VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"Jnana Sangama",Belagavi- 590 018

A PROJECT REPORT

ON

"SENTIMENTAL ANALYSIS OF POLITICAL TWEETS"

Submitted in partial fulfillment of the requirements for the Degree of

BACHELOR OF ENGINEERING

IN

Computer Science & Engineering (DATA SCIENCE)

Submitted by

Chinmay CS (4AI22CD013) **K Shivadarshan (4AI22CD028)**

Sunil Kumar AB (4AI22CD056) **Yashas Xavier (4AI22CD062)**

Under the Guidance of

Mrs. PALLAVI CS B.E.,M.Tech

Assistant Professor,
Department of CS&E (DATA SCIENCE)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING (DATA SCIENCE)
ADICHUNCHANAGIRI INSTITUTE OF TECHNOLOGY
(Affiliated to V.T.U., Accredited by NAAC)
CHIKKAMAGALURU-577102, KARNATAKA

2025- 2026

ADICHUNCHANAGIRI INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi)
Chikkamagaluru, Karnataka, India-577102.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)

CERTIFICATE

This is to certify that the Major Project Phase-II (BCD786) work entitled **“Sentimental Analysis of Political Tweets”** is a bonafide work carried out by **CHINMAY CS (4AI22CD013), K SHIVADARSHAN (4AI22CD028), SUNIL KUMAR AB (4AI22CD056), YASHAS XAVIER (4AI22CD062)** students of 7th semester B.E, in partial fulfillment for the award of Degree of **Bachelor of Engineering in Computer Science and Engineering (Data Science)** of the Visvesvaraya Technological University, Belagavi during the academic year **2025-26**. It is certified that all corrections and suggestions indicated for Internal Assessment have been incorporated in the report deposited in the department library. The project report has been approved, as it satisfies the academic requirements in respect of Project Work prescribed for the said Degree.

Signature of the Guide
Mrs. Pallavi CS **B.E.,M.Tech**
Assistant Professor
Dept. of CS&E(DATA SCIENCE)

Signature of the Coordinator
Mrs. Shilpa K V **B.E., M.Tech**
Assistant Professor
Dept. of CS&E(DATA SCIENCE)

Signature of the HOD
Dr. Adarsh M J **B.E., M.Tech., Ph.D**
Associate Professor & Head
Dept. of CS&E(DATA SCIENCE)

Signature of the Principal
Dr. C.T Jayadeva **B.E., M.Tech., Ph.D**
Principal,
A.I.T, Chikkamagaluru

External Examiners	**Signature with Date**
1. ______________________	______________________
2. ______________________	______________________

ADICHUNCHANAGIRI INSTITUTEOFTECHNOLOGY
(Affiliated to Visvesvaraya TechnologicalUniversity,Belagavi)
Chikkamagaluru, Karnataka, India – 577102

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
(DATA SCIENCE)

APPROVAL

The **Major Project Phase II – BCD786** entitled **“SENTIMENTAL ANALYSIS OF POLITICAL TWEETS”** is here by approved as a credible study of Engineering subject carried out and presented in a satisfactory manner for acceptance as a pre-requisite to the Degree of **BACHELOR OF ENGINEERING** in **COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)** during the academic year 2025-26.

Submitted By :

1. Chinmay CS	**(4AI22CD013)**
2. K Shivadarshan	**(4AI22CD028)**
3. Sunil Kumar AB	**(4AI22CD056)**
4. Yashas Xavier	**(4AI22CD062)**

Signature of the Guide
Mrs. Pallavi CS **B.E., M.Tech**
Assistant Professor
Dept. of CS&E (DATA SCIENCE)

Signature of the Coordinator
Mrs. Shilpa K V **B.E., M.Tech.**
Assistant Professor
Dept. of CS&E (DATA SCIENCE)

Signature of the HOD
Dr. Adarsh M J **B.E., M.Tech., Ph.D.**
Associate Professor & Head
Dept. of CS&E (DATA SCIENCE)

ABSTRACT

In the digital era, social media platforms such as Twitter have become powerful tools for expressing political opinions and public sentiments. This project, Sentimental Analysis of Political Tweets, aims to analyze and classify the sentiments expressed in political tweets as positive, negative, or neutral. The system utilizes Python and Streamlit to create an interactive user interface for data input, processing, and visualization.

The workflow involves several modules data acquisition, preprocessing, feature extraction, classification, and visualization. Tweets are collected either manually or through uploaded datasets, cleaned using preprocessing techniques, and converted into numerical features through vectorization methods. Machine learning algorithms such as Naive Bayes and Logistic Regression are applied to classify sentiments accurately. The results are displayed graphically, allowing users to interpret sentiment trends easily and clearly. Overall, this project provides an effective approach to understanding public opinion on political issues, supporting data-driven decision-making and sentiment-based political insights.

ACKNOWLEDGEMENTS

We express our humble Pranamas to his holiness **Parama Poojya Jagadguru Padmabushana Sri Sri Sri Dr. Balagangadharanatha Mahaswamiji** and **Parama Poojya Jagadguru Sri Sri Sri Dr. Nirmalanandanatha Mahaswamiji** and also to **Sri Sri Gunanatha Swamiji** Sringeri Branch Chikkamagaluru who have showered their blessings on us for framing our career successfully.

We are deeply indebted to our honorable Director **Dr. C K Subbaraya** for creating the right kind of care and ambience.

We are thankful to our beloved Principal **Dr. C T Jayadeva** for inspiring us to achieve greater endeavors in all aspects of Learning.

We express our deepest gratitude to **Dr. Adarsh M J,** Associate Professor & Head, Department of CS&E (DATA SCIENCE) for her valuable guidance, suggestions and constant encouragement without which success of our Project work would have been difficult.

We are grateful to our coordinator **Prof. Shilpa K V,** for her excellent guidance, constant encouragement, support and constructive suggestions.

We are thankful to our guide **Prof. Pallavi C S,** for her inspiration and lively correspondence for carrying our project work.

We would like to thank our beloved parents for their support, encouragement and blessings.

And last but not least, I would be pleased to express our heartfelt thanks to all teaching and non-teaching staff of CS&E (DATA SCIENCE) Department and our friends who have rendered their help, motivation and support.

Chinmay C S (4AI22CD013)
K Shivadarshan (4AI22CD028)
Sunil Kumar A B(4AI22CD056)
Yashas Xavier(4AI22CD062)

CONTENTS

ABSTRACT	**i**
ACKNOWLEDGEMENTS	**ii**
CONTENTS	**iii**
LIST OF FIGURES	**vi**
LIST OF SNAPSHOTS	**v**
CHAPTERS	Page no.
1. INTRODUCTION	
1.1 Introduction	1
1.2 Motivation	3
1.3 Problem Statement	3
1.4 Scope of the Project	4
1.5 Objectives	4
1.6 Overview of Simple ML models and Transformers	4
1.7 Review of Literature	5
1.8 Organization of Report	6
1.9 Summary	6
2. SOFTWARE REQUIREMENTS SPECIFICATION	
2.1 Hardware Requirements	7
2.2 Software Requirements	7
2.3 Functional Requirements	7
2.4 Non-Functional Requirements	8
2.5 Summary	8
3. HIGH LEVEL DESIGN	
3.1 Design Consideration	9
3.2 System Architecture	10
3.3 Specification using Use Case diagrams	11
3.4 Module Specification	12
3.4.1 Data Collection Module	12
3.4.2 Pre-processing Module	13

3.4.3 Classification Module 13
3.4.4 Feature Extraction module 15
3.4.5 Visualization Module 17
3.4.6 User Interface Module 18
3.5 Data flow diagram 20
3.5.1 Data flow diagram for Data Acquisition and Storage Module 21
3.5.2 Data Flow diagram for Data Preprocessing (Cleaning). 21
3.5.3 Data flow Diagram for Feature Extraction & Classification 22
3.5.4 Data flow Diagram for Reporting & Visualization 23
3.6 State chart diagram 24
3.7 Summary 25

4. DETAIL DESIGN

4.1 Structural chart of the Sentiment Analysis of Political Tweets 26
4.2 Flow chart of Pre-processing Module 27
4.3 Flow chart of Classification Module 28
4.4 Flow chart of Result and Visualization Module 29
4.5 Summary 30

5. IMPLEMENTATION

5.1 Implementation 31
5.2 Implementation Requirements 31
5.3 Programming Language Selection 32
5.3.1 Key Features of Programming Language Selected 32
5.4 Pseudocode for Each Module with Description 36
5.4.1 Pseudocode for Data Acquisition & Storage Module 36
5.4.2 Pseudocode for Pre-Processing Module 37
5.4.3 Pseudocode for Feature extraction Module 37
5.4.4 Pseudocode for Classification Module 38
5.4.5 Pseudocode for Reporting & Visualization Module 39
5.5 Summary 40

6. SYSTEM TESTING

6.1 Test Procedures 41
6.1.1 Unit Testing 41

6.1.2 Integration Testing 42

6.1.3 Functional Testing 42

6.1.4 Performance Testing 43

6.1.5 User Interface Testing 43

6.1.6 Validation and Accuracy Testing 44

6.1.7 Unit Test 44

6.2 Summary 45

7. RESULTS AND DISCUSSIONS

7.1 Experimental Results 46

7.2 Summary 52

8. CONCLUSION AND FUTURE ENHANCEMENTS

8.1 Conclusion 53

8.2 Future Enhancements 53

REFERENCES 55

LIST OF FIGURES

Figure No.	Description	Page No.
3.1	Architectural diagram of the Sentimental analysis of political	10
	tweets	
3.2	Use case diagram of Sentimental analysis of political tweets	11
3.3	Use case diagram of Data Collection Module	12
3.4	Use case diagram for preprocessing module	13
3.5	Usecase diagram of Data Collection Module	14
3.6	Use case diagram of Feature extraction module	16
3.7	Use case diagram of Visualization Module	17
3.8	Use case diagram of Visualization Module	19
3.9	Data flow diagram of Sentimental analysis system	20
3.10	Data flow diagram for Data Acquisition and Storage Module	21
3.11	Data flow diagram of Data preprocessing module	22
3.12	Data flow diagram of Feature Extraction Module	23
3.13	Data flow diagram of Reporting and visualization module	24
3.14	State chart diagram	24
4.1	Structural chart diagram of the sentimental analysis of political tweets	26
4.2	Flow chart of Pre-processing module	27
4.3	Flow chart of Classification	28
4.4	Flow chart of Result and Visualization	29

LIST OF SNAPSHOTS

Snapshot No.	Description	Page No.
7.1	User authentication page via login and Signup	46
7.2	Home Page of the System	46
7.3	Manual input of tweet	47
7.4	Upload CSV file of labelled data	47
7.5	The graph of sentiments vs tweets	48
7.6	Model performance of LR	48
7.7	Model performance analysis of MNB	49
7.8	Model performance analysis of BERT	49
7.9	Comparision of F1-Scores across all models	50
7.10	Confusion Matrix	50
7.11	Inter-model agreement	51
7.12	Count and visualization of the unlabelled data prediction	51
7.13	Inter-model agreement and Consistency scores	52

CHAPTER 1

INTRODUCTION

1.1 Introduction

Millions of individuals use social media, claims network sites to reveal their thoughts, feelings, and emotions regarding the people's daily existence. However, anything can be written about, including social interactions or product reviews. By means of Online communities offer a dialogue forum where Consumers inform and sway other people. Furthermore, social media offers a chance for business that gives a social networking platform for connecting with its customers to promote or communicate with consumers directly in order to the viewpoint of the customer regarding goods and services.

Contrarily, consumers have complete authority over purchasing decisions. what viewers want to see and how viewers react With this, the business's success and failure are made public resulting through word of mouth The social network, however may alter consumer behavior and decision-making, As an illustration, notes that 87% of internet users are Customers' choices and purchases are impacted by them review.

In order for the organization to catch up more quickly with what Customers believe that organizing would be more advantageous should be quick to respond and develop a winning plan competitors of theirs People are using social media sites like Twitter, which produce large volumes of opinion writings in the form of tweets and are available for sentiment analysis, while the World Wide Web expands quickly.

From a human perspective, this corresponds to a vast volume of information, making it challenging to quickly extract sentences, read them, analyze tweet by tweet, summarize them, and organize them into an understandable style. Additionally, social media gives businesses a chance by offering them a platform to engage with their customers for advertising. People heavily rely on user-generated content from the internet when making decisions. For instance, if someone wants to purchase a good or use a service, they will research it online and discuss it on social media before making a choice.

The volume of user-generated content is too great for a typical user to process. Since this needs to be automated, many different sentiment analysis approaches are in use Before requirements. The fundamental goals of textual information retrieval strategies are to process, search for, or examine the factual material that is already there.

Even if facts have an objective component, some other literary contents exhibit subjective traits. Sentiment Analysis's fundamental components opinions, sentiments, assessments, attitudes, and emotions are primarily represented by these contents (SA). In large part because of the enormous expansion in the amount of information available online from sources like blogs and social networks, it presents many challenging chances to design new applications.

For instance, by using SA and taking into account factors such as positive or negative attitudes about the goods, recommendations of items proposed by a recommendation system can be predicted. This technique could be used for different purposes such as politicians could use it for analyzing what kind of sentiments people from different areas are carrying towards him/her and hence could invest more in those areas.

An example of this is recent Trump elections, where he hired a group of analysts for this specific purpose. Sentiment analysis could also be applied in the field of business marketing. With the help of this technology different business organizations capture the feelings of people regarding their products and of that of their competitors. Organizations employ their strategies with accordance to this knowledge only. Leaving market research aside, analysis of sentiments could play a vital part in Service industries as it could analyze a full fledged customer experience and could reveal customer feeling, which could prove to be very beneficial.

The political landscape on social media presents unique technical hurdles that differ from standard commercial sentiment analysis. Political tweets are often saturated with sarcasm, hyperbole, and complex linguistic metaphors that can easily mislead basic algorithms. Furthermore, the presence of "echo chambers" and coordinated bot activity can skew perceived sentiment, requiring sophisticated filtering to distinguish between organic human opinion and artificial manipulation. Despite these challenges, political sentiment analysis remains an invaluable tool for election forecasting, crisis management, and policy testing, offering a granular view of how specific demographics react to shifting political narratives in the heat of a campaign. Political discourse on social media introduces technical challenges that are significantly different from those found in conventional commercial sentiment analysis. Tweets related to politics frequently contain sarcasm, exaggeration, and complex linguistic metaphors, which can confuse basic sentiment analysis algorithms. These linguistic features often cause models to misinterpret the true emotional tone of the content.

1.2 Motivation

In recent years, social media platforms like Twitter have emerged as powerful tools for political communication and public discourse. Political leaders, parties, journalists, and citizens increasingly use these platforms to express their views, share information, and engage in real-time discussions. The sheer volume and immediacy of tweets make them a rich source of data for understanding public opinion on political issues.

Sentiment analysis of political tweets offers a unique opportunity to tap into this data and uncover trends, opinions, and emotional reactions related to political events, policies, and personalities. By analyzing the sentiments expressed in tweets, we can gain valuable insights into public perception, which can be useful for media analysis, campaign strategy, and even policymaking.

Moreover, the ability to automate sentiment classification using machine learning and natural language processing techniques allows for the processing of large datasets quickly and efficiently. This scalability is particularly important during election periods or major political events when public opinion shifts rapidly.

The motivation behind this project stems from the desire to apply data science techniques to real-world political scenarios and contribute to the growing field of computational social science. By developing a reliable sentiment analysis model for political tweets, this project aims to bridge the gap between social media discourse and political analysis, enabling a deeper understanding of public sentiment and its implications·

1.3 Problem Statement

To design and develop a sentiment analysis system that processes political tweets using NLP to classify them as positive, negative, or neutral, with a user friendly interface for input and sentiment visualization.

Input: Manual text and Upload of a CSV dataset.

Processing:

- **Pre-processing:** The entered text performs several preprocessing steps to clean the text.
- **Analyzation:** The text will be analyzed and predicted using the models used.

Output: The tweets will be classified as positive, negative or neutral and displays the visual ization and factors for F1 score, precision and other factors by comparing with the 3 models.

1.4 Scope of the Project

The project will be helpful to companies, political parties, as well as common people. It will assist political parties in reviewing public opinions about the programs they plan to organize or those that have already been conducted. Similarly, companies can use the system to analyze feedback on their newly released hardware or software products. Movie makers can also benefit by reviewing audience reactions to currently running films. By analyzing the collected tweets, the sentiment analyzer provides insights into how positive, negative, or neutral people's opinions are regarding a particular topic or event.

1.5 Objectives

The main objectives of the project are:

- Gather relevant political tweets using manual input and the CSV datasets to clean the data by removing noise such as spam, URLs, special characters, and non- informative content.
- Apply Natural Language Processing (NLP) techniques and machine learning models used to classify the tweets into sentiment categories such as positive, negative, or neutral.
- Identify sentiment patterns and trends across different timeframes, political events,or user hashtags to understand public reaction to specific political topics or leaders.
- Assess the accuracy, precision, recall, and F1-score of the sentiment analysis model using labeled datasets to ensure its effectiveness.

1.6 Overview of Simple ML models and Transformers

The Traditional Machine Learning (ML) pipeline, crucial for providing a robust baseline in this comparative project, utilizes simpler, more interpretable algorithms like Naive Bayes (NB) and Logistic Regression (LR). Naive Bayes, a probabilistic classifier based on Bayes' theorem, assumes feature independence and excels at handling the high-dimensional, sparse datasets generated from text, making it computationally efficient and ideal for an initial assessment of sentiment based on word probabilities.

Logistic Regression, though named 'regression,' acts as a powerful linear classifier that uses a sigmoid function to output the probability of a text belonging to a specific sentiment class. LR is valued for its interpretability, as its calculated coefficients directly show the positive or negative weight of individual words (features), transparently explaining the classification decision. Both models are paired with TF-IDF feature engineering and collectively establish a reliable benchmark against which the performance gains and increased complexity of the advanced BERT model are measured and justified.

The Transformer architecture represents the pinnacle of current Natural Language Processing (NLP) advancements, fundamentally reshaping the field by relying entirely on the self-attention mechanism and abandoning sequential data processing. The standard model, consisting of Encoder and Decoder stacks, uses Multi-Head Attention to calculate the contextual relevance between every word in a sequence simultaneously, allowing it to capture long-range dependencies far more effectively than predecessors like RNNs.

The core innovation lies in using Query (Q), Key (K), and Value (V) vectors to compute attention scores, enabling the model to determine *what* to focus on in the input sentence. Models derived from this architecture, such as BERT (Bidirectional Encoder Representations from Transformers), leverage this power by being pre-trained on massive datasets using tasks like Masked Language Modeling. BERT, which is encoder-only, then uses its large parameter count and Positional Encodings to generate rich, contextualized word embeddings, providing the superior semantic understanding necessary to handle the complex nuances, slang, and implied meanings present in political social media text.

1.7 Review of Literature

A,Go,R.Bhayani,and L.Huang, Twitter Sentiment Classification Using Distant Supervision.[1]

The result of the analysis showed that most of the articles applied the opinion-lexicon method to analyse text sentiment in social media, extracted data on microblogging sites, mainly Twitter and sentiment analysis applications can be seen in world events, healthcare, politics and business. Their study further concluded that if the data structure is messy and a small amount of data and limited time is available for analysis, it is recommended to go for the lexicon-based method while for a large amount of data machine learning-based method is suitable as it requires more time and data to train. To improve the quality and accuracy of the result, it is suggested to combine both the lexicon and machine learning methods.

B. Liu, Sentiment Analysis and Opinion Mining,Synthesis Lectures On Human Language.[2]

In his paper on Sentiment Analysis in Social Media Text, emphasised tweet pre-processing. Pre-processing was done to normalize the language and generalize the vocabulary employed to express sentiment. The use of such generalized features significantly improves the results of sentiment classification .They used supervised learning and a feature set consisting of subjectivity-lexicon-based scores and achieved an accuracy of 61.6%.

The inaccuracies in prediction by the NLTK sentiment classifier and other resources available in Python. His deep learning-based model is a multiclass classification model with 10 classes, one for each decile. He build a Long ShortTerm Memory (LSTM) model in Keras and achieved an accuracy of 48.6%. The accuracy of this model will be much higher on a binary class dataset. they used retrieve data using Hadoop component called Flume and stores the tweets in a buffer storage called Channel which is a component of Flume. After that, using Flume there is some preprocessing work is done. Finally, the results are stored in a Data warehouse called Hive which is one of the Hadoop components There are many cases like making Twitter Sentiment analysis using other different technologies like Flume, Spark, Python, and etc. Among all the other data pipelines, Twitter-Kafka-Spark have the most efficient, scalable, consistency, reliable and failover benefits. We are mainly using this integration to achieve better performance and failover when compared to other technologies.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,".[3]

The paper introduces BERT (Bidirectional Encoder Representations from Transformers), a novel language representation model designed to improve performance on a wide range of Natural Language Processing (NLP) tasks. Unlike previous models that process text either left-to-right or right-to-left, BERT uses a deep bidirectional Transformer architecture, allowing it to understand the full context of a word by considering both its preceding and following words simultaneously.

BERT is pre-trained on large unlabeled text corpora (Wikipedia and BookCorpus) using two key objectives. The first is Masked Language Modeling (MLM), where random words in a sentence are masked and the model learns to predict them based on surrounding context. This enables true bidirectional learning. The second is Next Sentence Prediction (NSP), which trains the model to determine whether one sentence logically follows another, helping it understand sentence relationships.

1.8 Organization of Report

Organization of these is presented below. The report includes 4 chapters.

Chapter 1 – Introduction

Briefs about the introduction of personalized fashion recommendation system. It also presents motivation, scope of the project, Objectives and literature review.

Chapter 2 – System Requirements Specification

The chapter includes a brief description of the software technologies used in project along with other software, hardware requirements.

Chapter 3 – High Level Design

The chapter presents a brief discussion about the high-level design of proposed System.

Chapter 4 – Detailed Design

Thechapter presents the flowchart for personalized fashion recommendation system.

Chapter 5 – Implementation Details

The chapter shows us that how we implemented the project.

Chapter 6 – System Testing

The chapter tells us the accuracy,stability and usability of the models.

Chapter 7 – Results and Discussions

The chapter shows us the snapshots of the results that are obtained in the proposed work.

Chapter 8 – Conclusion and Future Enhancements

The chapter tells us the conclusion and the future enhancements of the project.

1.9 Summary

First chapter gives the introduction: about the basic idea of an individual getting virtual recommendation in an easy and understanding way which is described in section 1.1, Motivation for the project is presented in section 1.2. Section 1.3 describes problem statement of the project. Scope of the project is described in section 1.4 and objectives of the proposed technique are presented in section 1.5. Lastly review of literature: the number of reference papers referred is presented in section 1.7.

CHAPTER 2

SYSTEM REQUIREMENT SPECIFICATION

System requirement specification (SRS) presents the clear, brief also unambiguous explanation of the proposed methods. SRS allows the developers to check the system whether it satisfy the particular requirement or not. It stipulates the hardware and software prerequisite of the system. SRS assures that system is designed as mandatory at a given time. It also describes the functional and non-functional requirements for the system.

2.1 Hardware Requirements

- Processor: modern multi-core, e.g., Intel i5 / AMD Ryzen 5 (4-6 cores)
- RAM: ~ 16 GB
- Storage: SSD 1 TB+ (for dataset, intermediate results, model checkpoints)
- GPU: strongly recommended — e.g., NVIDIA-CUDA capable GPU with 8-16+ GB VRAM
- Good network / possibly GPU memory if using cloud accelerators

2.2 Software Requirements

● Programming Language	: Python 3.5 or higher.
● Front-end	: Python 3.5 or higher (NLTK, Transformers).
● Operating System	: Windows 10.
● IDE	: Vs code, Streamlit, Pytorch

2.3 Functional Requirements

The flow of the project is as follows:

- **User Input:** User enters or uploads political tweets for sentiment analysis.
- **Data Collection:** Tweets are fetched from Kaggle and uploaded via CSV file.
- **Data Preprocessing:** Tweets are cleaned by removing noise like URLs, mentions, and punctuation.
- **Tokenization:** Each tweet is split into individual words for analysis.
- **Feature Extraction:** Text data is converted into numerical format using NLP techniques
- **Model Loading/Training:** A pre-trained ML or deep learning model is loaded or trained on tweet data.
- **Data Overview:** The uploaded dataset will be overviewed whether the data is labelled or unlabelled.

- **Sentiment Prediction:** The model predicts whether each tweet is Positive, Negative, or Neutral.
- **Result Display:** Predicted sentiment is shown on the Streamlit web interface.
- **Visualization:** Graphs and charts display sentiment distribution for all analyzed tweets.
- **Data Storage/Export:** Results are saved or downloaded as a CSV file for further analysis.

2.4 Non-Functional Requirements

- **Performance:** The system provides fast and efficient performance by analyzing a single tweet within a few seconds. It can also handle multiple tweets uploaded in bulk without significant delay.
- **Accuracy:** The sentiment classification model maintains high accuracy, ensuring correct detection of positive, negative, and neutral tweets. Model performance is measured using metrics such as Precision, Recall, and F1-Score.
- **Usability:** The system is designed to be simple and user-friendly using the Streamlit interface. Users can easily input tweets, view results, and understand the visual outputs without any technical knowledge.
- **Relabilty:** The system performs consistently without crashing and handles invalid or empty inputs gracefully. It provides stable outputs even when multiple requests are made.
- **Compatability:** The project is compatible with the latest versions of Python, Streamlit, and major web browsers such as Chrome and Edge. It ensures smooth functioning across different devices and environments.

2.5 Summary

The system requirement specification is described in this chapter. Section 2.1 describes hardware requirement and 2.2 presents the software requirements. The 2.3 and 2.4 sections describe about the functional and non-functional requirements.

CHAPTER 3

HIGH LEVEL DESIGN

High Level Design (HLD) explains the overall architecture that is used for developing the Sentiment Analysis of Political Tweets system. The architecture diagram provides a broad overview of the system, identifying the major components such as data collection, preprocessing, sentiment classification, and visualization modules, along with their interactions. The HLD focuses on how these modules communicate with each other to achieve the desired output. The design uses simplified technical terminology so that the system architecture is understandable to developers, analysts, and administrators.

The High Level Design describes how data flows through the different components of the system. The process begins with data extraction from Twitter using APIs, followed by text preprocessing where irrelevant elements like hashtags, mentions, and special characters are removed. The cleaned data is then passed to the sentiment analysis model (such as BERT or other machine learning models), which classifies each tweet into positive, negative, or neutral categories. The results are stored and visualized through charts and dashboards for easy interpretation.

3.1 Design consideration

The design consideration describes how the system behaves under various operational conditions and what measures are taken in case of abnormal situations. It defines the essential steps involved in analyzing political tweets and ensures that the system performs efficiently and accurately. Some of the key design considerations are data collection, pre-processing, classification, data visualization, and recommendation.

- **Data collection:** Tweets are fetched from Kaggle and uploaded by csv file
- **Pre-Processing:** Unwanted text like links, hashtags, and symbols are removed to clean the data.
- **Classification:** Cleaned tweets are analyzed using NLP models to detect positive, negative, or neutral sentiments.
- **Visualization:** Results are displayed using charts to show overall sentiment trends.

- **Recommendation:** The system suggests insights such as trending topics or parties with higher positive sentiment.

3.2 System architecture:

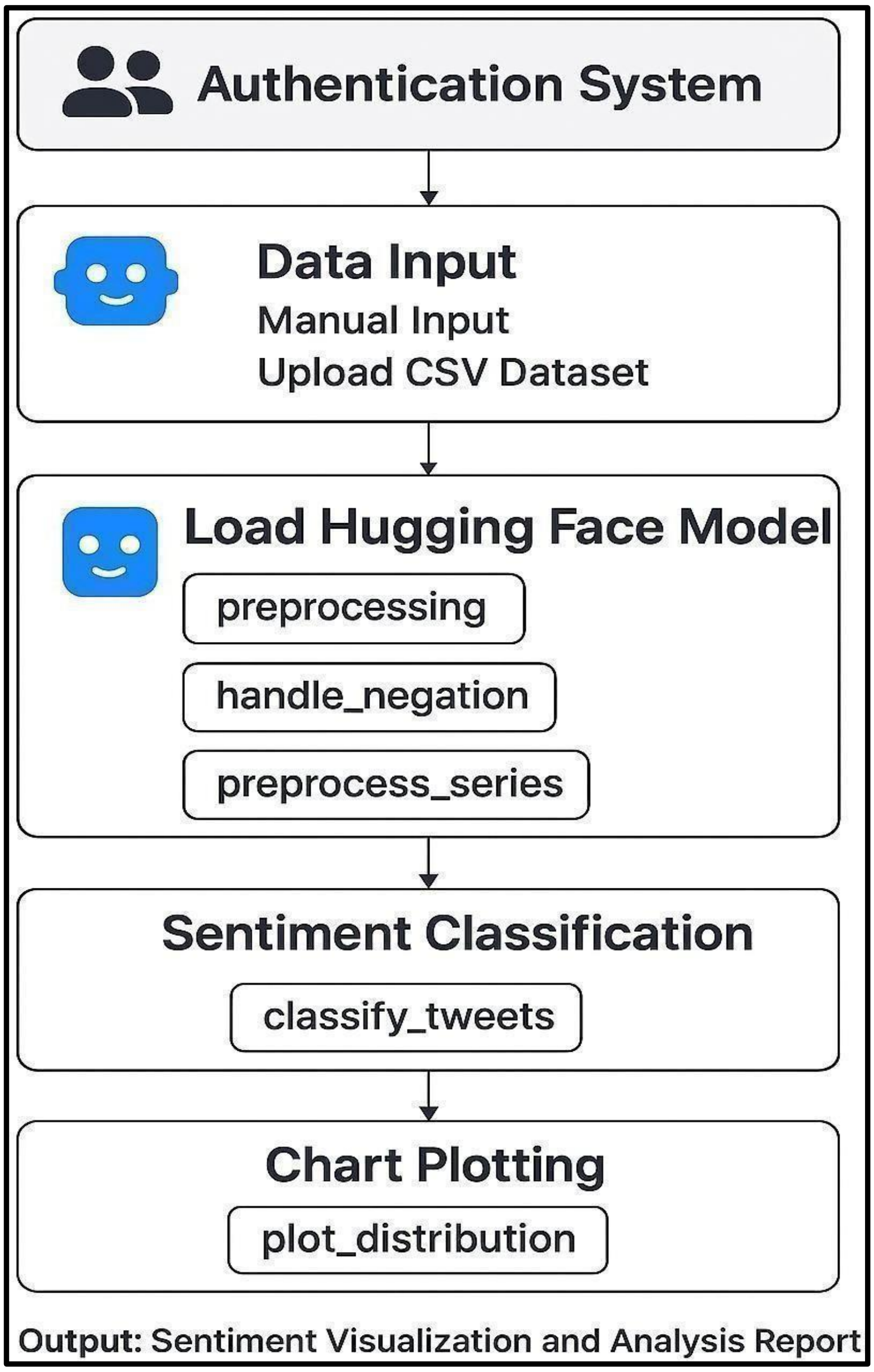


Figure 3.1: Architectural diagram

Figure 3.1 shows the Architectural diagram An architectural diagram refers to a high-level visual representation of a system’s structure. It shows how major components are organized, how they interact, and how data flows between them.

3.3 Specification using Use Case Diagrams:

The relationship among the user and the system is shown in a simple way using use cases. A name within the ellipse indicates the use cases. A stickman notation is used for actors, with the name being placed below and a solid line connecting actor.

.

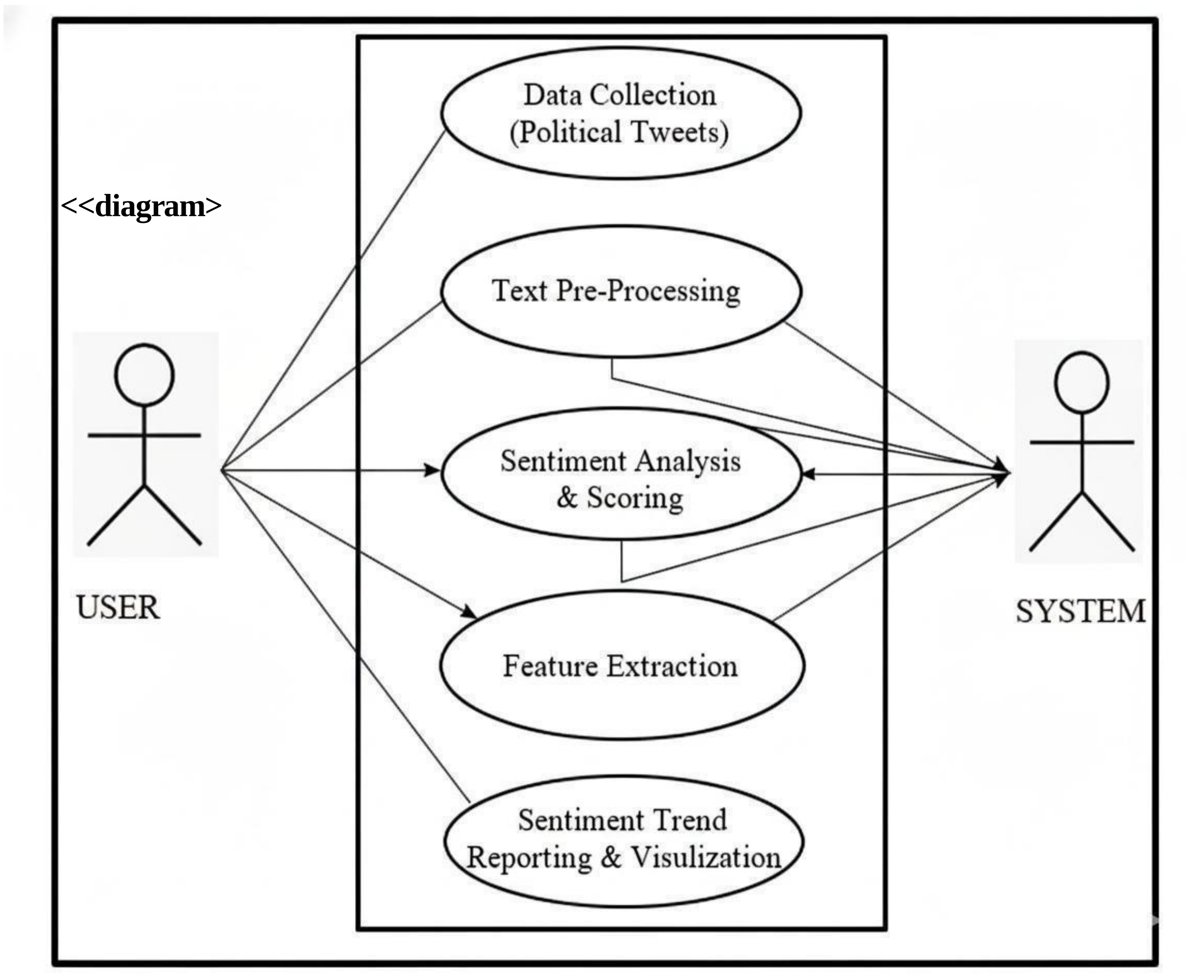

Data Collection (Political Tweets)
<<diagram>
Text Pre-Processing
Sentiment Analysis & Scoring
USER
SYSTEM
Feature Extraction
Sentiment Trend Reporting & Visulization

Figure 3.2: Use case diagram

Figure 3.2 shows the use case diagram of the Sentimental analysis of political tweets. In this use case diagram, there are five sets of use cases and two actors. Each use case represents the functionality provided by the system. The actor actively takes part in the first use case by collecting and sending it for preprocessing. The second phase is to pre-process the dataset. The pre-processing techniques used here for cleaning texts. The third phase is to classify the text into positive, negative and neutral and fourth is to feature extraction of texts. In the last phase visualization takes place and the result is provided for the user accordingly.

3.4 Module Specification

Module Specification is the way to improve the structure design by breaking down the system into modules and solving it as an independent task. By doing so the complexity is reduced and the modules can be tested independently. The number of modules for this model is five, namely Data collection module, Preprocessing module, Classification module, Feature extraction module, Visualization module and lastly User interface module.

3.4.1 Data Collection Module

Name of the module: Data collection

Actors: User, System

Use cases: Enter tweets, Upload a Csv dataset

Functionality: The main functionality of this module is to collect political tweets from the user or to upload an CSV file to store them in the system for further analysis.

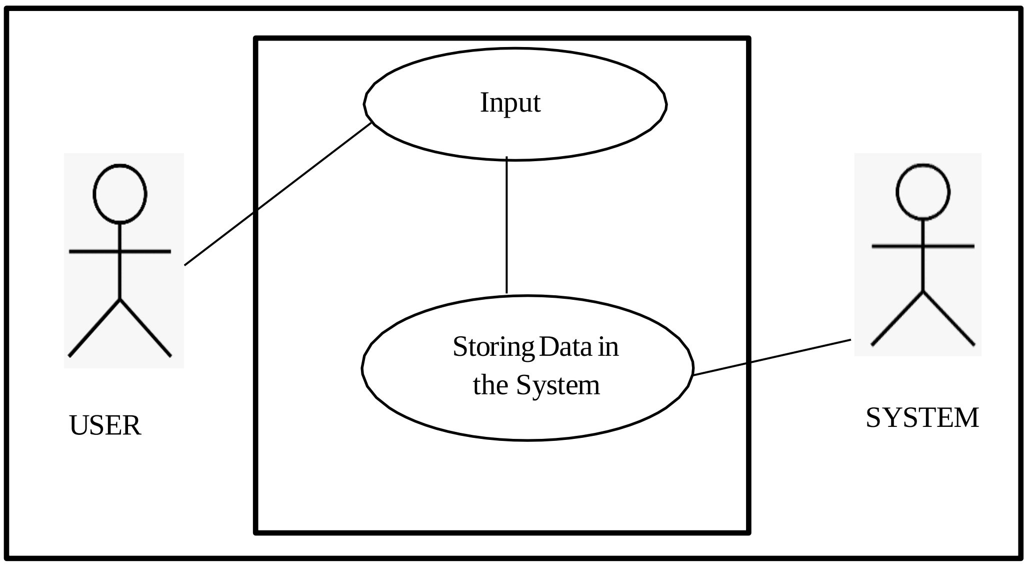


Figure 3.3: Use case diagram of Data Collection Module

Description: Figure 3.3 illustrates the **Data Collection Module**, which involves two main actors — the **User** and the **System**. In this module, the user provides input by either typing political tweets manually or uploading a dataset containing political tweets. The system then stores this collected data in a structured format such as a CSV file or database for further processing. This module ensures that all tweet data is properly gathered and stored in the system for future use of preprocessing the data.

3.4.2 Pre-processing Module

Name of the module: Pre-processing Module.

Actors: User, System.

Use cases: Input tweets, Clean and preprocess data

Functionality: The main functionality of this module is to clean and preprocess the collected political tweets to improve the accuracy of sentiment classification.

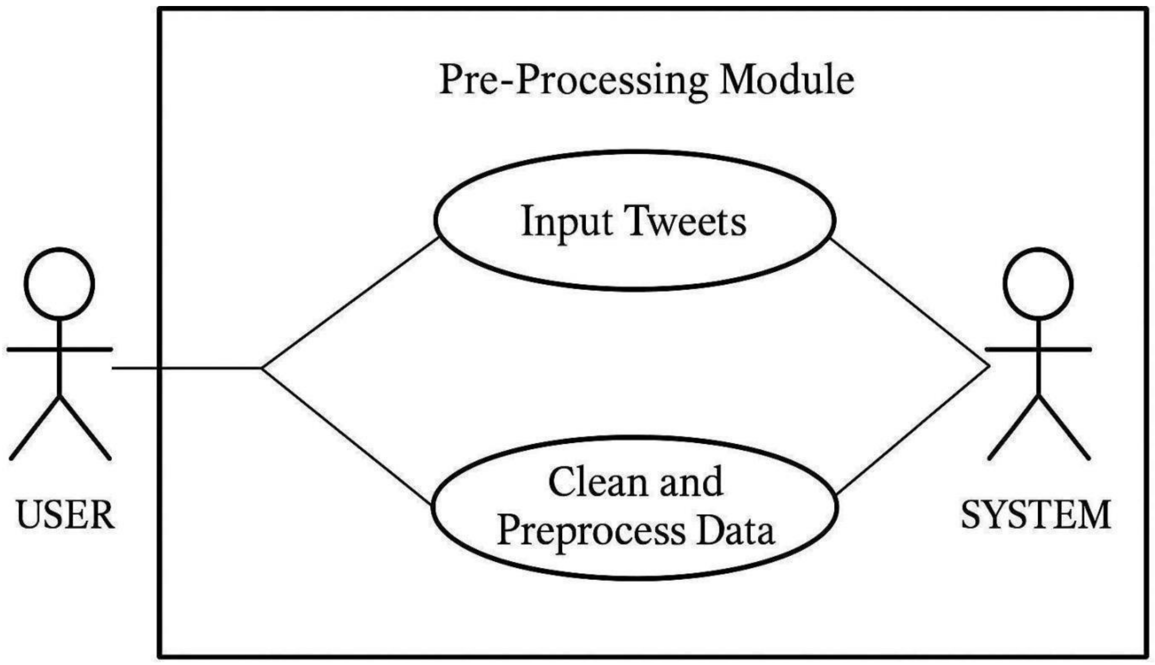


Fig 3.4: Use case diagram for preprocessing module

Description: Figure 3.4 shows the use case diagram of the Pre-processing Module. This module involves two actors — the User and the System — and two main use cases. In the first use case, the user provides the input tweets, which are collected from various sources such as Twitter or datasets. In the second use case, the system cleans and preprocesses the data by removing unwanted characters, URLs, stop words, and special symbols, and converts the text into a suitable format for sentiment analysis. This process enhances the quality and reliability of the data before classification.

3.4.3 Classification Module

Name of the module: Classification Module.

Actors: User, System

Use cases: Input Preprocessed Data, Feature Extraction, Train Model, Classify sentiment.

Functionality: The main functionality of this module is to classify each political tweet into sentiment categories such as positive, negative, or neutral using a machine learning or deep learning model.

Description: Figure 3.5 shows the use case diagram of the **Classification Module**, which includes two actors — the User and the System — and two main use cases. In the first use case, the user provides preprocessed tweet data as input for analysis. In the second use case, the system processes this data using the trained sentiment analysis model and classifies each tweet based on its sentiment polarity. The output of this module helps in understanding the overall public opinion and plays a vital role in further visualization and result interpretation.

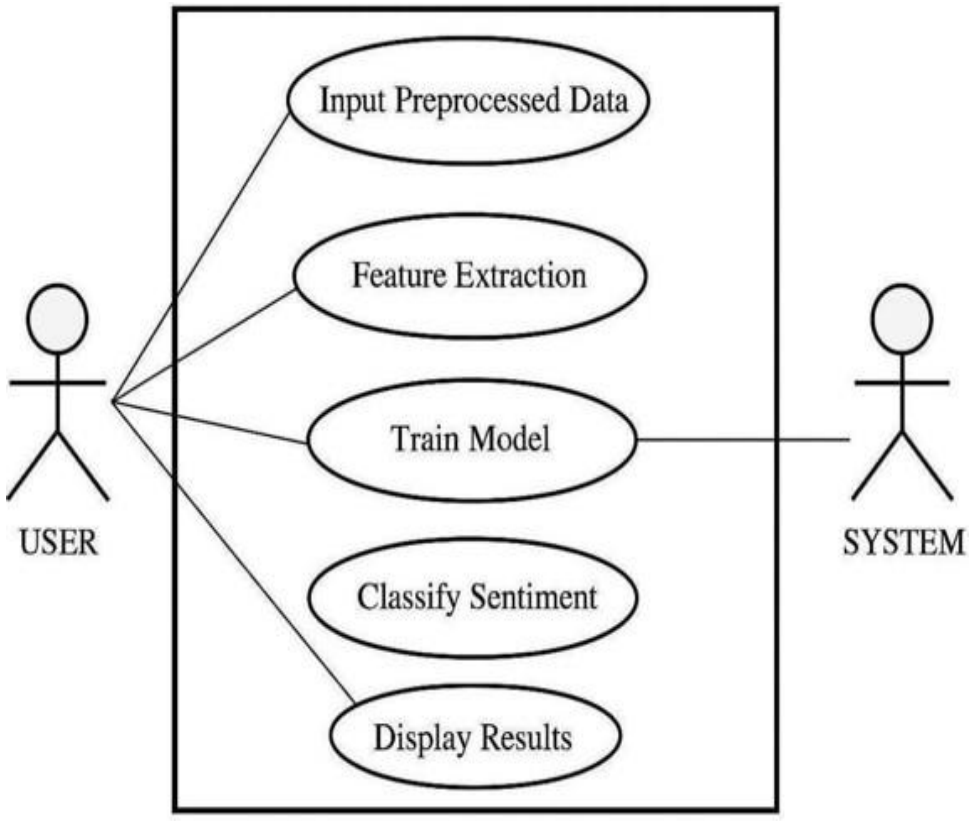


Fig 3.4: Use case diagram classification model

Figure 3.5 explains the following

- **Input Preprocessed Data:**

  The module receives cleaned and preprocessed tweets (tokenized, lowercased, with URLs/mentions/stopwords removed and optionally lemmatized). This ensures the model works on standardized text free of noise.

- **Feature Extraction:**

  Text is transformed into numerical representations the model can use for example TF-IDF vectors, Bag-of-Words counts, or dense embeddings (contextual embeddings from BERT/Transformer). Choice of features affects accuracy and computational cost.

- **Train Model:**

  A supervised learning algorithm (e.g., Logistic Regression, SVM, Naïve Bayes, or fine-tuned BERT) is trained on labeled tweets to learn patterns that map features to sentiment labels. Training includes splitting data, hyperparameter tuning, and validation to avoid overfitting.

- **Classify Sentiment:**

  The trained model predicts sentiment labels (Positive / Negative / Neutral) for incoming tweets; predictions may include confidence scores or probabilities to indicate certainty. Batch or real-time inference is supported depending on deployment.

- **Display Results:**

  Predicted labels and optional confidence scores are presented to the user via the UI (tables, single-tweet view) and visual summaries (bar charts, pie charts, or time-series plots) for easy interpretation and further analysis.

3.4.4 Feature Extraction model

Name of the module: Feature

Extraction **Actors:** User, System

Use cases: Input Text, Clean Text, Extract Features

Functionality: The main functionality of this module is to convert preprocessed political tweets into numerical feature representations that can be used by machine learning models for classification.

Description: Figure 3.6 includes the **Feature Extraction Module** which plays a vital role in converting raw text data into meaningful numerical representations that can be understood by machine learning models. After preprocessing, the cleaned political tweets are passed into this module. Key features such as word frequency, sentiment-bearing terms, and contextual relationships are extracted using techniques like Bag of Words (BoW), TF-IDF (Term Frequency–Inverse Document Frequency), or Word2Vec embeddings. These extracted features help capture the sentiment patterns in tweets, forming the basis for accurate classification in the next module. This process ensures that the classifier can effectively differentiate between positive, negative, and neutral sentiments.

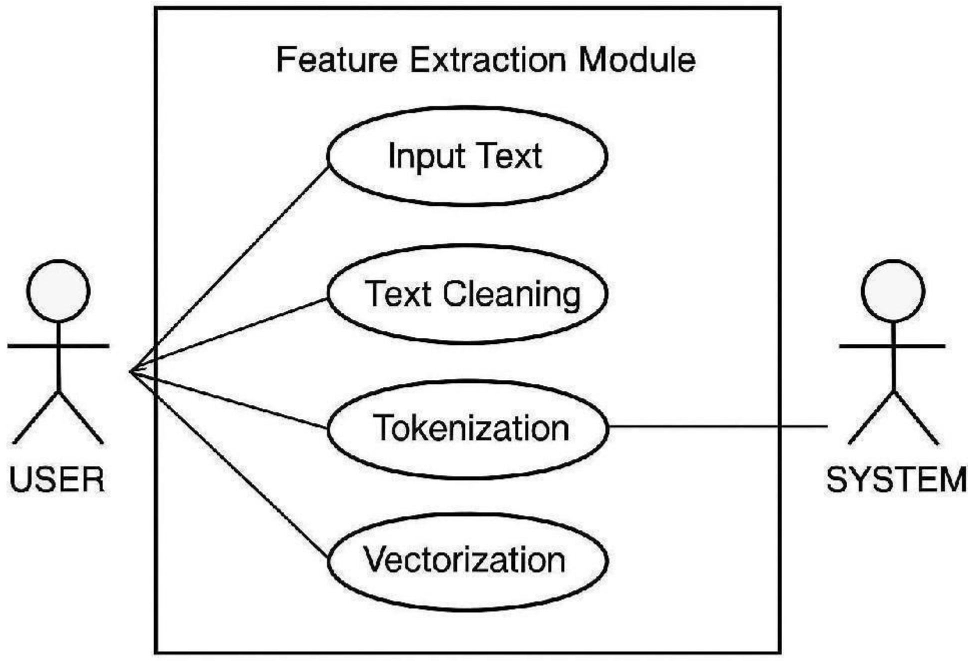


Fig 3.6: Use case diagram of Feature extraction module

Figure 3.6 explains the following

- **Input Text:** In this step, the user provides preprocessed political tweet data as input. The data is taken from previous modules and passed into the system for further feature extraction.The stage acts as the foundation for the entire text- processing workflow because the quality and organization of the input directly influence the performance of the following steps.
- **Text cleaning:** This step removes unwanted elements such as URLs, hashtags, emojis, stop words, and punctuation. It ensures that only meaningful text remains for accurate analysis**.** These components do not contribute to the understanding of sentiment or political meaning, so they must be removed or normalized.
- **Tokenization:** Once the text is cleaned, it is broken down into smaller meaningful units known as tokens. These tokens are usually words, but depending on the approach, they can also be phrases or sub-word units. Tokenization helps the system interpret the structure of the sentences, making it easier to analyze relationships between words.
- **Vectorization:** The tokens are converted into numerical form using techniques like Bag of Words, TF-IDF, or word embeddings. This allows the machine learning model to process and analyze the textual data effectively.

3.4.5 Visualization Module

Name of the Module: Visualization Module

Actors: User, system

Use Cases: Input Classified Data, Generate Graphs, Display Insights

Functionality:

The main functionality of this module is to present the classified sentiment results in an easy- to-understand visual format using charts and graphs.

Description:

Figure 3.7 shows the use case diagram of the Visualization Module, which includes two main actors — the User and the System — and three primary use cases. In the first step, the system receives the classified sentiment data (positive, negative, neutral) from the classification module. The system then generates visual representations such as bar graphs, pie charts, or line charts to show sentiment distribution and trends. Finally, the results are displayed to the user in an interactive dashboard for better understanding and interpretation of public opinion on political topics. This module helps in identifying sentiment trends, comparisons, and overall insights from the analyzed tweets.

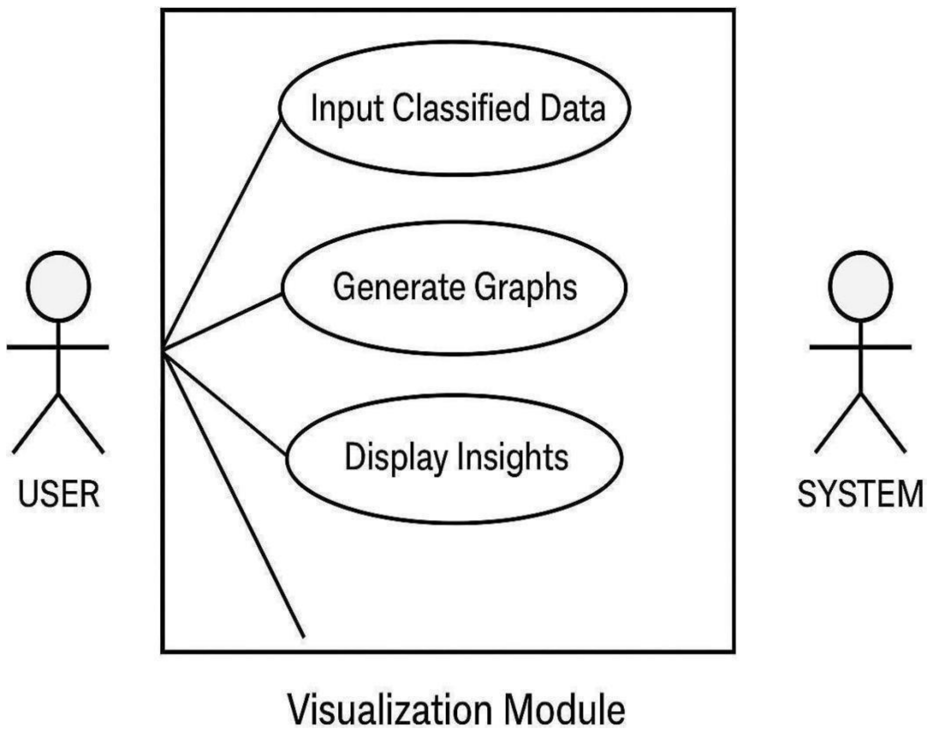


Fig 3.7: Use case diagram of Visualization Module

- **Input Classified Data:** In this step, the system takes the classified sentiment data (positive, negative, and neutral) generated by the sentiment classification module. The data is collected in a structured format for easy visualization and further analysis.
- **Generate Graphs:** The system processes the classified data to create visual representations such as bar charts, pie charts, or line graphs. These visuals help in understanding the sentiment distribution and trends related to political tweets more effectively.
- **Display Insights:** The final step involves presenting the generated visual results to the user through an interactive dashboard or interface. This allows users to easily interpret the sentiment patterns, compare results, and draw meaningful conclusions about public opinion.

3.4.6 User Interface Module

Name of the Module: User interface module.

Actors: User, System

Use cases: Provide Input, View Output

Functionality: The main functionality of this module is to provide an interactive platform for the user to interact with the system. It enables the user to input political tweets, initiate sentiment analysis, and view the classified results in an easy-to-understand format.

Description: The User Interface Module serves as the communication bridge between the user and the system. The user interacts with the application through a graphical or web-based interface.

In the first step, the user can enter tweets manually or upload a dataset of political tweets. The system then processes these inputs and displays the analyzed sentiments (positive, negative, or neutral). This module focuses on enhancing user experience by ensuring smooth navigation, clarity in result presentation, and responsiveness across different devices. The user interface provides a simple, intuitive, and efficient environment for interacting with the sentiment analysis system, making the overall process user-friendly and accessible. Overall the system works more efficiently with the user friendly interface which helps the user to easily use all the options and functions available in the project and hence they can enhance the better visualization of the graphs which is present in the output and all the other details.

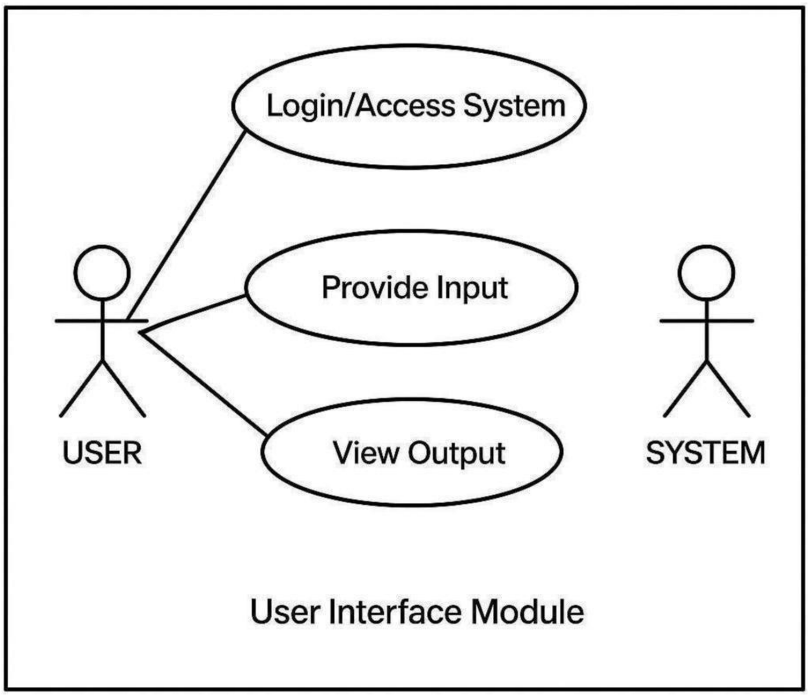


Fig 3.8: Use case diagram of User Interface Module

- **Login/Access System:** The step allows the user to log in or access the application interface, ensuring that only authorized individuals can operate the system. It helps maintain data privacy and prevents misuse of sensitive political information. In some cases, users may directly access the interface without authentication if public data input is allowed. The login process ensures that user actions can be tracked and managed efficiently
- **Provide Input:** In this step, the user provides input to the system, which may include entering a single tweet manually or uploading an entire dataset of political tweets for analysis. The system accepts the text input and automatically forwards it to the preprocessing and classification modules. The interface may support multiple file formats like CSV, Excel to improve usability. This step acts as the starting point for data processing and model-driven sentiment evaluation.
- **View Output:** After processing the input, the system displays the results to the user in a clear and structured manner. The output typically shows the sentiment classification of tweets—positive, negative, or neutral—along with possible visualizations such as charts or graphs. It helps users interpret the political sentiment trends and gain meaningful insights from the analyzed data.

3.5 Data Flow Diagram

Data flow diagram explains how the data flows between different processes. The flow diagram is composed of the input, process and the output. After each of the processes the data that flows between the systems need to be specified and hence called the data flow diagram. It is the initial step for designing the system to be implemented. It also shows where the data has originated, where it flows and where it is stored.

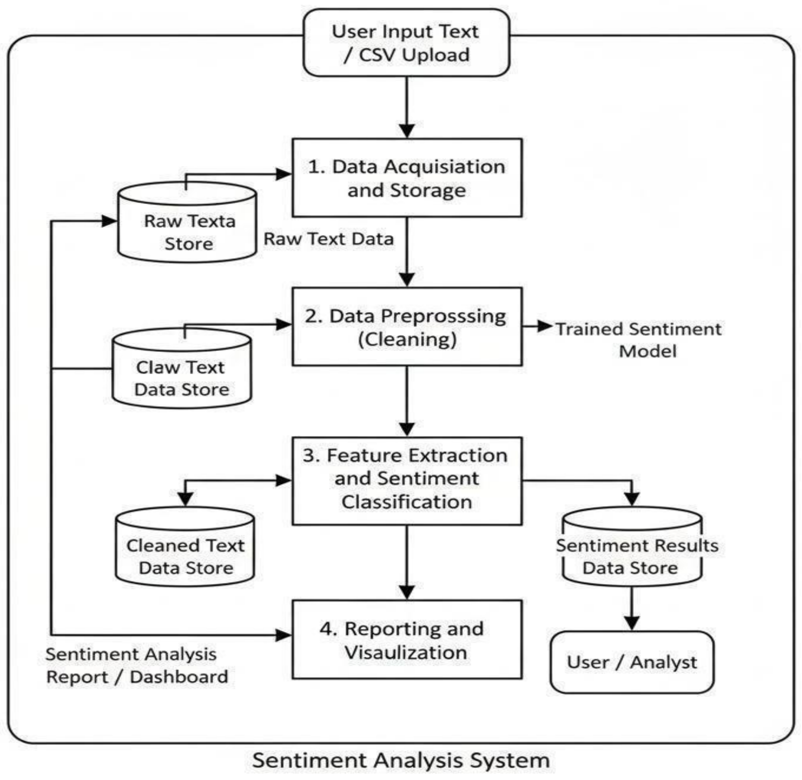

User Input Text / CSV Upload
1. Data Acquisiation and Storage
Raw Texta Store
Raw Text Data
2. Data Preprosssing (Cleaning)
Trained Sentiment Model
Claw Text Data Store
3. Feature Extraction and Sentiment Classification
Cleaned Text Data Store
Sentiment Results Data Store
4. Reporting and Visaulization
Sentiment Analysis Report / Dashboard
User / Analyst
Sentiment Analysis System

Fig 3.9: Data flow diagram of Sentimental analysis system

The Figure 3.9 refers the Data Flow Diagram for Sentiment Analysis of Political Tweets illustrates the overall process of analyzing sentiments from collected political tweets. The system begins with **data collection**, where tweets are gathered manually or from a dataset. The collected data then undergoes **preprocessing**, which includes cleaning, tokenization, and text normalization to prepare it for analysis. Next, in the **feature extraction** phase, important features such as keywords and sentiment-related terms are extracted. These features are then passed to the **classification module**, where machine learning models classify tweetsinto **positive, negative, or neutral** categories. Finally, the **visualization module** displays the sentiment results through graphs or charts, providing clear insights into the overall trends.

3.5.1 Data flow diagram for Data Acquisition and Storage

The Data Acquisition & Storage module initiates the sentiment analysis process by focusing on the reliable ingestion and secure storage of the raw data. This module receives Raw Text Data directly from the Input Text / CSV Upload source, acting as the system's entry point. Upon receipt, the central process, Receive and Store Data, first performs data validation to ensure the input is in a usable format and then executes data parsing to correctly extract the core textual content (the political tweets or text) along with any available metadata. The final and most critical step is the archival of this unedited, original data into the Raw Text Data Store. This repository serves as the single point of truth, ensuring that the source data is maintained securely and remains available for verification or future processing without alteration.

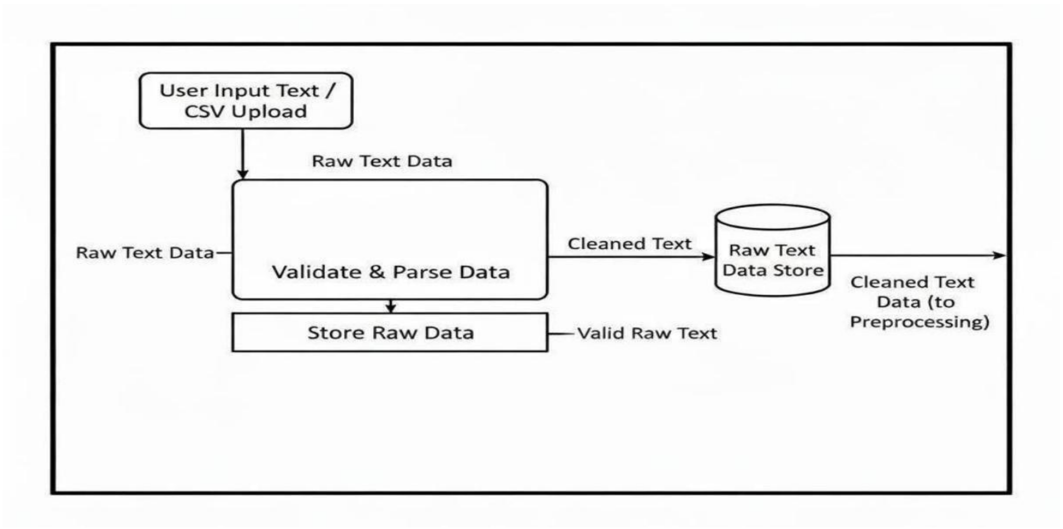


Fig 3.10 Data flow diagram for Data Acquisition and Storage Module

3.5.2 Data Flow diagram for Data Preprocessing (Cleaning).

The figure 3.10 shows how the Data Preprocessing (Cleaning) Module functions as the essential intermediate step that transforms raw, noisy text into a structured format ready for analysis. It starts by retrieving the Raw Text Data from its store, immediately subjecting it to a series of sequential Natural Language Processing (NLP) transformations. These transformations include removing unnecessary elements such as URLs, hashtags, emojis, user mentions, numbers, and punctuation to reduce noise within the datasets and the module may also convert all text to lowercase to maintain consistency and normalize variations in spelling or formatting.

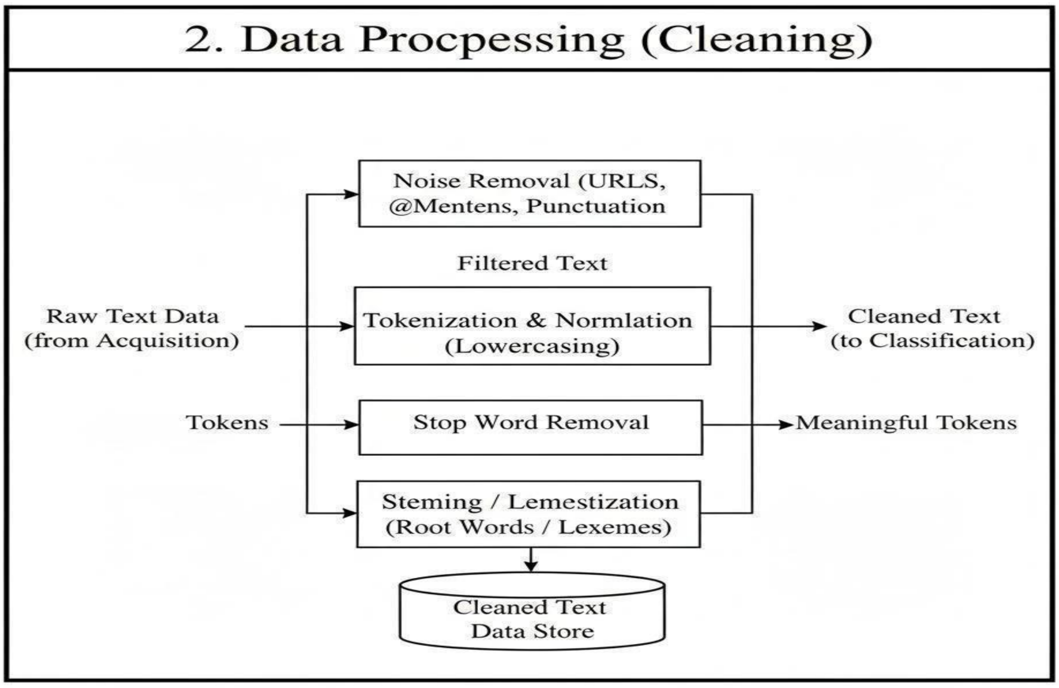

2. Data Procpessing (Cleaning)
Noise Removal (URLS, @Mentens, Punctuation
Filtered Text
Raw Text Data (from Acquisition)
Tokenization & Normlation (Lowercasing)
Cleaned Text (to Classification)
Tokens
Stop Word Removal
Meaningful Tokens
Steming / Lemestization (Root Words / Lexemes)
Cleaned Text Data Store

Fig 3.11 : Data flow diagram of Data preprocessing module

The figure 3.11 refers us the first processes are Noise Removal and Tokenization, which eliminate irrelevant characters (like URLs, @mentions, and special symbols) and break the text down into individual, manageable word units, respectively. Following this, Stop Word Removal eliminates common, non-meaningful words (e.g., 'the', 'is', 'a') to increase the analytical weight of key terms, while Normalization standardizes all text by converting it to a uniform case. Finally, Stemming/Lemmatization reduces words to their base or root form, ensuring that variants of the same word (e.g., 'running', 'ran', 'runs') are treated as a single feature. The output of this entire cleaning sequence is the Cleaned Text Data, which is then stored for direct use by the subsequent sentiment classification algorithms.

3.5.3 Data flow Diagram for Feature Extraction & Classification

The Feature Extraction & Classification Module constitutes the analytical core of the sentiment analysis project, responsible for converting human-readable text into mathematical features and assigning sentiment polarity. This process starts by retrieving the Cleaned Text Data and immediately channeling it through Feature Extraction (Vectorization). This transformation converts the linguistic tokens into Feature Vectors (numerical arrays) using methods like Bag-of-Words or TF-IDF, making the data quantifiable for machine learning.

The resulting numerical features are then passed to the Trained Sentiment Model, which applies its learned algorithm to process these vectors. The final, critical task is Sentiment Classification, where the model outputs a definitive Sentiment Label (e.g., Positive, Negative, Neutral) and a corresponding Confidence Score for each piece of text. These definitive results are then stored in the Sentiment Results Data Store, creating the final classified dataset ready for reporting and visualization.

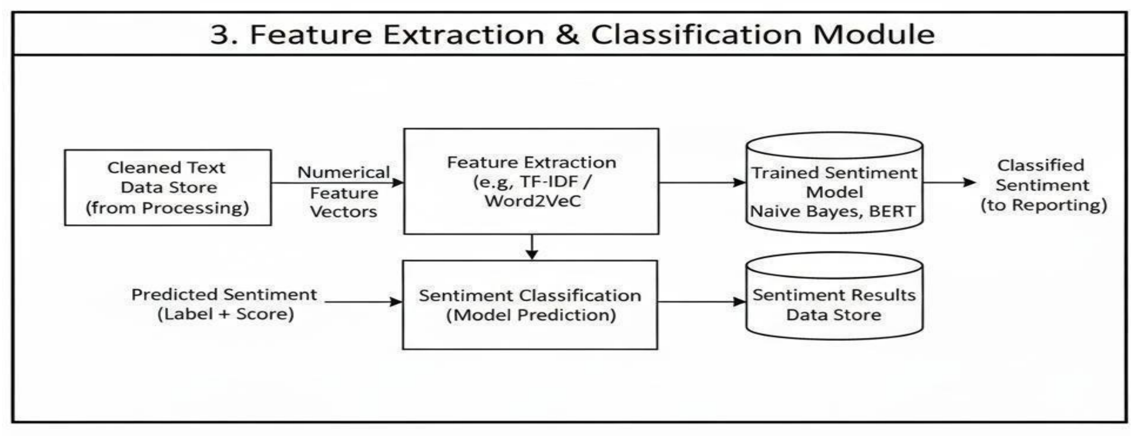

3. Feature Extraction & Classification Module
Cleaned Text Data Store (from Processing)
Numerical Feature Vectors
Feature Extraction (e.g, TF-IDF / Word2VeC
Trained Sentiment Model Naive Bayes, BERT
Classified Sentiment (to Reporting)
Predicted Sentiment (Label + Score)
Sentiment Classification (Model Prediction)
Sentiment Results Data Store

Fig 3.12: Data flow diagram of Feature Extraction Module

3.5.4 Data flow diagram for Reporting & Visualization

The figure 3.12 refers us the Reporting & Visualization Module serves as the system's final output engine, translating complex analytical results into actionable and easily interpretable insights for the user. This process begins by accessing the Sentiment Results Data Store and immediately subjecting the classified data to Result Aggregation, where individual tweet sentiments are summarized to calculate overall statistics, such as the total percentage breakdown of positive, negative, and neutral opinions. This aggregated data then flows into the Visualization Generation step, where various chart types including bar charts for overall sentiment, line graphs for trend analysis over time, and word clouds for dominant features are created. The final output is the Report Compilation, where these visual elements, along with key performance indicators and a summary analysis, are assembled into a coherent Sentiment Analysis Report/Dashboard, which is the final product delivered back to the external user/analyst. The module may generate downloadable formats such as PDF, Excel, or interactive dashboards, allowing users to store, share, or present the findings easily. The report may also include comparative insights that highlight how sentiment changes across different time periods, political events, or specific keywords. Detailed tables summarizing tweet counts, sentiment distributions.

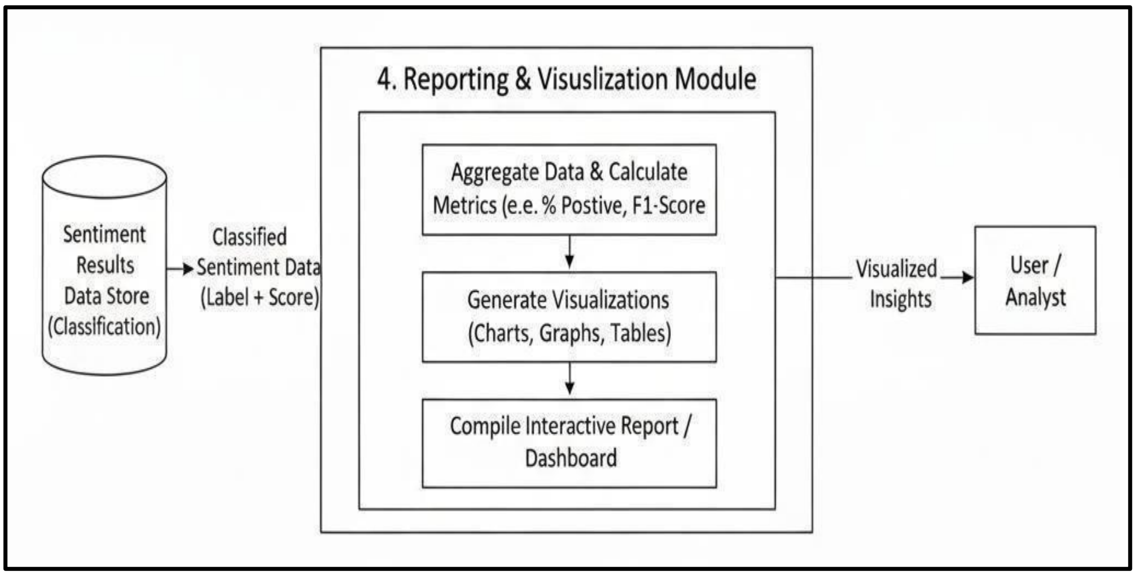

4. Reporting & Visuslization Module
Sentiment Results Data Store (Classification)
Classified Sentiment Data (Label + Score)
Aggregate Data & Calculate Metrics (e.e. % Postive, F1-Score
Generate Visualizations (Charts, Graphs, Tables)
Compile Interactive Report / Dashboard
Visualized Insights
User / Analyst

Fig 3.13: Data flow diagram of Reporting and visualization module

3.6 State chart diagram

The figure 3.13 is a state chart diagram is also named as a state diagram. It is a popular one among five UML diagrams and it is used to model the dynamic nature of the system. State chart defines various states of an object during its lifetime. The state chart diagram is a composition of a finite number of states and the functionalities describe the functioning of each module in the system.

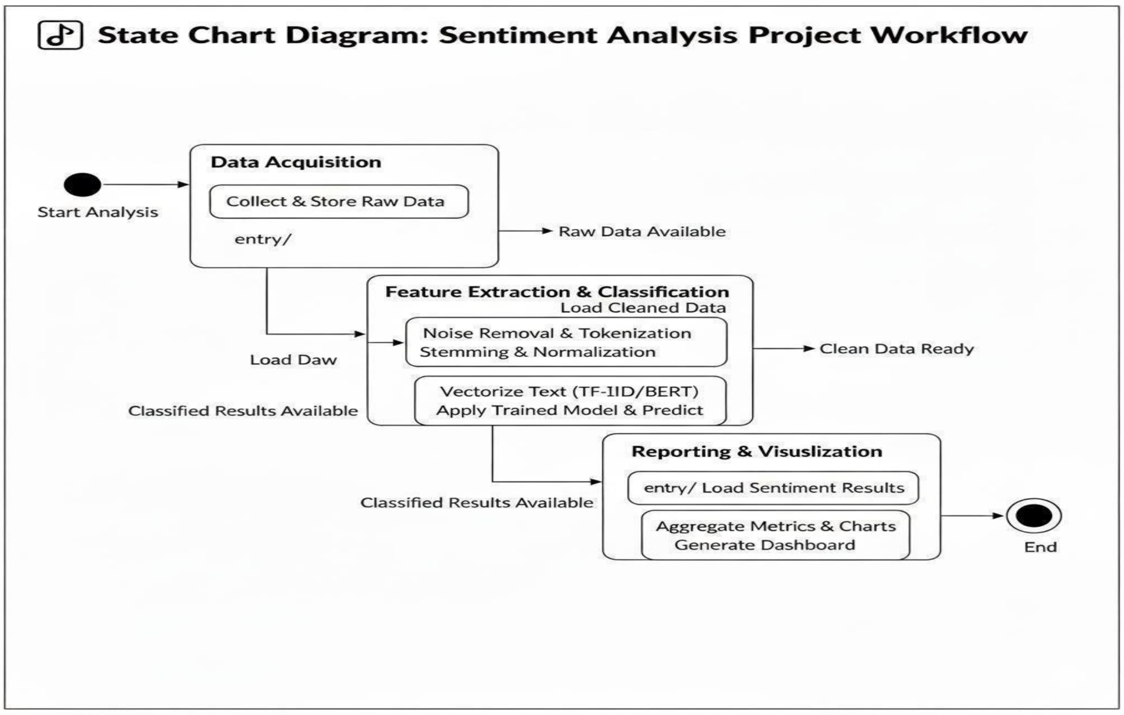

State Chart Diagram: Sentiment Analysis Project Workflow
Start Analysis
Data Acquisition
Collect & Store Raw Data
entry/
Raw Data Available
Load Daw
Classified Results Available
Feature Extraction & Classification
Load Cleaned Data
Noise Removal & Tokenization
Stemming & Normalization
Vectorize Text (TF-IID/BERT)
Apply Trained Model & Predict
Clean Data Ready
Classified Results Available
Reporting & Visuslization
entry/ Load Sentiment Results
Aggregate Metrics & Charts
Generate Dashboard
End

Fig 3.14: State chart diagram

It is a graph where each state is a directed edge and represented by a node and these directed edges represent transition between states. Each state name must be a unique name.

The Sentiment Analysis project begins at the Start State (solid circle) and transitions to Raw Data Received upon [Data Uploaded], where the unprocessed text is stored. The system then moves to the Data Preprocessing state upon [Preprocessing Initiated], where text is cleaned and normalized. Once [Preprocessing Complete], the system enters Clean Data Ready. This clean data is passed to the Model Classification state, triggered by [Classification Initiated], where the model detects polarity and assigns scores. After [Classification Complete], the system reaches Results Classified. Finally, triggered by [Reporting Initiated], the system enters Reporting & Visualization to aggregate and visualize the data, delivering the final Sentiment Analysis Report/Dashboard upon [Report Output], concluding the process at the End State (bullseye symbol).

3.7 Summary

In the third chapter, high level design of the proposed method is discussed. Section 3.1 presents the design considerations for the project. The system architecture of the proposed system is illustrated in section 3.2 , the next section 3.3 describes specification using use case diagram. Section 3.4 describes module specification using use case diagrams for all the five modules. The data flow diagram for the system is explained in section 3.5. The state chart diagram for the system is explained in section 3.6.

CHAPTER 4

DETAILED DESIGN

A detailed design explains the structure and working of each individual module used in the project. It is the second stage of development after the high-level design, focusing on how each module performs its function. This phase helps in organizing the system effectively and makes the implementation process easier and faster.

4.1 Structural chart

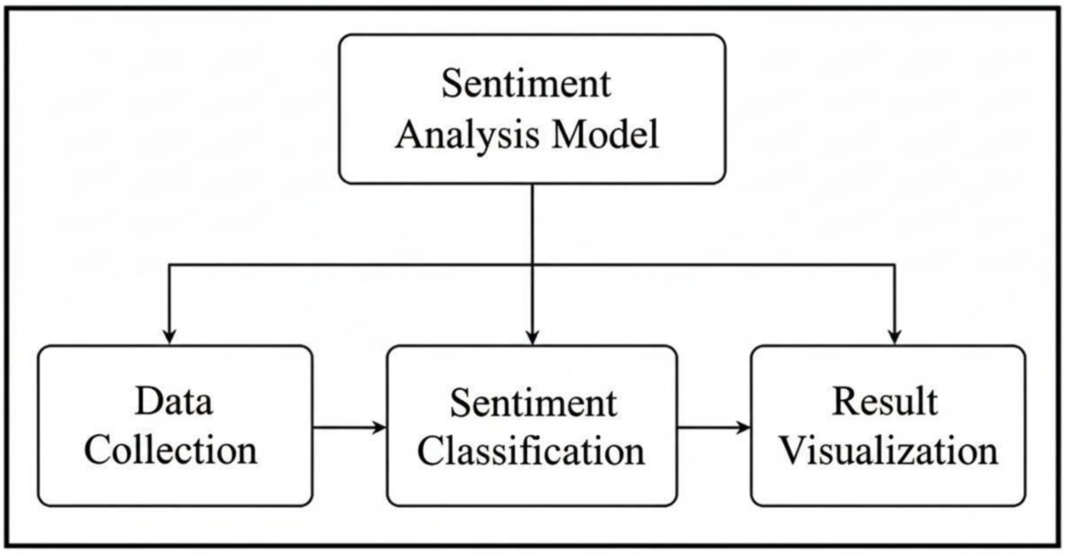


Figure 4.1: Structural chart diagram

The figure 4.1 illustrates the structural chart of the Sentiment Analysis Model consists of several modules such as Data Collection, Pre-Processing, Sentiment Classification, and Result Visualization. The Data Collection module gathers political tweets from various sources. The Pre-Processing module cleans and prepares the data by removing unwanted characters and stop words. The Sentiment Classification module analyzes each tweet and classifies it as positive, negative, or neutral using machine learning or BERT models. Finally, the Result Visualization module displays the sentiment distribution through charts and graphs for better understanding. This systematic, modular approach ensures accurate and interpretable analysis of political sentiment. Each module works independently yet seamlessly with the others, ensuring smooth data flow throughout the entire system. The structured workflow enhances reliability by reducing errors at each stage and ensuring that the final sentiment outputs are both consistent and meaningful.

4.2 Flow chart of Preprocessing

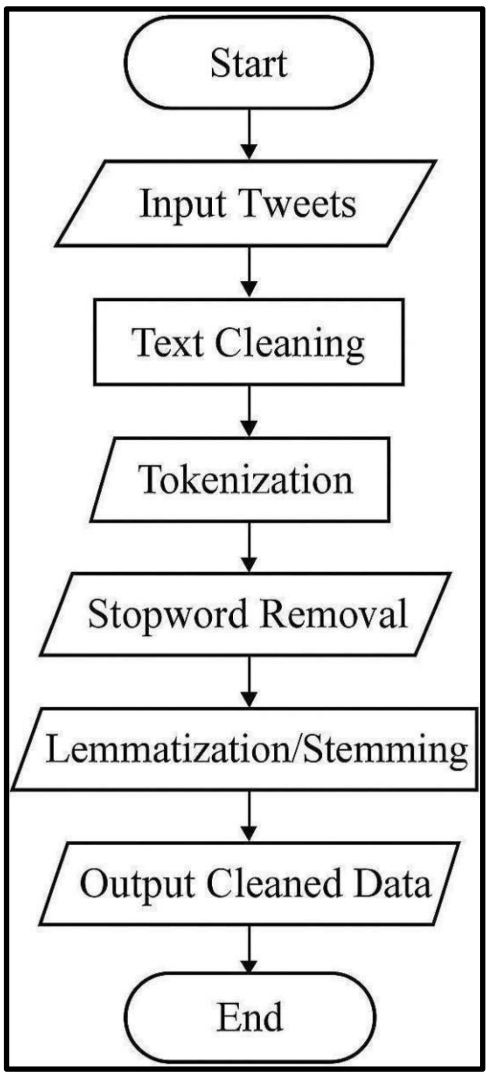


Figure 4.2: Flow chart of Pre-processing module

Figure 4.2 illustrates the preprocessing workflow for sentiment analysis of political tweets. It begins with the Start step, where raw tweet data is collected as input. The next step, Input Tweets, represents loading the textual data from sources like social media platforms. The Text Cleaning process removes unwanted elements such as URLs, mentions, hashtags, punctuation, and stopwords to refine the text.

The Preprocessing Module is a foundational component of the sentiment analysis system, engineered to transform raw, unstructured political tweet data into a clean, structured, and numerical format suitable for machine learning.

Then followed by Tokenization, which splits the text into smaller units or tokens for easier analysis. The final step, End, marks the completion of preprocessing, resulting in clean and structured data ready for model training and sentiment classification.

4.3 Flow chart of Classification

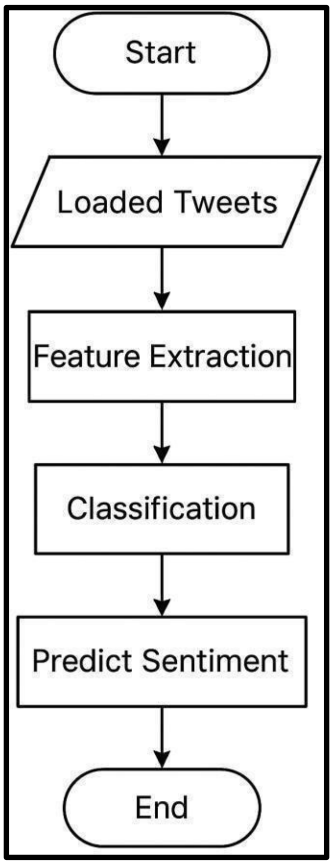


Figure 4.3: Flow chart of Classification

Figure 4.3 illustrates the classification process for sentiment analysis of political tweets begins after preprocessing the text data. Initially, the cleaned tweets are fed into the feature extraction stage, where meaningful numerical representations of words are created using techniques such as TF-IDF or word embeddings like BERT. These extracted features are then passed to the classification model, which may include machine learning algorithms such as Logistic Regression, SVM, or advanced deep learning models like BERT and LSTM.

The model performs training and testing phases during training, the model performance on unseen tweets. The classifier then predicts the sentiment category labeling each tweet as positive, negative, or neutral.

Finally, the results are visualized using graphs or charts to compare model accuracy, F1-score, and other performance metrics. This process ensures accurate and efficient sentiment detection, providing valuable insights into political opinions expressed on social media platforms.

4.4 Flow chart of Result and Visualization Module

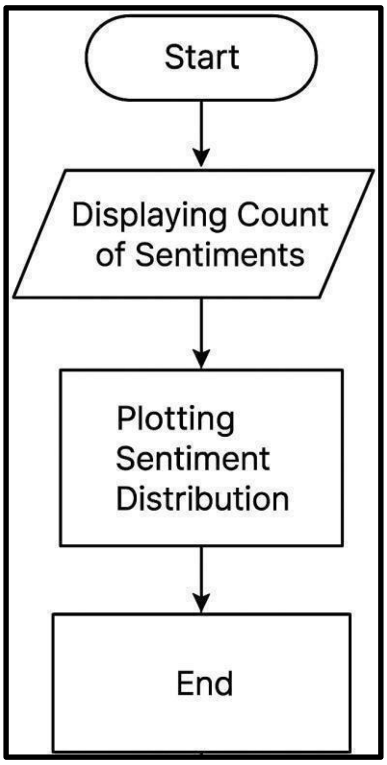

Start
Displaying Count of Sentiments
Plotting Sentiment Distribution
End

Figure 4.4: Flow chart of Result and Visualization Module

Figure 4.4 illustrates the Result and Visualization Module for sentiment analysis of political tweets is responsible for presenting the processed outcomes of the classification model in an interpretable and user-friendly manner. The process begins by taking the classified sentiment results which can be positive, negative, or neutral from the sentiment classification model. These results are then passed into the result processing unit, The aggregated data is then fed into the visualization engine to create dynamic dashboards, featuring interactive pie charts, bar graphs, and sentiment trend lines. This allows users to instantly grasp the overall public mood and drill down into sentiment polarity regarding specific topics or candidates.

The module aggregates and summarizes the sentiment distribution. The module includes a data visualization component, which converts numerical sentiment data into graphical representations such as bar charts, pie charts, or trend graphs.

These visualizations help users easily understand the overall political sentiment trends across different parties, topics, or time periods. Finally, the result display interface presents these visual insights to the user through dashboards or reports, enabling meaningful interpretation and decision-making based on the analyzed political tweets.

4.5 Summary

The Detailed Design of the Sentiment Analysis of Political Tweets project outlines the internal structure and flow of the system, describing how each module contributes to accurate sentiment prediction. It begins with collecting political tweets using APIs, followed by preprocessing to clean and prepare the text data. The cleaned tweets are then converted into numerical form through feature extraction techniques like TF-IDF or BERT embeddings. These features are passed to machine learning or deep learning models for classification into positive, negative, or neutral sentiments. Finally, the results are visualized using charts and graphs to present clear insights into public political opinions. This design ensures efficient data processing, accurate analysis, and easy interpretation of sentiment trends.

CHAPTER 5

IMPLEMENTATION

5.1 Implementation

The implementation phase of project development is the most important phase as it adds the final solution, which solves the problem at hand. The implementation phase solves the actual materialization of the ideas, which are expressed in the analysis document and developed in the design phase. Implementation should be perfect mapping of the design document in a suitable programming language in order to achieve the necessary final product. Often the product is ruined due to incorrect programming language chose for implementation or unsuitable method for programming. It is better for the coding phase to be directly linked to the design phase in the sense if the design is in terms of object-oriented way. The factors concerning the programming language and platform chosen are described in the next couple of sections.

Implementation of any software is always preceded by important decision regarding selection of the platform, the language used, etc. these decisions are often included by several factors such as real environment in which the system works, the aspect that is required, the security concerns and other implementation specific details. These are two major implementation decisions that have been made before the implementation decisions that have been made before the implementation of this project. They are as follows:

- Selection of platform (Operating system).
- Selection of programming language for development of the application.

5.2 Implementation Requirements

To execute training data the used software is as follows:

- Python language is used as Programming language
- Windows 10 is the operating system
- Political datasets
- Python IDLE for testing
- Streamlit for front end (GUI)

5.3 Programming Language Selection

The ideal programming language for this project is Python. It is selected due to its rich ecosystem of specialized libraries (e.g., NLTK, Scikit-learn, and Hugging Face for NLP and Machine Learning), its excellent community support, and its ease of integration with data visualization tools necessary for the final report.

5.3.1 KeyFeatures of Programming Language Selected

Python is a general purpose, dynamic, high-level, and interpreted programming language. It supports Object Oriented programming approach to develop applications. It is simple and easy to learn and provides lots of high-level data structures. Python is easy *to* learn yet powerful and versatile scripting language, which makes it attractive for Application Development. Python's syntax and dynamic typing with its interpreted nature make it an ideal language for scripting and rapid application development. Python supports multiple programming pattern, including object-oriented, imperative, and functional or procedural programming styles.

Python is not intended to work in a particular area, such as web programming. That is why it is known as multipurpose programming language because it can be used with web, enterprise, 3D CAD, etc. We don't need to use data types to declare variable because it is dynamically typed so we can write a=10 to assign an integer value in an integer variable. Python makes the development and debugging fast because there is no compilation step included in Python development, and edit-test-debug cycle is very fast.

Python provides many useful features to the programmer. These features make it most popular and widely used language. We have listed below few-essential feature of Python.

- Easy to use and Learn
- Expressive Language
- Interpreted Language
- Object-Oriented Language
- Open-Source Language
- Extensible
- Learn Standard Library

- Integrated
- Embeddable
- Dynamic Memory Allocation
- Wide Range of Libraries and Frameworks

Where is Python used?

Python is a general-purpose, popular programming language and it is used in almost every technical field. The various areas of Python use are given below.

- Data Science
- Date Mining
- Desktop Applications
- Console-based Applications
- Mobile Applications
- Software Development
- Artificial Intelligence
- Web Applications
- Enterprise Applications
- 3D CAD Applications
- Machine Learning
- Computer Vision or Image Processing Applications.
- Speech Recognitions

Python Basic Syntax

There is no use of curly braces or semicolon in Python programming language. It is English-like language. But Python uses the indentation to define a block of code. Indentation is nothing but adding whitespace before the statement when it is needed.

In the below example, the statements that are same level to right belong to the function. Generally, users can use four whitespaces to define indentation.
The following is the example given below :

For example -

```
def func():
    statement 1
    statement 2
    …………………
    …………………
     statement N
```

Python Popular Frameworks and Libraries

Python has wide range of libraries and frameworks widely used in various fields such as machine learning, artificial intelligence, web applications, etc. We define some popular frameworks and libraries of Python as follows.

- **Web development (Server-side) -** Django Flask, Pyramid, CherryPy
- **GUI based applications -** Tk, PyGTK, PyQt, PyJs, etc.
- **Machine Learning -** TensorFlow, PyTorch, Scikit-learn, Matplotlib, Scipy, etc.
- **Mathematics -** Numpy, Pandas, etc.

Python Features

1) Easy to Learn and Use

Python is easy to learn as compared to other programming languages. Its syntax is straightforward and much the same as the English language. There is no use of the semicolon or curly-bracket, the indentation defines the code block. It is the recommended programming language for beginners.

2) Expressive Language

Python can perform complex tasks using a few lines of code. A simple example, the hello world program you simply type **print ("Hello World")**. It will take only one line to execute, while Java or C takes multiple lines. Python is easy to learn yet powerful and versatile scripting language, which makes it attractive for Application Development. Python's syntax and dynamic typing with its interpreted nature make it an ideal language for scripting and rapid application development.

3) Interpreted Language

Python is an interpreted language; it means the Python program is executed one line at a time. The advantage of being interpreted language, it makes debugging easy and portable.

4) Cross-platform Language

Python can run equally on different platforms such as Windows, Linux, UNIX, and Macintosh, etc. So, we can say that Python is a portable language. It enables programmers to develop the software for several competing platforms by writing a program only once.

5) Free and Open Source

Python is freely available for everyone. It is freely available on its official website www.python.org. It has a large community across the world that is dedicatedly working towards make new python modules and functions. Anyone can contribute to the Python community. The open-source means, "Anyone can download its source code without paying any penny."

6) Object-Oriented Language

Python supports object-oriented language and concepts of classes and objects come into existence. It supports inheritance, polymorphism, and encapsulation, etc. The object-oriented procedure helps to programmer to write reusable code and develop applications in less code.

7) Extensible

It implies that other languages such as C/C++ can be used to compile the code and thus it can be used further in our Python code. It converts the program into byte code, and any platform can use that byte code.

8) Large Standard Library

It provides a vast range of libraries for the various fields such as machine learning, web developer, and also for the scripting. There are various machine learning libraries, such as Tensor flow, Pandas, Numpy, Keras, and Pytorch, etc. Django, flask, pyramids are the popular framework for Python web development. Python is also widely used in data visualization through libraries like Matplotlib, Seaborn, and Plotly , making it easier to interpret complex datasets. In addition, its simple syntax and readability make it an excellent choice for beginners as well as experienced programmers. Python’s strong community support and extensive documentation further enhance its usability across diverse domains, including automation, artificial intelligence, data science, and backend development.

9) GUI Programming Support

Graphical User Interface is used for the developing Desktop application. PyQT5, Tkinter, Kivy are the libraries which are used for developing the web application.

10) Integrated

It can be easily integrated with languages like C, C++, and JAVA, etc. Python runs code line by line like C, C++ Java. It makes easy to debug the code.

11. Embeddable

The code of the other programming language can use in the Python source code. We can use Python source code in another programming language as well. It can embed other language into our code.

12. Dynamic Memory Allocation

In Python, we don't need to specify the data-type of the variable. When we assign some value to the variable, it automatically allocates the memory to the variable at run time. Suppose we are assigned integer value 15 to **x,** then don't need to write **int x = 15.** Just write x = 15.

5.4 Pseudocode for Each Module with Description

5.4.1 Pseudocode for Data Acquisition & Storage Module

The Data Acquisition & Storage Module is responsible for ingesting political text data from three primary sources: CSV file, raw text file upload, or direct manual text input. It validates the input data, assigns a unique ID for tracking, and securely stores the original, unprocessed data in the designated data store to establish the baseline dataset.

Procedure: Manual text and upload a CSV file

Input: Text and CSV file

Output: Path of raw text data

Begin

```
        1. Display option to the user:
              a. Enter tweets manually
              b. Upload CSV file

        2. If manual input selected then
              Prompt the user to enter tweets.

              Upload a csv file
        3. Else if CSV upload selected then
              Prompt user to upload the dataset
              file
              Read the file and extract tweet text, username, and date
              Store extracted data in Tweet_List
        4. Verify collected data for completeness
              If any missing or invalid entries, prompt re-entry or skip
        5. Return Tweet_List as raw input data for the next module
  End
```

5.4.2 Pseudocode for Pre-Processing Module

This pseudocode details the necessary steps for cleaning the raw political text data, which is essential before sentiment analysis can begin.

Procedure: Pre_Processing (dataset, manual text)

Input: Raw tweets entered/uploaded

Output: Preprocessed dataset data

```
Begin
  1. Initialize Clean_Tweet_List as an empty list
  2. For each tweet T in Tweet_List do
        a. Convert T to lowercase
        b. Remove URLs, hashtags, mentions, and special characters
        c. Remove extra spaces and punctuation marks
        d. Tokenize the text into individual words
        e. Remove stop words (e.g., "is", "the", "at")
        f. Perform stemming or lemmatization on the remaining words
        g. Rejoin tokens into cleaned text string
        h. Append cleaned text to Clean_Tweet_List
  3. Validate the preprocessed data for empty or invalid entries
        If any, remove or correct
  4. Return Clean_Tweet_List
End
```

5.4.3 Pseudocode for Feature extraction Module

The **Feature Extraction Module** transforms cleaned tweet data into numerical transforms.

This includes tokenization, vectorization, and feature scaling.

```
Procedure: Extract_ Features (Clean_Tweet_List)
Input: Preprocessed tweet dataset
(Clean_Tweet_List) Output: Numerical feature
matrix (Feature_Vector) Begin
1. Initialize Feature_Vector as an empty list
2. For each cleaned tweet T in Clean_Tweet_List do
    a. Tokenize the text into words
    b. Apply vectorization using TF-IDF to convert words into  numeric form.
    c. Store the resulting feature vector in Feature_Vector
3. Normalizeor scale Feature_Vector to ensure uniform value ranges
4. Return Feature_Vector
End
```

5.4.4 Pseudocode for Classification Module

The Classification Module is responsible for training and testing the machine learning model on extracted features.

It classifies political tweets into three sentiment categories — Positive, Negative, or Neutral.

```
Procedure: Classify_Sentiment(Feature_Vector, Labels)
Input: Extracted features (Feature_Vector), Sentiment labels (Labels)
Output: Predicted sentiment categories (Predicted_Labels)
Begin
  1. Split dataset into Training_Set and Testing_Set (e.g., 80% training, 20% testing)
  2. Select classification model (e.g., Naive Bayes, Logistic Regression, or BERT)
  3. Train model on Training_Set
      Model = Train(Feature_Vector, Labels)
  4. Test the model on Testing_Set
      Predicted_Labels = Model.Predict(Testing_Set)
  5. Evaluate performance using metrics:
      a. Precision
      b. Recall
      c. F1-Score
      d. Accuracy
  6. Store results and performance metrics
  7. Return Predicted_Labels
```

End

5.4.5 Pseudocode for Reporting & Visualization Module

The Reporting & Visualization Module presents the final sentiment analysis results in a clear and interactive format.
It displays metrics such as accuracy, precision, recall, and F1-score, along with graphical visualizations like bar charts or pie charts representing the distribution of sentiments (Positive, Negative, Neutral).

Procedure: Generate_Report_And_Visualize(Results, Metrics)

Input:

- Results: Classified sentiment data
- Metrics: Model evaluation metrics (Accuracy, Precision, Recall, F1-score)

Output:

- Visualization graphs and summary report

```
Begin
   1. Initialize visualization framework (e.g., Matplotlib, Seaborn, or Plotly)

   2. Calculate total counts of each sentiment category:
      Positive_Count = Count(Results == "Positive")
      Negative_Count = Count(Results == "Negative")
      Neutral_Count = Count(Results == "Neutral")

   3. Generate bar chart for sentiment distribution
         Plot_Bar(["Positive", "Negative", "Neutral"], [Positive_Count,
Negative_Count, Neutral_Count])

  4. Generate pie chart for sentiment percentage visualization
      Plot_Pie([Positive_Count, Negative_Count, Neutral_Count])

  5. Display model performance metrics
         Print "Accuracy:", Metrics.Accuracy
         Print "Precision:", Metrics.Precision
         Print "Recall:", Metrics.Recall
         Print "F1-Score:", Metrics.F1Score

  6. Savevisualizations and metrics as a summary report (PDF or HTML format)

  7. Display all results to the user interface

End
```

5.5 Summary

This chapter describes the implementation of the Sentimental Analysis of Political tweets by comparing with three models. Implementation requirement is deliberated in Section 5.2; Section 5.3 briefs about the programming language selected. Section 5.4 describes the pseudocode for implementation of modules.

CHAPTER 6

SYSTEM TESTING

System testing is performed to ensure that the sentiment analysis system functions correctly and meets all project requirements. It involves testing the entire system as a whole to verify that all modules including data collection, preprocessing, feature extraction, classification, and visualization work together smoothly. The testing process checks whether tweets are correctly processed, classified into positive, negative, or neutral categories, and accurately displayed in visual formats. Performance, accuracy, and user interface are also tested to confirm that the system responds efficiently to large datasets and provides reliable sentiment insights. Overall, system testing ensures the accuracy, stability, and usability of the sentiment analysis model for political tweet evaluation.

6.1 Test Procedures

System testing ensures that the sentiment analysis system functions correctly as a complete and integrated application. It verifies that all modules from data collection to visualization work together to produce accurate and reliable results.

The below are the test procedures:

- Unit Testing
- Integration Testing
- Functional Testing
- Performance Testing
- User Interface Testing
- Validation and Accuracy Testing

6.1.1 Unit Testing

Each module such as data preprocessing, feature extraction, classification, and visualization is tested individually to confirm that it performs its intended function without errors. For example, tokenization and stopword removal are checked for correct text cleaning. Unit tests are applied to ensure that edge cases such as empty inputs, unusual characters, or missing values are handled correctly by each module. Dependencies between modules are also verified so that changes in one component do not introduce unexpected failures in another. This process ensures that the entire system functions seamlessly, producing accurate and reliable sentiment results from end to end.

6.1.2 Integration Testing

Integration testing is conducted after unit testing to ensure that all the individual modules of the system work together as a single, unified application. In the Sentiment Analysis of Political Tweets project, this phase verifies the proper interaction between modules such as data collection, preprocessing, feature extraction, classification, and visualization. The main objective is to confirm that the output of one module correctly serves as the input for the next, maintaining smooth data flow and system consistency.

During integration testing, various test cases are executed to check if tweets collected from the data module are accurately processed in the preprocessing stage, features are correctly passed to the classifier, and the predicted sentiments are properly visualized in the output module. Any mismatches, data loss, or communication errors between modules are identified and corrected. This process ensures that the entire system functions seamlessly, producing accurate and reliable sentiment results from end to end.

6.1.3 Functional Testing:

Functional testing is performed to verify that the Sentimental Analysis of Political Tweets system meets all functional requirements and performs each task as expected. It ensures that every feature of the system from data input to sentiment visualization works correctly according to the project objectives.

In this testing phase, the system is checked to confirm that tweets are properly collected from the dataset or Twitter API, cleaned through preprocessing steps, and converted into numerical features accurately. The classification module is tested to ensure that it correctly identifies the sentiment of each tweet as positive, negative, or neutral. Additionally, the visualization module is tested to verify that sentiment results are displayed clearly through charts or graphs.

Functional testing also validates user interactions, such as entering input data, triggering model analysis, and viewing results, ensuring that the system responds accurately to all user actions. Overall, it confirms that the application performs its intended operations efficiently and delivers accurate, meaningful sentiment insights on political tweets. The system's scalability is also examined, verifying that it can support additional features or larger datasets in future expansions. These combined testing efforts ensure a robust, reliable, and user-friendly sentiment analysis application for political tweet evaluation.

The functional testing clearly refers us how the system function step by step and give the results.

6.1.4 Performance Testing:

Performance testing is conducted to evaluate how efficiently the Sentiment Analysis of Political Tweets system performs under various workloads and data conditions. The main goal of this testing is to ensure that the system maintains speed, stability, and accuracy even when processing large volumes of political tweets. It measures key performance factors such as response time, processing speed, memory usage, and throughput.

During this phase, the system is tested by inputting large tweet datasets to check how quickly data is pre-processed, features are extracted, and sentiments are classified. The performance of the classification model (such as BERT, SVM, or LSTM) is analyzed to ensure it produces results within an acceptable time frame without compromising accuracy. Stress testing is also carried out to determine how the system behaves under heavy data loads or network delays.

Results from performance testing help identify bottlenecks and optimize components for faster execution. Overall, this testing ensures that the sentiment analysis system is robust, scalable, and capable of delivering quick and reliable sentiment insights even when analyzing thousands of political tweets simultaneously.

6.1.5 User Interface Testing:

Interface testing is performed to ensure that all modules in the Sentiment Analysis of Political Tweets system communicate and interact correctly with each other and with the user interface. The main goal of this testing is to verify that data flows smoothly between different components. such as data collection, preprocessing, feature extraction, classification, and visualization without any errors or data loss.

During this phase, the connections between the backend modules and the front-end interface are thoroughly tested. It checks whether user inputs (like uploading datasets or entering tweet text) are properly received and processed by the system, and whether the corresponding output (classified sentiments or visual graphs) is displayed correctly. The consistency of data exchange through APIs or database connections is also verified to ensure accuracy and reliability.

Additionally, interface testing validates the usability and responsiveness of the graphical interface, ensuring that buttons, menus, and visualization elements function as intended. Overall, this testing guarantees that the system provides a smooth, error-free.

6.1.6 Validation and Accuracy Testing:

Validation and accuracy testing play a crucial role in ensuring that the Sentiment Analysis of Political Tweets system produces reliable and meaningful results. The main objective of this testing is to verify that the model's predictions closely match the actual sentiments of political tweets and to evaluate its overall performance using various accuracy metrics.

In this phase, a validation dataset containing pre-labeled political tweets is used to test the trained model. The predicted sentiment labels (positive, negative, or neutral) are compared with the actual labels to calculate performance measures such as accuracy, precision, recall, and F1-score. A confusion matrix is also generated to analyze correct and incorrect predictions for each sentiment class.

This testing helps identify whether the model is overfitting or underfitting the data and ensures that it performs well on unseen tweets. The process validates the robustness, consistency, and correctness of the sentiment analysis system. Overall, validation and accuracy testing confirm that the developed model provides dependable, data-driven insights into political opinions expressed on social media platforms.

6.1.7 Unit Test

Unit testing focuses on verifying the functionality of individual modules or components of the system before integration. Each sub-component is tested independently to ensure it performs its specific task correctly. The main sub-components involved in unit testing for this project are as follows:

Data Collection Module:

Tests whether tweets are successfully fetched from the Twitter API or dataset, ensuring that only relevant political tweets are collected without duplication or missing data.

Data Preprocessing Module:

Validates cleaning operations such as removal of URLs, mentions, emojis, and special characters, as well as tokenization, stop word removal, and lemmatization accuracy.

User Interface Module:

Checks that input fields, buttons, and output displays in the user interface work correctly and interact seamlessly with the backend system.

Feature Extraction Module:

Ensures that textual data is correctly converted into numerical form using methods like TF-IDF or BERT embeddings and checks that feature dimensions are consistent.

Classification Module:

Tests the training and prediction functions of the sentiment classification model to verify correct labeling of tweets into *positive*, *negative*, or *neutral* classes

Performance Evaluation Module:

Confirms that evaluation metrics such as accuracy, precision, recall, and F1-score are computed correctly for model assessment.

Result and Visualization Module:

Tests whether the output results are properly visualized using bar charts, pie charts, or graphs and that visual data matches the classification output.

6.2 Summary

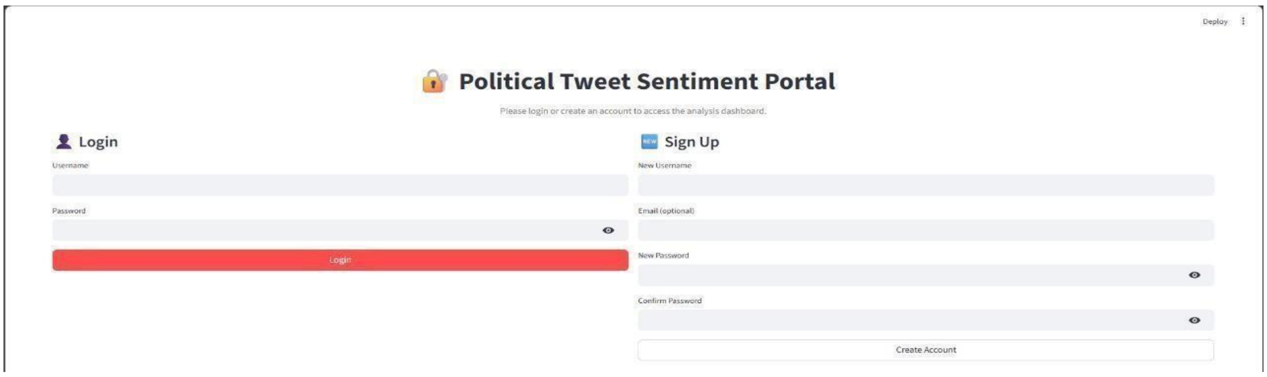
The sixth chapter describes the system testing part of the proposed work. Testing part is discussed in section 6.1. Section 6.1.7 describes the unit testing for each and every individual modules such as test case for different sensors used in the model and its working.System testing serves as a rigorous audit of the software's overall quality and reliability before it reaches the end user. It encompasses a wide variety of testing types, including performance testing to check for speed and stability, security testing to safeguard data, and regression testing to ensure that recent fixes haven't introduced new errors. By validating the product against the original System Requirement Specifications (SRS), this process provides stakeholders with the necessary confidence that the application is "feature-complete" and ready for the final stage of User Acceptance Testing (UAT). Ultimately, it acts as the primary defense against system-wide failures.

CHAPTER 7

RESULTS AND DISCUSSIONS

7.1 Experimental Results

The snapshot gives an overall idea of working of the system. These are the shots taken when the system is actively executing, that is when the system is executed step by step according to the project working pattern.



Deploy

Political Tweet Sentiment Portal

Please login or create an account to access the analysis dashboard.

Login

Username

Password

Login

Sign Up

New Username

Email (optional)

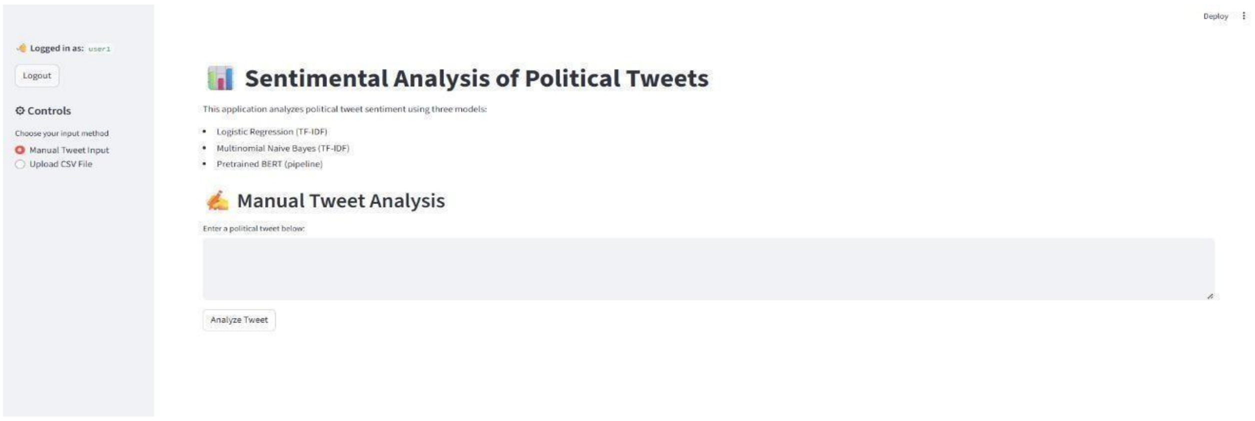
New Password

Confirm Password

Create Account

Snapshot 7.1: User authentication page via login and Signup

Snapshot 7.1 shows the Login and Sign-Up Page serves as the entry point to the Political Tweet Sentiment Portal. The interface is designed to be clean, minimalistic, and user-friendly, allowing new users to register easily while providing existing users with a quick login option.



Logged in as: user1

Logout

Controls

Choose your input method

[✓ Manual Tweet Input]
Upload CSV File

Deploy

Sentimental Analysis of Political Tweets

This application analyzes political tweet sentiment using three models:

- Logistic Regression (TF-IDF)
- Multinomial Naive Bayes (TF-IDF)
- Pretrained BERT (pipeline)

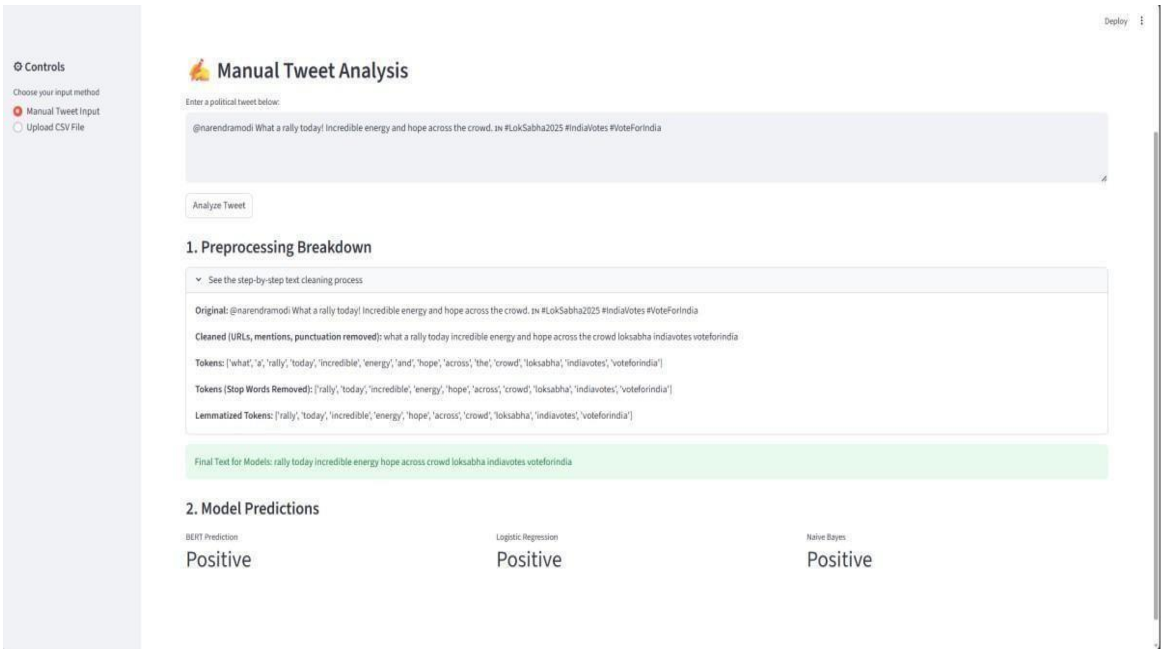
Manual Tweet Analysis

Enter a political tweet below:

Analyze Tweet

Snapshot 7.2 Home page of the System

Snapshot 7.2 shows the home page consists of two controls buttons. One is for Manual tweet input and other is to Upload CSV file. The different buttons are used for two different purposes. And the GUI consist of the details of the controls and the title of the project.



Manual Tweet Analysis

Enter a political tweet below:

@narendramodi What a rally today! Incredible energy and hope across the crowd. IN #LokSabha2025 #IndiaVotes #VoteForIndia

1. Preprocessing Breakdown

See the step-by-step text cleaning process

Original: @narendramodi What a rally today! Incredible energy and hope across the crowd. IN #LokSabha2025 #IndiaVotes #VoteForIndia

Cleaned (URLs, mentions, punctuation removed): what a rally today incredible energy and hope across the crowd loksabha indiavotes voteforindia

Tokens: ['what', 'a', 'rally', 'today', 'incredible', 'energy', 'and', 'hope', 'across', 'the', 'crowd', 'loksabha', 'indiavotes', 'voteforindia']

Tokens (Stop Words Removed): ['rally', 'today', 'incredible', 'energy', 'hope', 'across', 'crowd', 'loksabha', 'indiavotes', 'voteforindia']

Lemmatized Tokens: ['rally', 'today', 'incredible', 'energy', 'hope', 'across', 'crowd', 'loksabha', 'indiavotes', 'voteforindia']

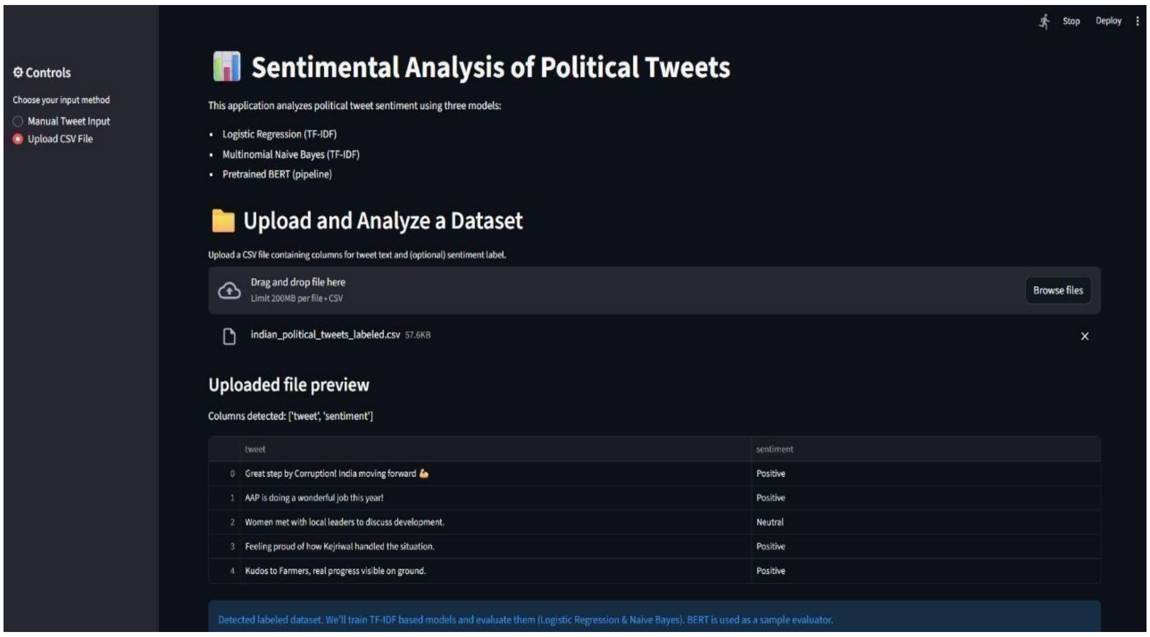
Final Text for Models: rally today incredible energy hope across crowd loksabha indiavotes voteforindia

2. Model Predictions

BERT Prediction	Logistic Regression	Naive Bayes
Positive	Positive	Positive

Snapshot 7.3: Manual input of tweet

Snapshot 7.3 shows the manual text of the tweet. When a user enters a manual input in the box then click on the analyze tweet button, it continues with preprocessing the tweet into different steps as shown in the snapshot. Later the cleaned text will be used for the sentiment classification by the model and shows the result.



Sentimental Analysis of Political Tweets

This application analyzes political tweet sentiment using three models:

- Logistic Regression (TF-IDF)
- Multinomial Naive Bayes (TF-IDF)
- Pretrained BERT (pipeline)

Upload and Analyze a Dataset

Upload a CSV file containing columns for tweet text and (optional) sentiment label.

Uploaded file preview

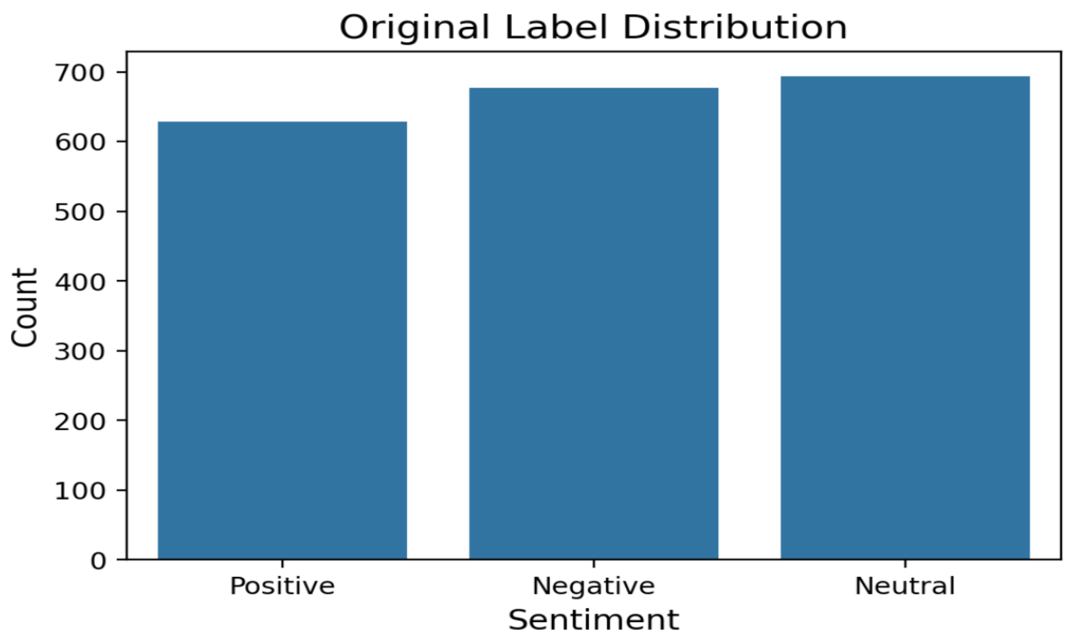
Columns detected: ['tweet', 'sentiment']

	tweet	sentiment
0	Great step by Corruption! India moving forward 💪	Positive
1	AAP is doing a wonderful job this year!	Positive
2	Women met with local leaders to discuss development.	Neutral
3	Feeling proud of how Kejriwal handled the situation.	Positive
4	Kudos to Farmers, real progress visible on ground.	Positive

Detected labeled dataset. We'll train TF-IDF based models and evaluate them (Logistic Regression & Naive Bayes). BERT is used as a sample evaluator.

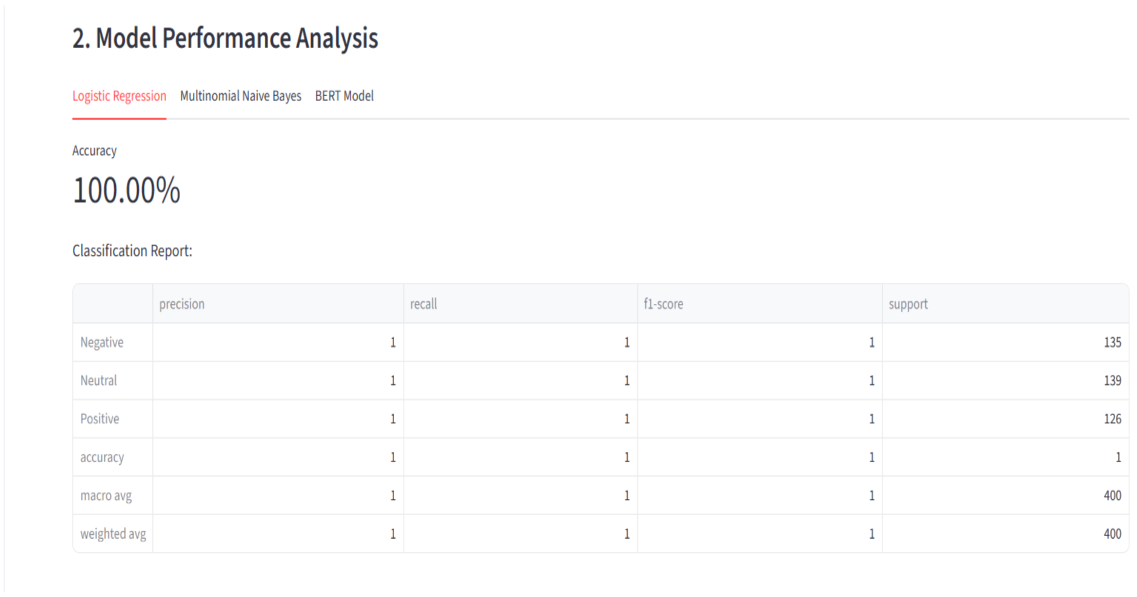
Snapshot 7.4: Upload CSV file of labelled data

Snapshot 7.4 shows the uploading a csv file with labelled data such that it trains th e models and detects if the dataset is labelled or unlabelled.



Snapshot 7.5: The graph of sentiments vs tweets

Snapshot 7.5 shows the original label distribution of how the sentiments are classified into positive, negative and neutral classification and shows the counts as there are same collection of tweets the graph is shown like this.



2. Model Performance Analysis

Accuracy

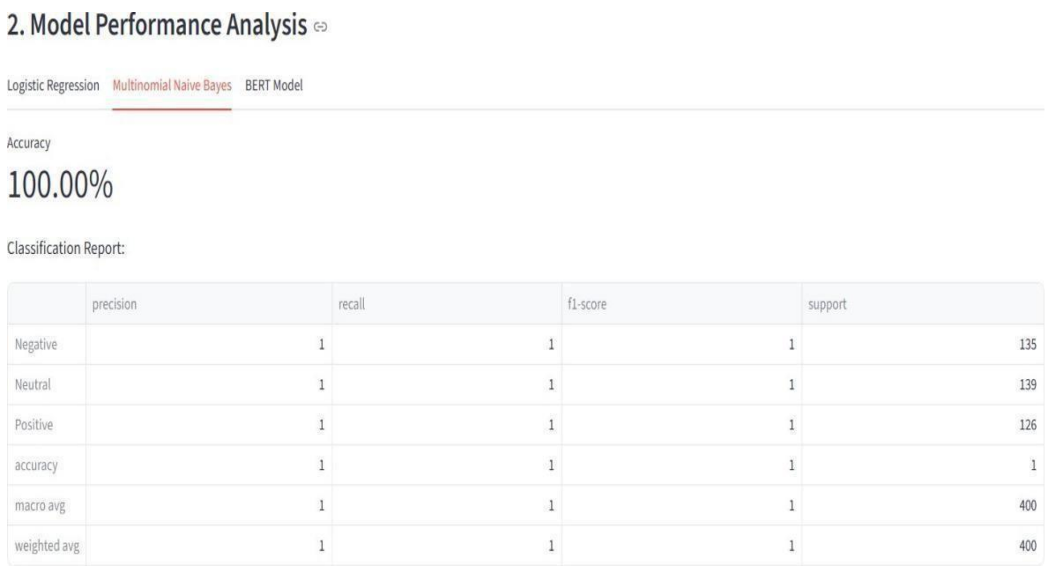
100.00%

Classification Report:

	precision	recall	f1-score	support
Negative	1	1	1	135
Neutral	1	1	1	139
Positive	1	1	1	126
accuracy	1	1	1	1
macro avg	1	1	1	400
weighted avg	1	1	1	400

Snapshot 7.6: Model performance of LR

Snapshot 7.6 shows the model performance analysis and the classification report of Logistic Regression model with the factors, precision, recall, f1_score and support.



2. Model Performance Analysis

Logistic Regression | Multinomial Naive Bayes | BERT Model

Accuracy

100.00%

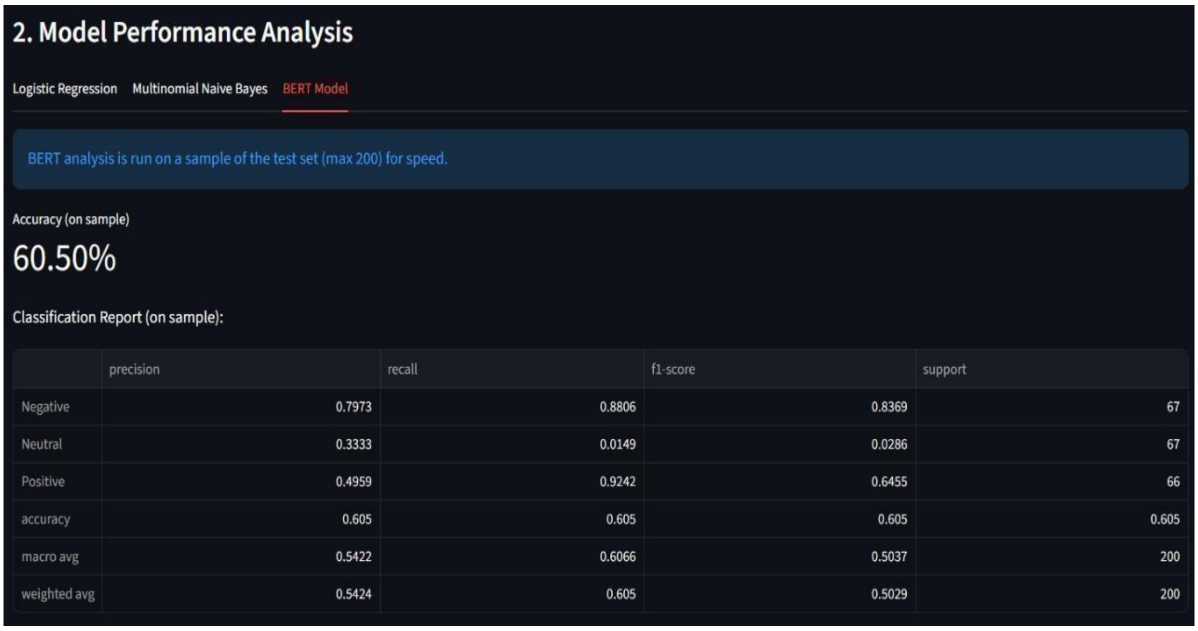
Classification Report:

	precision	recall	f1-score	support
Negative	1	1	1	135
Neutral	1	1	1	139
Positive	1	1	1	126
accuracy	1	1	1	1
macro avg	1	1	1	400
weighted avg	1	1	1	400

Snapshot 7.7: Model performance analysis of MNB

Snapshot 7.7 shows the model performance analysis and the classification report of Multinomial Naïve Bayes model with the factors, precision, recall, f1_score and support.

.



2. Model Performance Analysis

Logistic Regression | Multinomial Naive Bayes | BERT Model

BERT analysis is run on a sample of the test set (max 200) for speed.

Accuracy (on sample)

60.50%

Classification Report (on sample):

	precision	recall	f1-score	support
Negative	0.7973	0.8806	0.8369	67
Neutral	0.3333	0.0149	0.0286	67
Positive	0.4959	0.9242	0.6455	66
accuracy	0.605	0.605	0.605	0.605
macro avg	0.5422	0.6066	0.5037	200
weighted avg	0.5424	0.605	0.5029	200

Snapshot 7.8: Model performance analysis of BERT

Snapshot 7.8 shows the model performance analysis and the classification report of BERT model with the factors, precision, recall, f1_score and support.

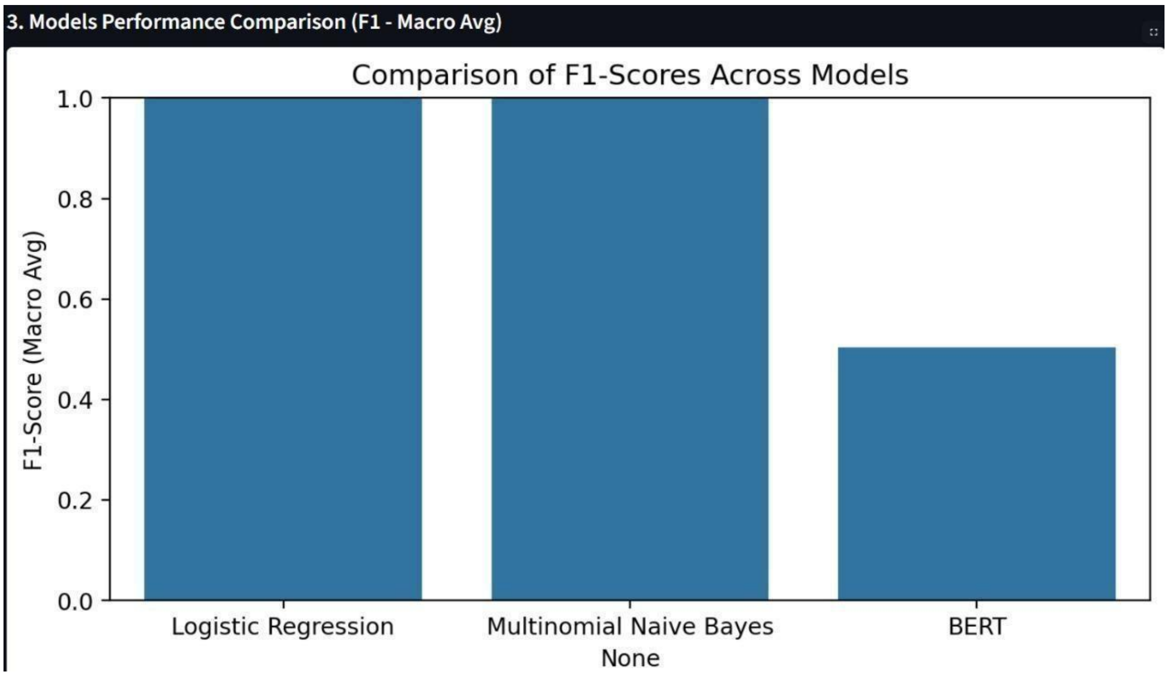


Fig 7.9: Comparision of F1-Scores across all models

Snapshot 7.9 shows the comparision of the F1-Scores with all the three models and with the macro avg scores of the data.

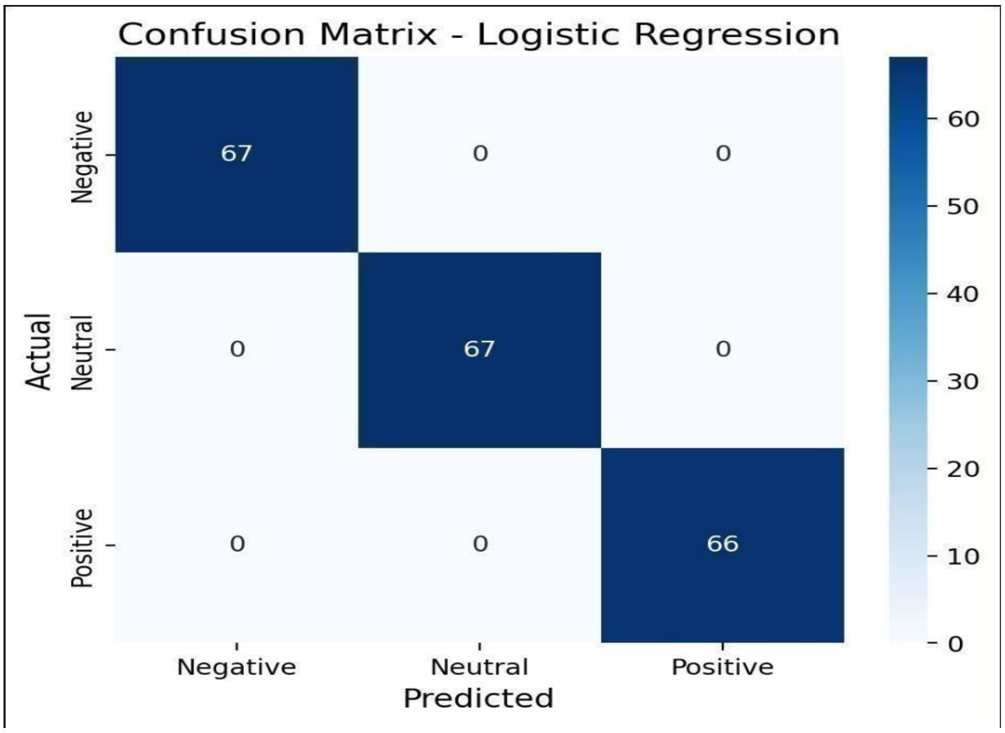
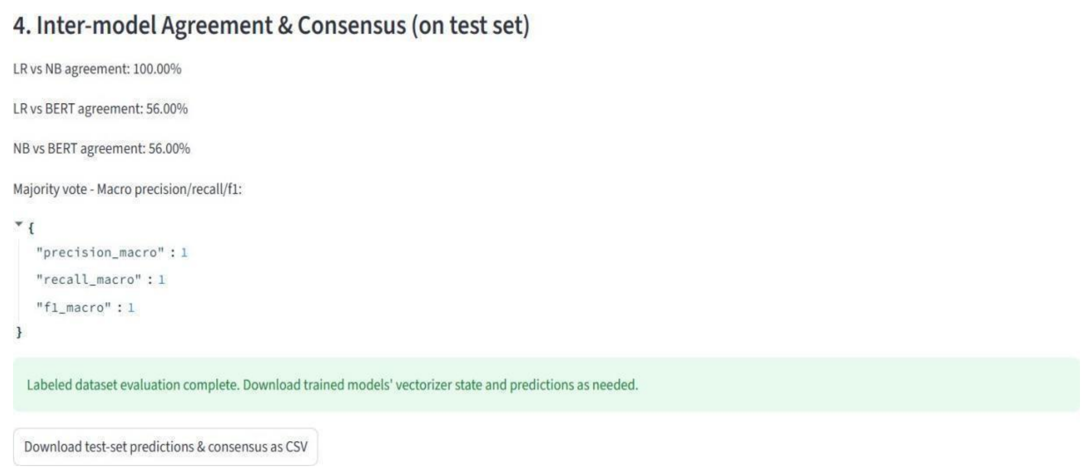


Fig 7.10: Confusion matrix

Snapshot of 7.10 shows the confusion matrix for the following dataset uploaded.

+



4. Inter-model Agreement & Consensus (on test set)

LR vs NB agreement: 100.00%

LR vs BERT agreement: 56.00%

NB vs BERT agreement: 56.00%

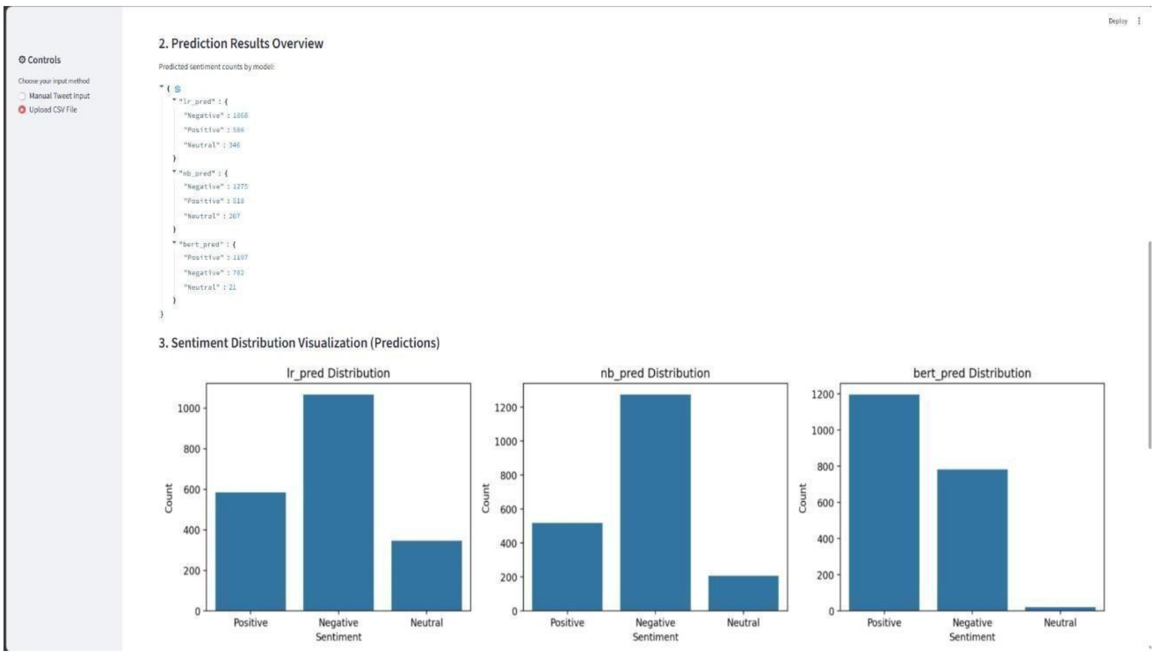
Majority vote - Macro precision/recall/f1:

```
{
  "precision_macro" : 1
  "recall_macro" : 1
  "f1_macro" : 1
}
```

Labeled dataset evaluation complete. Download trained models' vectorizer state and predictions as needed.

Snapshot 7.11: Inter-model agreement

Snapshot 7.11 shows the inter-model agreement classification and consensus between the three models with the percentage an it shows the majority vote prediction of all the sentiments and we can download the predictions in the CSV format.



2. Prediction Results Overview

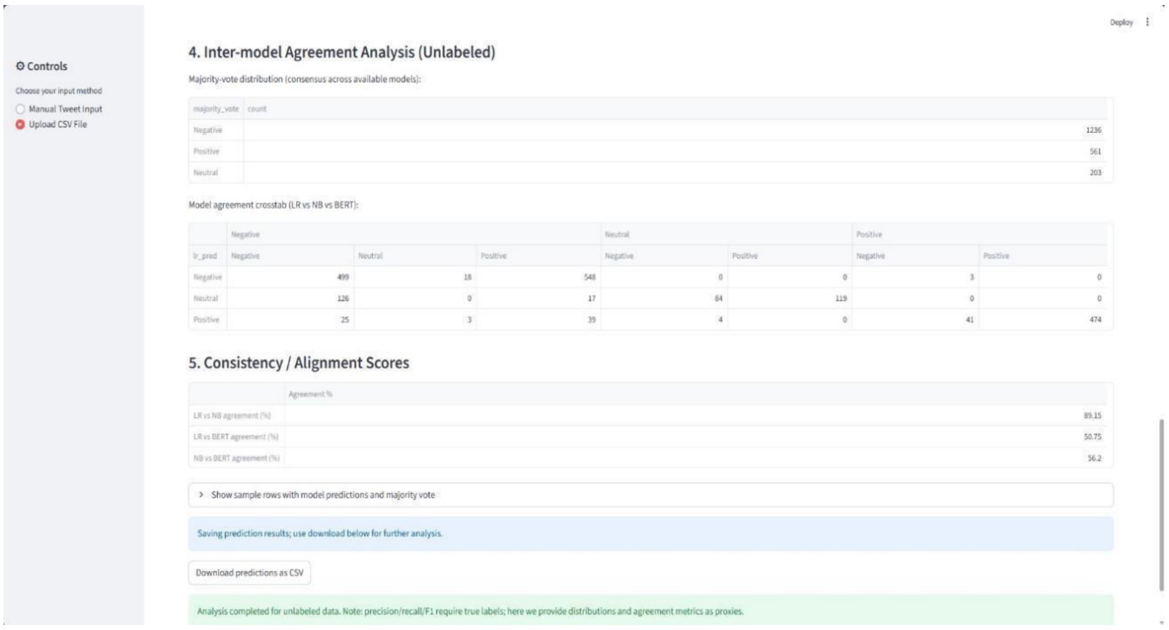
Predicted sentiment counts by model:

```
{
  "lr_pred" : {
    "Negative" : 1068
    "Positive" : 586
    "Neutral" : 346
  }
  "nb_pred" : {
    "Negative" : 1275
    "Positive" : 518
    "Neutral" : 207
  }
  "bert_pred" : {
    "Positive" : 1197
    "Negative" : 782
    "Neutral" : 21
  }
}
```

3. Sentiment Distribution Visualization (Predictions)

Snapshot 7.12: Count and visualization of the unlabelled data prediction.

Snapshot 7.12 shows the count and visualization of the un labelled data in which the three models are trained and predicts the sentiments and displays the graph in bar graph.



4. Inter-model Agreement Analysis (Unlabeled)

Majority-vote distribution (consensus across available models):

majority_vote	count
Negative	1236
Positive	561
Neutral	203

Model agreement crosstab (LR vs NB vs BERT):

	Negative			Neutral		Positive	
lr_pred	Negative	Neutral	Positive	Negative	Positive	Negative	Positive
Negative	499	18	548	0	0	3	0
Neutral	126	0	17	84	119	0	0
Positive	25	3	39	4	0	41	474

5. Consistency / Alignment Scores

	Agreement %
LR vs NB agreement (%)	89.15
LR vs BERT agreement (%)	50.75
NB vs BERT agreement (%)	56.2

› Show sample rows with model predictions and majority vote

Saving prediction results; use download below for further analysis.

Download predictions as CSV

Analysis completed for unlabeled data. Note: precision/recall/F1 require true labels; here we provide distributions and agreement metrics as proxies.

Snapshot 7.13: Inter-model agreement and Consistency scores

Snapshot 7.13 shows the inter-model agreement analysis of un labelled data and the consistency scores of the data which is being predicted for the un-labelled data.

7.2 Summary

This chapter describes the snapshots of the results that are obtained in the proposed work that gives an overall idea of working of the system.

CHAPTER 8

CONCLUSION AND FUTURE ENHANCEMENTS

8.1 Conclusion

The project “Sentiment Analysis of Political Tweets” successfully demonstrates how Natural Language Processing (NLP) and Machine Learning can be applied to understand public opinion on political matters through social media platforms such as Twitter. By collecting, preprocessing, and analyzing political tweets, the system identifies sentimentsPositive, Negative, or Neutral using models like Logistic Regression, Naive Bayes, and BERT.

Through comparative evaluation, it was observed that traditional machine learning models such as Logistic Regression and Naive Bayes perform reasonably well when trained on well-balanced and preprocessed datasets. However, transformer-based models like BERT provided more context-aware predictions, showing higher accuracy and robustness in understanding the nuanced tone of political discussions.

The results indicate that sentiment analysis can serve as a valuable tool for gauging public emotions, political trends, and voter attitudes in real time. This can help policymakers, analysts, and researchers gain insights into citizens’ perspectives and responses toward political events or leaders.

In conclusion, the integration of NLP with modern AI techniques provides a scalable and effective solution for analyzing political sentiments on social media. Future enhancements may include expanding the dataset, integrating regional language processing, and using deep learning models fine-tuned specifically on Indian political discourse to achieve even more accurate and insightful results.

8.2 Future Enhancements

In the future, the Sentiment Analysis of Political Tweets project can be enhanced in several ways to improve its accuracy, scope, and applicability Additionally, the system can be integrated with the Twitter API to enable real-time tweet collection and analysis, allowing continuous monitoring of public sentiment during election campaigns or major political events. Another important improvement would be to expand sentiment categories beyond positive, negative, and neutral, enabling the detection of complex emotions such as anger, joy, fear, or sarcasm using advanced deep learning models. Incorporating topic modeling and trend analysis would also help identify popular political discussions and emerging issues over time. Moreover, the project could benefit from a more interactive visualization dashboard built using tools like Streamlit, Plotly, or Tableau to present insights in an engaging and dynamic way. Future work can also focus on refining data preprocessing techniques to handle slang, emojis, code-mixed language, and political hashtags that often appear in tweets. Ultimately, these enhancements would make the system more powerful, real-time, and capable of predicting future political trends based on sentiment shifts within social media data.

REFERENCES

[1] A. Go, R. Bhayani, and L. Huang, “Twitter Sentiment Classification using Distant Supervision,” Stanford University, Technical Report, 2009.

[2] B. Liu, Sentiment Analysis and Opinion Mining, Synthesis Lectures on Human Language Technologies, Morgan & Claypool Publishers, 2012.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” in *Proc. NAACL-HLT*, 2019, pp. 4171–4186.

[4] S. S. Suresh and A. S. Reddy, “Sentiment Analysis on Indian Political Tweets Using Machine Learning Approaches,” International Journal of Advanced Computer Science and Applications (IJACSA), vol. 11, no. 4, pp. 322–329, 2020.

[5] A. Pak and P. Paroubek, “Twitter as a Corpus for Sentiment Analysis and Opinion Mining,” in *Proc. LREC*, Valletta, Malta, 2010.

[6] C. J. Hutto and E. Gilbert, “VADER: A Parsimonious Rule-Based Model for Sentiment Analysis of Social Media Text,” in *Proc. ICWSM*, 2014.

[7] S. M. Mohammad and P. D. Turney, “Crowdsourcing a Word–Emotion Association Lexicon” (NRC Emotion Lexicon), National Research Council Canada / EMNLP-related publications, 2013.

[8] X. Zhang, J. Zhao, and Y. LeCun, “Character-level Convolutional Networks for Text Classification,” in *Proc. NeurIPS*, 2015.

[9] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, L. Zettlemoyer, and V. Stoyanov, “RoBERTa: A Robustly Optimized BERT Pretraining Approach,” arXiv preprint, 2019.

[10] OpenAI, ChatGPT (GPT-5.1 Model), Large Language Model, 2025

[11] Google, Gemini Advanced, Large Language Model, Google DeepMind, 2024

[12] Microsoft, Copilot AI Assistant, Microsoft Corporation, 2024.

Github Links

Chinmay CS - https://github.com/Chinmaychandrashekar
K Shivadarshan - https://github.com/shivdarshh
Sunil Kumar AB - https://github.com/Sunilkumarab
Yashas Xavier - https://github.com/yashasxavier