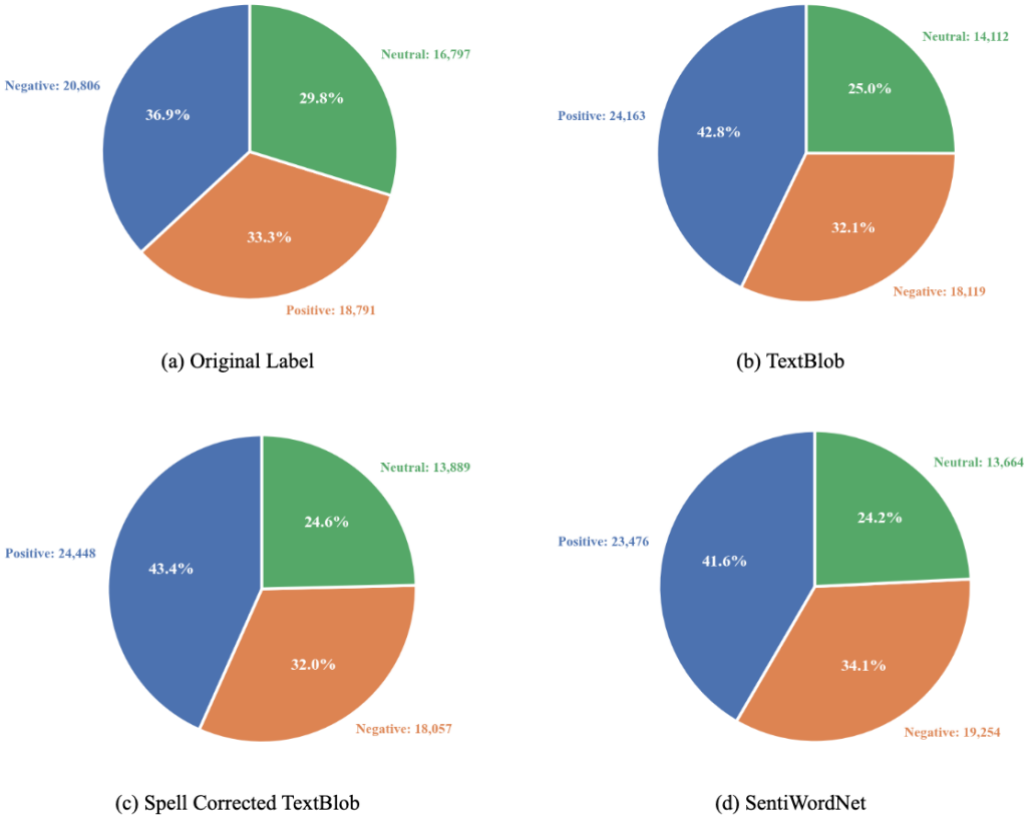


Figure 5.2. TextBlob, spell corrected TextBlob, and SentiWordNet classification of our pre-processed tweets.

5.2.4 TextBlob vs VADER vs SentiWordNet

All the three lexicon-based techniques had considerable variation between them. Table 5.4 shows some of the disparities between the three techniques. when we compare TextBlob with VADER, TextBlob‘s positive polarity was almost 5.3% lesser, the negative polarity was also 5% lesser, and the neutral polarity was 11% higher than VADER. TextBlob and SentiWordNet didn’t had much difference in their classification. They had almost 1%, 2%, and 0.8% difference in positive, negative, and neutral polarities respectively. Hence, SentiWordNet and VADER classification difference was almost same as TextBlob and VADER classification difference.

Table 5.4. TextBlob vs VADER vs SentiWordNet annotation disparity.

Original Label	Tweet	TextBlob Sentiment	VADER Sentiment	SentiWordNet Sentiment
Positive	come borderland murder	Neutral	Negative	Negative
Positive	come meet one beautiful god gamble	Positive	Positive	Neutral
Negative	version look like late zombie	Negative	Positive	Neutral
Negative	amazon really make dirty yes okay get	Positive	Positive	Positive
Neutral	time cyberpunk remake bit ly	Neutral	Positive	Negative
Neutral	anyone expect little disappointed schedule wholeheartedly endorse decision	Negative	Negative	Negative

As we see in table 5.5, all the three lexicons with five variations didn’t perform well in our dataset as their accuracies was not even 50%. And out of the three polarities, neutral polarity was very badly predicted by the lexicons with the precision, recall, and f1-score of less than 35%. Both the VADER variations delivered similar results and on the other hand, both the TextBlob variations also delivered similar results. So, we can assure that, neither spelling correction nor emojis played major role in sentiment classification in our dataset.

VADER had the best accuracy i.e., 48%. Then TextBlob and SentiWordNet had 47% and 42% accuracy respectively. On the other hand, TextBlob had the best f1-score and precision i.e., 46%. Then VADER and SentiWordNet had 44% and 41% f1-score; 45% and 41% precision respectively.

So, we can conclude that VADER had the best accuracy and TextBlob had the best f1-score. But both of them was less than 50% may be because of the dataset in hand was very noisy and had too many slang words, acronyms and negations. In future, some of the advanced pre-processing techniques will be analysed to handle these slang words, acronyms, and negations etc., which might help us to improve the performance of the lexicon-based techniques. So, since the lexicon-based techniques didn’t deliver the expected performance. For ML classifiers, just the original label given was considered as our target variable to train the ML models.

Table 5.5. Evaluation metrics of lexicon-based techniques.

Lexicon Used	Accuracy	Precision				Recall				F1-Score			
		Neg	Neu	Pos	Macro	Neg	Neu	Pos	Macro	Neg	Neu	Pos	Macro
TextBlob	47.299	0.56	0.34	0.49	0.46	0.49	0.29	0.62	0.47	0.52	0.31	0.55	0.46
Spell Corrected TextBlob	47.297	0.56	0.34	0.48	0.46	0.48	0.28	0.63	0.47	0.52	0.31	0.55	0.46
VADER with Emojis	47.893	0.55	0.34	0.46	0.45	0.56	0.15	0.69	0.47	0.56	0.21	0.55	0.44
VADER without Emojis	47.975	0.55	0.34	0.46	0.45	0.56	0.15	0.69	0.47	0.56	0.21	0.55	0.44
SentiWordNet	42.213	0.49	0.31	0.43	0.41	0.46	0.25	0.54	0.41	0.48	0.28	0.48	0.41

5.3 Results based on Feature Extraction Techniques Combinations

For feature extraction, TF-IDF and BoW approach were implemented (going forward in this study, to refer to BoW approach, BoW and CountVectorizer was alternatively used). As discussed in section 4.7, each of the feature extraction techniques was experimented with eight different combinations i.e., each ngram_range: [(1,3),(1,1),(2,2),(3,3)] was combined with each of the min_df values 2 and 5. And in this section, we tried to figure out the best feature extraction combination based on the accuracy of the ML models experimented on it.

Table 5.6 and 5.7 shows the TF-IDF and BoW results of nine different classifiers on the unbalanced dataset of 70:30 and 80:20 train-test split ratios respectively. In all the cases, Unigrams and all three N-grams combination i.e., ngram_range of (1,1) and (1,3) performed better than the Bigrams and Trigrams i.e., ngram_range of (2,2) and (3,3). When we compare TF-IDF and CountVectorizer with their respective combinations, MNB, SVC, KNN, and AdaBoost had better performance with TF-IDF vectorizer. On the other hand, BNB had better performance with CountVectorizer but not with big difference.

When we compare all the eight combinations of TF-IDF with each other, LR, BNB, MNB, and SVC had better performance with ngram_range of (1,3) and min_df of 2. And has mentioned earlier, all the nine ML models with this combination outperformed, when they were combined with Bigrams and Trigrams. RF had best performance with ngram_range of (1,1) and min_df of 5. XGBoost had best performance with ngram_range of (1,3) and min_df of 5. KNN and AdaBoost with 70:30 split ratio had the best performance with ngram_range of (1,1) and min_df of 5.

When we compare all the eight combinations of CountVectorizer with each other, LR, BNB, MNB, DT, and XGBoost had better performance with ngram_range of (1,3) and min_df of 2. RF and AdaBoost with 80:20 split ratio, KNN with 70:30 split ratio, and SVC had best performance with ngram_range of (1,1) and min_df of 5. KNN with 80:20 split ratio, RF and AdaBoost with 70:30 split ratio had the best performance with ngram_range of (1,1) and min_df of 2.

To conclude, out of the eight different combinations of TF-IDF and BoW approach, four of them were consistently performed better i.e., ngram_range: [(1,3),(1,1)] combined with min_df values 2 and 5. ML models LR, MNB, SVC and AdaBoost with TF-IDF (ngram_range of (1,3) and min_df of 2) outperformed all the other combinations of both TF-IDF and BoW. Similarly, in balanced datasets as well, these four combinations performed better. So, for further analysis only these four combinations of features will be used i.e., 128 combinations mentioned earlier was reduced to 64 combinations.

Table 5.6. Accuracies of classifiers on different TF-IDF combinations in the unbalanced dataset of 70:30 and 80:20 train-test split ratios.

Split Ratio	Classifier	min_df = 2, N-grams = (1,3)	min_df = 2, N-grams = (1,1)	min_df = 2, N-grams = (2,2)	min_df = 2, N-grams = (3,3)	min_df = 5, N-grams = (1,3)	min_df = 5, N-grams = (1,1)	min_df = 5, N-grams = (2,2)	min_df = 5, N-grams = (3,3)
70:30	LR	92.192	79.774	88.250	77.292	85.738	77.664	68.674	45.369
	BNB	87.765	74.177	86.335	76.772	75.607	71.901	63.520	44.896
	MNB	90.685	75.371	87.700	77.061	78.580	73.296	65.630	45.056
	DT	53.579	53.679	41.575	39.281	53.502	53.703	42.112	39.157
	RF	51.859	59.330	42.869	39.559	57.586	59.507	43.360	40.162
	SVC	93.516	93.676	88.894	77.126	93.232	93.416	71.635	45.700
	XGBoost	83.297	83.628	68.792	58.751	84.036	83.569	64.667	45.044
	KNN	90.390	93.097	86.890	75.312	92.287	93.250	70.282	44.855
	AdaBoost	66.499	66.582	49.028	42.166	66.493	66.682	49.607	42.207
80:20	LR	92.854	80.406	89.476	79.502	87.712	78.243	71.877	46.768
	BNB	88.962	74.741	87.880	79.582	77.276	72.701	65.467	46.112
	MNB	91.879	75.627	89.175	79.785	80.184	73.863	68.109	46.298
	DT	54.429	53.099	41.369	38.833	54.358	53.480	41.688	39.011
	RF	49.543	59.855	42.131	39.436	57.337	59.926	42.380	39.587
	SVC	94.060	94.033	90.017	79.927	93.652	93.741	73.996	47.008
	XGBoost	85.061	84.839	69.324	61.761	85.327	84.804	66.876	46.263
	KNN	91.879	94.370	88.164	78.030	93.093	94.281	73.163	38.940
	AdaBoost	66.274	66.034	48.985	41.972	66.566	66.513	49.339	42.078

Table 5.7. Accuracies of classifiers on different CountVectorizer combinations in the unbalanced dataset of 70:30 and 80:20 train-test split ratios.

Split Ratio	Classifier	min_df = 2, N-grams = (1,3)	min_df = 2, N-grams = (1,1)	min_df = 2, N-grams = (2,2)	min_df = 2, N-grams = (3,3)	min_df = 5, N-grams = (1,3)	min_df = 5, N-grams = (1,1)	min_df = 5, N-grams = (2,2)	min_df = 5, N-grams = (3,3)
70:30	LR	90.514	80.537	87.659	77.026	85.832	78.326	68.928	45.328
	BNB	87.765	74.177	86.335	76.772	75.607	71.901	63.520	44.896
	MNB	89.627	74.100	86.731	76.736	76.311	72.132	63.644	44.861
	DT	54.714	54.666	41.941	39.080	54.649	54.678	41.935	39.122
	RF	51.351	59.466	42.225	39.228	56.806	59.294	43.425	40.079
	SVC	61.404	74.413	74.620	68.326	69.886	75.040	69.218	45.340
	XGBoost	83.793	83.202	68.952	58.798	83.415	83.220	63.177	44.760
	KNN	87.340	91.075	79.674	68.272	89.988	91.199	67.025	44.447
	AdaBoost	65.630	65.784	48.975	42.018	65.678	65.749	48.839	42.012
80:20	LR	91.320	81.036	88.740	79.502	87.260	79.191	71.948	46.830
	BNB	88.962	74.741	87.880	79.582	77.276	72.701	65.467	46.112
	MNB	90.779	74.439	88.288	79.537	77.773	72.568	65.742	46.121
	DT	53.905	53.719	41.466	38.869	53.843	53.710	41.458	38.780
	RF	47.690	59.651	42.256	39.516	55.262	59.934	42.451	39.569
	SVC	61.646	75.290	75.131	70.751	70.343	75.822	71.345	46.582
	XGBoost	85.398	84.609	70.086	61.770	84.662	84.600	65.626	45.979
	KNN	88.953	92.739	81.568	69.643	91.187	92.597	69.793	38.195
	AdaBoost	65.440	65.529	48.710	41.848	65.520	65.688	48.728	41.750

5.4 Results based on Data Balancing Techniques

For data balancing, SMOTE, ROS, and RUS were implemented. With these three balanced datasets and the original unbalanced dataset, we have four in total. In this section, we tried to find the best one out of them. The performance of these four datasets were evaluated using the accuracy of the ML models prediction on them. In our hand, we had four different feature combinations for both TF-IDF and BoW. Then two different train-test split ratios. Hence in total, we had 16 different combinations on which these four data sets' performances were evaluated.

Table 5.8 and 5.9 shows the results of nine ML models on four different datasets i.e., unbalanced, SMOTE, ROS, and RUS with TF-IDF's and BoW's feature combinations (min_df= 2 and ngram_range = [(1,3),(1,1)]) in 70:30 and 80:20 split ratios. In most of the cases, RUS was the least performed data balancing technique when compared with SMOTE, ROS and Unbalanced dataset. This may be because of the dataset in hand was just slightly unbalanced and RUS under samples and reduces the size of the dataset, the performance might be reduced. The TF-IDF's and BoW's feature combinations with min_df of 5 also delivered the similar performance.

SMOTE, ROS, and Unbalanced datasets were alternatively performed well in each of the classifiers on different combinations. So, further experiments were made on these three datasets. Hence, the 64 combinations was reduced to 48 combinations.

Table 5.8. Accuracies of classifiers on the unbalanced and balanced datasets using SMOTE, ROS, and RUS balancing techniques of 70:30 and 80:20 train-test split ratios on TF-IDF features with min_df = 2.

Split Ratio	Classifier	N-grams = (1,3)				N-grams = (1,1)			
		Unbalanced	SMOTE	ROS	RUS	Unbalanced	SMOTE	ROS	RUS
70:30	LR	92.192	92.080	92.257	91.578	79.774	79.638	79.603	79.372
	BNB	87.765	87.440	87.482	87.139	74.177	73.929	74.106	73.805
	MNB	90.685	90.655	90.561	90.265	75.371	75.229	75.471	75.140
	DT	53.579	51.209	51.090	50.884	53.679	50.955	50.712	50.334
	RF	51.859	63.520	63.804	63.763	59.330	63.071	63.243	63.792
	SVC	93.516	92.659	93.516	93.037	93.676	92.500	93.688	93.120
	XGBoost	83.297	84.012	84.006	82.393	83.628	83.983	83.983	82.576
	KNN	90.390	90.573	90.390	89.497	93.097	93.097	93.097	92.322
	AdaBoost	66.499	65.719	65.636	65.489	66.582	65.719	65.554	65.536
80:20	LR	92.854	92.641	93.076	92.473	80.406	80.601	80.051	79.848
	BNB	88.962	88.439	88.625	87.951	74.741	74.448	74.758	74.554
	MNB	91.879	91.852	92.021	91.347	75.627	75.911	76.009	75.539
	DT	54.429	51.512	51.201	51.130	53.099	51.600	51.352	51.042
	RF	49.543	64.075	64.226	63.942	59.855	63.933	64.066	63.153
	SVC	94.060	93.546	94.157	93.803	94.033	93.315	94.069	93.723
	XGBoost	85.061	85.389	85.451	83.899	84.839	85.274	85.584	83.491
	KNN	91.879	91.790	91.879	90.416	94.370	94.246	94.370	93.324
	AdaBoost	66.274	65.786	65.360	65.254	66.034	65.751	65.981	65.910

Table 5.9. Accuracies of classifiers on the unbalanced and balanced datasets using SMOTE, ROS, and RUS balancing techniques of 70:30 and 80:20 train-test split ratios on CountVectorizer features with min_df = 2.

Split Ratio	Classifier	N-grams = (1,3)				N-grams = (1,1)			
		Unbalanced	SMOTE	ROS	RUS	Unbalanced	SMOTE	ROS	RUS
70:30	LR	90.514	89.887	90.561	90.200	80.537	80.235	80.460	80.046
	BNB	87.765	87.263	87.482	87.139	74.177	74.053	74.106	73.805
	MNB	89.627	89.586	89.462	89.278	74.100	74.230	74.171	74.006
	DT	54.714	51.475	51.244	50.570	54.666	51.646	51.197	50.476
	RF	51.351	63.520	63.591	63.124	59.466	62.882	63.970	63.266
	SVC	61.404	63.154	61.422	61.930	74.413	76.458	74.396	74.803
	XGBoost	83.793	84.532	84.444	82.659	83.202	83.900	83.693	82.162
	KNN	87.340	87.411	87.340	85.898	91.075	91.063	91.075	89.905
	AdaBoost	65.630	64.791	65.140	65.412	65.784	64.827	65.358	65.453
80:20	LR	91.320	90.877	91.338	90.930	80.920	80.920	80.876	80.610
	BNB	88.962	88.270	88.625	87.951	74.439	74.439	74.758	74.554
	MNB	90.779	90.762	90.957	90.318	74.359	74.359	74.546	74.253
	DT	53.905	51.654	51.538	51.361	51.618	51.618	51.494	51.370
	RF	47.690	63.906	63.924	62.851	61.965	61.965	63.756	63.321
	SVC	61.646	63.366	61.761	63.126	77.675	77.675	75.335	77.445
	XGBoost	85.398	86.142	85.903	84.059	85.229	85.229	85.318	83.775
	KNN	88.953	89.272	88.953	86.453	92.721	92.721	92.739	90.788
	AdaBoost	65.440	64.181	65.334	65.538	65.618	65.618	65.493	65.130

5.5 Results based on Train-Test Split Ratios

As mentioned in section 4.6, 70:30 and 80:20 train-test split ratios were used in this study. In this section, we tried to find the best performing split for each of the ML models. And to evaluate the performance of them, 24 different combinations of datasets were used i.e., four different feature combinations for both TF-IDF and BoW on SMOTE, ROS and Unbalanced datasets.

Table 5.10 shows the accuracy of nine ML models on unbalanced dataset. When 70:30 and 80:20 split ratios were compared with their respective feature combination. ML classifiers LR, BNB, MNB, SVC, XGBoost, and KNN had better performance with 80:20 split ratio. The other ML classifiers DT, RF, and AdaBoost had better performances in 70:30 and 80:20 split ratios alternatively, may be because, all the three classifiers didn't have higher accuracies in both the train-test split ratios. And hence, there was just slight difference in the accuracy of 70:30 and 80:20 split ratios.

The observations found on the table 5.10 i.e., on unbalanced dataset was exactly observed on both the SMOTE and ROS datasets as well, which was shown on tables 5.11 and 5.12. To summarize, the six ML classifiers LR, BNB, MNB, SVC, XGBoost, and KNN had better performance with 80:20 split ratio. And the other three classifiers DT, RF, and AdaBoost didn't have better performances on both 70:30 and 80:20 split ratios.

Table 5.10. Comparing accuracies of classifiers on the unbalanced dataset of 70:30 and 80:20 train-test split ratios.

Vectorizer	Classifier	70:30 ratio				80:20 ratio			
		min_df = 2, N-grams = (1,3)	min_df = 2, N-grams = (1,1)	min_df = 5, N-grams = (1,3)	min_df = 5, N-grams = (1,1)	min_df = 2, N-grams = (1,3)	min_df = 2, N-grams = (1,1)	min_df = 5, N-grams = (1,3)	min_df = 5, N-grams = (1,1)
TF-IDF	LR	92.192	79.774	85.738	77.664	92.854	80.406	87.712	78.243
	BNB	87.765	74.177	75.607	71.901	88.962	74.741	77.276	72.701
	MNB	90.685	75.371	78.580	73.296	91.879	75.627	80.184	73.863
	DT	53.579	53.679	53.502	53.703	54.429	53.099	54.358	53.480
	RF	51.859	59.330	57.586	59.507	49.543	59.855	57.337	59.926
	SVC	93.516	93.676	93.232	93.416	94.060	94.033	93.652	93.741
	XGBoost	83.297	83.628	84.036	83.569	85.061	84.839	85.327	84.804
	KNN	90.390	93.097	92.287	93.250	91.879	94.370	93.093	94.281
	AdaBoost	66.499	66.582	66.493	66.682	66.274	66.034	66.566	66.513
BoW	LR	90.514	80.537	85.832	78.326	91.320	81.036	87.260	79.191
	BNB	87.765	74.177	75.607	71.901	88.962	74.741	77.276	72.701
	MNB	89.627	74.100	76.311	72.132	90.779	74.439	77.773	72.568
	DT	54.714	54.666	54.649	54.678	53.905	53.719	53.843	53.710
	RF	51.351	59.466	56.806	59.294	47.690	59.651	55.262	59.934
	SVC	61.404	74.413	69.886	75.040	61.646	75.290	70.343	75.822
	XGBoost	83.793	83.202	83.415	83.220	85.398	84.609	84.662	84.600
	KNN	87.340	91.075	89.988	91.199	88.953	92.739	91.187	92.597
	AdaBoost	65.630	65.784	65.678	65.749	65.440	65.529	65.520	65.688

Table 5.11. Comparing accuracies of classifiers on the SMOTE dataset of 70:30 and 80:20 train-test split ratios.

Vectorizer	Classifier	70:30 ratio				80:20 ratio			
		min_df = 2, N-grams = (1,3)	min_df = 2, N-grams = (1,1)	min_df = 5, N-grams = (1,3)	min_df = 5, N-grams = (1,1)	min_df = 2, N-grams = (1,3)	min_df = 2, N-grams = (1,1)	min_df = 5, N-grams = (1,3)	min_df = 5, N-grams = (1,1)
TF-IDF	LR	92.080	79.638	85.466	77.617	92.641	80.601	87.180	77.959
	BNB	87.440	73.929	75.714	71.588	88.439	74.448	77.134	72.506
	MNB	90.655	75.229	78.533	73.225	91.852	75.911	80.113	74.067
	DT	51.209	50.955	51.587	50.381	51.512	51.600	52.026	51.529
	RF	63.520	63.071	63.999	63.432	64.075	63.933	63.906	64.385
	SVC	92.659	92.500	92.245	92.192	93.546	93.315	93.049	93.173
	XGBoost	84.012	83.983	84.396	83.752	85.389	85.274	85.770	85.176
	KNN	90.573	93.097	92.222	93.227	91.790	94.246	92.863	94.148
	AdaBoost	65.719	65.719	66.003	66.032	65.786	65.751	66.025	66.034
BoW	LR	89.887	80.235	85.431	78.344	90.877	80.920	87.100	79.165
	BNB	87.263	74.053	75.619	71.984	88.270	74.439	76.939	72.675
	MNB	89.586	74.230	76.151	72.043	90.762	74.359	77.729	72.524
	DT	51.475	51.646	51.528	51.640	51.654	51.618	51.645	51.547
	RF	63.520	62.882	62.711	62.545	63.906	61.965	62.816	63.153
	SVC	63.154	76.458	72.238	76.984	63.366	77.675	72.897	78.234
	XGBoost	84.532	83.900	83.823	83.598	86.142	85.229	85.185	84.883
	KNN	87.411	91.063	89.999	91.288	89.272	92.721	91.285	92.659
	AdaBoost	64.791	64.827	65.093	64.685	64.181	65.618	64.509	65.564

Table 5.12. Comparing accuracies of classifiers on the ROS dataset of 70:30 and 80:20 train-test split ratios.

Vectorizer	Classifier	70:30 ratio				80:20 ratio			
		min_df = 2, N-grams = (1,3)	min_df = 2, N-grams = (1,1)	min_df = 5, N-grams = (1,3)	min_df = 5, N-grams = (1,1)	min_df = 2, N-grams = (1,3)	min_df = 2, N-grams = (1,1)	min_df = 5, N-grams = (1,3)	min_df = 5, N-grams = (1,1)
TF-IDF	LR	92.257	79.603	85.318	77.174	93.076	80.051	87.162	78.349
	BNB	87.482	74.106	75.832	71.972	88.625	74.758	77.276	72.781
	MNB	90.561	75.471	78.427	73.267	92.021	76.009	80.690	74.111
	DT	51.090	50.712	51.652	50.807	51.201	51.352	51.503	51.370
	RF	63.804	63.243	64.496	64.230	64.226	64.066	64.084	64.057
	SVC	93.516	93.688	93.221	93.428	94.157	94.069	93.679	93.732
	XGBoost	84.006	83.983	84.497	83.675	85.451	85.584	85.797	85.344
	KNN	90.390	93.097	92.287	93.250	91.879	94.370	93.093	94.281
	AdaBoost	65.636	65.554	65.861	65.784	65.360	65.981	65.875	65.892
BoW	LR	90.561	80.460	85.673	78.149	91.338	80.876	87.410	78.917
	BNB	87.482	74.106	75.832	71.972	88.625	74.758	77.276	72.781
	MNB	89.462	74.171	76.187	72.031	90.957	74.546	78.136	72.693
	DT	51.244	51.197	51.256	51.185	51.538	51.494	51.538	51.538
	RF	63.591	63.970	63.420	63.869	63.924	63.756	63.268	63.516
	SVC	61.422	74.396	69.921	74.993	61.761	75.335	70.379	75.813
	XGBoost	84.444	83.693	83.959	83.758	85.903	85.318	85.531	85.016
	KNN	87.340	91.075	89.988	91.199	88.953	92.739	91.187	92.597
	AdaBoost	65.140	65.358	65.518	65.400	65.334	65.493	65.414	65.644

5.6 Best Performing ML Models

In this study, we experimented with nine ML classifiers viz. LR, BNB, MNB, DT, RF, SVC, KNN, XGBoost, and AdaBoost. These nine ML classifiers were experimented with 128 different combinations. Out of that, top 3 combinations of each ML models based on accuracy was given in table 5.13, which once again reconfirms the observations made in previous sections i.e.,

1. 80:20 split ratio had better performance for LR, BNB, MNB, SVC, KNN, and XGBoost.
2. DT, RF, and AdaBoost didn't performed well on our dataset.
3. Unbalanced, SMOTE, and ROS outperformed RUS.
4. TF-IDF and BoW feature combinations - ngram_range of (1,3) and (1,1) with min_df of 2 and 5 performed better than the other bigrams and trigrams combinations.

Table 5.13. Top 3 combinations of all nine classifiers based on accuracy.

Classifier	Top 1		Top 2		Top 3	
	Combination	Accuracy	Combination	Accuracy	Combination	Accuracy
LR	80:20 + ROS + TF-IDF (min_df=2, N-grams=(1,3))	93.076	80:20 + Unbalanced + TF-IDF (min_df=2, N-grams=(1,3))	92.854	80:20 + SMOTE + TF-IDF (min_df=2, N-grams=(1,3))	92.641
BNB	80:20 + Unbalanced + TF-IDF (min_df=2, N-grams=(1,3))	88.962	80:20 + Unbalanced + BoW (min_df=2, N-grams=(1,3))	88.962	80:20 + ROS + BoW (min_df=2, N-grams=(1,3))	88.625
MNB	80:20 + ROS + TF-IDF (min_df=2, N-grams=(1,3))	92.021	80:20 + Unbalanced + TF-IDF (min_df=2, N-grams=(1,3))	91.879	80:20 + SMOTE + TF-IDF (min_df=2, N-grams=(1,3))	91.852
DT	70:30 + Unbalanced + BoW (min_df=2, N-grams=(1,3))	54.714	70:30 + Unbalanced + BoW (min_df=5, N-grams=(1,1))	54.678	70:30 + Unbalanced + BoW (min_df=2, N-grams=(1,1))	54.666
RF	70:30 + ROS + TF-IDF (min_df=5, N-grams=(1,3))	64.496	70:30 + RUS + TF-IDF (min_df=5, N-grams=(1,3))	64.490	80:20 + SMOTE + TF-IDF (min_df=5, N-grams=(1,1))	64.385
SVC	80:20 + ROS + TF-IDF (min_df=2, N-grams=(1,3))	94.157	80:20 + ROS + TF-IDF (min_df=2, N-grams=(1,1))	94.069	80:20 + Unbalanced + TF-IDF (min_df=2, N-grams=(1,3))	94.060
XGBoost	80:20 + SMOTE + BoW (min_df=2, N-grams=(1,3))	86.142	80:20 + ROS + BoW (min_df=2, N-grams=(1,3))	85.903	80:20 + ROS + TF-IDF (min_df=5, N-grams=(1,3))	85.797

KNN	80:20 + ROS + TF-IDF (min_df=2, N-grams=(1,1))	94.370	80:20 + Unbalanced + TF-IDF (min_df=2, N-grams=(1,1))	94.370	80:20 + ROS + TF-IDF (min_df=5, N-grams=(1,1))	94.281
AdaBoost	70:30 + Unbalanced + TF-IDF (min_df=5, N-grams=(1,1))	66.682	70:30 + Unbalanced + TF-IDF (min_df=2, N-grams=(1,1))	66.582	80:20 + Unbalanced + TF-IDF (min_df=5, N-grams =(1,3))	66.566

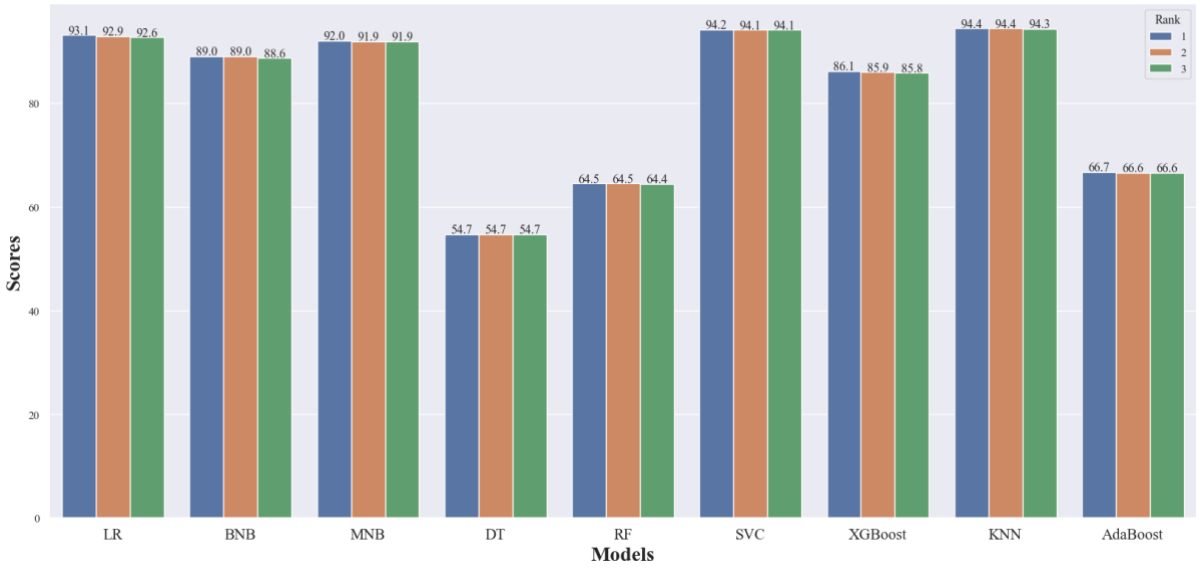


Figure 5.3. Accuracy of top three combinations of all 9 ML classifiers.

As shown in figure 5.3, KNN and SVC outperformed all the other classifiers. And all the nine ML models were finetuned with their parameters as shown in table 4.21. From that, the best performed parameters of each ML models were shown in table 5.14. Along with that, the best results produced by each of the nine model were shown in table 5.15.

Based on accuracy, KNN and SVC outperformed other ML classifiers with the accuracy of 94.37% and 94.16%. Next in line was LR and MNB with the accuracy of 93.08% and 92.02%. And for f1-score as well, KNN and SVC had the best performance with the f1-score of 94.37% and 94.10%. Again, next in line was LR and MNB with the f1-score of 93.04% and 91.96%. Even for precision and recall, KNN and SVC are the best performing models with the precision of 94.47% and 94.19% and with the recall of 94.36% and 94.04%. DT, RF, and AdaBoost were the least performed models with the accuracy of 54.71%, 64.50%, and 66.68%. The accuracy, precision, recall, and f1-score of the best performing combination of all the nine classifiers were shown in figure 5.4.

Table 5.14. Best parameters combinations of all 9 ML classifiers based on accuracy.

Classifier	Best Parameters
LR	C= 10, random_state= 42
BNB	alpha= 0.5, fit_prior= False
MNB	alpha= 0.5, fit_prior= False
DT	max_depth=20, min_samples_leaf=5, criterion= 'entropy', min_samples_split= 2, random_state=42
RF	max_depth=20, min_samples_leaf=5, criterion= 'gini', n_estimators': 100, oob_score=True, random_state=42
SVC	C=100, gamma=1, random_state=42
XGBoost	n_estimators= 750, learning_rate= 0.25, min_child_weight= 1, gamma= 1.5, subsample= 0.8, colsample_bytree= 0.8, max_depth= 10, n_jobs= -1, random_state= 42.
KNN	n_neighbors=1
AdaBoost	learning_rate=0.5, n_estimators=500, random_state=42

Table 5.15. The best result produced by all 9 classifiers.

Classifier	Combination	Accuracy	Class	Precision	Recall	F1-Score
LR	80:20 + ROS + TF-IDF (min_df=2, N-grams=(1,3))	93.076	0	0.93	0.92	0.93
			1	0.93	0.94	0.94
			2	0.93	0.92	0.93
			Macro avg	0.93	0.93	0.93
			Weighted avg	0.93	0.93	0.93
BNB	80:20 + Unbalanced + TF-IDF (min_df=2, N-grams=(1,3))	88.962	0	0.80	0.97	0.87
			1	0.94	0.90	0.92
			2	0.98	0.79	0.88
			Macro avg	0.90	0.89	0.89
			Weighted avg	0.90	0.89	0.89
MNB	80:20 + ROS + TF-IDF (min_df=2, N-grams=(1,3))	92.021	0	0.91	0.93	0.92
			1	0.94	0.92	0.93
			2	0.92	0.90	0.91
			Macro avg	0.92	0.92	0.92
			Weighted avg	0.92	0.92	0.92
DT	70:30 + Unbalanced + BoW (min_df=2, N-grams=(1,3))	54.714	0	0.70	0.42	0.53
			1	0.47	0.88	0.62
			2	0.71	0.27	0.39
			Macro avg	0.63	0.53	0.51
			Weighted avg	0.62	0.55	0.52
RF	70:30 + ROS + TF-IDF (min_df=5, N-grams=(1,3))	64.496	0	0.66	0.66	0.66
			1	0.63	0.76	0.69
			2	0.66	0.49	0.56
			Macro avg	0.65	0.64	0.64
			Weighted avg	0.65	0.64	0.64
SVC	80:20 + ROS + TF-IDF (min_df=2, N-grams=(1,3))	94.157	0	0.93	0.94	0.94
			1	0.94	0.96	0.95
			2	0.95	0.92	0.94
			Macro avg	0.94	0.94	0.94
			Weighted avg	0.94	0.94	0.94
XGBoost	80:20 + SMOTE + BoW (min_df=2, N-grams=(1,3))	86.142	0	0.84	0.88	0.86
			1	0.87	0.88	0.88
			2	0.88	0.81	0.84
			Macro avg	0.86	0.86	0.86
			Weighted avg	0.86	0.86	0.86
KNN	80:20 + ROS + TF-IDF (min_df=2, N-grams=(1,1))	94.370	0	0.91	0.96	0.94
			1	0.97	0.93	0.95
			2	0.96	0.93	0.95
			Macro avg	0.94	0.94	0.94
			Weighted avg	0.94	0.94	0.94
AdaBoost	70:30 + Unbalanced + TF-IDF (min_df=5, N-grams=(1,1))	66.682	0	0.70	0.66	0.68
			1	0.65	0.79	0.71
			2	0.65	0.53	0.58
			Macro avg	0.67	0.66	0.66
			Weighted avg	0.67	0.67	0.66

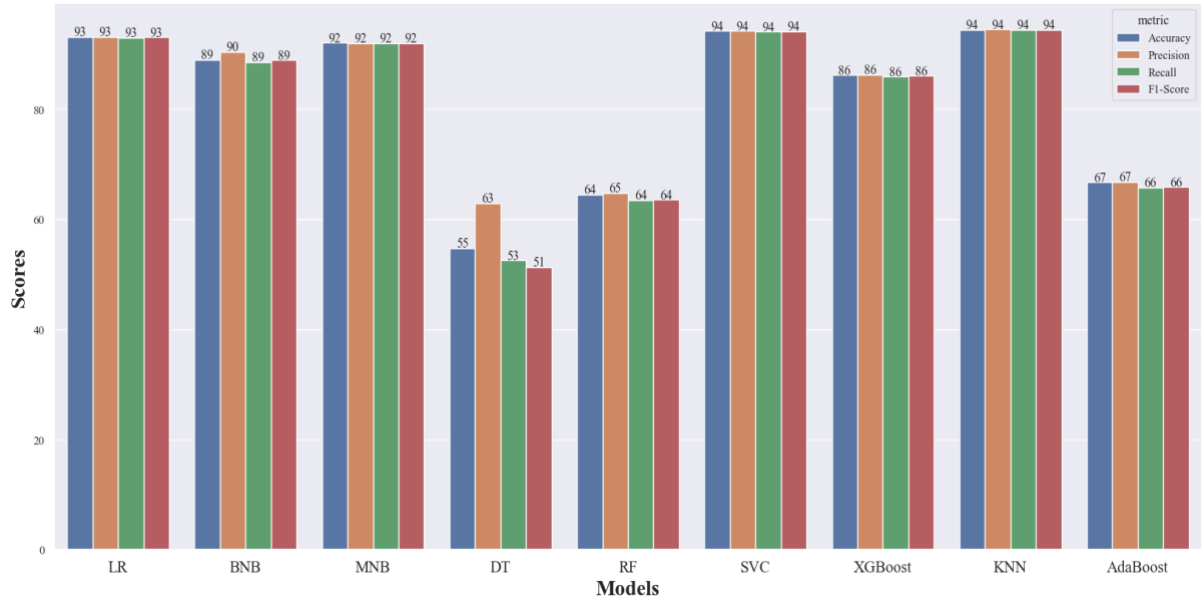


Figure 5.4. Accuracy, Precision, Recall, and F1-Score of the best performing combination of all nine classifiers.

Finally, with the nine classifiers and 128 different combinations of datasets, around 1152 models were built. We already seen that KNN and SVC were the best performing classifiers as shown in figure 5.4. Now, from that 1152 models, the top 10 best performing models were shown in table 5.16 where all the ten models were KNN and SVC with different combinations of datasets. All the top 5 models were KNN models with 80:20 split ratio and unigrams features. In the next five, four of them were SVC with 80:20 split ratio and either unigrams (1,1) or all N-grams combination (1,3). All the ten models had accuracy and F1-Score of 94% or above as shown in figure 5.5 and 5.6. The confusion matrix of the best performed model overall i.e., KNN with 80:20 split ratio was shown in figure 5.7. KNN with unbalanced and ROS datasets shared the first place as both of them had the same accuracy, precision, recall, and f1-score.

Table 5.16. The results of top 10 best performing models overall.

Rank	Classifier	Combination	Accuracy	Class	Precision	Recall	F1-Score
1	KNN(n_neighbors=1)	80:20 + Unbalanced + TF-IDF (min_df=2, ngram_range=(1,1))	94.370	0	0.91	0.96	0.94
				1	0.97	0.93	0.95
				2	0.96	0.93	0.95
				Macro avg	0.94	0.94	0.94
				Weighted avg	0.94	0.94	0.94
2	KNN(n_neighbors=1)	80:20 + ROS + TF-IDF (min_df=2, ngram_range=(1,1))	94.370	0	0.91	0.96	0.94
				1	0.97	0.93	0.95
				2	0.96	0.93	0.95
				Macro avg	0.94	0.94	0.94
				Weighted avg	0.94	0.94	0.94
3	KNN(n_neighbors=1)	80:20 + Unbalanced + TF-IDF (min_df=5, ngram_range=(1,1))	94.281	0	0.91	0.96	0.93
				1	0.96	0.94	0.95
				2	0.96	0.93	0.94
				Macro avg	0.94	0.94	0.94
				Weighted avg	0.94	0.94	0.94
4	KNN(n_neighbors=1)	80:20 + ROS + TF-IDF (min_df=5, ngram_range=(1,1))	94.281	0	0.91	0.96	0.93
				1	0.96	0.94	0.95
				2	0.96	0.93	0.94
				Macro avg	0.94	0.94	0.94
				Weighted avg	0.94	0.94	0.94

5	KNN(n_neighbors=1)	80:20 + SMOTE + TF-IDF (min_df=2, ngram_range=(1,1))	94.246	0	0.94	0.95	0.94
				1	0.97	0.93	0.95
				2	0.92	0.95	0.94
				Macro avg	0.94	0.94	0.94
				Weighted avg	0.94	0.94	0.94
6	SVC(C=100, gamma=1)	80:20 + ROS + TF-IDF (min_df=2, ngram_range=(1,3))	94.157	0	0.93	0.94	0.94
				1	0.94	0.96	0.95
				2	0.95	0.92	0.94
				Macro avg	0.94	0.94	0.94
				Weighted avg	0.94	0.94	0.94
7	KNN(n_neighbors=1)	80:20 + SMOTE + TF-IDF (min_df=5, ngram_range=(1,1))	94.148	0	0.94	0.94	0.94
				1	0.97	0.93	0.95
				2	0.92	0.95	0.94
				Macro avg	0.94	0.94	0.94
				Weighted avg	0.94	0.94	0.94
8	SVC(C=100, gamma=1)	80:20 + ROS + TF-IDF (min_df=2, ngram_range=(1,1))	94.069	0	0.93	0.94	0.94
				1	0.95	0.95	0.95
				2	0.95	0.93	0.94
				Macro avg	0.94	0.94	0.94
				Weighted avg	0.94	0.94	0.94
9	SVC(C=100, gamma=1)	80:20 + Unbalanced + TF-IDF (min_df=2, ngram_range=(1,3))	94.060	0	0.93	0.94	0.94
				1	0.94	0.96	0.95
				2	0.95	0.92	0.93
				Macro avg	0.94	0.94	0.94
				Weighted avg	0.94	0.94	0.94
10	SVC(C=100, gamma=1)	80:20 + Unbalanced + TF-IDF (min_df=2, ngram_range=(1,1))	94.033	0	0.93	0.94	0.94
				1	0.95	0.95	0.95
				2	0.95	0.93	0.94
				Macro avg	0.94	0.94	0.94
				Weighted avg	0.94	0.94	0.94

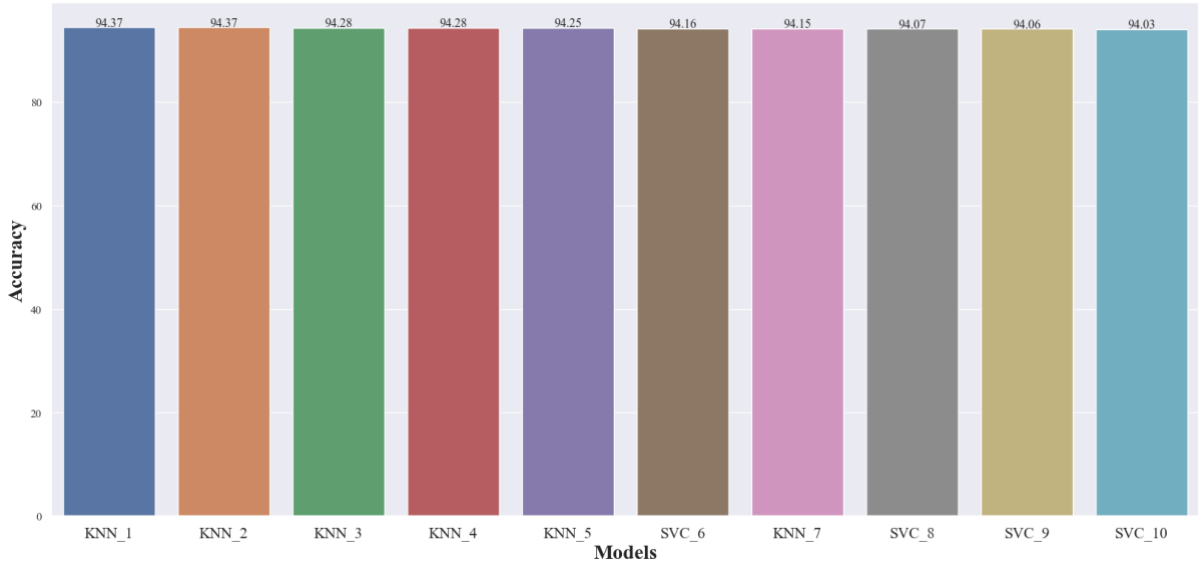


Figure 5.5. Accuracy of the top 10 best performing models overall.

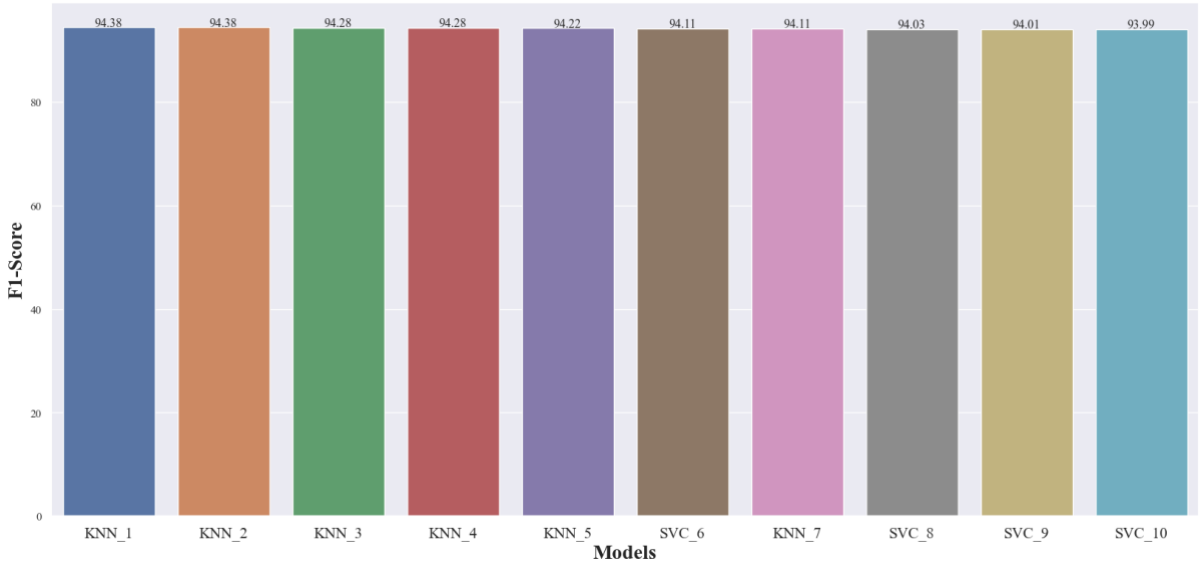


Figure 5.6. F1-Score of the top 10 best performing models overall.

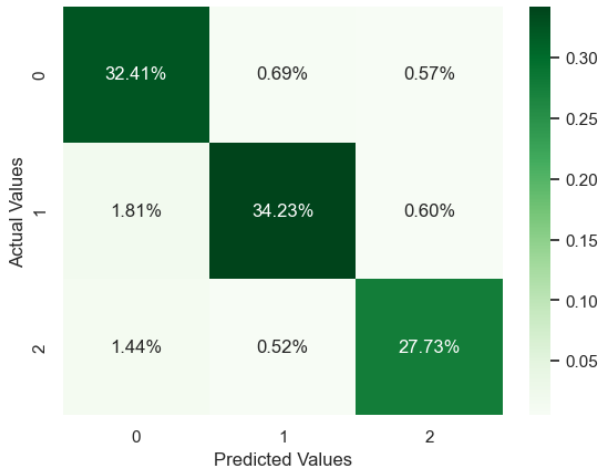


Figure 5.7. Confusion matrix of the best model - KNN with 80:20 split ratio.

5.7 Summary

The lexicon-based techniques implemented in this study i.e., TextBlob, VADER, and SentiWordNet didn't performed well on our dataset as all the three techniques got less than 50% accuracy. The reason may be because of the dataset in hand was very noisy and had too many slang words, acronyms and negations. Hence, for ML classifiers, the annotations of lexicon-based methods were didn't used and just the original label given was considered as our target variable to train the ML models. From the eight different feature combinations of both TF-IDF and BoW, four combinations (ngram_range: [(1,3), (1,1)] and min_df: 2, 5) performed consistently better than the other four combinations of bigrams and trigrams (ngram_range: [(2,2), (3,3)] and min_df: 2, 5). Among the data balancing techniques, SMOTE and Random Over Sampling (ROS) consistently performed better than Random Under Sampling (RUS). In this study, we designed and developed 1152 different models based on vectorization, data balancing, train-test splits and nine ML classifiers. From that, KNN and SVC with 80:20 split ratio outperformed all the other combinations with the best accuracy of 94.37% and 94.16%. Followed by them, LR and MNB with 80:20 split ratio on random oversampled dataset attained the best accuracy of 93.08% and 92.02%.

CHAPTER 6

CONCLUSIONS AND RECOMMENDATIONS

In this chapter, the overall conclusion of this study was discussed by stating the results of each objective of this study, answering all the research questions, indicating major limitations, and pointing out the best performed techniques. Along with that, the contribution this study had given to the existing knowledge and the future recommendations were specified.

6.1 Introduction

Twitter, a renowned microblogging platform allows the users to express their thoughts and emotions. And many people express their opinion on various issues through social media platforms like Twitter. If we can collectively predict the overall opinion or sentiment of people in a specific topic, it can help in many different ways like, to know the target audience for brands/businesses, take necessary precautions for a pandemic or trading shares for investors etc., Hence, this study focused on building a combination of lexicon-based and machine learning-based algorithm which can classify the sentiment of the tweets correctly. But the major limitations of this study were, it is focused only on the English language and the algorithms used for sentiment classification were lexicon-based, ML based but not deep learning-based as the dataset in hand was small, which does not allow enough learning paths for an overall stable system that can handle test points.

6.2 Discussion and Conclusion

For this study, a dataset from Kaggle.com called "Twitter Sentiment analysis" was taken. It had raw tweets along with their respective sentiments. The dataset was cleaned and pre-processed using various pre-processing methods such as lowercasing, URLs removal, stop words removal, negation handling, POS tagging, tokenisation, lemmatization, stemming etc.., Then the sentiments of the pre-processed tweets were predicted using the lexicon-based methods viz. TextBlob, VADER, and SentiWordNet. VADER was experimented with and without the emojis, but the difference in accuracy among the two was just 0.09%, which conforms, emojis didn't played a major role in the dataset used for the sentiment classification task. Sadly, the lexicon-based techniques used didn't perform well on our dataset as all the three techniques got less than 50% accuracy. The reason may be because of the dataset in hand was very noisy and had too many slang words, acronyms and negations. The best accuracy was obtained by VADER without the emojis i.e., 47.98%. Hence, for ML classifiers, the annotations of lexicon-based methods were didn't used and just the original label given was considered as our target variable to train the ML models.

For the machine learning-based approach, a total of 1152 models were developed and evaluated. The models were generated from nine different ML algorithms with a combination of different data balancing approach, feature extraction techniques, and train-test split ratios. For data balancing, SMOTE, ROS (Random Over Sampling), and RUS (Random Under Sampling) were implemented. For feature extraction, TF-IDF and BoW (Bag of Words) approach with eight different parameter combinations each (ngram_range: [(1,3), (1,1), (2,2), (3,3)] with min_df: 2 and 5) was implemented. For train-test split, 70:30 and 80:20 split ratios were used. And the nine different ML algorithms used for sentiment classification were Logistic Regression (LR), Bernoulli Naïve Bayes (BNB), Multinomial Naïve Bayes (MNB), Decision Trees (DT), Random Forest (RF), Support Vector Classifier (SVC), K-Nearest Neighbour (KNN), eXtreme Gradient Boosting (XGBoost), and AdaBoost.

The results of the 1152 models were evaluated primarily using accuracy and also with precision, recall, and F1-Score. The results had shown that, Oversampling had better performance than undersampling. And unigrams had better performance than bigrams and trigrams. In most of the cases, TF-IDF vectorizer and 80:20 train-test split ratio had performed better than BoW and 70:30 split ratio. As shown in table 6.1, among the nine ML algorithms, KNN and SVC with 80:20 split ratio outperformed all the other combinations with the best accuracy of 94.37% and 94.16%. Followed by them, LR and MNB with 80:20 split ratio on random oversampled dataset attained the best accuracy of 93.08% and 92.02%. The least performed ML classifiers were DT, RF and AdaBoost.

As shown in table 6.2, the result of the best performing model of this study was compared with the SoA models. And it can be observed that our model performed considerably well by performing better than most of them. We can also observe that, SVM was the best performing model in most of the cases. And in our study, SVM was the second-best model which outperformed almost every one of the SoA SVM models.

Even though the main aim of this study to build a combination of lexicon-based and ML based algorithm was not achieved. Because of the performance of lexicon-based techniques didn't achieve good accuracy. The ML models developed in this study achieved SoA models' performance. Hence, the top models KNN and SVC can be used to predict the sentiment of the people's opinion shared on social media platforms like twitter with the accuracy of 94%. This might help the brands, businesses, governments, or investors to know their target audience, competitors, take precautionary measures, investment plans etc.,

Table 6.1. The top 4 best performing algorithms of this study.

Rank	Classifier	Accuracy	Precision	Recall	F1-Score
1	KNN	94.37	94.47	94.36	94.38
2	SVC	94.16	94.19	94.04	94.11
3	LR	93.08	93.08	93.00	93.04
4	MNB	92.02	91.99	91.95	91.96

Table 6.2. Comparison of the state-of-the-art methods with the best result of this study.

Citation	Model Used	Dataset Used	Number of Tweets	Accuracy (%)
(Dagar et al., 2021)	Linear SVM	Sentiment140 - A labelled dataset publicly available on Kaggle	16,00,000	83.71
(Neogi et al., 2021)	Random Forest	A python library Twippy was used to extract the tweets.	18,000	96.60
(Bengesi et al., 2023)	SVM	A python library Twippy was used to extract the multilingual tweets.	1,07,396	93.48
(Mujahid et al., 2021)	DT, SVM and RF	Tweets were extracted using Twitter API.	17,155	95.00
(Chandralekha et al., 2022)	SVM	Tweets were extracted using Twippy library.	1,24,000	93.00
(Remali et al., 2022)	SVM	Tweets were extracted using Twitter Intelligent Tools (TWINT)	38,602	90.41
(Gupta et al., 2021)	Linear SVC	A python library Twippy was used to extract the tweets.	12,741	84.40
Current Study	KNN	Twitter Sentiment Analysis - A labelled dataset publicly available on Kaggle	75,682	94.37

6.3 Contribution to knowledge

As mentioned in section 6.2 since our main aim of the study to build a combination of lexicon-based and ML based algorithm was not achieved. The study didn't bring much to the existing knowledge. But the ML algorithms KNN and SVC of our study did achieve the SoA model's performance with the decent volume of dataset. Hence, it reconfirms that Support Vector Classifier (SVC) consistently performs well on classification problems as most of the best performing SoA models was SVC. The results of the study show that, over sampling techniques consistently performs better than the under-sampling technique. And unigrams consistently performs better than bigrams and trigrams.

6.4 Future Recommendations

In future, we can work on the sarcastic comments or sarcastic emoticons. And our work can be extended to test the framework's performance with other datasets which are specific to this domain. And improve the performance of lexicon-based techniques by using some of the advanced pre-processing techniques to handle the slang words, acronyms, and negations in the dataset better. And also experiment with other lexicon-based techniques like SentiStrength and RapidMiner for a more robust data labelling process. Since twitter allows multiple languages and people express their thoughts and opinions in their regional languages, we can extend this study to work on other regional languages as well. One interesting direction for future work is to work on deep learning models to improve their performance on small datasets and transformer algorithms. Another interesting direction will be, to rework on the hyperparameter tunning to further enhance and improve the accuracy level of best performing models. Finally, we can move one step forward and work on emotion detection, which conveys the exact feeling of the people whether they feel happy, sad, angry or surprised etc., This will help us to know the people's thoughts and opinions even better.

REFERENCES

Abdullah, N.A., Feizollah, A., Sulaiman, A. and Anuar, N.B., (2019) Challenges and Recommended Solutions in Multi-Source and Multi-Domain Sentiment Analysis. *IEEE Access*, 7, pp.144957–144971.

Ahuja, R., Chug, A., Kohli, S., Gupta, S. and Ahuja, P., (2019) The Impact of Features Extraction on the Sentiment Analysis. *Procedia Computer Science*, 152, pp.341–348.

Alam, S. and Yao, N., (2019) The impact of preprocessing steps on the accuracy of machine learning algorithms in sentiment analysis. *Comput Math Organ Theory*, [online] 25, pp.319–335. Available at: https://doi.org/10.1007/s10588-018-9266-8 [Accessed 24 Apr. 2023].

Al-Shabi, M.A. and Al-Shabi, M.A., (2020) Evaluating the performance of the most important Lexicons used to Sentiment analysis and opinions Mining. *IJCSNS International Journal of Computer Science and Network Security*. [online] Available at: https://www.researchgate.net/publication/343473213 [Accessed 17 Apr. 2023].

Anon (2023) *Chat / Internet Slang | Abbreviations | Acronyms | Kaggle*. [online] Available at: https://www.kaggle.com/datasets/gowrishankarp/chat-slang-abbreviations-acronyms [Accessed 6 May 2023].

Anon (2023) *Twitter Sentiment Analysis | Kaggle*. [online] Available at: https://www.kaggle.com/datasets/jp797498e/twitter-entity-sentiment-analysis?datasetId=1520310&sortBy=voteCount [Accessed 6 May 2023].

Arpita, Kumar, P. and Garg, K., (2020) Data Cleaning of Raw Tweets for Sentiment Analysis. *Indo - Taiwan 2nd International Conference on Computing, Analytics and Networks, Indo-Taiwan ICAN 2020 - Proceedings*, pp.273–276.

Aslam, N., Rustam, F., Lee, E., Washington, P.B. and Ashraf, I., (2022) Sentiment Analysis and Emotion Detection on Cryptocurrency Related Tweets Using Ensemble LSTM-GRU Model. *IEEE Access*, 10, pp.39313–39324.

Bengesi, S., Oladunni, T., Olusegun, R. and Audu, H., (2023) A Machine Learning - Sentiment Analysis on Monkeypox Outbreak: An Extensive Dataset to Show the Polarity of Public Opinion from Twitter Tweets. *IEEE Access*.

Bouazizi, M. and Ohtsuki, T., (2019) Multi-class sentiment analysis on twitter: Classification performance and challenges. *Big Data Mining and Analytics*, 23, pp.181–194.

Braig, N., Benz, A., Voth, S., Breitenbach, J. and Buettner, R., (2023) Machine Learning Techniques for Sentiment Analysis of COVID-19-Related Twitter Data. *IEEE Access*, PP, p.1.

Chandralekha, E., Jemin, V.M., Rama, P., Shanthini Watson Benjamin, M.I. and Antony Kumar, K., (2022) Sentiment Analysis of Corona Vaccination on Twitter Data using Machine Learning Techniques. *Proceedings - 6th International Conference on Computing Methodologies and Communication, ICCMC 2022*, pp.708–712.

Dagar, M., Kajal, A. and Bhatia, P., (2021) Twitter Sentiment Analysis using Supervised Machine Learning Techniques. *2021 5th International Conference on Information Systems and Computer Networks, ISCON 2021*, March.

Gandhi, U.D., Malarvizhi Kumar, P., Chandra Babu, G. and Karthick, G., (2021) Sentiment Analysis on Twitter Data by Using Convolutional Neural Network (CNN) and Long Short Term Memory (LSTM). *Wireless Personal Communications*, [online] 0123456789. Available at: https://doi.org/10.1007/s11277-021-08580-3.

Ghasiya, P. and Okamura, K., (2021) Investigating COVID-19 News across Four Nations: A Topic Modeling and Sentiment Analysis Approach. *IEEE Access*, 9, pp.36645–36656.

Gupta, I. and Joshi, N., (2021) Feature-Based Twitter Sentiment Analysis with Improved Negation Handling. *IEEE Transactions on Computational Social Systems*, 84, pp.917–927.

Gupta, P., Kumar, S., Suman, R.R. and Kumar, V., (2021) Sentiment Analysis of Lockdown in India during COVID-19: A Case Study on Twitter. *IEEE Transactions on Computational Social Systems*, 84, pp.939–949.
Hameed, Z. and Garcia-Zapirain, B., (2020) Sentiment Classification Using a Single-Layered BiLSTM Model. *IEEE Access*, 8, pp.73992–74001.
He, Q., (2022) Recent Works for Sentiment Analysis using Machine Learning and Lexicon Based Approaches. *Proceedings - 2022 5th International Conference on Advanced Electronic Materials, Computers and Software Engineering, AEMCSE 2022*, pp.422–426.
Ho, J., Ondusko, D., Roy, B. and Hsu, D.F., (2019) Sentiment analysis on tweets using machine learning and combinatorial fusion. *Proceedings - IEEE 17th International Conference on Dependable, Autonomic and Secure Computing, IEEE 17th International Conference on Pervasive Intelligence and Computing, IEEE 5th International Conference on Cloud and Big Data Computing, 4th Cyber Scienc*, 1, pp.1066–1071.
Hota, H.S., Sharma, D.K. and Verma, N., (2021) *Lexicon-based sentiment analysis using Twitter data*. [online] *Data Science for COVID-19 Volume 1: Computational Perspectives*. Elsevier Inc. Available at: http://dx.doi.org/10.1016/B978-0-12-824536-1.00015-0.
Jayasurya, G.G., Kumar, S., Singh, B.K. and Kumar, V., (2022) Analysis of Public Sentiment on COVID-19 Vaccination Using Twitter. *IEEE Transactions on Computational Social Systems*, 94, pp.1101–1111.
Jin, N., Wu, J., Ma, X., Yan, K. and Mo, Y., (2020) Multi-task learning model based on multi-scale CNN and LSTM for sentiment classification. *IEEE Access*, 8, pp.77060–77072.
Juanita, S., Adiyarta, K. and Syafrullah, M., (2022) Sentiment analysis on E-Marketplace User Opinions Using Lexicon-Based and Naïve Bayes Model. *International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*, 2022-Octob, pp.379–382.
Kamiş, S. and Goularas, D., (2019) Evaluation of Deep Learning Techniques in Sentiment Analysis from Twitter Data. *Proceedings - 2019 International Conference on Deep Learning and Machine Learning in Emerging Applications, Deep-ML 2019*, pp.12–17.
Kaya, A., Keceli, A.S., Catal, C. and Tekinerdogan, B., (2019) The impact of feature types, classifiers, and data balancing techniques on software vulnerability prediction models. *Journal of Software: Evolution and Process*, 319.
Khan, K. and Yadav, S., (2021) Sentiment analysis on covid-19 vaccine using Twitter data: A NLP approach. *IEEE Region 10 Humanitarian Technology Conference, R10-HTC*, 2021-Septe.
Khanam, R. and Sharma, A., (2021) Sentiment Analysis using Different Machine Learning Techniques for Product Review. *2021 International Conference on Computational Performance Evaluation, ComPE 2021*, pp.646–650.
Kolchyna, O., Souza, T.T.P., Treleaven, P. and Aste, T., (2015) Twitter Sentiment Analysis: Lexicon Method, Machine Learning Method and Their Combination. [online] Available at: http://arxiv.org/abs/1507.00955.
Kumar, S. and Zymbler, M., (2019) A machine learning approach to analyze customer satisfaction from airline tweets. *Journal of Big Data*, [online] 61, pp.1–16. Available at: https://journalofbigdata.springeropen.com/articles/10.1186/s40537-019-0224-1 [Accessed 23 Mar. 2023].
Kumar, S.S. and Rajini, A., (2019) Extensive Survey on Feature Extraction and Feature Selection Techniques for Sentiment Classification in Social Media. *2019 10th International Conference on Computing, Communication and Networking Technologies, ICCCNT 2019*.
Kumaresh, N., Naulegari, J., Bonta, V. and Janardhan, N., (2019) A Comprehensive Study on Lexicon Based Approaches for Sentiment Analysis. *Asian Journal of Computer Science and*

Technology, [online] 8S2, pp.1–6. Available at: www.rottentomatoes. [Accessed 17 Apr. 2023].
Lin, Y., Li, J., Yang, L., Xu, K. and Lin, H., (2020) Sentiment Analysis with Comparison Enhanced Deep Neural Network. *IEEE Access*, 8, pp.78378–78384.
Mandloi, L. and Patel, R., (2020) Twitter sentiments analysis using machine learninig methods. *2020 International Conference for Emerging Technology, INCET 2020*, pp.1–5.
Mishev, K., Gjorgjevikj, A., Vodenska, I., Chitkushev, L.T. and Trajanov, D., (2020) Evaluation of Sentiment Analysis in Finance: From Lexicons to Transformers. *IEEE Access*, 8, pp.131662–131682.
Mishra, P., Patil, S.A., Shehroj, U., Aniyeri, P. and Khan, T.A., (2022) Twitter Sentiment Analysis using Naive Bayes Algorithm. *3rd International Informatics and Software Engineering Conference, IISEC 2022*.
Moussa, M.E., Mohamed, E.H. and Haggag, M.H., (2018) A generic lexicon-based framework for sentiment analysis. *https://doi.org/10.1080/1206212X.2018.1483813*, [online] 425, pp.463–473. Available at: https://www.tandfonline.com/doi/abs/10.1080/1206212X.2018.1483813 [Accessed 17 Apr. 2023].
Mrozek, P., Panneerselvam, J. and Bagdasar, O., (2020) Efficient resampling for fraud detection during anonymised credit card transactions with unbalanced datasets. *Proceedings - 2020 IEEE/ACM 13th International Conference on Utility and Cloud Computing, UCC 2020*, pp.426–433.
Mujahid, M., Lee, E., Rustam, F., Washington, P.B., Ullah, S., Reshi, A.A. and Ashraf, I., (2021) Sentiment Analysis and Topic Modeling on Tweets about Online Education during COVID-19. *Applied Sciences 2021, Vol. 11, Page 8438*, [online] 1118, p.8438. Available at: https://www.mdpi.com/2076-3417/11/18/8438/htm [Accessed 11 Feb. 2023].
Naresh, A. and Venkata Krishna, P., (2021) An efficient approach for sentiment analysis using machine learning algorithm. *Evolutionary Intelligence*, [online] 142, pp.725–731. Available at: https://doi.org/10.1007/s12065-020-00429-1.
Neogi, A.S., Garg, K.A., Mishra, R.K. and Dwivedi, Y.K., (2021) Sentiment analysis and classification of Indian farmers' protest using twitter data. *International Journal of Information Management Data Insights*, [online] 12, p.100019. Available at: https://doi.org/10.1016/j.jjimei.2021.100019.
Neshan, S.A.S. and Akbari, R., (2020) A Combination of Machine Learning and Lexicon Based Techniques for Sentiment Analysis. *2020 6th International Conference on Web Research, ICWR 2020*, pp.8–14.
Nugroho, D.K., (2021) US presidential election 2020 prediction based on Twitter data using lexicon-based sentiment analysis. *Proceedings of the Confluence 2021: 11th International Conference on Cloud Computing, Data Science and Engineering*, pp.136–141.
Onan, A., (2021) Sentiment analysis on massive open online course evaluations: A text mining and deep learning approach. *Computer Applications in Engineering Education*, 293, pp.572–589.
Pandya, V., Somthankar, A., Shrivastava, S.S. and Patil, M., (2021) Twitter Sentiment Analysis using Machine Learning and Deep Learning Techniques. *Proceedings of the 2021 2nd International Conference on Communication, Computing and Industry 4.0, C2I4 2021*.
Pereira, J. and Saraiva, F., (2020) A Comparative Analysis of Unbalanced Data Handling Techniques for Machine Learning Algorithms to Electricity Theft Detection. *2020 IEEE Congress on Evolutionary Computation, CEC 2020 - Conference Proceedings*.
Pereira, J. and Saraiva, F., (2021) Convolutional neural network applied to detect electricity theft: A comparative study on unbalanced data handling techniques. *International Journal of Electrical Power and Energy Systems*, 131.

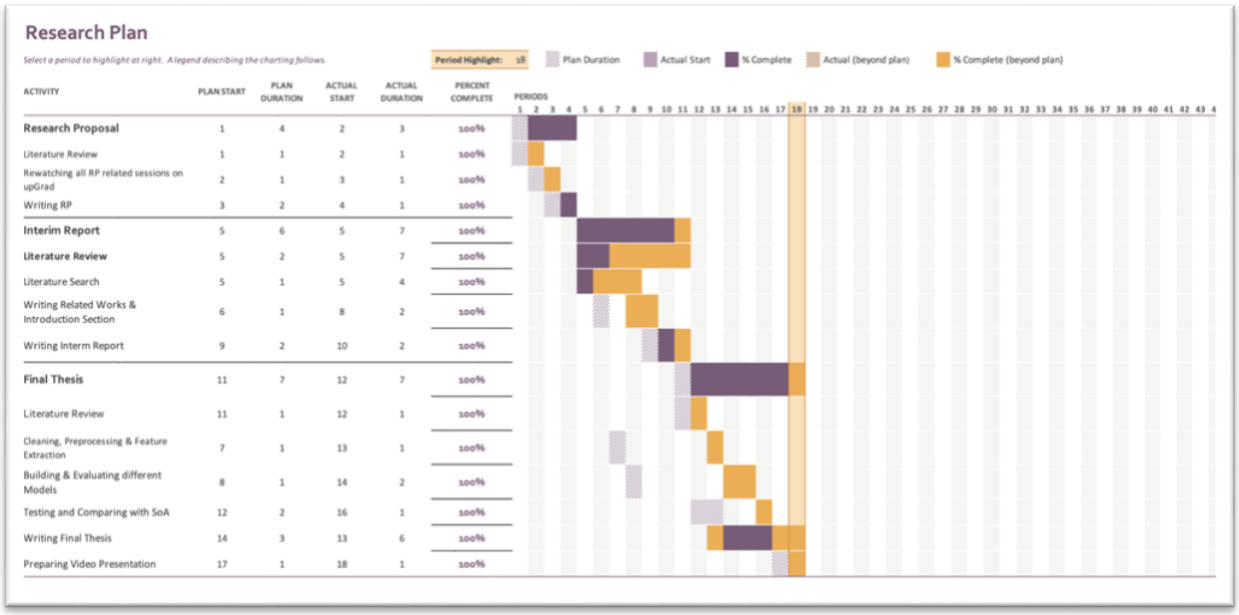
Pradha, S., Halgamuge, M.N. and Tran Quoc Vinh, N., (2019) Effective text data preprocessing technique for sentiment analysis in social media data. *Proceedings of 2019 11th International Conference on Knowledge and Systems Engineering, KSE 2019*.
El Rahman, S.A., Alotaibi, F.A. and Alshehri, W.A., (2019) Sentiment Analysis of Twitter Data. *2019 International Conference on Computer and Information Sciences, ICCIS 2019*.
Remali, N.A.S., Shamsuddin, M.R. and Abdul-Rahman, S., (2022) Sentiment Analysis on Online Learning for Higher Education During Covid-19. *2022 3rd International Conference on Artificial Intelligence and Data Sciences: Championing Innovations in Artificial Intelligence and Data Sciences for Sustainable Future, AiDAS 2022 - Proceedings*, pp.142–147.
Roy, A. and Ojha, M., (2020) Twitter sentiment analysis using deep learning models. *2020 IEEE 17th India Council International Conference, INDICON 2020*, June, pp.0–17.
Rustam, F., Khalid, M., Aslam, W., Rupapara, V., Mehmood, A. and Choi, G.S., (2021) A performance comparison of supervised machine learning models for Covid-19 tweets sentiment analysis. *PLoS ONE*, [online] 162, pp.1–23. Available at: http://dx.doi.org/10.1371/journal.pone.0245909.
Samuel, J., Ali, G.G.M.N., Rahman, M.M., Esawi, E. and Samuel, Y., (2020) COVID-19 public sentiment insights and machine learning for tweets classification. *Information (Switzerland)*, 116, pp.1–22.
Satrya, R.N., Pratiwi, O.N., Farifah, R.Y. and Abawajy, J., (2022) Cryptocurrency Sentiment Analysis on the Twitter Platform Using Support Vector Machine (SVM) Algorithm. *Proceedings - International Conference Advancement in Data Science, E-Learning and Information Systems, ICADEIS 2022*.
Seo, S., Kim, C., Kim, H., Mo, K. and Kang, P., (2020) Comparative Study of Deep Learning-Based Sentiment Classification. *IEEE Access*, 8, pp.6861–6875.
Sharma, R. and Rohini, J., (2022) A Study on Lexicon Based Techniques of Twitter Sentiment Analysis. *2022 8th International Conference on Advanced Computing and Communication Systems (ICACCS)*, 1.
Singh, S., Kumar, K. and Kumar, B., (2022) Sentiment Analysis of Twitter Data Using TF-IDF and Machine Learning Techniques. *2022 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing, COM-IT-CON 2022*, pp.252–255.
Symeonidis, S., Effrosynidis, D. and Arampatzis, A., (2018) A comparative evaluation of pre-processing techniques and their interactions for twitter sentiment analysis. *Expert Systems with Applications*, [online] 110, pp.298–310. Available at: https://doi.org/10.1016/j.eswa.2018.06.022.
Taj, S., Shaikh, B.B. and Fatemah Meghji, A., (2019) Sentiment analysis of news articles: A lexicon based approach. *2019 2nd International Conference on Computing, Mathematics and Engineering Technologies, iCoMET 2019*.
Venkatesh, Hegde, S.U., Zaiba, A.S. and Nagaraju, Y., (2021) Hybrid CNN-LSTM model with glove word vector for sentiment analysis on football specific tweets. *Proceedings of the 2021 1st International Conference on Advances in Electrical, Computing, Communications and Sustainable Technologies, ICAECT 2021*.
Vyas, P., Reisslein, M., Rimal, B.P., Vyas, G., Basyal, G.P. and Muzumdar, P., (2021) Automated Classification of Societal Sentiments on Twitter With Machine Learning. *IEEE Transactions on Technology and Society*, 32, pp.100–110.
Wunderlich, F. and Memmert, D., (2020) Innovative Approaches in Sports Science—Lexicon-Based Sentiment Analysis as a Tool to Analyze Sports-Related Twitter Communication. *Applied Sciences 2020, Vol. 10, Page 431*, [online] 102, p.431. Available at: https://www.mdpi.com/2076-3417/10/2/431/htm [Accessed 17 Apr. 2023].

Xiao, Z., Wang, L. and Du, J.Y., (2019) Improving the performance of sentiment classification on imbalanced datasets with transfer learning. *IEEE Access*, 7, pp.28281–28290.

APPENDIX A: RESEARCH PLAN

The updated plan of work for this study was given below

- One Period = One Week.
- Updated the actual time taken from the interim report to final thesis submission (June 07).



APPENDIX B: RESEARCH PROPOSAL

Building A Combination of Lexicon Based and Machine Learning Based Model for Twitter Sentiment Analysis

Bharath Raj S

Research Proposal

February 2023

Abstract

Sentiment analysis is a booming research field where many research works are being done. Analysing the given data and finding the overall sentiment or emotion of it and classifying it has positive, negative or neutral is sentiment analysis also known as opinion mining. Twitter, a renowned microblogging platform allows the users to share their thoughts, opinion on anything like products, movies etc., If we can collectively predict the overall opinion or sentiment of the people for a particular product or brand using some techniques, it can help to know the target audience for brands, take necessary precautions for a pandemic or get reviews for the movie etc., The aim of this study is to build a lexicon-based machine learning model which can classify the sentiment of the tweets correctly. A publicly available 'Twitter Sentiment Analysis' dataset from Kaggle is taken for this study. Textblob, a lexicon-based technique used to classify the text as positive, negative or neutral based on the semantic orientation of the words in the text, is used to label the data. Since the tweets will contain some irregular data, it needs to be cleaned and pre-processed. The process includes lemmatization, stemming, removal of stop words, symbols, non-English words, numerical values, Hashtags, URLs and User Mentions. BoW (Bag of Words) and TF-IDF (Term Frequency - Inverse Document Frequency) is used for feature extraction. Finally, most commonly used machine learning algorithms for classification problems are implemented namely Naïve Bayes, Logistic Regression, SVM, Decision Trees and Random Forest. The performance of the models is evaluated on accuracy, recall, precision and F-measure.

1. Background

The online social media platforms such as Facebook, Twitter and Instagram make the people to share their perspective, thoughts or opinions about online products, latest movies, government elections, social issues etc.,(El Rahman et al., 2019). In Twitter for every day 550M tweets were posted. With this huge amount of data, we can easily find out, what is the publics opinion about something whether it is positive, negative or neutral (Mandloi and Patel, 2020). To mention a few,

- Film makers can know how their movie is received by the sentiment of the public reviews.
- A mobile brand can find how their new mobile model launched is performing with the sentiment of the product reviews.
- Youtubers can find out how well their new video is received by the viewers with the comments section's sentiment.
- An investor or a trader can find wheatear to invest/buy in a particular stock or not with the sentiment of the stock in the twitter.

Huge companies/ brands/ political parties will definitely have an account in twitter, so that they can find out the sentiment of their products or services with their own tweet's comments and retweets and plan their next steps (El Rahman et al., 2019).

The reason for choosing twitter for the sentiment analysis are as follows,

- As mentioned earlier, the amount of data we get every day in twitter.
- The restriction of 280 characters per tweet allows us to capture the context of the message better (Mandloi and Patel, 2020).

In recent years, the sentiment analysis research area is growing rapidly (Naresh and Venkata Krishna, 2021). And different kinds of techniques have been implemented. In this paper, we are going to focus only on the machine learning models because of the size of the data. If the dataset we have in hand is not so big, there is no scope for Deep learning algorithms to train well (Rustam et al., 2021).

We are getting 'Twitter Sentiment Analysis' dataset from Kaggle.com which has tweets with their corresponding sentiments. But we are going to find the sentiment of the tweets again using lexicon-based approach. So that we can have some comparison between the two results with two different target variables for the ML models we are going to train, which may help us to improve the performance of the model (Rustam et al., 2021).

For now, we are going to try out five different machine learning algorithms viz., logistic regression, Naïve Bayes, SVM, Decision Trees and Random Forest. For feature extraction, most widely used two techniques BoW and TF-IDF were utilized (Mujahid et al., 2021). Surely, we will work on more techniques by the end of this study. The performance of the ML models is evaluated with four evaluation metrics viz., accuracy, precision, recall and f1-score (Rustam et al., 2021).

In this study, we are going to try to find the answers for the following questions,

1. Will the emoticons in the tweets help us to find the sentiment of the tweet better?
2. What are all the benefits from this study for the business community?
3. Which feature engineering technique will give us better results?
4. Can the process of finding the sentiment of the day to day tweets automated completely?

2. Problem Statement and Related Work

In the field of sentiment analysis, massive amount of research work has been done and

- both the supervised and non-supervised techniques have been used,
- different types of feature extraction methods have been tried,
- contrasting pre-processing techniques have been applied,
- different kinds of evaluation metrics have been used

by the researchers. We are going to discuss about a few of them here. In which, some of them

- have used a small dataset, so the proposed model won't perform well on all kinds of data.
- lack on computational power, so didn't tried different approaches.
- fixed on very few algorithms for no reason.

By undoing all these things, at the end of this study, after trying out as many techniques as possible, will figure out which one gives us better results.

In this study (Roy and Ojha, 2020) the authors implemented both machine learning and deep learning algorithms like Naive Bayes, Maximum Entropy, Decision Tree, Random Forest, XGBoost, SVM, Multi-Layer Perceptron, RNN and CNNs. And finally built an ensemble of their best 5 models and achieved an accuracy of 83.58%. In another study (Kolchyna et al., 2015) they proposed a combination of lexicon based and Machine learning based method and got F1 score of 73%.

While most of the papers focused only on textual data, in this study (Roy and Ojha, 2020) the authors additionally used the emoticons in the tweets. In another study (Gandhi et al., 2021) a deep learning model (LSTM) had been used in the movie review dataset (IMDB) and got an accuracy of 88.02%.

An optimization-based machine learning technique have been introduced in this paper (Naresh and Venkata Krishna, 2021) called SMODT. Sequential Minimal Optimization Decision Trees (SMO+DT) got an accuracy of 89.47%. In 2020, the Indian farmers protested against the three farm acts passed, for which Twitter reacted in both ways. In this study to capture the semantic orientation, the authors used lexicon-based approach and labelled the data, then applied multiple machine learning techniques. Out of which, Random Forest outperformed with accuracy of 96.6% (Neogi et al., 2021).

While most of the studies used Evaluation metrics as accuracy, along with it, in this study the authors have used AUROC curves to evaluate the model (Dagar et al., 2021). In another study about which one is popular KFC or McDonalds, the author used F1 score and cross validation as evaluation metric (El Rahman et al., 2019). In another study (Samuel et al., 2020) about the coronavirus specific tweets authors have used specificity as one of the main evaluation metrics.

In a study of MOOC (massive open online courses) reviews, the author had tried three different term weighting schemes (TP, TF and TF-IDF), three N-gram techniques (Unigram, Bigram and trigram), five supervised methods and five ensemble methods. For deep learning models, three word embedding schemes (word2ved, fastText, GloVe) were employed and five different DL models were tried. After evaluated using accuracy and F-score, Random subspace ensemble obtained the highest performance. Out of the nine configurations, Unigram with TF based representation had highest accuracy (Onan, 2021).

In the recent years of sentiment analysis on Twitter data, one of the most popular topics is coronavirus. So, it won't be complete without talking about that. In this paper (Rustam et al., 2021) for feature extraction, author had used combination of BoW and TF-IDF representation. In that study, ETC (Randomized Decision Trees) model attained the best accuracy of 0.92. In another paper, (Samuel et al., 2020) the author noticed that the model's accuracy reducing when the length of the tweets was increased. In another paper, (Mujahid et al., 2021) the author proved that,

- for annotation TextBlob is the best one to choose when compared to VADER and SentiWordNet.
- Data balanced with SMOTE gives higher accuracy.
- Due to the small dataset ML models outperformed DL models.
- In BoW feature engineering technique DT, SVM and RF had best accuracy of 95% and in TF-IDF technique, SVM outperformed others with the same accuracy.

In sentiment analysis, various ML and DL techniques had been used. So, to get a better understanding and view, a tabular form of the same is given below.

Table 1. A summary of methodology used in the studies with the dataset and the evaluation metrics.

Citation	Dataset	Methodology	Evaluation Metric
(Mujahid et al., 2021)	Total of 17,155 tweets collected from Twitter	DT, SVM and RF	Accuracy, Sensitivity, Precision and F1 Score
(Roy and Ojha, 2020)	Training: 800,000 tweets, Test: 200,000 tweets	Ensemble of deep learning models	Accuracy
(Dagar et al., 2021)	1,600,000 tweets extracted from Twitter API	LR, Multinominal Naïve Bayes, Linear SVM	Accuracy, AUROC curve
(Kolchyna et al., 2015)	SemEval-2013 competition, Task 2-B dataset	Combination of Lexicon (OL+EMO) and ML (Cost-Sensitive SVM) based model	F1 Score
(Rustam et al., 2021)	Total of 7528 tweets collected from Twitter	ETC (Randomized DTs)	Accuracy, Recall, Precision and F1 Score
(Naresh and Venkata Krishna, 2021)	Airline Twitter dataset	SMO (Sequential Minimal Optimization) +DT	Accuracy, Recall, Precision and F1 Score

(Neogi et al., 2021)	18,000 tweets collected from Twitter	Random Forest	Accuracy, Recall, Precision and Confusion Matrix
(Samuel et al., 2020)	Total of 900,000 tweets collected using Twitter API	Naïve Bayes Classifier, Logistic Regression	Accuracy, Sensitivity, Specificity
(Mandloi and Patel, 2020)	Data gathered using Twitter API	Naïve Bayes Classifier	Accuracy, Precision
(Gandhi et al., 2021)	IMDB dataset from Kaggle	CNN and LSTM	Accuracy
(Onan, 2021)	93000 reviews from coursetalk	Random Subspace Ensemble	Classification accuracy and F-score

3. Research Questions

In this study about sentiment analysis of twitter data, what are all the questions which the study going to answer at the end of it is given below.

7. Will the emoticons in the tweets help us to find the sentiment of the tweets better?
8. What are all the benefits from this study for the business community?
9. Which feature engineering technique will give us better results?
10. Which one of the Machine Learning Models will win the battle of best performing model?
11. What is the best evaluation metric, to attain our aim?
12. Can the process of finding the sentiment of the day to day tweets automated completely?

4. Aim and Objectives

The main aim of this research is to propose a lexicon based ML model which classifies the sentiment of the tweets correctly as positive, negative or neutral. The identification of the correct sentiment of the tweets allow us to better understand the society/target audience or to make necessary precautionary measures for any surprises.

The research objectives are formulated based on the aim of this study which are as follows:

- To go through the State-of-the-art papers and techniques used among them.
- To find the best feature engineering technique which will lead us to the best results.
- To develop a ML model which can classify the sentiment of the tweets correctly.
- To evaluate the performance of the ML model.
- Compare our best model with the SoA.

5. Significance of the Study

As mentioned earlier, in twitter at least 500M tweets were sent per day. So, the data of twitter can tell us all about the society's thoughts, emotions and sentiments (Mandloi and Patel, 2020) which can help us to,

- get better reviews - E.g., Opinion about an election results, brands, products or movies.
- make necessary precautions - E.g., In 2019-2020, It would have helped to eliminate the mass fear from incomplete or wrong information about coronavirus (Samuel et al., 2020).
- know the target audience - E.g., A upcoming OTT platform can target people who prefers web series.
- Make investment plan - E.g., Know the recent trends of share market and which stocks to invest in.

To achieve all this, a good sentiment analysis algorithm/methodology of twitter data would help. And at the end of this study, a good lexicon based ML model is going to be built and it will do the job for us.

6. Scope of the Study

In this study, there are some boundaries/limitations for many reasons, which has been shared in detail below.

6.1. In Scope

- Sentiment analysis on Twitter.
- Machine Learning Models.
- Textual Data.
- English words.

6.2. Out of Scope

- Any other social media platforms except Twitter like Facebook, YouTube etc., Since the Tweets are restricted to 280 characters, it is easier to get the context out of it (Mandloi and Patel, 2020).
- Deep learning Models. Because the dataset we have in hand is not so big (Rustam et al., 2021).
- Numerical data, symbols, images and videos. The study only involves textual data, and numbers or symbols won't add much to predict the sentiment.
- Non-English words. Even though Twitter allows different languages, in this study we are focusing only on English words to identify the sentiment of the tweet.

7. Research Methodology

In this section, we are going to discuss about the process we used to find the sentient of the tweets in detail. Firstly, about the data we have in hand which is gathered from Twitter. Then the techniques used to clean the data and after that how it is processed to get in the understandable shape of machine language. Further the machine learning techniques used to classify the data into positive, negative and neutral correctly. At last, the evaluation metrics used to evaluate the performance of the ML models and choosing the best one (Neogi et al., 2021).

7.1. Data Description

The dataset used in this study is taken from kaggle.com which consists of two csv files, one is to train the data and other is to validate the performance. The dataset has tweets with their respective sentiments and the columns were tweet-id, sentiment and tweet content. The tweet content is the tweet posted which has textual data with Hashtags, URLs, User mentions, Emoticons, Symbols and Numerical values (Roy and Ojha, 2020). Also, the textual data will have misspelled words or text slang. So before applying any algorithms the data need to be cleaned and pre-processed.

7.2. Data Pre-processing

The collected data is a tweet posted on social media, it will consist of irrelevant or noisy data. Before handling that we need to make sure there are no duplicates in the data (Neogi et al., 2021). Now, we clean the improper tweet content in the following ways,

- Removal of Hashtags, URLs, User mentions, Punctuations, Numbers and Retweets: In the process of understanding the meaning of the tweet and finding its sentiment, Hashtags (#), URLs, User mentions (@name), Retweets (RT) and Punctuations [!?()”,’] won’t add any kind of meaning. So those should be removed using regular expression also shortly called as RegEx (Mujahid et al., 2021).

- Removal of non-English words: Even though, Twitter allows different kinds of languages, since we are focusing only on English, we remove all the non-English words (Mandloi and Patel, 2020).

- Converting all the words in a tweet to lowercase. And removing unwanted space in the tweet.

- Removal of stop words: The stop words (a, I, an, the, to etc.,) won’t be more significant to find the sentiments in a sentence. So, it can be removed (Mandloi and Patel, 2020).

- Stemming and Lemmatization: To convert all the words in the tweet into its base form we use stemming and lemmatization. Because it reduces the complexity of the feature extraction process. For example: For the words ‘reduces’, ‘reduced’ and ‘reducing’ three different features will be created instead if we replace all the three words with its base form ‘reduce’, we will have only one feature (Mujahid et al., 2021).

- Handling Emoticons: Instead of removing the emoticons, we can use it in a better transformed way. Because the user who tweets will express his/her emotion using emoticons, which vitally help us to figure out the sentiment of the tweet. In a study, (Roy and Ojha, 2020) Using regular expression the author transformed some of the commonly used positive and negative emoticons with EMO_POS and EMO_NEG respectively.

7.3. Lexicon based Sentiment Computation

In our dataset, we already have the sentiment score for each of the tweets. But we are going to use lexicon-based approach and find the sentiment of the tweets again and label it has positive, negative or neutral. The benefit of doing it is, now we can build ML models with two target variables in hand. The one i.e., already given in the dataset and another one created using lexicon-based approach. When we compare the results of both approaches, it will help us to improve the performance of the ML models (Rustam et al., 2021).

In the lexicon-based approach we are capturing the semantic orientation of the words in the tweet, and labelling it as positive, negative or neutral (Neogi et al., 2021). TextBlob, a python package helps us to do this job and returns the polarity score between [-1,1]. If the score is >0 then the sentiment of the tweet is positive, if the score is >0 then negative and if it is =0 then neutral (Mujahid et al., 2021). This score is computed by taking average of the word's polarities in the tweet (Kolchyna et al., 2015).

7.4. Feature Engineering

Before implementing the Machine Learning techniques, we need to extract the features from the pre-processed data. Most widely used two techniques for feature extraction are BoW and TF-IDF (Mujahid et al., 2021).

7.4.1. TF-IDF (Term Frequency-Inverse Document Frequency)

TF-IDF is a feature extraction technique which gives us the importance/weightage of each term in a document by multiplying the Term Frequency (TF) with the Inverse Document Frequency (IDF).

TF is the number of times a term t occurred in the document d divided by the total length of the document.

$$TF_{t,d} = \text{count}_{t,d} / \text{total count}_{d}$$

IDF is the number of documents N divided by number of documents which has the term t.

$$IDF = N / Df_t$$

If a term is frequently occurred in the dataset, then the IDF value of the term will be less. Finally, TF-IDF is defined as $TF_{t,d}$ * log (IDF) (Rustam et al., 2021).

7.4.2. BoW (Bag of Words)

BoW is a feature extraction technique which will form a feature vector which has the number of occurrences of each unique word. And it will create a vocabulary of all the unique words in the dataset with their occurrences which will help us to train the machine learning models (Mujahid et al., 2021).

To make the performance of the ML models even better we try to concatenate the Bow and TF-IDF features (Rustam et al., 2021).

7.5. Proposed Methodology

In general, for classification problems, Machine Learning techniques have been preferred which is a supervised method, that inspect the data labelled as positive, negative or neutral (Kolchyna et al., 2015). For now, we have considered five popular ML techniques, namely Naïve Bayes, SVM, Logistic Regression, Decision Trees and Random Forest. Going forward, we will try to cover as many ML algorithms or Ensembles as possible.

7.5.1. Naïve Bayes

Naïve Bayes is a simple machine learning technique which is mostly used to classify high-dimensional data (Mandloi and Patel, 2020). It uses the Bayes Theorem mentioned below for the classification problems,

$$P(A/B) = (P(B/A) * P(A)) / P(B)$$

Here, P(A/B) is posterior probability, P(B/A) is likelihood probability and P(A) & P(B) are prior probability (Neogi et al., 2021).

7.5.2. Support Vector Machine (SVM)

SVM is a popular linear model used for both the classification and regression problems (Mandloi and Patel, 2020). In most of the sentiment analysis works, SVM is included (Rustam et al., 2021). “SVM finds a hyperplane in the higher dimensional space to separate instances of different classes” (Onan, 2021).

7.5.3. Logistic Regression (LR)

Logistic Regression is a supervised machine learning model which performs well on binary outcomes but also good in multi-class classification (Mujahid et al., 2021). It takes the values between 0 and 1 as input and produces output using sigmoid function (Dagar et al., 2021).

7.5.4. Decision Trees (DT)

Decision Trees are a Tree-structured classifier model which has root node, internal nodes, branches and leaf node (Rustam et al., 2021). Root node is the extracted features we gave into the model. Internal nodes represent the attributes of the dataset. Branches are the rules used to make that split. Finally, the leaf node is the result i.e., the sentiment of the tweet (positive, negative or neutral) (Neogi et al., 2021).

7.5.5. Random Forest (RF)

Random Forest is an ensemble technique used for both regression and classification (Roy and Ojha, 2020). A RF is a collection of decision trees, if we have more and more decision trees, the prediction power of the RF increases. We say 'Random' forest because we select the number of trees randomly with the random number of data samples and include them in our forest. For the final prediction, we take voting to get the aggregated results from the trees (Mujahid et al., 2021).

7.6. Performance Evaluation

To evaluate the performance of the machine learning models we are considering four evaluation metrics: Accuracy, Precision, Recall and F-measure (Rustam et al., 2021).

Accuracy: Accuracy is the most widely used evaluation metric which is the ratio of correctly classified prediction to the overall prediction (Naresh and Venkata Krishna, 2021).

$$\text{Accuracy} = (TN + TP) / (TP + FP + TN + FN)$$

Here, TP = True Positive, TN = True Negative, FP = False Positive and FN = False Negative (Onan, 2021).

Precision: Precision shows the proportion of correctly classified to the total number of positive samples (Naresh and Venkata Krishna, 2021).

$$\text{Precision} = TP / (TP + FP)$$

Recall: Recall indicates the completeness of the classifier i.e., the ratio of correctly predicted positive samples (Naresh and Venkata Krishna, 2021; Rustam et al., 2021).

$$\text{Recall} = TP / (TP + FN)$$

F-Measure: F-measure is another widely used evaluation metric for machine learning models (Onan, 2021). It is a harmonic mean of Precision and Recall (Rustam et al., 2021).

$$\text{F-measure} = (2 * \text{Precision} * \text{Recall}) / (\text{Precision} + \text{Recall})$$

8. Required Resources

To carry out this study of Twitter Sentiment Analysis using Lexicon based ML model, the following are the necessary software and hardware requirements.

Hardware Requirements	Software Requirements
CPU - AMD Ryzen 4th Gen (4000 series) or an equivalent Intel Core processor will also be optimal. **GPU** - GPU with a size of 8GB minimum. **RAM** - 8 GB Minimum. **Operation System** - Microsoft Windows 10 or higher. **Storage** - SSD of 256 GB is minimum.	A python environment - Anaconda or Jupyter Notebook. Microsoft Word, Excel. Mendeley. Turnitin. Important python libraries required to execute our work: Scikit Learn, Pandas, NumPy, SciPy, SpaCy, TensorFlow, Keras, Matplotlib, NLTK, TextBlob.

9. Research Plan

The plan of work for this study was shown in the below figure 'Research plan for the study'.

- One Period = One Week.
- Plan was created from the topics approval (Jan 10) to Final Thesis Submission (May 31).
- The timings were estimated by keeping in mind about the modules and unanticipated events.

Fig.1 Research plan for the study

References

Dagar, M., Kajal, A. and Bhatia, P., (2021) Twitter Sentiment Analysis using Supervised Machine Learning Techniques. *2021 5th International Conference on Information Systems and Computer Networks, ISCON 2021*, March.

Gandhi, U.D., Malarvizhi Kumar, P., Chandra Babu, G. and Karthick, G., (2021) Sentiment Analysis on Twitter Data by Using Convolutional Neural Network (CNN) and Long Short Term Memory (LSTM). *Wireless Personal Communications*, [online] 0123456789. Available at: https://doi.org/10.1007/s11277-021-08580-3.

Kolchyna, O., Souza, T.T.P., Treleaven, P. and Aste, T., (2015) Twitter Sentiment Analysis: Lexicon Method, Machine Learning Method and Their Combination. [online] Available at: http://arxiv.org/abs/1507.00955.

Mandloi, L. and Patel, R., (2020) Twitter sentiments analysis using machine learninig methods. *2020 International Conference for Emerging Technology, INCET 2020*, pp.1–5.

Mujahid, M., Lee, E., Rustam, F., Washington, P.B., Ullah, S., Reshi, A.A. and Ashraf, I., (2021) Sentiment Analysis and Topic Modeling on Tweets about Online Education during COVID-19. *Applied Sciences 2021, Vol. 11, Page 8438*, [online] 1118, p.8438. Available at: https://www.mdpi.com/2076-3417/11/18/8438/htm [Accessed 11 Feb. 2023].

Naresh, A. and Venkata Krishna, P., (2021) An efficient approach for sentiment analysis using machine learning algorithm. *Evolutionary Intelligence*, [online] 142, pp.725–731. Available at: https://doi.org/10.1007/s12065-020-00429-1.

Neogi, A.S., Garg, K.A., Mishra, R.K. and Dwivedi, Y.K., (2021) Sentiment analysis and classification of Indian farmers' protest using twitter data. *International Journal of Information Management Data Insights*, [online] 12, p.100019. Available at: https://doi.org/10.1016/j.jjimei.2021.100019.

Onan, A., (2021) Sentiment analysis on massive open online course evaluations: A text mining and deep learning approach. *Computer Applications in Engineering Education*, 293, pp.572–589.

el Rahman, S.A., Alotaibi, F.A. and Alshehri, W.A., (2019) Sentiment Analysis of Twitter Data. *2019 International Conference on Computer and Information Sciences, ICCIS 2019*.

Roy, A. and Ojha, M., (2020) Twitter sentiment analysis using deep learning models. *2020 IEEE 17th India Council International Conference, INDICON 2020*, June, pp.0–17.

Rustam, F., Khalid, M., Aslam, W., Rupapara, V., Mehmood, A. and Choi, G.S., (2021) A performance comparison of supervised machine learning models for Covid-19 tweets sentiment analysis. *PLoS ONE*, [online] 162, pp.1–23. Available at: http://dx.doi.org/10.1371/journal.pone.0245909.

Samuel, J., Ali, G.G.M.N., Rahman, M.M., Esawi, E. and Samuel, Y., (2020) COVID-19 public sentiment insights and machine learning for tweets classification. *Information (Switzerland)*, 116, pp.1–22.