A
PROJECT REPORT
ON

Twitter Sentiment Analysis

Submitted in partial fulfilment for the award of the degree in

BACHELOR OF TECHNOLOGY

IN

Computer Science & Engineering

(Maulana Abul Kalam Azad University of Technology)

Submitted by:

MD LABIB ALAM (University Roll No**: 33200121090**)

Other Members:

MD SHARIQUE JAWAID KHAN (University Roll No**: 33200121078**)

SYED AMMAR ALI (University Roll No**: 33200121085**)

SNIGDHA HALDER (University Roll No**: 33200121010**)

MD FIROZE KHAN (University Roll No**: 33200121087**)

TIYASA DAS (University Roll No**: 33200121015**)

Under the Guidance of:

Prof. Subhankar Guha

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

TECHNO INTERNATIONAL BATANAGAR
Maheshtala, Kolkata-700141, West Bengal, India (2025)

Table Of Contents **Page No.**

Overview of Project	**3**
Literature Review	**4**
System Block Diagram	**6**
Project Methodology	**7**
Technical Platform	**13**
Expected Result	**15**
Scope of Further Improvements	**17**
References	**19**

1. Overview of the Project

1.1 Introduction

The project focuses on implementing a sentiment analysis system capable of classifying text data, such as tweets, into sentiment categories like positive, negative, or neutral. By combining natural language processing (NLP) techniques with machine learning models, this project aims to process, clean, and analyze textual data to extract valuable insights regarding public opinions and sentiments.

1.2 Problem Statement

In today's digital age, social media platforms generate an enormous amount of text data every second. However, extracting meaningful insights from this unstructured data poses a significant challenge. Understanding the sentiment behind this data is crucial for applications like brand reputation monitoring, customer feedback analysis, and product review evaluation. The difficulty lies in handling issues such as noisy data, varying linguistic patterns, and context-specific meanings. This project addresses these challenges by designing a pipeline for effective sentiment classification.

1.3 Significance

Sentiment analysis has far-reaching implications in various domains. Businesses can leverage this technology to gauge customer opinions, improve products, and design effective marketing strategies. Governments and organizations can use sentiment analysis to track public sentiment on policies, events, or social movements. By analyzing public emotions in real time, decision-makers can respond promptly to emerging trends or concerns. Furthermore, this project highlights the potential of machine learning in solving real-world problems, emphasizing its practicality and relevance in today’s data-driven landscape.

1.4 Goal

The primary goal of this project is to develop a sentiment analysis system that accurately classifies textual data into predefined sentiment categories. The system should demonstrate high performance in terms of accuracy, precision, and recall. Ultimately, the project aims to provide a robust, scalable, and interpretable solution for

sentiment analysis, with potential applications in various industries, including marketing, customer service, and social research.

2. Literature Review

2.1 Existing Research

Sentiment analysis has been extensively studied and implemented using various techniques over the years. Traditional approaches like lexicon-based methods rely on predefined dictionaries of words annotated with sentiment values. While simple and interpretable, these methods often fail to capture the nuances of context and complex sentence structures. With the advent of machine learning, algorithms such as Support Vector Machines (SVM), Naive Bayes, and Random Forest have gained popularity due to their ability to learn patterns from labeled data. More recently, deep learning techniques, including Long Short-Term Memory (LSTM) networks and transformers like BERT, have revolutionized sentiment analysis by leveraging contextual word embeddings and sequence modeling.

2.2 Tools and Technologies

Sentiment analysis has greatly benefited from the development of various tools and technologies. These tools and frameworks facilitate the application of advanced techniques to process and analyze text data effectively. Below are some of the key tools and technologies used in sentiment analysis:

- **Text Representation Techniques**: The TF-IDF (Term Frequency-Inverse Document Frequency) is a statistical measure used to evaluate the importance of a word in a document relative to a collection of documents (corpus). It combines two components that is Term Frequency (TF) and Inverse Document Frequency (IDF) and results in TF-IDF which is the product of Term Frequency and Inverse Document Frequency

$$TF(t,d) = \frac{number\ of\ times\ t\ appears\ in\ d}{total\ number\ of\ terms\ in\ d}$$

$$IDF(t) = log\frac{N}{1+df}$$

$$TF-IDF(t,d) = TF(t,d) * IDF(t)$$

- **Machine Learning Models**: Random Forest is a widely used ensemble learning model that combines multiple decision trees to improve classification performance. By aggregating the predictions from various trees, Random Forest provides a robust and accurate sentiment classification, making it well-suited for sentiment analysis tasks.

2.3 Gaps Identified

Despite significant progress, several limitations persist in existing methods:

- **Lexicon-based methods** struggle with ambiguity and context-specific meanings, such as sarcasm or idiomatic expressions.
- **Traditional machine learning models** require extensive feature engineering and may fail to generalize well to unseen data.
- **Deep learning models**, while powerful, often require large datasets, significant computational resources, and are sometimes challenging to interpret. Handling noisy and unstructured data, such as social media text, remains a critical challenge.

2.4 Relevance

The reviewed studies provide a foundation for the current project by highlighting the strengths and weaknesses of existing methods. For instance, TF-IDF is used to represent text features effectively, while Random Forest is chosen for its robustness and interpretability. This combination aligns with the objective of developing a reliable sentiment analysis system for text data, particularly social media content. By addressing identified gaps and building on established techniques, this project contributes to advancing the field of sentiment analysis and its practical applications.

3. System Block Diagram

The diagram illustrates the structured flow of the sentiment analysis pipeline. Starting with data import using Pandas, the process includes exploration to handle missing values and derive summary statistics. Feature engineering, including text length analysis and visualizations, is followed by cleaning the data using TweetCleaner. The cleaned dataset is split into training and testing sets, while feature transformation using TF-IDF vectorization prepares the textual data for model training. A classifier, such as Random Forest, is trained and evaluated using metrics like the confusion matrix. Additionally, the system supports predictions on custom inputs, completing the sentiment analysis process efficiently.

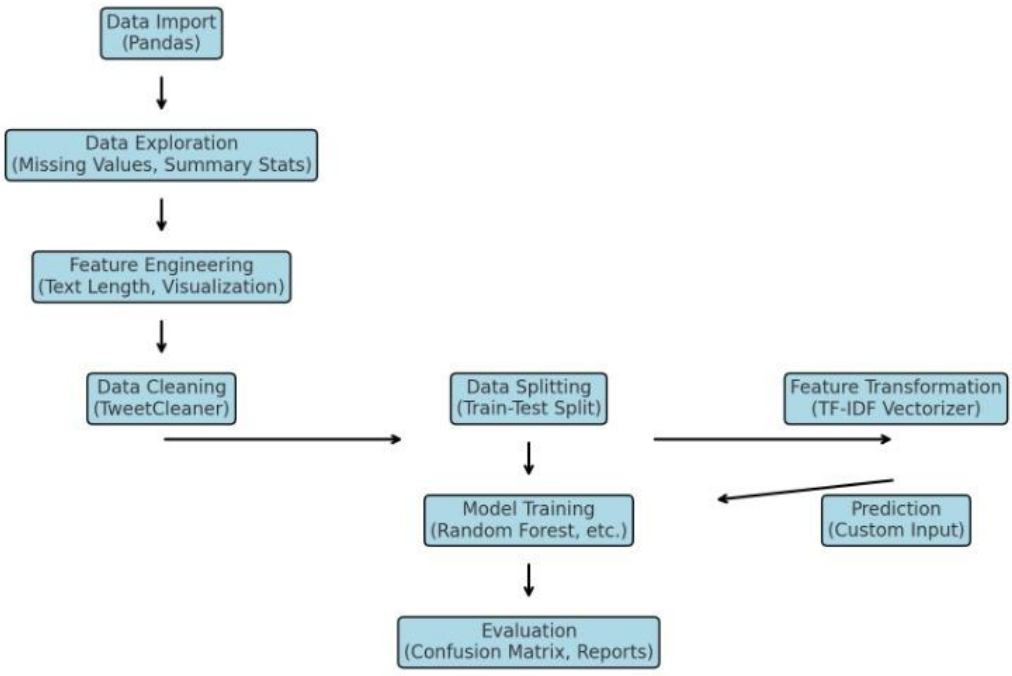

Fig : System Block Diagram

4. Project Methodology

4.1 Data Collection

The project utilizes a labeled dataset of tweets, collected from publicly available resources. Each entry consists of text and a corresponding sentiment label (e.g., positive, neutral, or negative). The dataset ensures a balanced distribution of sentiment classes to train the model effectively.

4.2 Exploratory Data Analysis (EDA)

In the exploratory data analysis (EDA) phase, a thorough examination of the dataset was conducted to gain insights and prepare it for effective modeling. We started by analyzing the dataset's structure, focusing on key attributes such as ‘text’ and ‘sentiment’. The dataset was evaluated for missing values, duplicates, and irrelevant data, followed by cleaning to ensure its quality and reliability. A critical aspect of the analysis was understanding the distribution of sentiments across the dataset. Upon visualization, we observed that the dataset was balanced, with no significant skew toward any particular sentiment category (positive, neutral, or negative). This indicated that there was no need for countermeasures like oversampling or undersampling to address class imbalance, allowing us to move forward with the model as is.

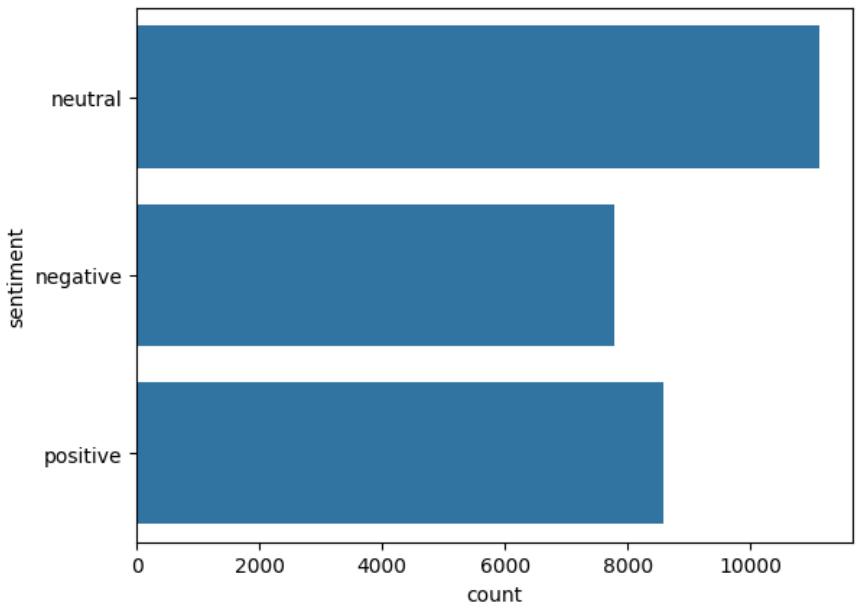

Fig : Sentiment Distribution

Another essential analysis involved exploring the length of tweets, where it was found that the maximum tweet length in the dataset was 33 words. Most tweets were concise, and their lengths fell within a consistent range, making them manageable for text processing tasks. This analysis of tweet lengths provided a good understanding of the text complexity in the dataset. To provide a clearer picture of these findings, visualizations were created, including a diagram depicting the tweet length distribution and a figure showcasing the sentiment distribution. These insights offered a solid foundation for feature engineering and model development, ensuring the dataset was ready for the next phases of the project.

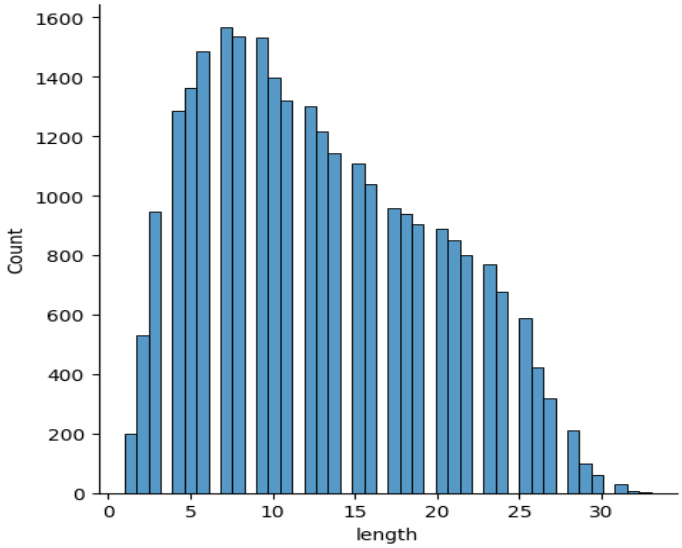

Fig : Tweet Length Distribution

4.3 Data Preprocessing

The text preprocessing pipeline is implemented using the TweetCleaner class, which applies various cleaning techniques to the raw tweets. The following methods are part of the preprocessing:

- **Removing URLs**: URLs often appear in tweets, but they do not contribute meaningfully to sentiment analysis. The remove_urls method uses a regular expression to remove URLs starting with "http" or "www". This ensures that any external links in the tweet text are excluded.
- **Removing Mentions (@user)**: Mentions of users (e.g., @user) are irrelevant to sentiment analysis, so the remove_mentions method removes these handles

from the text. This is achieved by identifying patterns that start with @ followed by a word character.

- **Cleaning Hashtags**: While hashtags provide context and sentiment, the "#" symbol itself does not add useful information for analysis. The clean_hashtags method removes the "#" symbol while keeping the associated hashtag text. This allows the analysis to focus on the key terms in the hashtag.
- **Removing Punctuation and Numbers**: Punctuation marks and numbers are typically not relevant for sentiment analysis in this context. The remove_punctuation_and_numbers method removes both punctuation and numbers using regular expressions and the str.translate() method, which removes characters that are not alphabetic.
- **Stopword Removal**: Stopwords are common words (e.g., "the", "is", "in") that do not carry significant meaning in sentiment analysis. The remove_stopwords method removes such words using a predefined set of English stopwords from the NLTK library. This method also excludes "negation words" (e.g., "not", "no", "never") from being removed, as these words are crucial for maintaining the context of negation in sentiment.
- **Stemming**: Stemming is the process of reducing words to their root forms (e.g., "running" becomes "run"). The remove_stopwords method also incorporates stemming using the PorterStemmer from NLTK. This helps standardize the words by reducing them to their base forms, improving the generalization ability of the model.
- **Handling Negation in Text**: Negation words such as "not," "no," and "never" can change the sentiment of a sentence. To address this, we implemented a method that adds a **NEG_** prefix to words following a negation word, until the next punctuation or after a certain number of words. This ensures that words like "good" in "not good" are correctly interpreted as negative. Handling negation allows the model to more accurately analyze sentiment in sentences containing negations.

4.4Feature Engineering

To convert textual data into a numerical format, TF-IDF (Term Frequency-Inverse Document Frequency) is employed. This method emphasizes important terms by considering their frequency in a document relative to their occurrence across the entire dataset. It helps highlight sentiment-related terms while reducing the weight of commonly used words.

4.5 Model Development

A Random Forest Classifier, an ensemble learning method, is used for sentiment prediction due to its robustness and ability to handle noisy data. The dataset is split into training and testing subsets, ensuring the model learns patterns from one subset and generalizes effectively to unseen data.

- **Training:** The model learns sentiment-related patterns from the training data.
- **Testing:** Performance is validated on unseen testing data.

4.6 Evaluation

For evaluating the performance of your sentiment analysis model, the **confusion matrix** and **classification report** provides the following metrics:

Confusion matrix: The confusion matrix is a structured table with N rows and N columns, where each cell reports the frequency of actual versus predicted values for classification. It provides insights into four key metrics: true positives (correctly predicted positive cases), true negatives (correctly predicted negative cases), false positives (cases incorrectly classified as positive), and false negatives (cases incorrectly classified as negative). This matrix serves as a fundamental tool for assessing the accuracy and reliability of a classification model**.**

Precision: Precision is the ratio of correctly predicted positive observations to the total predicted positives. It answers the question: *Of all the instances predicted as a certain class (e.g., positive), how many were actually that class?*

$$\text{Precision} = \frac{tp}{tp + fp}$$

Recall (Sensitivity or True Positive Rate): Recall is the ratio of correctly predicted positive observations to the total actual positives. It answers the question: *Of all the actual instances of a class, how many were correctly predicted?*

$$\text{Recall} = \frac{tp}{tp + fn}$$

F1-Score: The F1-score is the harmonic mean of precision and recall. It balances the trade-off between precision and recall. It is useful when you need a single metric to compare models, especially when there is an imbalance between classes.

$$F_1 = \left(\frac{2}{\text{recall}^{-1} + \text{precision}^{-1}} \right) = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Accuracy: Accuracy measures the overall correctness of the model by calculating the ratio of correct predictions (both true positives and true negatives) to the total number of predictions.

$$\text{Accuracy} = \frac{tp + tn}{tp + tn + fp + fn}$$

By analyzing these metrics, you can assess where your model performs well and where it needs improvement, such as improving recall for certain classes or optimizing

precision.

	precision	recall	f1-score	support
negative	0.74	0.59	0.66	1572
neutral	0.66	0.73	0.70	2236
positive	0.75	0.78	0.76	1688
accuracy			0.71	5496
macro avg	0.72	0.70	0.71	5496
weighted avg	0.71	0.71	0.71	5496

4.7 Deployment

The trained model, along with the preprocessing pipeline, is packaged into a deployable system. It can be integrated into applications, such as customer feedback analysis tools, to predict sentiment in real-time. Deployment considerations include scalability, user interface design, and model retraining to accommodate new data trends.

5. Technical Platform

5.1 Programming Language

the Python programming language is the foundation for the implementation. Python's simplicity and powerful libraries make it an ideal choice for data analysis, machine learning, and natural language processing (NLP).

5.2 Libraries/Frameworks

Pandas: We use pandas to load, manipulate, and preprocess the dataset. It helps in tasks like cleaning text, dropping unnecessary columns, handling missing values, and generating basic statistics, such as text length analysis.

NLTK (Natural Language Toolkit): Since sentiment analysis involves understanding the structure and meaning of text, nltk plays a crucial role in this project. We use it for tasks like:

- **Tokenization**: Breaking down text into smaller components (words).
- **Stopword removal**: Filtering out common words (e.g., "the", "and") that do not contribute much to sentiment.
- **Stemming**: Reducing words to their base form (e.g., "running" to "run") to standardize the data and improve model efficiency.

scikit-learn: This library is essential for implementing the machine learning pipeline. It helps with:

- **Feature extraction**: We use the TF-IDF method from scikit-learn to convert text data into numerical features suitable for machine learning models.
- **Model training**: We use Random Forest Classifier from scikit-learn to train a model that predicts sentiment (positive, neutral, negative) based on the preprocessed text data.
- **Evaluation**: We use tools from scikit-learn to assess model performance through accuracy and other metrics such as precision, recall, and F1-score.

Matplotlib: This library is used to create visualizations for:

- **Data analysis**: Plotting the distribution of text lengths or the distribution of sentiments across the dataset.

- **Model evaluation**: Visualizing performance metrics such as confusion matrices or accuracy scores.

5.3 Development Environment

Jupyter Notebook: During the project, Jupyter Notebook was used for interactive coding and experimentation. It allowed for easy documentation, inline visualizations, and quick testing of different parts of the project, such as preprocessing steps or model evaluation.

Google Colab: For more computationally intensive tasks, like model training with large datasets, Google Colab provided cloud-based resources that sped up the process. It also allowed for leveraging free GPU resources for faster processing.

6. Expected Result

6.1 Accuracy

The expected result in terms of accuracy is a model that can effectively predict the sentiment of text with a reasonable degree of precision. Based on the performance metrics like accuracy, precision, recall, and F1-score, the model should show solid performance in classifying text into the three sentiment categories: positive, neutral, and negative. As demonstrated in the evaluation, an accuracy of around 70-75% is expected, indicating that the model is able to correctly predict the sentiment of the majority of the test data. However, accuracy alone is not the sole criterion; the balance of precision and recall across different classes is also crucial. Here is a heat map of the confusion matrix

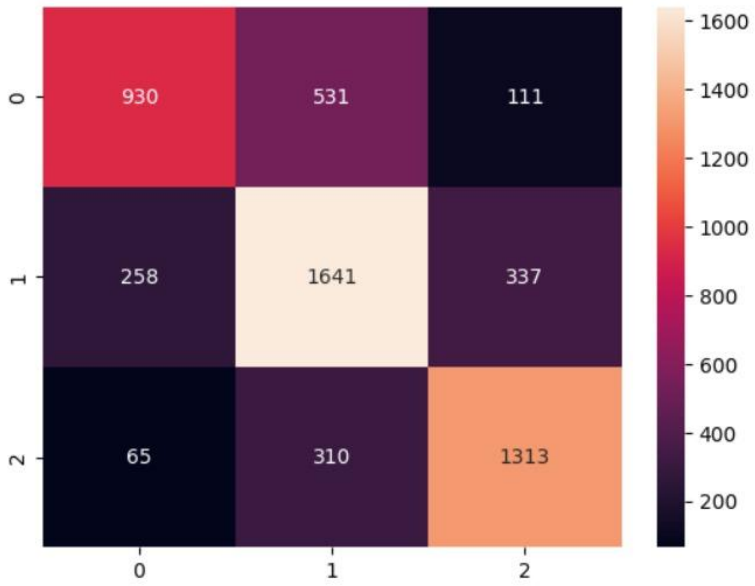

Fig : Confusion Matrix

6.2 Functionality

The functionality of the sentiment analysis model in this project revolves around its ability to:

- **Preprocess text data**: Handle text preprocessing steps, such as cleaning, tokenization, stopword removal, stemming, and negation handling.
- **Classify sentiment**: Correctly classify sentiment into one of three categories (positive, neutral, or negative) based on the cleaned input text.

- **Provide feedback**: Offer meaningful feedback in the form of sentiment labels (e.g., "positive," "negative," "neutral") for new, unseen text.
- **Model evaluation**: Evaluate the performance of the model using metrics like accuracy, precision, recall, and F1-score to ensure it works as expected across different sentiment categories.

6.3 Applications

This sentiment analysis model can be applied in several real-world scenarios where understanding the sentiment of textual data is valuable:

- **Social Media Monitoring**: It can be used to analyze public opinions expressed on platforms like Twitter, Facebook, or Instagram, providing insights into how people feel about certain topics, brands, or products.
- **Customer Feedback Analysis**: Organizations can use this model to process customer feedback from reviews, surveys, or emails, helping them understand customer sentiment and improve their services.
- **Brand Sentiment Analysis**: Companies can monitor the sentiment surrounding their brand by analyzing tweets, online reviews, and news articles to identify potential PR issues or opportunities.
- **Market Research**: The model can be used to gauge public sentiment about a specific event, product launch, or political issue, helping companies or organizations tailor their strategies accordingly.
- **Opinion Mining**: News outlets, research firms, or analysts can use sentiment analysis to monitor political discourse or media coverage, extracting opinions on various subjects.

7. Scope of Further Improvements

7.1 Enhanced Preprocessing

While the current preprocessing pipeline handles key text-cleaning tasks (such as stopword removal, stemming, and negation handling), further enhancements could be made to improve the accuracy and robustness of the model:

- **Lemmatization**: Instead of stemming, which reduces words to their root form, lemmatization could be used to convert words to their base form, considering their context. This can help in retaining the correct meaning of words.
- **Advanced Stopword Handling**: Custom stopword lists could be created based on the specific domain of the text to remove noise that is specific to the dataset (e.g., specific terms in customer reviews, social media posts, etc.).
- **Handling Sarcasm and Irony**: Sarcastic or ironic statements often carry sentiments that are opposite to their literal meaning. Implementing techniques to identify sarcasm and irony could help improve the model's ability to classify sentiments more accurately.
- **Contextual Word Embeddings**: Instead of using traditional bag-of-words methods like TF-IDF, integrating pre-trained word embeddings (e.g., **Word2Vec**, **GloVe**, or **BERT**) could improve text representation and lead to better understanding of semantic meaning.

7.2 Real-Time Analysis

Currently, the model works on a static dataset, but real-time sentiment analysis could be an area for improvement:

- **Live Social Media Monitoring**: The model can be integrated into platforms like Twitter or Reddit to monitor real-time tweets or posts, identifying emerging trends and public opinion shifts.
- **Instant Feedback Systems**: In a customer service or e-commerce context, real-time sentiment analysis can help organizations respond immediately to customer complaints or positive feedback, improving user engagement.

- **Event-Based Sentiment Tracking**: Real-time sentiment analysis could be used to track sentiment changes related to live events, product launches, or news developments, providing instant insights for decision-makers.

7.3 Scalability

The current model works well for a limited dataset, but scalability is essential for large-scale sentiment analysis tasks:

- **Handling Large Datasets**: As the volume of textual data grows (e.g., millions of customer reviews or social media posts), the model needs to be optimized for efficient processing. Techniques such as distributed computing or GPU acceleration can be employed to scale up training and inference.
- **Multi-Language Support**: To scale beyond English, the model could be enhanced to support multiple languages, adapting to different linguistic structures and cultural nuances. This would require collecting datasets in other languages and possibly training separate models or using multi-lingual embeddings.
- **Model Optimization**: To handle real-time analysis with large volumes of data, model optimization techniques such as quantization or pruning could be explored to reduce model size and speed up inference time without sacrificing too much accuracy.

8. References

8.1 Research Papers

Here, you would cite papers that provide foundational theories or approaches relevant to sentiment analysis, text processing, and machine learning models. Example references include:

- **Pang, B., & Lee, L. (2008).** "Opinion Mining and Sentiment Analysis." *Foundations and Trends® in Information Retrieval, 2(1–2), 1-135.*
  This paper provides a comprehensive overview of sentiment analysis, covering methods for extracting sentiments from text, including rule-based and machine learning-based approaches.
- **Garg, P., & Vohra, R. (2018).** "Sentiment Analysis of Twitter Data using Natural Language Processing." *Proceedings of the 2018 International Conference on Data Science and Engineering, 154-158.*
  This paper discusses the application of sentiment analysis to Twitter data, using Natural Language Processing (NLP) methods for sentiment classification. It highlights various challenges and techniques applied in analyzing user opinions.
- **Gupta, V., & Kumar, P. (2017).** "A Survey of Sentiment Analysis Techniques in Social Media." *International Journal of Computer Science and Information Technologies, 8(5), 1023-1027.*
  This paper provides an overview of various sentiment analysis techniques and methods applied to social media text. It explores different algorithms, including machine learning and deep learning models, and compares their effectiveness for sentiment classification tasks.

8.2 Libraries and Tools

This section would list the key libraries and frameworks that were used in the project. Some examples:

- **pandas**: *https://pandas.pydata.org/*
  A powerful library for data manipulation and analysis, used for loading datasets, preprocessing, and basic operations on dataframes.
- **nltk**: *https://www.nltk.org/*

A toolkit for working with human language data, providing functionalities like tokenization, stopword removal, and stemming.

- **scikit-learn**: *https://scikit-learn.org/*
  A versatile machine learning library that supports model training, feature extraction (like TF-IDF), and evaluation metrics.
- **matplotlib**: *https://matplotlib.org/*
  A plotting library used for data visualization, especially for generating plots related to model performance.

8.3 Dataset

The dataset used in this project is crucial for training and evaluating the sentiment analysis model. It's important to credit the source of the dataset and its specific characteristics. For example:

Training Dataset: The project uses a labeled dataset (train.csv) for sentiment classification.