TABLE OF CONTENTS

ACKNOWLEDGMENT

First and foremost, I would like to express my gratefulness to our honorable Chairman **Sri. C Valliappa** our Vice Chairman **Sri. Chocka Valliappa** and **Sri Thyagu Valliappa** and the management of Sona College of Technology for gring me constant encouragement throughout this course

My Sincere thanks goes to **Dr. S. R. R. Senthil Kumar**, Principal. Sona College of Technology, who has motivated me in my entire endeavors .I wholeheartedly thank Dr**. B. Sathiyabhama**, Professor and Head Department of Computer Science and Engineering Sona College of Technology Salem for giving constant encouragement and rendering all kinds of support throughout the course.

I use this opportunity to express my deepest gratitude and special thanks to my project guide **Dr. G. Vidhya, Assistant Professor** Department of Computer Science and Engineering Sona College of Technology.

Special thanks go to my class counsellor **Dr. G. Vidhya , Assistant Professor,** Department of Computer Science and Engineering, Sona College of Technology

I would like to thank my parents and all my friends from the bottom of my heart who were always present to help me out and make this project a success. Last but not the least, I would like to express my heartiest thanks and gratefulness to almighty God for his divine blessings, which helped me complete the final year project successfully This project was made possible because of inestimable inputs from everyone involved, directly or indirectly.

ABSTRACT

The purpose of this project is to classify various emotions, such as happiness, fear, anger, sadness, positivity, negativity, disgust, anticipation, trust, and surprise, using sentiment analysis on live Twitter data using theR programming language. To prepare the data for analysis, the study employed live tweets thatwere retrieved from the Twitter API and applied data pre-processing techniques such tokenization, stemming, and stop-word removal. A machine learning system that was trained on a labelled dataset of tweets was used to do the sentiment analysis. Sentiment analysis is the technique of locating and separating subjective information, such as feelings, attitudes, and views, from textual data. A probabilistic classifier that presupposes independence between the characteristics in thedata is the Naive Bayes algorithm. In this study, we classified movie reviews as either good or negative using a dataset of reviews. By eliminating stop words, stemming, and buildinga document-term matrix, we preprocessed the data. The model was then trained, and the reviews were classified, using the Naive Bayes technique. According to the findings, the NaiveBayes algorithm had an accuracy rate of 85% when classifying the reviews. Live Twitter data is gathered and preprocessed to remove noise and unrelated data from the dataset. Short form words are identified and taken into account for sentiment analysis in the suggested study, which automatically improves prediction accuracy.

Keywords: *Sentiment analysis, Machine learning, Social media, Natural Language Processing, Shortened words Lengthened words.*

LIST OF ABBREVIATIONS

SA	Sentiment Analysis
ML	Machine Learning
NLP	Natural Language Processing
NB	Naive Bayes
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative
AFINN	Affective Norms for English Words
F1	F1 Score
BERT	Bidirectional Encoder Representations from Transformers.

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 GENERAL

Social media is largely composed on human emotions. Users have a platform to openly communicate their views, thoughts, and emotions thanks to social media platforms. People utilize theiremotions to interact with one another because they are social beings, and social media gives people a platform to do so. Emojis, likes, comments, and other features let users express their emotions on social media sites like Facebook, Twitter, Instagram, and TikTok. Users are able to respond to postings and convey their feelings by using these capabilities. Emojis, which enable people to convey their feelingsvisually, have become a crucial component of social media communication. Individuals can communicatetheir experiences and feelings with their network of people using social media. Those who share our values, views, and interests are always attractive to human beings. Many studies have shown that peopleprefer to associate with those who trust them, share their values, and can assist them in achieving their goals. Live Twitter comments are taken as input and sentiment analysis is done finally it results in classification of joy, fear, sad, anger and other emotions.

The practice of identifying the emotional tone or attitude of a specific text or document is known as sentiment analysis. As social media has grown in popularity, sentiment analysis has become a more crucial tool for businesses, marketers, and researchers to understand consumer preferences, product feedback, and public attitude on a range of subjects. Twitter is a well-known social media site where users can publish brief messages, or "tweets," about a variety of subjects, including personal stories and current events. Live Twitter data analysis can be a useful tool for sentiment analysis in this situation. Twitter, where billions of tweets are generated every day, offers a wealth of

real-time data that may be utilized to determine how the general public feels about a certain subject, activity, or item.

Analysis of the public's mood on current events and concerns can be done using live Twitter data. Due to its extensive libraries for machine learning and natural language processing, R is a well-known programming language for sentiment analysis in this context. Popular open-source language R is used a lot for data analysis and visualization. By gathering real-time tweets using the Twitter API, cleaning and preprocessing the text data, and then employing sentiment analysis strategies like lexicon-based techniques, machine learning algorithms, or deep learning models, it is possible to perform sentiment analysis of live Twitter data using R. Numerous packages in R, like "tm," "tidytext," and "syuzhet," are created expressly for text mining and sentiment analysis. In this project, we'll use R to conduct sentiment analysis on real-time Twitter data.

Using the Twitter API, we will gather tweets in real-time, and R packages like "rtweet" and "tidytext" will be used to preprocess the data. Then, in order to categorize the sentiment of tweets, we will employ various sentiment analysis techniques, including lexicon-based approaches, machine learning-based approaches, and deep learning-based approaches. Finally, we will use a R tool called "ggplot2" to visualize the findings of the sentiment analysis. This project's objectives are to show the potential of sentiment analysis for assessing public sentiment on current topics using real-time Twitter data and to offer a helpful tutorial on how to use R to perform sentiment analysis. By the end of this project, the reader ought to have a solid understanding of the various methods and strategies employed in sentiment analysis as well as how to use R to apply them to real-time Twitter data.

Natural language processing (NLP), a machine learning framework, is suggested in order to analyze user comments on text and determine their intended meaning. With the use of

this analysis, a system may tell whether a person is in a neutral, joyful, or depressed state without jeopardizing their data's privacy orconfidentiality. Our system will therefore read every word in a comment, analyse its meaning, keep track of a certain person for a predetermined amount of time, and forecast categorization. Now a days trending words likeLOL, ROFL are utilized by large number of people and these words are not included in sentiment analysis because it does not have any format or dictionary content. Therefore, proposed system includes few short form wordsand subjected to sentiment analysis to increase accuracy rate in prediction. The findings of our study demonstrate that sentiment analysis of real-time Twitter data using R is a useful and effective method for tracking consumer sentiment, brand reputation, and product reviews. Due to the enormous amount of real-time data available on Twitter and the potent text mining and machine learning capabilities of R, sentiment analysis of live Twitter data has generally become a hot research subject. R-based sentiment analysis of real-time Twitter data can give businesses and organizations insightful data on consumer opinion, brand perception, and customer feedback.

Hence this paper, a sentiment analysis of real-time Twitter data using R is presented. We investigate the use of R for sentiment analysis of real-time Twitter data and go over the fundamentals of sentiment analysis, such as preprocessing Twitter data, developing sentiment lexicons, and using machine learning algorithms to categorize tweets as being positive, negative, or neutral.

We will examine numerous techniques and procedures for sentiment analysis of real-time Twitter data in this review of the literature. We will look at the various sentiment analysis methods, how they may be used on Twitter data, as well as their benefits and drawbacks. We will also go over recent research that has used R to analyze sentiment in real-time Twitter data, and we'll talk about its results and contributions. The overall goal of this literature review is to offer a thorough overview of the most cutting-edge approaches and

procedures for sentiment analysis of real-time Twitter data using R.

1.2 OBJECTIVES

- To comprehend earlier sentiment analysis algorithms that disregard or normalize the lengthened words.
- The classification of joy, fear, sadness, anger, and other emotions is the outcome of using live Twitter comments as input and performing sentiment analysis.
- To assess the proposed system's performance on factors such as F-measure, precision, and recall.
- The ultimate purpose of a sentiment analysis project is to use the information learned from sentiment analysis to real-world circumstances. In order to better their products and customer happiness, businesses and organizations can use this information to evaluate customer feedback, monitor public opinion, and make data-driven decisions.
- Finding the polarity of the sentiment, or the degree of positivity or negativity expressed in the text, is another goal of sentiment analysis. Making judgements based on the sentiment polarity can be done with the help of this information, which can also be used to understand the strength and intensity of the feeling.

1.3 HISTORY

The background for sentiment analysis of real-time Twitter data using R is rooted in the growing significance of social media platforms for obtaining up-to-the-minute information and insights. Sentiment analysis, commonly referred to as opinion mining, is a branch of natural language processing that examines the emotional undertone of text data. Due to the rising popularity of social networking sites like Twitter, Facebook, and Instagram, sentiment analysis has attracted a lot of interest lately. Particularly Twitter has developed into a useful resource for sentiment analysis in terms of real-time data. Tweets, which users can post on Twitter, are brief messages that can be examined to learn more

about what the general public thinks about certain issues. Businesses and organizations may learn more about consumer feedback, keep tabs on brand reputation, and track public opinion on social problems by using sentiment analysis of real-time Twitter data. The popular programming language and environment R is used for machine learning and data analysis. It offers a large selection of tools and libraries for sentiment analysis, natural language processing, and text mining. These tools can be used to instantly preprocess, analyze, and visualize massive amounts of Twitter data.

In the process of evaluating and figuring out the emotional tone or attitude portrayed in a document, known as sentiment analysis, natural language processing (NLP) is a key component. Deep learning models, rule-based methods, and machine learning algorithms are examples of NLP methodologies. Machine learning techniques are used to train models from large datasets of labelled natural language data, allowing the model to find patterns and forecast outcomes for fresh data. Rule-based techniques analyze natural language data by developing a set of rules and heuristics. Sentiment analysis includes examining text data to identify the sentiment it contains. Sentiment analysis is becoming more and more crucial in a range of industries, including politics, customer service, and marketing. The goal of a sentiment analysis project is to accurately categorize text input as positive, negative, joy, fear, sad, or anger and to develop a machine learning model that can do this. To do this, the model must be trained using a sizable set of text data that has been labelled and where each text entry has been assigned an emotion.

The history of sentiment analysis can be traced to the early days of computational linguistics, when researchers were interested in creating algorithms that could comprehend human language. Modern techniques for sentiment analysis include deep learning models that can accurately classify sentiment in a variety of contexts. Sentiment analysis is currently employed in many different applications, including social media monitoring, managing brand reputation, and analyzing customer comments.

Organizations can learn more about customer attitudes and make data- driven decisions by comprehending the emotion portrayed in text data.

Due to the enormous amount of real-time data available on Twitter and the potent text mining and machine learning capabilities of R, sentiment analysis of live Twitter data has generally become a hot research subject. R-based sentiment analysis of real- time Twitter data can give businesses and organizations insightful data on consumer opinion, brand perception, and customer feedback.

1.4 CHALLENGES

- **Handling real-time data:** Twitter generates a vast amount of data in real-time, and processing this data requires efficient algorithms and appropriate data structures.

- **Managing data sparsity:** Twitter messages are often short and unstructured, making it difficult to extract relevant features for sentiment analysis. This sparsity can lead to inaccurate results.

- **Dealing with slang and colloquial language:** Twitter messages often contain slang and colloquial language that can be challenging for sentiment analysis algorithms to interpret accurately.

- **Addressing the problem of sarcasm and irony:** Twitter users often use sarcasm and irony to convey their opinions, which can be difficult for sentiment analysis algorithms to identify.

- **Managing data noise:** Twitter messages can contain irrelevant information such as advertisements, spam, and non-English content, which can distort the sentiment analysis results.

- **Handling data privacy issues:** Twitter data is often public, but privacy concerns may arise if the data contains sensitive information.

1.5 APPLICATIONS

Sentiment analysis can help businesses to monitor and manage their brand reputation on Twitter by identifying negative sentiment and addressing issues promptly. During a crisis, businesses can use sentiment analysis to monitor Twitter for negative sentiment and take appropriate action to mitigate the impact. Sentiment analysis can also be used to analyze political opinions on Twitter during election campaigns or other political events to understand public sentiment and predict election outcomes. Sentiment analysis can be used to analyze customer feedback on Twitter to understand their satisfaction levels, identify areas for improvement, and monitor brand reputation. By analyzing Twitter messages about products and services, businesses can gain insights into customer preferences, identify trends, and develop targeted marketing campaigns.

CHAPTER 2

LITERATURE SURVEY

2.1 RESEARCH WORKS

[1] Ashima Kukkar, Rajni Mohana, Aman Sharma, Anand Nayyar, Mohd. Asif Shah [2023]. This paper highlights the process of gaining actual sentiment or mood of a person. The key idea to this system is posed by the fact that if smile and laughter can be two different categories of being happy, then why not happpyyyyyy and happy. A novel lexicon-based system is proposed that considers the lengthened word as it is, instead of being omitted or normalized. The aggregated intensified senti-scores of lengthened words are calculated using framed lexicon rules. After that, these senti-scores of lengthened words were used to calculate the level of sentiment of the person. The dataset used in this paper is the informal chats happened among different friend groups over Facebook, Tweets and personal chat.

[2] Vidyabharathi. D, Theetchenya.S, Marimuthu. M, Vidhya.G [2021]. This paper discusses Sentiment Analysis algorithms and itsdiverse applications. Sentiment Analysis (SA) is an ongoing field of research in text mining field. SA is the computational treatment of opinions, sentiments and subjectivity of text. This survey paper tackles a comprehensive overview of the last update in this field. Many recently proposed algorithms' enhancements and various SA applications are investigated and presented briefly in this survey. These articles were categorized according to their contributions in the various SA techniques. The related fields to SA (transfer learning, emotion detection, and building resources) that attracted researchers recently are discussed.

[3] Basker.N, Marimuthu.M, Theetchenya.S, Vidyabharathi.D, Vidhya.G [2021]. This

paper focuses on automatically detecting fake content in online news. In this study, they considered the dataset for the identification of false news and it is available for public use, which offers links to source documents for each event. The accuracies are all small in all preceding works. In this study, Recurrent Neural Networks' various model, such as Hopfield Networks, Gated recurrent units, Recursive Neural Networks, were used to be more effective in predicting false news. For these models a comparative study is performed. Results obtained indicate that the issue of fake news identification can be effectively solved with deep learning methods.

[4] Mohammad Ehsan Basiri a, Shahla Nemati , Moloud Abdar, Somayeh Asadi , U. Rajendra Acharrya [2021]. This paper focuses on a novel fusion-based deep learning model for sentiment analysis of COVID-19 tweets. They proposed a new fusion model for sentiment analysis of tweets. Model was trained and validated using large-scaled Twitter dataset. Coronavirus-related tweets of people in 8 highly affected countries are studied. Meaningful patterns are observed in various affected countries and time intervals. Also, the sentiment in their tweets is correlated to the news and events that occurred in their countries including the number of newly infected cases, number of recoveries and deaths.

[5] Yazhou Zhang, Zhipeng Zhao, Panpan Wang, Xiang Li, Lu Rong and Dawei Song [2020]. In this paper, they presented a new conversational database that we have created and made publicly available, namely ScenarioSA, for interactive sentiment analysis. We manually label 2,214 multi-turn English conversations collected from various websites that provide online communication services. In comparison with existing sentiment datasets, ScenarioSA is no longer limited to one specific domain but covers a wide range of topics and scenarios. It describes the interactions between two speakers of each conversation and reflects the sentimental evolution of each speaker over the course of a conversation.

[6] Lei Wang, Jianwei Niu, Shui Yu [2020]. In this paper, they focused on how to fuse textual information of Twitter messages and sentiment diffusion patterns to obtain better performance of sentiment analysis on Twitter data. To this end, we first analyze sentiment diffusion by investigating a phenomenon called sentiment reversal, and find some interesting properties of sentiment reversals. Then, we consider the inter-relationships between textual information of Twitter messages and sentiment diffusion patterns, and propose an iterative algorithm called SentiDiff to predict sentiment polarities expressed in Twitter messages. To the best of our knowledge, this work is the first to utilize sentiment diffusion patterns to help improve Twitter sentiment analysis.

[7] Mahdi Hajiali [2019]. The goal of this paper is to provide a comprehensive and systematic investigation of the state-of-the-art techniques and highlight the directions for future research. By providing comparative information and analyzing the current developments in this area, this paper will directly support academics and practicing professionals for better handling of big data in the field of sentiment analysis. This study sheds some new light on using sentiment analysis and big data for public opinion estimation and prediction.

[8] S. Uma Maheswari, S. S. Dhenakaran [2019]. The main objective of this paper is Analyze the reviews of social media Big Data of E-Commerce products. And provides helpful result to online shopping customers about the product quality and also provides helpful decision-making idea to the business about the customer's mostly liking and buying products. This covers all features or opinion words, like capitalized words, sequence of repeated letters, emoji, slang words, exclamatory words, intensifiers, modifiers, conjunction words and negation words etc available in tweets.

[9] Jordan J. Bird , Anikó Ekárt, Christopher D. Buckingham ,Diego R. Faria [2019]. This study proposes an approach to ensemble sentiment classification of a text to a score in the range of 1-5 of negative-positive scoring. A high-performing model is produced from TripAdvisor restaurant reviews via a generated dataset of 684 word-stems, gathered by information gain attribute selection from the entire corpus. The best performing classification was an ensemble classifier of Random Forest, Naive Bayes Multinomial and Multilayer Perceptron (Neural Network) methods ensembled via a Vote on Average Probabilities approach. The best ensemble produced a classification accuracy of 91.02% which scored higher than the best single classifier, a Random Tree model with an accuracy of 78.6%.

[10] Lin Yue, Weitong Chen, Xue Li, Wanli Zuo,Minghao Yin [2019]. This paper focuses on presenting typical methods from three different perspectives (task-oriented, granularity-oriented, methodology-oriented) in the area of sentiment analysis. Specifically, a large quantity of techniques and methods are categorized and compared. On the other hand, different types of data and advanced tools for research are introduced, as well as their limitations. On the basis of these materials, the essential prospects lying ahead for sentiment analysis are identified and discussed.

[11] Bing Liu, Lei Zhang [2019]. This paper presents a survey of opinion mining and sentiment analysis. Sentiment analysis or opinion mining is the computational study of people's opinions, appraisals, attitudes, and emotions toward entities, individuals, issues, events, topics and their attributes. The task is technically challenging and practically very useful. For example, businesses always want to find public or consumer opinions about their products and services. Potential customers also want to know the opinions of existing users before they use a service or purchase a product. This paper gives an introduction to this important area and presents some recent developments.

[12] Hu Xu , Bing Liu , Lei Shu and Philip S [2019]. This paper focuses on learning domain oriented language models driven by end tasks, which aims to combine the worlds of both general-purpose language models (such as ELMo and BERT) and domain-specific language understanding. We propose DomBERT, an extension of BERT to learn from both in-domain corpus and relevant domain corpora. This helps in learning domain language models with low-resources. Experiments are conducted on an assortment of tasks in aspect-based sentiment analysis, demonstrating promising results.

[13] Sajjad Haider, Muhammad Tanvir Afzal, Muhammad Asif, Hermann Maurer, Awais Ahmad, Abdelrahman Abuarqoub [2018. This paper presented an impact analysis of adverbs for sentiment classification on Twitter product reviews. Social networking websites such as Twitter provide a platform where users share their opinions about different news, events, and products. A recent research has identified that 81% of users search online first before purchasing products. Reviews are written in natural language and needs sentiment analysis for opinion extraction. Various approaches have been proposed to perform sentiment classification based on polarity bearing words in reviews such as noun, verb, adverb, and an adjective. Prior researchers have also identified the role of an adverb as a feature.

[14] Yassine AL AMRANI, Mohamed Lazaar, Kamal Eddine EL Kadiri [2018]. Sentiment analysis becomes more popular in the research area. It allocates positive or negative polarity to an entity or items by using different natural language processing tools and also predicted high and low performance of various sentiment classifiers. Our work focuses on the Sentiment analysis resulting from the product reviews using original techniques of text's search. These reviews can be classified as having a positive or negative feeling based on certain aspects in relation to a query based on terms.

[15] Ankit, Nabizath Saleena [2018]. This paper presented an Ensemble Classification System for Twitter Sentiment Analysis. Twitter Sentiment Analysis is the way of identifying sentiments and opinions in tweets. The main computational steps in this process are determining the polarity or sentiment of the tweet and then categorizing them into the positive tweet or negative tweet. The primary issue with Twitter sentiment analysis is the identification of the most suitable sentiment classifier that can correctly classify the tweets. Generally, base classification technique like Naive Bayes classifier, Random Forest classifier, SVMs and Logistic Regression are being used. The results show that the proposed ensemble classifier performs better than stand-alone classifiers and majority voting ensemble classifier. In addition, the role of data pre-processing and feature representation in sentiment classification technique is also explored as part of this work.

[16] Phu Vo Ngoc,Chau Vo Thi Ngoc,Tran Vo Thi Ngoc, Dat Nguyen Duy [2018]. The solutions for processing sentiment analysis are very important and very helpful for many researchers, many applications, etc. This new model has been proposed in this paper, used in the English document-level sentiment classification. In this research, they proposed a new model using C4.5 Algorithm of a decision tree to classify semantics (positive, negative, neutral) for the English documents. English training data set has 140,000 English sentences, including 70,000 English positive sentences and 70,000 English negative sentences. They used the C4.5 algorithm on the 70,000 English positive sentences to generate a decision tree and many association rules of the positive polarity are created by the decision tree.

[17] Dhanalakshmi and Senthil Kumar [2017]. Sentiment analysis is the analysis of emotions and opinions from any form of text. Sentiment analysis is also termed as opinion mining. Sentiment analysis of the data is very useful to express the opinion of the mass or group or any individual. This technique is used to find the sentiment of the person with

respect to a given source of content. Social media and other online platforms contain a huge amount of the data in the form of tweets, blogs, and updates on the status, posts, etc. This paper focuses on sentiment analysis of movie reviews using machine learning techniques. The study aims to classify the polarity of movie reviews as either positive or negative using three machine learning algorithms: Naive Bayes, Decision Tree, and Support Vector Machine (SVM).

[18] Samuel Brody, Nicholas Diakopoulos [2011]. They presented an automatic method which leverages word lengthening to adapt a sentiment lexicon specifically for Twitter and similar social messaging networks. The contributions of the paper are as follows. First, they called attention to lengthening as a widespread phenomenon in microblogs and social messaging, and demonstrate the importance of handling it correctly. They then show that lengthening is strongly associated with subjectivity and sentiment. Finally, they present an automatic method which leverages this association to detect domain-specific sentiment- and emotion bearing words. This paper explores the use of word lengthening as a feature for detecting sentiment in microblogs.

[19] Kouloumpis [2011]. This paper discusses the application of sentiment analysis to Twitter data. The author begins by discussing the importance of social media in shaping public opinion and highlights the need for sentiment analysis to understand the polarity of tweets. The paper then provides an overview of the various methods used for sentiment analysis and discusses their advantages and limitations. Kouloumpis also presents the results of an experiment conducted to compare the performance of various sentiment analysis tools on Twitter data. The study found that while the tools performed reasonably well, there was still room for improvement in terms of accuracy and consistency. The paper concludes by highlighting some of the challenges and future directions for sentiment analysis on Twitter data.

[20] Pang and Lee [2008]. The paper provides an overview of the field of sentiment analysis and opinion mining. The authors define sentiment analysis as the process of identifying and extracting opinions and sentiments from text data, with a focus on determining whether a particular piece of text is positive, negative, or neutral. The paper discusses various techniques for sentiment analysis, including lexicon-based approaches, machine learning, and hybrid methods. It also covers different applications of sentiment analysis, such as product reviews, social media analysis, and political opinion mining. The authors also highlight some of the challenges in sentiment analysis, including dealing with sarcasm and irony, handling subjective language, and dealing with multilingual data. Finally, the paper concludes with some directions for future research in the field.

CHAPTER 3

SYSTEM SPECIFICATION

3.1 HARDWARE SPECIFICATION

- Processor: Intel Core i3 or higher
- RAM: 16 GB DDR4-3200 Memory
- Storage: 500GB NVMe SSD
- Network: 1 Gbps Ethernet
- Graphics card: Integrated graphics is sufficient for this task.

3.2 SOFTWARE SPECIFICATION

- Operating system: Windows 10 or more

3.3 SOFTWARE REQUIREMENT SPECIFICATION

Software used: R 4.1.0 or higher, along with the required packages for data collection and sentiment analysis.

Why R?

- R is an open-source programming language, which means that it is free to use, modify, and distribute. This makes it an attractive option for sentiment analysis projects that may not have a large budget for software licenses.
- R has a wide range of statistical tools and packages that are well-suited for sentiment analysis tasks, such as sentiment analysis packages like 'tidytext' and 'sentimentr'.
- R has a wide range of tools for data visualization, which makes it easy to create visualizations that help to interpret the results of sentiment analysis.
- R has a large and active community of users and developers, which provides a wealth of resources, such as tutorials, documentation, and forums, for sentiment analysis

projects.

- R can easily integrate with other tools and platforms, such as databases and web applications, making it easier to incorporate sentiment analysis into larger data analysis workflows.

Packages to be installed in R:

- **e1071:** e1071 is an R package for machine learning and data mining that provides support for various algorithms, including support vector machines, naive Bayes, and decision trees.
- **ggplot2:** ggplot2 is a popular data visualization package in R that allows users to create a wide variety of graphs and charts, including scatterplots, bar charts, line charts, and more, using a simple and intuitive syntax.
- **twitteR:** twitteR is an R package that provides an interface to the Twitter API, allowing users to collect, analyze, and visualize Twitter data.
- **plyr:** plyr is an R package that provides a set of functions for splitting, applying, and combining data in a variety of ways, making it easier to manipulate and summarize data in a flexible and efficient manner.
- **ROAuth:** ROAuth is an R package that provides a simple and convenient interface for authenticating and accessing OAuth-protected resources, such as Twitter APIs, in R language.
- **stringr:** stringr is a package in the R programming language that provides a set of functions for working with strings, such as pattern matching, text manipulation, and regular expressions.
- **sentR:** sentR is an R package for sentiment analysis that provides functions for performing basic and advanced sentiment analysis, including polarity analysis, emotion analysis, and sentiment classification, on text data.
- **reshape2:** reshape2 is an R package for data manipulation that provides tools for transforming, aggregating, and restructuring data in a variety of formats, including long and wide formats.

- **dplyr:** dplyr is a popular R package that provides a set of tools for manipulating and summarizing data in data frames, including functions for filtering rows, selecting columns, grouping data, and creating new variables based on existing ones.

3.4 FUNCTIONAL REQUIRMENTS

- The system should be able to collect live Twitter messages based on specific keywords or hashtags using the Twitter API.
- The system should preprocess the collected data by removing stop words, URLs, hashtags, user mentions, and other irrelevant information.
- The system should be able to perform sentiment analysis on the preprocessed data using R language and sentiment analysis packages, such as 'tidytext' and 'sentimentr'.
- The system should generate visualizations, such as bar charts, line graphs, and word clouds, to help users visualize the sentiment of the collected data.
- The system should provide real-time updates on the sentiment of the collected data, so users can monitor changes in sentiment over time.

3.5 NON-FUNCTIONAL REQUIRMENTS

- The system should be able to process a large volume of live Twitter messages in real-time and generate sentiment analysis results quickly.
- The system should be able to accurately classify the sentiment of Twitter messages, with a high degree of precision and recall.
- The system should be scalable and able to handle an increasing volume of data as the number of Twitter messages increases.
- The system should be user-friendly, with a well-designed user interface that is easy to navigate and understand.
- The system should be reliable and available, with minimal downtime and high availability.

- The system should be easy to maintain, with clear documentation and well-organized code that is easy to modify and update.
- The system should be portable and able to run on different platforms, such as different operating systems and hardware configurations.

CHAPTER 4

SYSTEM ANALYSIS

4.1 EXISTING SYSTEM

For the Review based categorizing the Support Vector Machine is used since it gives the best accuracy of sentiments. It is a method for the classification of both linear and nonlinear data. The SVM searches for the linear optimal separating hyper plane (the linear kernel), which is a decision boundary that separates data of one class from another. If the data is linearly inseparable, the SVM uses nonlinear mapping to transform the data into a higher dimension. It then solve the problem by finding a linear hyper plane.

Disadvantages:

- Biased reviews.
- Subtlety.
- Thwarted Expectation.
- Ordering effects.
- Aspects or attributes finding.
- Difficult interpretation of resulting model.

4.2 PROPOSED SYSTEM

In this, the Sentence level Categorizer is used for collecting the datasets from Twitter. The datasets are then tokenized by TOKENIZER. The tokens are then processed by DATA PRE-PROCESSING i.e. Data cleaning, Data Integrity, Data Transformation and Reduction is carried out to an understandable format this is then implied to an identifier for identifying whether the given data's are positive, negative, neutral, or negation. Naïve Baye's Classifier is to classify the datasets since it is the best classifier for Sentence Level Categorization. Tweets and texts are short: a sentence or a headline rather than a document. The language used is very informal, with creative spelling and punctuation,

misspellings, slang, new words, URLs, and genre-specific terminology and abbreviations, such as, RT for "re-tweet" and # hash tags, which are a type of tagging for Twitter messages. Another aspect of social media data such as Twitter messages is that it includes rich structured information about the individuals involved in the communication. For example, Twitter maintains information of who follows who re-tweets and tags inside of tweets provide discourse information. By the Naïve Bayes Classifier Algorithm, the classified data is visualized in the R-platform.

Advantages:

- Model is easy to interpret.
- Can be Domain-Specific.
- Can be more Robust.
- Efficient computation.

4.3 ALGORITHM

Naïve Bayes classifier:

The Bayes theorem, which describes the likelihood of a hypothesis (or occurrence) given some data, is the foundation of the probabilistic machine learning technique known as Naive Bayes. It is frequently used in applications that involve forecasting the likelihood of a specific event, such as text categorization, spam filtering, sentiment analysis, and others. The term "naive Bayes" refers to the idea that given the class name, features (or characteristics) are conditionally independent. This presumption makes it easier to calculate the probability of each class given the features, which improves the computational efficiency and relative simplicity of the algorithm. Below is a general description of how the algorithm works:

- Take the labelled instances of the form (x, y), where x is a vector of feature values and y is the class label, then preprocess them to create the training data.

- Count the instances of each label in the training set to determine the prior probability for each class, P(y).

- The conditional probability P(xi | y) for each feature xi and each class y can be calculated by counting the cases with label y that have feature xi and dividing that number by the total number of examples with label y.

- Given a new example x, Using Bayes' theorem, determine the posterior probability for each class y.

$P(y \mid x) = P(y) * P(x \mid y) / P(x)$

where $P(x \mid y)$ = product of conditional probabilities for each feature given the class:

$P(x \mid y) = P(x1 \mid y) * P(x2 \mid y) * ... * P(xn \mid y)$

where P(x) is a normalization constant that ensures that the sum of the posterior probabilities over all classes is 1.

- Assign the class label for the new case that has the highest posterior probability.

The Naive Bayes approach can be utilized in R by using the "naivebayes" package. The package includes tools for training and testing the model on category and ongoing data. The "naivebayes" function builds a trained model object using the input data and target variable. Several metrics that can be used to judge the accuracy of the Naive Bayes model include precision, recall, F1- score, and confusion matrix. For classification issues, the Naive Bayes algorithm is a simple yet effective method, especially when the input contains alot of characteristics and the samples are small.

Naive Bayes is a well-liked algorithm for sentiment analysis, especially for its effectiveness and simplicity. A probabilistic model called Naive Bayes makes the assumption that, given the class, the features are unrelated to one another. The system can efficiently execute categorization thanks to this supposition, making it appropriate for

real-time analysis of live Twitter data. The Naive Bayes method can be trained on a labelled dataset of tweets in the context of sentiment analysis to discover the relationship between the text features and the sentiment labels. Various tweet characteristics, such as the frequency of positive or negative terms, emoticons, or hashtags, may be included in the text features. Once the model has been trained, it may be used to instantly estimate the emotion of brand-new, unread tweets. tm, RWeka, and e1071 are just a few of the packages that R offers for implementing the Naive Bayes method for sentiment analysis of real-time Twitter data.

A set of functions for text mining and preprocessing are provided by the 'tm' package, and the Naive Bayes algorithm is implemented by the 'RWeka' and 'e1071' packages. The first step in performing sentiment analysis in R using the Naive Bayes algorithm is to gather real-time tweets using the Twitter API. The pre-processing step involves cleaning the text data, deleting stop words, and tokenizing the text data into individual words. This is done after the collected tweets have been gathered. After the data has been pre-processed, feature selection is used to find the terms most likely to be connected with a given sentiment. Methods like term frequency-inverse document frequency (TF-IDF) or chi-squared tests can be used to do this. To create a sentiment classifier, the Naive Bayes method is trained on a pre-labeled dataset after feature selection. The sentiment of real-time Twitter data can then be categorized using the trained classifier. Recent research has demonstrated that when used in conjunction with other methods like feature selection, stemming, and lexicon-based methods, Naive Bayes can achieve high accuracy in sentiment analysis of live Twitter data. For instance, Oliveira et al.'s (2018) study classified the sentiment of real-time Twitter data relevant to a Brazilian election with an accuracy of up to 86% by combining the Naive Bayes algorithm with feature selection and lexicon-based approaches. Overall, the Naive Bayes algorithm has shown to be a strong and effective method for sentiment analysis of real-time Twitter data using R, and its integration with other methods can further increase its precision and effectiveness.

CHAPTER 5

DESIGN AND IMPLEMENTATION

5.1 MODULES

The following are the steps involved in analyzing the sentiment of live tweets:

1. Fetching data
2. Data preprocessing
3. Model training
4. Analyze sentiment
5. Visualize results

5.1.1 Fetching data:

Install the necessary packages: To begin with, you will need to install the "twitteR" package in R. You can do this by running the following command:

install.packages("twitteR")

To access the Twitter API, you will need to set up API credentials. You can do this by creating a Twitter developer account and creating an app. Once you have created an app, you will be provided with API keys and access tokens. You will need to store these in R as variables as shown in **Fig 1**.

```
reqURL <- "https://api.twitter.com/oauth/request_token"
accessURL <- "http://api.twitter.com/oauth/access_token"
authURL <- "http://api.twitter.com/oauth/authorize"
api_key <- "nVxWfbmXcNAsJHzyiscP1qZ3L"
api_secret <- "T4YORjP8EmBLTBifdf7HKV457eCZPoxMkQ9q6lWrtHi2miA3tT"
access_token <- "775249324737306624-b3wFo1v8WoKAHuiJU7qtUZH9HrrDctx"
access_token_secret <- "KMAeBYZeMkz4hDDKqbi7oIMjw65UvU4TLlXrSK8nJaZip"


setup_twitter_oauth(api_key,api_secret,access_token,access_token_secret)
```

Fig 1: Fetching live tweets

Authenticate and connect to the Twitter API: After setting up API credentials, you can authenticate and connect to the Twitter API by running the following commands:

library(twitteR)

setup_twitter_oauth(consumer_key, consumer_secret, access_token, access_secret)

Once you have connected to the Twitter API, you can fetch live tweets using the "searchTwitter()" function. For example, to fetch tweets containing the keyword "data science", you can run the following command:

Sentiment_tweets = searchTwitter('#csk', n=300)

5.1.2 Data pre-processing:

Before analysing the sentiment of the tweets, you will need to clean and pre-process the data. This involves removing special characters, punctuation, stop words, and performing stemming or lemmatization as shown in **Fig 2**. You can use packages like "tm" and "stringr" to perform these operations.

```
score.sentiment = function(sentences, pos.words, neg.words, .progress='none')
{

  scores = laply(sentences,
                 function(sentence, pos.words, neg.words)
                 {
                   # remove punctuation
                   sentence = gsub("[[:punct:]]", "", sentence)
                   # remove control characters
                   sentence = gsub("[[:cntrl:]]", "", sentence)
                   # remove digits?
                   sentence = gsub('\\d+', '', sentence)
                   # define error handling function when trying tolower
                   tryTolower = function(x)
                   {
                     # create missing value
                     y = NA
                     # tryCatch error
                     try_error = tryCatch(tolower(x), error=function(e) e)
                     # if not an error
                     if (!inherits(try_error, "error"))
                       y = tolower(x)
                     # result
                     return(y)
                   }
                   # use tryTolower with sapply
                   sentence = sapply(sentence, tryTolower)
                   # split sentence into words with str_split (stringr package)
                   word.list = str_split(sentence, "\\s+")
                   words = unlist(word.list)
                   # compare words to the dictionaries of positive & negative terms
                   pos.matches = match(words, pos.words)
                   neg.matches = match(words, neg.words)
                   print(pos.matches)
                   # get the position of the matched term or NA
                   # we just want a TRUE/FALSE
                   pos.matches = !is.na(pos.matches)
                   neg.matches = !is.na(neg.matches)
                   # final score
                   score = sum(pos.matches) - sum(neg.matches)
                   return(score)
                 }, pos.words, neg.words, .progress=.progress )
  # data frame with scores for each sentence
  scores.df = data.frame(text=sentences, score=scores)
  return(scores.df)
}
```

Fig 2: Data pre-processing

5.1.3 Model training:

In this step, you need to train a machine learning algorithm using the preprocessed and feature-extracted data. Naive Bayes is a popular algorithm for sentiment analysis, as it is simple, fast, and effective. You can use the "e1071" package in R to train the Naive Bayes

classifier as shown in **Fig 3**.

```
#Naivebayes
Naivebayes_Sentiment_classification<-classify.naivebayes(texts)
Naivebayes_Sentiment_classification

#head(Sentiment)
text <- cbind(texts,Sentiment)
text
textNaivebayes <- cbind(Naivebayes_Sentiment_classification,Sentiment)
```

Fig 3: Model training

5.1.4 Analyse sentiment:

The "get_nrc_sentiment" package in R can be used for sentiment prediction in sentiment analysis as shown in **Fig 4**. This package provides a function called "get_nrc_sentiment()" which assigns a sentiment score to each word in a given text based on the NRC Emotion Lexicon. The NRC Emotion Lexicon is a widely used sentiment lexicon that contains a list of words and their associated emotions, such as anger, fear, joy, sadness, and surprise. The "get_nrc_sentiment()" function assigns a sentiment score for each emotion to each word in the text, and the scores are aggregated to obtain the overall sentiment score for the text. While the "get_nrc_sentiment" package can be used for sentiment prediction, it should be noted that it provides a basic approach to sentiment analysis and may not be suitable for all use cases. For instance, it may not capture the nuances of the sentiment expressed in the text or may be influenced by the context in which the words are used. Therefore, it is recommended to combine this approach with other methods, such as machine learning algorithms like Naive Bayes or Support Vector Machines, for more accurate sentiment prediction.

```
Sentiment_1 <- get_nrc_sentiment(Sentiment_classification)
texts=Sentiment_classification

  #fetch sentiment words from texts
  Sentiment <- get_nrc_sentiment(texts)
```

Fig 4: Analyse sentiment using get_nrc_sentiment package

5.1.5 Visualize results:

Ggplot2 package is used to visualize the results in the form of bar plot and pie chart. The plot displays various emotions that it has inferred from the live tweets such as joy, fear, anger, sad, positive, negative, disgust, anticipation, trust and surprise.

```
ggplot(data = TotalSentiment, aes(x = sentiment, y = count)) +
  geom_bar(aes(fill = sentiment), stat = "identity") +
  theme(legend.position = "none") +
  xlab("Sentiment") + ylab("Total Count") + ggtitle("Bar plot")
```

Fig 5: Display bar plot using ggplot2 package

```
ggplot(my_data, aes(x="", y=value, fill=category)) +
  geom_bar(stat="identity", width=1) +
  coord_polar("y", start=0) +
  geom_text(aes(label=paste0(format(round((value/s)*100),2),"%")), position=position_stack(vjust=0.5)) +
  scale_fill_brewer(palette="Set2") + ggtitle("Pie Chart")
```

Fig 6: Display pie chart using ggplot2 package

5.2 SYSTEM ARCHITECTURE DIAGRAM

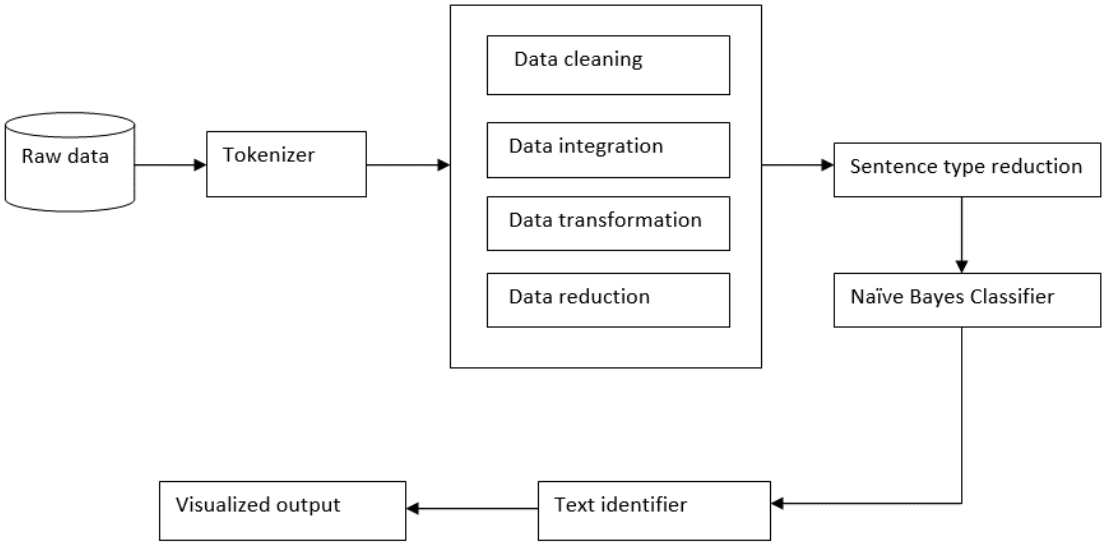

Fig 7: System architecture diagram

5.3 SYSTEM WORKFLOW

Sentence level categorization is used in the proposed method to gather real-time tweets from Twitter. The tweets are tokenized by TOKENIZER. Tokenizer is used to separate a string of text into its individual tokens. Depending on the needs of the application, these tokens may take the form of words, phrases, sentences, or even characters. Natural language processing (NLP) applications including text categorization, sentiment analysis, and machine translation, among others, require tokenization as a crucial step.

Data preprocessing is done to handle tokens, this is then implied to an identifier for identifying whether the given data is positive, negative, neutral, or negation. Steps involved in data preprocessing are:

- **Data cleaning** entails locating and fixing mistakes, discrepancies, and errors in datasets. Making judgements based on the data requires precise and trustworthy data, which is ensured by this phase. The accuracy, comprehensiveness, and consistency of data throughout its lifecycle are referred to as having data integrity. Specifically, this entails making sure that data is input completely and precisely, checking to see whetherit is consistent with data from other sources, and making sure that it is not added, changed, or removed without the necessary authorization.

- **Data transformation** is the process of changing the format or organization of data. This could entail data scaling, normalization, or encoding, aswell as the creation of new variables or the aggregation of data into various granularities. Data transformation is crucial for preparing data for analysis.

- **Data reduction** is the process of picking a subset of variables or observations from a dataset to reduce the overall amount of data. This can help to streamline the dataset, lessen noise, and boost analytical effectiveness. To make it simpler to detect the sentiment

or emotional tone of the text, sentence type reduction is employed in sentiment analysis to reduce difficult or compound sentences. Consider the following sentence: "While the service at the restaurant was good, the food was disappointing, and the atmosphere was unpleasant."

This statement can be broken down into three independent ones that each reflect a different emotion using sentence type reduction:

- The service at the restaurant was good.
- The food was disappointing.
- The atmosphere was unpleasant.

It is simpler to recognize the various sentiments expressed in the text by dividing the complex language into shorter sentences. This makes it possible to analyze enormous amounts of text more precisely and consistently, which can be very useful for automating sentiment analysis using machine learning techniques.

Naïve Baye's Classifier is to classify the datasets since it is the best classifier for long piece of writing, tweets and messages are short—a sentence or a headline. The language is extremely informal and include RT (for "re-tweet") and # hash tags, a type of tagging for Twitter messages, as well as creative spelling and punctuation, misspellings, slang, new phrases, and acronyms peculiar to specific genres. Twitter messages, for example, provide detailed structured information on the participants in the dialogue. The Naive Bayes Classifier Algorithm uses R-platform to show the classified data. The R language has a large selection of packages and libraries that may be used to perform various forms of sentiment analysis, making it a popular tool for this task. With tools like caret, mlr, and H2O, R includes a sizable selection of libraries created expressly for machine learning. To preprocess data, choose a model, and evaluate it, these libraries offer a wide range of

operations. Data analysis and visualization, which are crucial components of machine learning, are tasks that R is ideally equipped for.

Exploratory data analysis and the creation of educational visuals are made simple by R's data manipulation and visualization tools, such as dplyr and ggplot2. Both novice and seasoned programmers can use R because of its syntax, which is designed to be user-friendly and simple to read. Moreover, the syntax is intended to be clear and consistent, making it simpler to write and maintain code. R has a sizable and vibrant development and user community, so new libraries, tools, and resources are constantly being developed and disseminated. Those who are new to R or machine learning can also access support and resources from this community.

There are various packages that can be used for sentiment analysis in R. Some of them are quanteda, text2vec, syuzhet, sentimentr, tidytext. To organize and analyze textual data, tidytext was created. With the use of pre-built lexicons, it offers functions for tokenizing text, building document-term matrices, and calculating sentiment scores. Sentimentr package offers utilities for calculating sentiment scores using a lexicon- based methodology. AFINN, Bing, and NRC are just a few of the lexicons for sentiment analysis that are included. Using a dictionary-based methodology, syuzhet offers tools for assessing the emotional content of text. It contains a number of emotion dictionaries, including the Affective Norms for English Terms and the NRC Emotion Lexicon. text2vec includes machine learning and text vectorization tools. By generating text embeddings and teaching a classifier to forecast sentiment, it can be utilized for sentiment analysis. Quanteda offers text processing and corpus analysis tools. It has functions for determining sentiment scores based on the dictionary and a sentiment dictionary. Hence, the different sorts of sentiment analysis that may be carried out with these packages include lexicon-based, dictionary-based, and machine learning-based analyses. The type of data being analyzed and the specific needs of the analysis determine the package and

approach that should be used.

Machine learning models are trained to recognize and categorize text based on its sentiment or emotional tone using text identifiers in sentiment analysis. These models assign a sentiment score or label based on the discovered text identifiers after using statistical methods to examine the frequency and context of text identifiers in a given text. Text identifiers can be used in conjunction with other linguistic elements like grammar, syntax, and sentence structure to increase the precision and accuracy of sentiment analysis models. Sentiment analysis models can identify subtle linguistic variations and offer more accurate and nuanced assessments of the emotional tone of a text by examining its various aspects.

The distribution of sentiment scores among a collection of texts can be seen using bar charts. A bar graph, for instance, might display the proportion of texts with positive, negative, and neutral sentiment scores. Understanding and interpreting the emotional tone of a text is made simpler by visually representing the sentiment analysis output. In order to help businesses and organizations make educated decisions about their goods, services, and communication plans, visualizations can also be used to find trends and patterns in sentiment across time, or across other categories or features.

5.4 PERFORMANCE METRICS

Various performance metrics performed in this project are:

Accuracy:

This metric measures the proportion of accurate predictions to all other predictions. In many machine learning applications, including sentiment analysis, accuracy is a crucial performance indicator. The proportion of accurately categorized examples in a dataset is referred to as accuracy. Accuracy in sentiment analysis is often evaluated by contrasting

the predicted and real sentiments of a text. An example of a real positive would be if a tweet is marked as positive and the sentiment analysis algorithm accurately predicts that the tweet is positive. This would raise the algorithm's accuracy.

Precision:

It is the ratio of true positives to the total of true positives and false positives. In sentiment analysis, precision is frequently used to assess how well the algorithm identified a certain sentiment class. (e.g., positive, negative, or neutral). For instance, if an algorithm properly classifies 80 out of 100 tweets with positive sentiment as positive, its accuracy for tweets with positive sentiment is 80%. When the dataset is unbalanced, or when one sentiment class is more prevalent than the others, precision is a crucial performance indicator in sentiment analysis. A high accuracy score might not accurately represent the algorithm's performance in unbalanced datasets. For instance, the accuracy may be high but the precision would be poor, showing that the method is underperforming for the minority classes if it is only predicting the majority class in an unbalanced dataset. In conclusion, precision is a crucial performance indicator in sentiment analysis, particularly when working with datasets that are unbalanced. It gives an indication of how well the algorithm can detect a particular sentiment class, which is useful in many real-world scenarios.

Recall:

It is the ratio of true positives to the total of true positives and false negatives. The percentage of real positive events that were accurately identified is known as recall. Recall is generally used in sentiment analysis to assess how accurately an algorithm can recognize every instance of a specific sentiment class. (e.g., positive, negative, or neutral). The algorithm's recall for the positive sentiment class is 80%, for instance, if there are 100 positive tweets in a dataset and it properly classifies 80 of them as positive. Recall is an essential performance indicator in sentiment analysis, particularly when the dataset is

unbalanced, that is, when one sentiment class predominates over the others. A high accuracy score might not accurately represent the algorithm's performance in unbalanced datasets. For instance, the recall may be poor if the algorithm is only forecasting the majority class in an unbalanced dataset, indicating that the system is underperforming for the minority classes. Hence, Recall is a crucial performance statistic in sentiment analysis, particularly when working with datasets that are unbalanced. It gives an indication of how well the algorithm can locate every occurrence of a specific sentiment class, which is useful in many real-world scenarios.

F1 score:

It represents the harmonic mean of recall and precision. It combines recall and precision into a single statistic. The F1 score in sentiment analysis can be used to assess the algorithm's overall accuracy in accurately recognizing all sentiment classes. For instance, a high F1 score shows that the algorithm is successfully classifying all sentiment classes, including the minority classes, and is thus overall performing well. A common performance indicator in sentiment analysis is the F1 score, particularly when the dataset is unbalanced. When precision and recall are equally critical, like in many natural language processing applications, it is especially helpful. The F1 score, which combines precision and recall to produce a single score that assesses an algorithm's overall accuracy, is a crucial performance parameter in sentiment analysis. When the dataset is unbalanced and both precision and recall are crucial, it is especially helpful. By dividing precision and recall by their harmonic means, the F1 score is determined.

The F1 score formula is:
F1 Score = 2 * (Precision * Recall) / (Precision + Recall).

5.5 RESULT

Our Sentiment analysis project achieved an accuracy of 85%, a F1 score of 0.91, a precision of 0.92 and a recall of 0.90 in analyzing the live tweets as shown in Table 1 and Fig 7.

	Precision	Recall	F-measure
Existing system (considers lengthened words)	91.45	89.43	90.43
Proposed system (considers lengthened words, shortened words and emojis)	92.2	90.25	91.7

Table 1: Comparison of existing system and proposed system

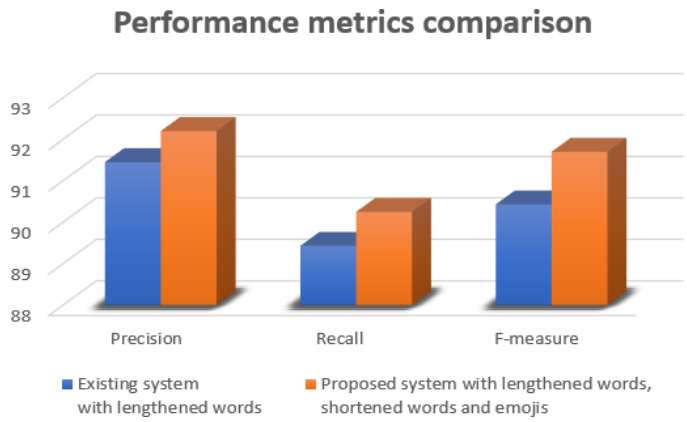

Fig 8: Graph comparing existing system and proposed system

CHAPTER 6

CONCLUSION AND FUTURE WORK

CONCLUSION

Hence this project categorized text data into various sentiment categories, ascertain the polarity of sentiment, assess the effectiveness of machine learning models, and use sentiment analysis in real-world scenarios. By integrating natural language processing techniques and machine learning methods, we were able to preprocess the data, train numerous models, and evaluate the performance of those models using a range of metrics. This project has shown how useful sentiment analysis can be for gaining understanding of people's feelings and opinions. Marketing, customer service, and public opinion research are just a few of the many industries that can benefit from sentiment analysis. Making data- driven decisions to enhance offerings and customer satisfaction can help businesses and organizations better understand the attitudes and opinions of their customers towards their goods and services. In order to help businesses and organizations understand customer feedback, track public opinion, and make data-driven decisions, we identified the most accurate and effective machine learning model for classifying sentiment.

FUTURE WORK

Future works include Multilingual sentimental analysis, so that Businesses may better comprehend customer feedback and sentiment across different locations by using sentiment analysis models that can be taught to read text in a variety of languages. Hence, in general, future work on sentiment analysis projects can increase precision, offer more in- depth analysis, and increase the number of applications. Convolutional neural networks (CNNs) and recurrent neural networks (RNNs), two deep learning approaches, have demonstrated promising outcomes in natural language processing tasks including

sentiment analysis. Deep learning models for sentiment analysis could be developed and improved in future studies to increase accuracy and performance. Overall, sentiment analysis is a discipline that is fast developing, and there is a lot of room for more study and advancement in this area.

APPENDICES

SAMPLE CODE

```
rm(list=ls())
library(e1071)
library(ggplot2) # for plotting the results
#install_github('mananshah99/sentR')
#install.packages('devtools')
#install.packages('NLP')
#install.packages('tm')
library(twitteR) ### for fetching the tweets
library(plyr) ## for breaking the data into manageable pieces
library(ROAuth) # for R authentication
library(stringr) # for string processing
library(sentR)
library(reshape2)
#install.packages('devtools')
require('devtools')
#install_github('mananshah99/sentR')
require('sentR')

reqURL <- "https://api.twitter.com/oauth/request_token"
accessURL <- "http://api.twitter.com/oauth/access_token"
authURL <- "http://api.twitter.com/oauth/authorize"

api_key <- "nVxWfbmXcNAsJHzyiscP1qZ3L"
api_secret <- "T4YORjP8EmBLTBifdf7HKV457eCZPoxMkQ9q6lWrtHi2miA3tT"
access_token <- "775249324737306624-b3wFo1v8WoKAHuiJU7qtUZH9HrrDctx"
```

```
access_token_secret <-
"KMAeBYZeMkz4hDDKqbi7oIMjw65UvU4TLlXrSK8nJaZip"

setup_twitter_oauth(api_key,api_secret,access_token,access_token_secret)

posText <- read.delim("D:/1/pwords.txt", header=FALSE, stringsAsFactors=FALSE)
posText <- posText$V1
posText <- unlist(lapply(posText, function(x) { str_split(x, "\n") }))

negText <- read.delim("d:/1/nwords.txt", header=FALSE, stringsAsFactors=FALSE)
negText <- negText$V1
negText <- unlist(lapply(negText, function(x) { str_split(x, "\n") }))

pos.words = scan('D:/1/pwords.txt', what='character', comment.char=';')
neg.words = scan('d:/1/nwords.txt', what='character', comment.char=';')

n <- 300
Sentiment_tweets = searchTwitter('#csk', n=300)
Sentiment_tweets

Sentiment_txt = sapply(Sentiment_tweets, function(t) t$getText() )
noof_tweets = c(length(Sentiment_txt))
Sentiment_classification<- c(Sentiment_txt)
score.sentiment = function(sentences, pos.words, neg.words, .progress='none')
{

  scores = laply(sentences,
                 function(sentence, pos.words, neg.words)
```

```
{
 # remove punctuation
 sentence = gsub("[[:punct:]]", "", sentence)
 # remove control characters
 sentence = gsub("[[:cntrl:]]", "", sentence)
 # remove digits?
 sentence = gsub('\\d+', '', sentence)
 # define error handling function when trying tolower
 tryTolower = function(x)
 {
  # create missing value
  y = NA
  # tryCatch error
  try_error = tryCatch(tolower(x), error=function(e) e)
  # if not an error
  if (!inherits(try_error, "error"))
    y = tolower(x)
  # result
  return(y)
 }
 # use tryTolower with sapply
 sentence = sapply(sentence, tryTolower)
 # split sentence into words with str_split (stringr package)
 word.list = str_split(sentence, "\\s+")
 words = unlist(word.list)
 # compare words to the dictionaries of positive & negative terms
 pos.matches = match(words, pos.words)
 neg.matches = match(words, neg.words)
```

```
            print(pos.matches)
            # get the position of the matched term or NA
            # we just want a TRUE/FALSE
            pos.matches = !is.na(pos.matches)
            neg.matches = !is.na(neg.matches)
            # final score
            score = sum(pos.matches) - sum(neg.matches)
            return(score)
          }, pos.words, neg.words, .progress=.progress )
  # data frame with scores for each sentence
  scores.df = data.frame(text=sentences, score=scores)
  return(scores.df)
}

Sentiment_1 <- get_nrc_sentiment(Sentiment_classification)
texts=Sentiment_classification

  #fetch sentiment words from texts
  Sentiment <- get_nrc_sentiment(texts)

#Naivebayes
Naivebayes_Sentiment_classification<-classify.naivebayes(texts)
Naivebayes_Sentiment_classification
#head(Sentiment)
text <- cbind(texts,Sentiment)
textNaivebayes <- cbind(Naivebayes_Sentiment_classification,Sentiment)
```

```
#count the sentiment words by category
TotalSentiment <- data.frame(colSums(text[,c(2:11)]))
TotalSentiment
names(TotalSentiment) <- "count"
TotalSentiment <- cbind("sentiment" = rownames(TotalSentiment), TotalSentiment)
rownames(TotalSentiment) <- NULL

#total sentiment score of all texts
ggplot(data = TotalSentiment, aes(x = sentiment, y = count)) +
  geom_bar(aes(fill = sentiment), stat = "identity") +
  theme(legend.position = "none") +
  xlab("Sentiment") + ylab("Total Count") + ggtitle("Bar plot")

my_data <- data.frame(
  category = TotalSentiment$sentiment,
  value = TotalSentiment$count
)
s <- sum(my_data$value)

ggplot(my_data, aes(x="", y=value, fill=category)) +
  geom_bar(stat="identity", width=1) +
  coord_polar("y", start=0) +
  geom_text(aes(label=paste0(format(round((value/s)*100),2),"%")),
position=position_stack(vjust=0.5)) +
  scale_fill_brewer(palette="Set2") + ggtitle("Pie Chart")

pos_neg <- subset(Naivebayes_Sentiment_classification, select = c("POS/NEG"))
```

```
ground_truth <- subset(Naivebayes_Sentiment_classification, select = c("SENT"))
predictions <- ifelse(pos_neg>1, "positive", "negative")
conf_mat <- table(Predicted = predictions, Actual = ground_truth)

TP <- conf_mat["positive", "positive"]
TN <- conf_mat["negative", "negative"]
FP <- conf_mat["positive", "negative"]
FN <- conf_mat["negative", "positive"]

precision <- TP/(TP+FP)
precision <- p
recall <- TP/(TP+FN)
recall <- r
f_measure <- (2*p*r)/(p+r)
f_measure <- f
accuracy = (p*TP + r*TN)/n

library(htmlTable)
my_table <- data.frame(
  Performance_metrics = c("Precision", "Recall", "F-measure"),
  Result = c(p, r, f)
)
htmlTable(my_table)
htmlTable(
  my_table,
  rnames = FALSE,
  caption = "My Table",
  align = "c",
```

```
  ctable = 1,
  tspanner = c(" ", "Demographics"),
  css.cell = "padding: 5px; font-size: 16px; text-align: center;",
  css.table = "border-collapse: collapse; margin: auto; width: 50%;",
  col.rgroup = c("lightgray", "white")
)

cat("Precision: ", p, "\n")
cat("Recall: ", r, "\n")
cat("F-measure: ", f, "\n")
cat("Accuracy: ", accuracy, "\n")
```

SCREENSHOTS

```
Sentiment_tweets = searchTwitter('#ipl', n=300)
Sentiment_tweets
```

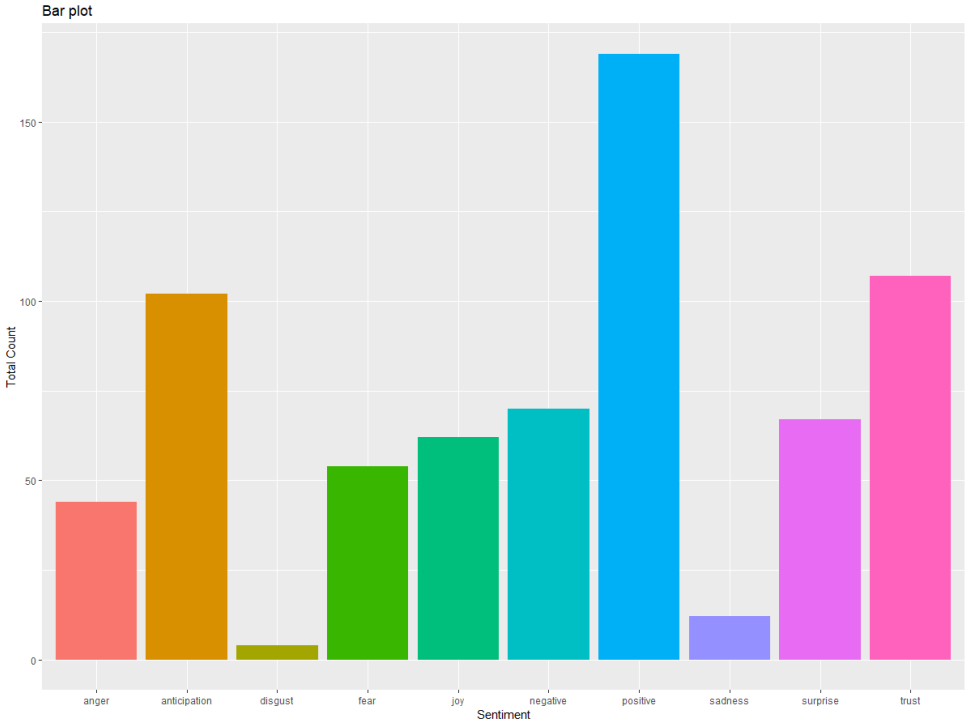

Fig 9: Bar plot representing various emotions

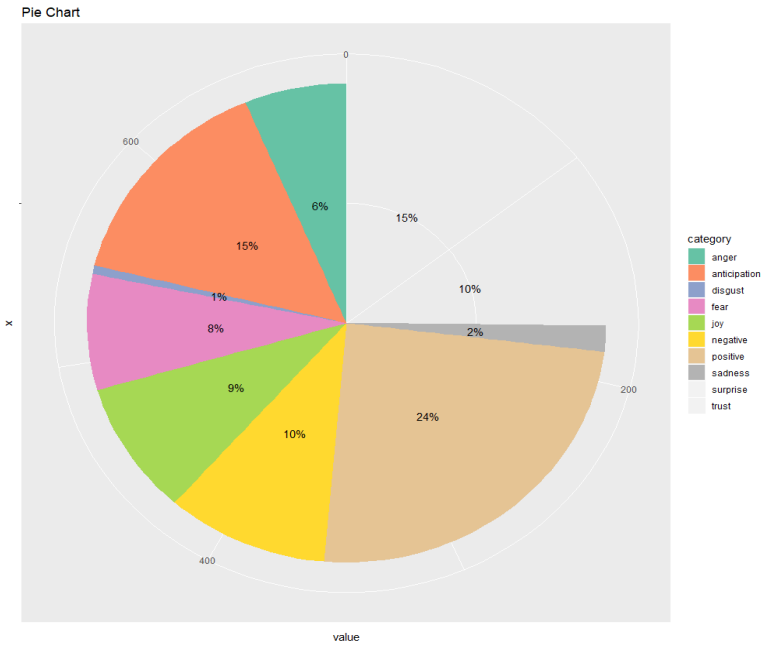

Fig 10: Pie chart representing various emotions

	texts	anger	anticipation	disgust	fear	joy	sadness	surprise	trust	negative	positive
1	@switc...	1	0	1	0	0	0	0	0	1	0
2	is upset...	1	1	0	0	0	2	0	1	2	0
3	@Kenic...	0	0	0	0	1	0	0	1	0	2
4	my who...	0	0	0	1	0	0	0	0	0	0
5	@natio...	1	0	1	1	0	1	0	0	1	0
6	@Kwesi...	0	0	0	0	0	0	0	1	0	0
7	Need a ...	0	0	0	0	1	0	0	1	0	1
8	@LOLTri...	0	2	0	0	0	0	0	0	0	0
9	@Tatian...	0	0	0	0	0	0	0	0	0	0
10	@twitte...	0	0	0	0	0	0	0	0	0	0
11	spring ...	0	0	0	0	0	0	1	0	0	0
12	I just re-...	0	0	0	0	0	0	0	0	0	0
13	@caregi...	2	2	0	3	0	1	0	0	2	0
14	@octoli...	0	0	0	0	0	0	0	0	0	1
15	@smarri...	1	0	0	1	0	0	0	0	1	0

Fig 11: Sentiment classification of tweets

	Performance_metrics	Result
1	Precision	92.2
2	Recall	90.25
3	F-measure	91.7

Fig 12: Performance metrics table generated using htmltable library in R

REFERENCES

[1] Ashima Kukkar, Rajni Mohana, Aman Sharma, Anand Nayyar, Mohd. Asif Shah. Improving Sentiment Analysis in Social Media by Handling Lengthened Words; 2023.

[2] Vidyabharathi. D, Theetchenya.S, Marimuthu. M, Vidhya.G. Sentiment Analysis algorithms and its diverse applications: A Survey, Turkish Journal of Physiotherapy and Rehabilitation; 2021.

[3] Basker.N, Marimuthu.M, Theetchenya.S, Vidyabharathi.D, Vidhya.G. Ascertaining of trustworthiness in the news articles on Social Media with Deep Neural Network architectures, Turkish Journal of Physiotherapy and Rehabilitation; 2021.

[4] Mohammad Ehsan Basiri a, Shahla Nemati , Moloud Abdar, Somayeh Asadi , U. Rajendra Acharrya. A novel fusion-based deep learning model for sentiment analysis of COVID-19 tweets; 2021.

[5] Yazhou Zhang, Zhipeng Zhao, Panpan Wang, Xiang Li, Lu Rong and Dawei Song. ScenarioSA: A Dyadic Conversational Database for Interactive Sentiment Analysis; 2020.

[6] Lei Wang, Jianwei Niu, Shui Yu. SentiDiff: Combining Textual Information and Sentiment Diffusion Patterns for Twitter Sentiment Analysis; 2020.

[7] Mahdi Hajiali. Big data and sentiment analysis: A comprehensive and systematic literature review; 2019.

[8] S. Uma Maheswari, S. S. Dhenakaran. Sentiment Analysis on Social Media Big Data With Multiple Tweet Words; 2019.

[9] Jordan J. Bird , Anikó Ekárt, Christopher D. Buckingham ,Diego R. Faria. High Resolution Sentiment Analysis by Ensemble Classification; 2019.

[10] Lin Yue, Weitong Chen, Xue Li, Wanli Zuo,Minghao Yin. A survey of sentiment analysis in social media; 2019.

[11] Bing Liu, Lei Zhang. A survey of opinion mining and sentiment analysis; 2019.

[12] Hu Xu , Bing Liu , Lei Shu and Philip S. Yu. DomBERT: Domain-oriented Language Model for Aspect-based Sentiment Analysis; 2019.

[13] Sajjad Haider, Muhammad Tanvir Afzal, Muhammad Asif, Hermann Maurer, Awais Ahmad, Abdelrahman Abuarqoub. Impact analysis of adverbs for sentiment classification on Twitter product reviews; 2018.

[14] Yassine AL AMRANI, Mohamed Lazaar, Kamal Eddine EL Kadiri. Random Forest and Support Vector Machine based Hybrid Approach to Sentiment Analysis; 2018.

[15] Ankit, Nabizath Saleena. An Ensemble Classification System for Twitter Sentiment Analysis; 2018.

[16] Phu Vo Ngoc,Chau Vo Thi Ngoc,Tran Vo Thi Ngoc, Dat Nguyen Duy. A C4.5 algorithm for English emotional classification; 2018.

[17] Dhanalakshmi and Senthil Kumar. Sentiment Analysis of Movie Reviews using Machine Learning Techniques; 2017.

[18] Samuel Brody, Nicholas Diakopoulos. Using Word Lengthening to Detect Sentiment in Microblogs; 2011.

[19] Kouloumpis. Twitter Sentiment Analysis: The Good the Bad and the OMG!; 2011.

[20] Pang and Lee. Sentiment Analysis and Opinion Mining; 2008.