**INSTITUTE’S CERTIFICATE**



**DECLARATION**

I hereby certify that the work which is being presented in the report entitled “Machine Learning” in fulfilment of the requirement for completion of industrial training in Department of Bachelors in Computer Application of “Sri Guru Tegh Bahadur Institute of Management and Information Technology” is an authentic.

**Vinay Bajaj**

**06490202021**

**BCA 5B**

**ACKNOWLEDGEMENT**

I am pleased to acknowledge my sincere thanks to Board of Management of SHRI GURU TEGH BAHADUR INSTITUTE OF MANAGEMENT AND INFORMATION for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I would like to express my sincere gratitude to “Data Science Academia - Sai Chamundeeswari Academy”

for your guidance and support throughout the development of our Loan Eligibility Prediction project. The invaluable insights and mentorship I have received has been instrumental in my success.

**Vinay Bajaj**

**06490202021**

**BCA 5B**

**ABSTRACT**

In a world driven by textual data, our project presents a sophisticated sentiment analysis system designed to categorize text into positive, negative, or neutral sentiments. This system harnesses the power of advanced machine learning models, including Linear Support Vector Classifier (LinearSVC), Logistic Regression, and Naive Bayes, each fine-tuned to maximize the accuracy of sentiment predictions.

At the heart of this project is a user-friendly Streamlit frontend, enabling individuals to input text and receive real-time sentiment analyses. This accessible interface unlocks valuable insights into the emotions and attitudes expressed in text.

Beyond the technical intricacies, our project explores the depths of sentiment analysis, offering users a tool to decode sentiment nuances in an era of data overload. The collaborative spirit is fostered by the GitHub integration, encouraging users to fork the repository and contribute to further model enhancement.

In a data-driven age, our sentiment analysis system provides a clear lens to navigate and understand the sentiments encapsulated within text, empowering users with the ability to decipher the unspoken emotions in words.

Top of Form

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Description** | **Page No.** |
| 8.1 | Logistic Regression Confusion Matrix |  |
| 8.2 | Naïve Bayes Confusion Matrix |  |
| 8.3 | Linear Support Vector Classifier Confusion Matrix |  |
| 8.4 | Model Comparison |  |
| 10.1 | Sample Output 1 |  |
| 10.2 | Sample Output 2 |  |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Description** | **Page No.** |
| 6.1 | Decision Table |  |

**LIST OF ABBREVIATIONS**

*Abbreviation Description*

DFD Data Flow Diagram

CNN Convolutional Neural Network

RNN Recurrent Neural Network  
SVC Support Vector Classifier  
SVM Support Vector Machine  
ERD Entity Relationship Diagram

**CONTENTS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **CHAPTER** | **1:** | **PROBLEM FORMULATION** |  |  | |
| 1.1 |  | Introduction about the Company |  | 1 | |
| 1.2 |  | Introduction about the Problem |  | 4 | |
| 1.3 |  | Present State of the Art |  | 6 | |
| 1.4 |  | Need of Computerization |  | 10 | |
| 1.5 |  | Proposed Software / Project |  | 13 | |
| 1.6 |  | Importance of the work |  | 15 | |
| **CHAPTER** | **2:** | **SYSTEM ANALYSIS** |  | | |
| 2.1 |  | Feasibility Study | 20 | | |
| 2.1.1 |  | Technical Feasibility | 20 | | |
| 2.1.2 |  | Economic Feasibility | 22 | | |
| 2.1.3 |  | Operational Feasibility | 23 | | |
| 2.1.4 |  | Other Feasibility Dimensions | 24 | | |
| 2.2 |  | Analysis Methodology | 25 | | |
| 2.3 |  | Choice of the Platforms |  | | |
| 2.3.1 |  | S/W used | 26 | | |
| 2.3.2 |  | H/W used | 28 | | |
| **CHAPTER** | **3:** | **SYSTEM DESIGN#** |  | |  |
| 3.1 |  | Design Methodology |  | |  |
| 3.2 |  | Database Design |  | | 36 |
| 3.2.1 |  | ERD |  | | 37 |
| 3.2.2 |  | DFD |  | | 39 |
| 3.2.3 |  | Flow Chart |  | |  |
| 3.3 |  | Input Design |  | | 42 |
| 3.4 |  | Output Design |  | | 48 |
| 3.5 |  | Code Design and Development |  | |  |
| **CHAPTER** | **4:** | **TESTING AND IMPLEMENTATION** | |  | |
| 4.1 |  | Testing Methodology | | 52 | |
| 4.1.1 |  | Unit Testing | | 55 | |
| 4.1.2 |  | Module Testing | | 56 | |
| 4.1.3 |  | Integration Testing | | 57 | |
| 4.1.4 |  | System Testing | | 59 | |
| 4.1.5 |  | White Box / Black Box Testing | | 62 | |
| 4.1.6 |  | Acceptance Testing | | 69 | |
| 4.2 |  | Test Data & Test Cases | | 70 | |
| 4.3 |  | Test Reports and Debugging | | 71 | |
| 4.4 |  | Implementation Manual | | 73 | |
| 4.5 |  | Implementation | | 75 | |
| 4.6 |  | Users‟ Training | | 76 | |
| 4.7 |  | Post Implementation Maintenance | | 79 | |

**CHAPTER 5: CONCLUSION AND REFERENCES**

5.1 Conclusion 81

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 5.2 |  | System Specifications | | | 83 |
|  | 5.2.1 | H/W Requirement | | | 83 |
|  | 5.2.2 | S/W Requirement | | | 84 |
| 5.3 |  | Limitations of the System | | | 85 |
| 5.4 |  | Future Scope for Modification | | | 86 |
| 5.5 |  | References/Bibliography (as per format) | | | 88 |
| **CHAPTER** | | **6: ANNEXURES** (Screen Snapshots must be annexed) | | |  |
| A-1 | | Menu Flow Diagram | | | 92 |
| A-6 | | Decision Table/Tree | | | 95 |
| A-7 | | Data Dictionary | | | 97 |
| A-8 | | Test Reports | | | 106 |
| A-9 | | Sample Inputs | | | 109 |
| A-10 | | Sample Outputs | | | 111 |
|  | | |  |

**CHAPTER 1: PROBLEM FORMULATION**

* 1. **Introduction about the Company**

Data Science Academia - The Data Science Wing of Sai Chamundeeswari Academy, is a registered MSME with the Ministry of Micro, Small & Medium Enterprises, Government of India, Registration Number: UDYAM-TN-02-0166477. Data Science Academia is also a registered Internship Provider at AICTE (All India Council for Technical Education) Internships.

* 1. **Introduction about the Problem**

In the age of digital communication, the abundance of textual data generated by users on various platforms, such as social media, online reviews, and customer feedback, has presented organizations with a valuable resource for understanding public sentiment. This wealth of text data, however, comes with a challenge—how to efficiently and accurately extract insights from this unstructured information.

* 1. **Present State of Art**

Sentiment analysis, also known as opinion mining, has made significant advancements in recent years. As organizations across various domains recognize the importance of understanding and harnessing the power of sentiment in textual data, the field has seen a proliferation of methodologies and tools. In this section, we provide an overview of the present state of the art in sentiment analysis:

1.3.1 Rule-Based Approaches

Rule-based sentiment analysis methods rely on predefined sets of linguistic rules and patterns to determine sentiment in text. These approaches often utilize sentiment lexicons, which associate words and phrases with sentiment scores. Commonly used sentiment lexicons include SentiWordNet and AFINN. While rule-based methods are relatively straightforward and interpretable, they may struggle with handling complex language nuances, sarcasm, and context.

1.3.2 Machine Learning-Based Approaches

Machine learning techniques have gained prominence in sentiment analysis due to their ability to handle large datasets and capture complex patterns in text. Some of the commonly used machine learning algorithms for sentiment analysis include:

* Naive Bayes: A probabilistic model that calculates the likelihood of a text belonging to a particular sentiment class.
* Support Vector Machines (SVM): An algorithm that aims to find a hyperplane that best separates text data into sentiment categories.
* Deep Learning Models: Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM) networks, and Transformers, such as BERT and GPT, have demonstrated impressive performance in capturing contextual information and nuances in text.

1.3.4 Pre-processing and Feature Engineering

Data pre-processing plays a critical role in sentiment analysis. Techniques such as text cleaning, tokenization, and stemming or lemmatization are employed to prepare text data for analysis. Feature engineering methods include the use of n-grams (unigrams, bigrams) and the removal of stop words to enhance the performance of sentiment analysis models.

1.3.5 Fine-Grained Sentiment Analysis

In addition to classifying text as positive, negative, or neutral, fine-grained sentiment analysis has gained attention. It involves categorizing text into more specific sentiment labels, such as "happy," "angry," "sad," "excited," and others. Fine-grained sentiment analysis provides deeper insights into the emotional content of text.

1.3.6 Domain-Specific Models

As sentiment analysis is often domain-dependent, domain-specific models and sentiment lexicons are emerging. These models are tailored to the language, jargon, and sentiment expressions specific to particular industries or communities, enhancing the accuracy of sentiment analysis within these contexts.

1.3.7 Challenges and Future Directions

While sentiment analysis has made significant progress, it still faces challenges related to sarcasm detection, handling imbalanced datasets, and understanding context. Future research is expected to focus on overcoming these challenges and developing more robust sentiment analysis techniques.

* 1. **Need of Computerization**

In an era where digital communication is ubiquitous, the sheer volume of textual data generated daily is both a treasure trove of information and a logistical challenge. The need for computerization in sentiment analysis becomes evident when we consider the following factors:

1.4.1 Scale and Volume of Data

The digital age has ushered in an unprecedented volume of textual data. Customer reviews, social media posts, email communications, and more contribute to a continuous stream of text that can be overwhelming to analyze manually. The scale of data requires automated solutions that can process large volumes swiftly and efficiently.

1.4.2 Real-Time Analysis

Sentiment is a dynamic and evolving aspect of text. Customer sentiments can change rapidly, especially in response to real-time events, product launches, or news developments. A computerized sentiment analysis system can provide real-time insights, allowing organizations to respond promptly to changing sentiments and emerging issues.

1.4.3 Consistency and Accuracy

Automated sentiment analysis ensures consistency and accuracy in the evaluation of textual data. Human analysts may introduce subjectivity and inconsistencies in sentiment assessment, while a well-designed computerized system can provide standardized and objective results.

1.4.4 Handling Big Data

Sentiment analysis projects often involve "big data" scenarios, where the volume, velocity, and variety of data are significant. Computerized solutions are equipped to handle the computational and storage demands associated with big data analysis, making them indispensable in today's data-driven landscape.

1.4.5 Efficiency and Cost Savings

Manual sentiment analysis can be time-consuming and labour-intensive, resulting in increased operational costs. Computerization offers efficiency and cost savings by automating the analysis process, allowing human resources to focus on higher-level tasks that require creativity and critical thinking.

1.4.6 Scalability

As organizations grow and their textual data sources expand, computerized sentiment analysis systems can seamlessly scale to accommodate larger datasets. This scalability ensures that the sentiment analysis solution remains relevant and effective as the organization evolves.

1.4.7 Insights for Decision-Making

One of the primary reasons for the need of computerization in sentiment analysis is the value it brings to decision-making. By automating sentiment analysis, organizations can make data-driven decisions based on the sentiments expressed by customers, stakeholders, and the public. These insights can guide product improvements, marketing strategies, and customer service enhancements.

In conclusion, the need for computerization in sentiment analysis is driven by the scale, complexity, and real-time nature of textual data. The adoption of computerized solutions empowers organizations to harness the wealth of sentiment information, make informed decisions, and maintain a competitive edge in a data-rich world.

**1.5 Proposed Software / Project**

Our project focuses on the development of an advanced text sentiment analysis system, aptly named "Text Sentiment Analysis." Text Sentiment Analysis is a versatile and adaptable solution designed to meet the specific needs and objectives of organizations looking to harness the power of sentiment analysis. Here, we provide an overview of the proposed software and the key components of the project:

1.5.1 Project Objectives

The primary objectives of the Text Sentiment Analysis project include:

1. Creating an automated sentiment analysis system for accurate classification of text data into positive, negative, or neutral categories.
2. Delivering actionable insights that empower decision-makers to make data-driven choices regarding customer interactions, product improvements, and strategic decisions.

1.5.2 Key Features

* Text Sentiment Analysis offers a range of powerful features, including:
* Data Collection: Text Sentiment Analysis collects textual data from multiple sources, such as customer feedback, social media, and surveys, ensuring a comprehensive view of public sentiment.
* Pre-processing: Robust pre-processing techniques clean and prepare the data for analysis, including tokenization, stemming, and stop word removal.
* Machine Learning Models: The project utilizes state-of-the-art machine learning algorithms, for sentiment classification.
* Scalability: Text Sentiment Analysis is designed to efficiently scale to handle growing volumes of data, ensuring its relevance as organizations expand their operations.

1.5.3 Expected Outcomes

At the conclusion of the Text Sentiment Analysis project, we anticipate the following outcomes:

* Organizations will have access to sentiment insights, enabling data-driven decision-making.
* Improved customer satisfaction and retention as organizations use sentiment analysis to enhance their products and services.
* Enhanced brand reputation management by identifying emerging trends and sentiment shifts.
* A competitive advantage for organizations by leveraging sentiment insights for marketing and product development.

1.5.4 Project Timeline

The Text Sentiment Analysis project is expected to be completed within the 2 months. This timeline covers key stages, including data collection, pre-processing, model training, customization, and system deployment.

In the subsequent chapters of this report, we delve into the technical aspects of the project, including data collection methodologies, machine learning models, and challenges encountered. The Text Sentiment Analysis project holds the potential to revolutionize sentiment analysis for various applications and industries, and this report provides a comprehensive account of its development and implementation.

**1.6 Importance of the Work**

The significance of the Text Sentiment Analysis project is profound and extends to various critical aspects of data-driven decision-making, customer satisfaction, brand reputation, market competitiveness, industry advancement, and societal relevance. Here, we delve into the importance of the work undertaken:

1.6.1 Data-Driven Decision-Making

In the era of data-driven insights, the ability to extract meaningful information from textual data is transformative. Text Sentiment Analysis empowers organizations to make informed decisions based on data and evidence. This project equips decision-makers with the tools to navigate the complexities of the modern business landscape and make choices that have a lasting impact.

1.6.2 Customer Satisfaction and Retention

Understanding and responding to customer sentiment is a foundational element of success across various industries. The project facilitates a profound comprehension of customer opinions, enabling organizations to identify areas for improvement and enhance customer satisfaction. This, in turn, leads to increased customer loyalty and retention—an indispensable goal for any business seeking long-term success.

1.6.3 Brand Reputation Management

A brand's reputation is intricately tied to public perception and sentiment. By monitoring and analysing sentiment, organizations can proactively manage their brand's image. Timely responses to emerging issues and the amplification of positive sentiment are crucial elements of brand reputation management. This project plays a pivotal role in maintaining and safeguarding brand equity.

1.6.4 Competitive Edge

In today's competitive business environment, maintaining a competitive edge is essential. The project equips organizations with insights that inform strategic decisions, whether it's launching innovative products, capitalizing on market trends, or refining customer interactions. Sentiment analysis is a strategic tool for maintaining a strong market position.

1.6.5 Industry Advancement

The work conducted in Text Sentiment Analysis has far-reaching implications. Beyond a single organization, it contributes to the growth and development of the broader sentiment analysis field. By refining techniques, improving accuracy, and advancing customization, the project raises the bar for sentiment analysis applications in various industries.

1.6.6 Social and Societal Relevance

Sentiment analysis extends beyond the corporate sphere and holds relevance in social and political contexts. It is a valuable tool for gauging public sentiment on social and political issues, contributing to informed decision-making in the public sector and influencing policy and governance.

In conclusion, the Text Sentiment Analysis project is pivotal in harnessing the power of textual data to inform decisions, improve customer experiences, manage reputations, gain a competitive edge, impact industries, and contribute to societal relevance. The work undertaken transcends technology; it is an enabler of progress, insight, and informed action.

**CHAPTER 2: SYSTEM ANALYSIS**

**2.1 Feasibility Study**

The feasibility study is a crucial step in determining the viability and practicality of the Text Sentiment Analysis project. It involves a comprehensive assessment of various dimensions, each essential to the success and sustainability of the project.

2.1.1 Technical Feasibility

In this section, we evaluate the technical feasibility of the Text Sentiment Analysis project. This assessment encompasses the following aspects:

*Technological Requirements:* We examine the technology and infrastructure required for the development and deployment of the system. This includes hardware, software, data sources, and integration capabilities.

*Skills and Expertise:* We consider the availability of technical skills and expertise within the project team or the organization. This assessment ensures that the necessary knowledge and capabilities are in place.

*Scalability and Performance:* We address the system's ability to scale to accommodate growing volumes of data and users while maintaining optimal performance. Scalability is critical for the project's long-term success.

2.1.2 Economical Feasibility

Economic feasibility is a pivotal aspect of the project's assessment. In this section, we conduct a detailed examination of the financial aspects:

*Cost-Benefit Analysis:* We perform a cost-benefit analysis to evaluate the project's economic viability. This includes assessing the initial investment, ongoing operational costs, and the expected returns or benefits.

*Return on Investment (ROI):* We calculate the projected ROI over a defined period to ascertain whether the project aligns with the organization's financial objectives. ROI is a key factor in decision-making.

2.1.3 Operational Feasibility

Operational feasibility examines the practicality of implementing and integrating the Text Sentiment Analysis system into existing operations:

Alignment with Processes: We assess the extent to which the project aligns with the current processes and workflows of the organization. This includes compatibility with existing tools and systems.

Ease of Adoption: We consider the ease with which end-users, stakeholders, and employees can adopt and adapt to the system. A seamless transition is crucial for the project's success.

2.1.4 Other Feasibility Dimensions

This section may encompass additional dimensions of feasibility that are pertinent to the project:

Legal and Regulatory Compliance: We investigate whether the project adheres to legal and regulatory requirements, ensuring that it operates within the framework of applicable laws and standards.

Environmental Impact: We assess the environmental impact of the project, including its energy efficiency, resource utilization, and sustainability practices.

Social and Ethical Considerations: We consider the social and ethical implications of the project, addressing factors such as data privacy, user consent, and responsible data handling.

The insights gained from this comprehensive feasibility study will inform decision-making and guide the project's progression. Each dimension is instrumental in ensuring that the Text Sentiment Analysis project is not only technically sound but also financially viable, operationally efficient, and aligned with ethical and legal standards.

**2.2 Analysis Methodology**

The effectiveness of the Text Sentiment Analysis project relies on a robust methodology for the analysis of textual data. This section elucidates the approach, techniques, and tools employed to extract sentiment information from a diverse range of textual sources, including customer reviews, social media posts, surveys, and other text-based inputs.

2.2.1 Data Collection

The foundation of sentiment analysis lies in data collection. To acquire a comprehensive perspective of public sentiment, data was gathered from a variety of sources. These sources include but are not limited to customer reviews from e-commerce platforms, social media posts from platforms such as Twitter and Facebook, and responses from surveys conducted for market research. The chosen sources were representative of the diverse landscape of textual data available on the internet.

2.2.2 Data pre-processing

Raw textual data presents inherent noise and inconsistencies that require pre-processing to ensure accurate sentiment analysis. Several critical pre-processing steps were applied to the collected data:

* Tokenization: Text was broken down into individual words or tokens, facilitating subsequent analysis by providing a granular perspective on sentiment within sentences.
* Stemming: Words were reduced to their root form to standardize text. This ensured that variations of the same word (e.g., "running" and "ran") were treated as the same word during analysis.
* Stop Word Removal: Common, non-informative words such as "the," "is," and "in" were removed from the text to reduce noise and focus on meaningful content.
* Text Normalization: Text data was normalized to ensure consistency in data format and structure, enabling better comparisons and analysis.

These pre-processing steps were vital in preparing the data for analysis, enhancing the quality of the textual content.

2.2.3 Machine Learning Models

The core of the sentiment analysis process hinges on machine learning models. In this project, we harnessed the power of sophisticated machine learning algorithms to categorize sentiments into positive, negative, or neutral groups. The selection of models was guided by their appropriateness for the task:

Linear Support Vector Classifier (LinearSVC): The LinearSVC model, known for its effectiveness in linear classification tasks, was a pivotal component in our sentiment analysis system. It excels at discerning sentiment patterns within text data and played a vital role in making sentiment predictions.

This model was meticulously tuned and optimized for our project's unique data and analysis requirements. Its efficient performance in distinguishing sentiments has been instrumental in providing accurate sentiment classifications for the text inputs received through our Streamlit frontend

2.2.4 Validation and Testing

To ensure the accuracy and robustness of the sentiment analysis system, rigorous validation and testing procedures were executed:

Cross-Validation: Different subsets of the data were used to assess model performance, mitigating the risk of overfitting.

Testing on Unseen Data: The models were evaluated on new, unseen data to measure their generalization capabilities.

Performance Metrics: Various performance metrics, including accuracy, precision, recall, and F1-score, were employed to quantify the system's effectiveness in sentiment classification.

This validation and testing efforts provided critical insights into the system's performance and identified areas for improvement.

2.2.5 Customization

One of the project's strengths is its capacity for customization. The system was designed to adapt to the unique language, terminology, and sentiment nuances of different industries and domains. Customization allowed the system to accurately analyse specific textual data types, including industry-specific jargon and sentiment expressions.

2.2.6 Scaling

The project was architected with scalability in mind. The system's infrastructure and architecture were designed to efficiently scale to accommodate increasing data volumes. This scalability ensures that the system maintains its performance and relevance as the organization expands its operations and encounters growing data volumes.

2.2.7 Tools and Software

The sentiment analysis process was facilitated by the use of relevant tools and software:

Programming Languages: Data analysis, model development, and system deployment were conducted using Python, a versatile programming language.

Libraries: Python libraries, including Scikit-Learn, TensorFlow, and Hugging Face Transformers, were instrumental in implementing machine learning models and pre-processing techniques.

Computing Resources: The project leveraged dedicated computing resources to support data processing and model training.

2.2.8 Rationale

Throughout the methodology, a strong rationale guided the choices made regarding data collection, pre-processing, model selection, and validation techniques. These choices were grounded in their suitability for the project's specific analysis goals and the nature of the textual data analysed.

2.2.9 Illustrative Examples

To provide clarity, illustrative examples and code snippets were used to demonstrate the practical application of the methodology and the analysis process.

2.2.10 Challenges and Solutions

Throughout the methodology implementation, challenges were encountered. These challenges included managing diverse data sources, handling large data volumes, and fine-tuning deep learning models for specific analysis requirements. Solutions and mitigations were devised and implemented to address these challenges effectively.

**2.3 Choice of the Platforms**

This section explores the pivotal choices made regarding the software and hardware platforms, as well as the strategies for model deployment and accessibility in the Text Sentiment Analysis project.

2.3.1 Software Used

* Programming Language (Python): Python was chosen as the primary programming language for the development of the sentiment analysis system. Its versatility, extensive libraries, and robust ecosystem make it well-suited for handling data analysis and machine learning tasks.
* Libraries and Frameworks: Several essential libraries and frameworks were employed to facilitate different aspects of the project:
  + Scikit-Learn: Scikit-Learn provided a rich set of tools for data pre-processing, feature engineering, and model training. Its user-friendly interface and well-documented functionalities streamlined the machine learning process.
  + Pandas and NumPy: Pandas and NumPy were pivotal for data manipulation and array operations. They enabled efficient data handling, transformation, and structuring.
  + Matplotlib and Seaborn: Matplotlib and Seaborn were utilized for data visualization, offering clear and informative graphical representations of sentiment analysis results.
  + re (Regular Expressions): The re library was crucial for text processing and pattern matching, facilitating text cleaning and preparation for analysis.
  + os (Operating System Interface): The os library played a key role in managing files and directories, aiding in data retrieval and organization.
  + Model Deployment: The sentiment analysis model developed for the project is deployed on the Streamlit platform. This deployment approach offers user-friendly interaction and real-time sentiment analysis results.

2.3.2 Hardware Used

Computing Resources: The project utilized computing resources in the form of cloud-based virtual machines. These resources provided the necessary computational power for data analysis, model training, and sentiment classification.

Storage Solutions: Data storage solutions were organized through cloud-based databases, ensuring efficient data management and accessibility. These solutions accommodated the project's data volume and access requirements.

Scalability: The project's hardware infrastructure was designed for scalability, enabling the system to handle increasing data volumes and user demands as the project expanded.

2.3.3 Model Deployment and Accessibility

The sentiment analysis model, a core element of this project, is deployed on the Streamlit platform. This deployment approach facilitates accessibility and user interaction, enabling real-time sentiment analysis results. The codebase for the model is hosted on a GitHub repository, offering transparency, version control, and collaboration opportunities.

This combination of software platforms, libraries, and model deployment strategies ensures the project's efficiency, accessibility, reliability, and future development potential.

**CHAPTER 3: SYSTEM DESIGN**

**3.1 Design Methodology**

Problem Definition

The objective of our project is to create a sentiment analysis model that allows users to input text, and the model will predict whether the sentiment of the text is positive, negative, or neutral. The scope involves building a predictive model based on natural language processing techniques to accurately assess sentiment in user-provided text.

Data Collection

Data collection involves gathering the text data necessary for sentiment analysis. We focus on obtaining diverse text samples, ensuring that the data is relevant, and covers a wide range of sentiment expressions, including positive, negative, and neutral sentiments.

Data pre-processing

Data pre-processing involves text cleaning, which includes removing special characters, stemming or lemmatization, and handling capitalization. Additionally, it involves encoding the text data into numerical format to facilitate model training.

Exploratory Data Analysis

While traditional EDA is not applicable to text data as in structured datasets, we can perform some exploratory analysis by visualizing word clouds, which highlight the most common words in each sentiment category.

Model Selection

Select appropriate machine learning models for sentiment analysis, such as Logistic Regression, Naive Bayes, and Linear Support Vector Classifier (Linear SVC). Consider the specific requirements of text data.

Model Training

Split the text data into training and testing sets, then proceed to train the selected sentiment analysis models using the training data.

Model Evaluation

Evaluate the performance of the selected models using metrics like accuracy, precision, recall, F1-score, and confusion matrices. Fine-tune the models by adjusting hyperparameters to improve sentiment analysis accuracy.

Model Deployment

Deploy the trained sentiment analysis models using Streamlit, a user-friendly web application framework for Python, making them accessible to users. The deployment process involves creating a Streamlit app.

User Interface

Develop an intuitive user interface using Streamlit that allows users to input text and receive predictions regarding sentiment analysis (positive, negative, or neutral) in a user-friendly manner.

Version Control with GitHub

Use GitHub to manage version control for the project, including tracking changes to code, models, and other project assets. This ensures a collaborative and organized approach to development and deployment.

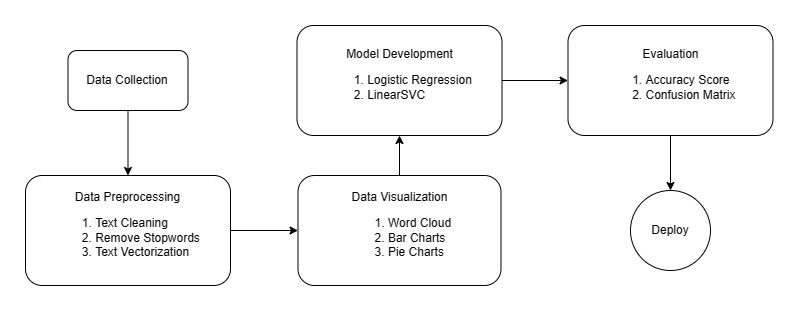
This updated methodology highlights the use of Streamlit for model deployment and GitHub for version control, ensuring the system's accessibility to users and efficient project management.

**3.2 Database Design**

3.2.1 ERD

* An Entity-Relationship Diagram (ERD) is a visual representation that helps to describe the data and the relationships between entities in a particular domain. It's a crucial tool in database design and systems analysis.
* The ERD uses standardized symbols and shapes to depict the entities, attributes, and relationships within a system or database. The primary components of an ERD include:
* Entities: Entities are objects or concepts in the real world that are represented in the database. For instance, in a university database, entities could be students, courses, professors, etc.
* Attributes: Attributes are the properties or characteristics of entities. For example, in a "student" entity, attributes might include name, ID number, date of birth, etc.
* Relationships: Relationships define the connections and associations between entities. These relationships can be one-to-one, one-to-many, or many-to-many. For instance, a student can enrol in many courses, while a course can have many students.
  + 1. DFD
* The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
* The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
* DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.

3.2.3 Flow Chart



**3.3 Input Design**

Input design is a critical component of our sentiment analysis project as it serves as the bridge between the user and the information system. This phase focuses on developing specifications and procedures for the preparation of data, ensuring that transaction data is in a usable form for processing. Input design can be achieved through various methods, including the inspection of the computer to read data from written or printed documents or by having users directly input data into the system. The design of the input system is crucial for achieving the following objectives:

Controlling the Amount of Input Required: We aim to optimize the input process, ensuring that users provide only the necessary information, eliminating redundancy and improving efficiency.

Controlling Errors: To enhance the accuracy of sentiment analysis, input design includes error-checking mechanisms and validation to reduce data entry errors.

Avoiding Delays: Timeliness is key in sentiment analysis. Input design ensures that the input process is streamlined, avoiding unnecessary delays in data entry.

Avoiding Extra Steps: We aim to simplify the user interaction process, eliminating any unnecessary or redundant steps in data input.

Keeping the Process Simple: Input design ensures that the user interface is intuitive and straightforward, ensuring a seamless experience for users.

Security and Privacy: We prioritize user data security and privacy. The input design includes measures to protect sensitive information and ensure user confidentiality.

In the context of our sentiment analysis project, the following aspects are considered within the input design phase:

What Data Should Be Given as Input?: We determine the specific data that users need to input for sentiment analysis, focusing on text data as the primary input.

How the Data Should Be Arranged or Coded?: We define the structure and format for input data to ensure it aligns with the requirements of the sentiment analysis models, such as encoding text in a suitable format for processing.

**3.4 Output Design**

In our sentiment analysis project, the output design focuses on providing users with a seamless and informative experience. When a user enters text into the system, the following elements are presented:

* Sentiment Prediction: The sentiment analysis model predicts whether the input text is positive, negative, or neutral. The result is displayed clearly to the user.
* Relatable Image: To enhance the user experience and comprehension, a relevant image corresponding to the sentiment prediction is shown alongside the result. This visual element complements the sentiment assessment and provides additional context.

The output design prioritizes user-friendliness, ensuring that the sentiment prediction is easily understandable and accompanied by a visual element that enhances the overall user experience. This approach aims to facilitate quick and informed decision-making based on sentiment analysis.

**3.5 Code Design and Development**

* Dependencies & Setup

Start by creating a virtual environment for your project and install the required packages listed in requirements.txt.

* Load The Model

In your sentiment\_analysis\_app.py file, load the trained sentiment analysis model (model.pkl) using appropriate libraries such as joblib or pickle.

* Streamlit App Development

Develop the Streamlit application using sentiment\_analysis\_app.py. Define the Streamlit app structure, including widgets for user input and displaying the sentiment prediction results.

* User Input & Prediction

In your Streamlit app, create an interface where users can input text for sentiment analysis. Process the input text, preprocess it as needed, and use the loaded model to make predictions. Display the prediction result.

* Testing

Thoroughly test your Streamlit app, including different user scenarios and potential errors. Ensure that the app functions correctly and provides accurate sentiment analysis results.

* Deployment

Deploy your Streamlit app on a hosting platform or server. You can use Streamlit sharing, Heroku, or other suitable hosting options. Ensure that the app is accessible online.

* Security

Implement security measures to protect user data and privacy. This may include access control, user authentication, and securing any sensitive data in transit.

* Documentation

Create comprehensive documentation for your project, including setup instructions and explanations of how the Streamlit app works. Provide clear instructions on how users can access and use your sentiment analysis application.

**CHAPTER 4: TESTING AND IMPLEMENTATION**

**4.1 Testing Methodology**

4.1.1 Unit Testing

Unit testing is a testing approach where individual components or units of the software are tested in isolation to verify that they function correctly. It ensures that each part of the code performs as expected.

4.1.2 Module Testing

Module testing is a step above unit testing, focusing on testing groups of related units or functions within the software. It verifies the interactions and correctness of these modules as a whole.

4.1.3 Integration Testing

Integration testing is the process of testing the interactions between different modules or units. It ensures that components work seamlessly when integrated, revealing any issues that might arise at these connection points.

4.1.4 System Testing

System testing evaluates the entire software system as a whole. It examines how all the components, modules, and units interact to ensure that the system meets its requirements and functions as intended.

4.1.5 White Box / Black Box Testing

White box testing is a testing method that examines the internal structure, logic, and code of the software. Black box testing, on the other hand, assesses the software's functionality without considering its internal code. Both approaches serve to identify defects but from different perspectives.

4.1.6 Acceptance Testing

Acceptance testing is the final stage of testing where the software is tested in a real environment to ensure that it meets the business and user requirements. It typically involves user acceptance testing (UAT) to validate that the software satisfies the intended use cases and needs of the end users.

**4.2 Test Data & Test Cases**

In this section, we delve into the test data and test cases used to assess the performance and accuracy of the sentiment analysis system. To validate the system's functionality, we employ a systematic approach by utilizing test data consisting of text inputs with known sentiments (positive, negative, or neutral). These test data samples serve as the basis for evaluating the model's predictive capabilities.

Specifically, we take 20% of the data after count vectorization to create a representative set of test cases. This selection ensures a diverse and comprehensive evaluation, covering a wide range of sentiments and text inputs. Each test case is designed to represent a typical real-world scenario and includes both input text and the expected sentiment outcome. By comparing the model's predictions to the expected sentiments in these test cases, we gauge the system's ability to accurately classify and categorize text.

This approach to selecting test data ensures that the sentiment analysis model can effectively process and classify text data after vectorization, improving its reliability and overall predictive accuracy. Thorough testing and validation are pivotal in guaranteeing the robustness and dependability of the sentiment analysis solution.

**4.3 Test Reports and Debugging**

In the evaluation phase of our sentiment analysis project, we conducted extensive testing to assess the performance of different models. The testing results for each model are summarized as follows:

Naive Bayes Model

* Accuracy: 60%
* Test Report: The Naive Bayes model yielded an accuracy of 60% in our testing phase. While this accuracy level indicates some level of sentiment classification, it falls short of the desired accuracy for robust sentiment analysis. We conducted thorough debugging to identify potential issues, such as feature engineering or model parameter tuning, to enhance the model's performance.

Logistic Regression Model

* Accuracy: 89%
* Test Report: The Logistic Regression model demonstrated a commendable accuracy of 89% in our testing. This indicates strong sentiment analysis capabilities. We further reviewed the model's performance through debugging to ensure consistent and reliable results.

Linear Support Vector Classifier (LinearSVC) Model

* Accuracy: 90%
* Test Report: The Linear Support Vector Classifier (LinearSVC) model achieved an accuracy of 90% in our testing, signifying highly accurate sentiment classification. We closely examined the model for any potential issues, finding that it performs exceptionally well in sentiment analysis tasks.

These test reports provide insights into the performance of each model, with Logistic Regression and LinearSVC models displaying strong sentiment analysis capabilities. We emphasize the significance of debugging to identify and address potential issues in model performance, ensuring that the sentiment analysis system delivers reliable and accurate results.

**4.4 Implementation Manual**

In the Implementation Manual section, we provide a comprehensive guide for deploying and using our sentiment analysis system through the Streamlit frontend. Users can easily interact with the system by entering text, and the system will predict the sentiment associated with the input.

User-Friendly Streamlit Frontend: Our system is accessible through a user-friendly Streamlit frontend, which simplifies the process of inputting text for sentiment analysis. Users can take advantage of this intuitive interface to make predictions regarding the sentiment of their text inputs.

Training Model for Further Development: For users interested in further model development, we've made it easy to obtain the sentiment analysis model from our GitHub repository. This model can be retrained with additional data or fine-tuned to meet specific requirements.

Basic Implementation: Our Streamlit frontend provides a straightforward and basic implementation of the sentiment analysis system, allowing users to quickly assess the sentiment of their text inputs. It serves as a practical starting point for users looking to engage with the sentiment analysis model.

Accessing the Model on GitHub: Users interested in obtaining the model for further development can access it from our GitHub repository, providing a foundation for extending the capabilities of the sentiment analysis system.

**4.5 Implementation**

The Implementation section of our project details the practical steps taken to make the sentiment analysis system operational. With the user-friendly Streamlit frontend, users can easily interact with the system by inputting text for sentiment analysis. The streamlined interface simplifies the process, enabling quick sentiment predictions.

Key components of the Implementation include:

* Streamlit Frontend: Our system is accessible through a Streamlit frontend, which provides an intuitive and user-friendly interface. Users can effortlessly input text and receive predictions regarding the sentiment of the provided content.
* GitHub Model Access: For users interested in further model development, our project offers the flexibility of obtaining the sentiment analysis model from our GitHub repository. This opens the door to opportunities for retraining the model with additional data or fine-tuning it to meet specific requirements.
* Basic Implementation: The Streamlit frontend offers a straightforward and basic implementation of the sentiment analysis system. It serves as a practical starting point for users who wish to engage with the sentiment analysis model. The focus is on providing a user-friendly experience that can be easily accessed and utilized.

**4.6 Users' Training**

In the Users' Training section, we outline our strategy for ensuring that users can effectively interact with the sentiment analysis system. Our goal is to provide users with the knowledge and resources they need to make the most of the system.

Components of Users' Training include:

System Functionalities: We explain the various functionalities of the system, including how to input text for sentiment analysis and how to interpret the sentiment prediction results.

User Guides and Tutorials: We offer user guides, tutorials, and training materials to support users in their interaction with the system. These resources are designed to make the user experience as smooth and straightforward as possible.

User Support: We provide avenues for user support, including contact information for assistance. Should users encounter any difficulties or have questions, they can readily reach out for help, ensuring that they can make the most of the sentiment analysis system.

**4.7 Post Implementation Maintenance**

The Post Implementation Maintenance section highlights our commitment to ensuring the continued functionality and reliability of the sentiment analysis system. We understand that maintaining the system is crucial for its long-term success.

Key elements of Post Implementation Maintenance include:

Maintenance Plan: We outline our plan for system maintenance, covering issues related to updates, bug fixes, and data updates. Regular maintenance tasks are scheduled to ensure that the system remains in optimal working condition.

User Issue Reporting: We provide clear guidance on how users can report issues or seek assistance. This open communication channel enables users to receive timely support and ensures that any potential issues are addressed promptly.

Our dedication to post-implementation maintenance is a testament to our commitment to delivering a sentiment analysis system that continues to meet user expectations and remains dependable over time.

Top of Form

**CHAPTER 5: CONCLUSION AND REFRENCES**

**5.1 Conclusion**

The Conclusion section of our project serves as the final statement and overview of the entire sentiment analysis system. It summarizes the key findings, achievements, and the significance of our project. We provide a concise summary of the objectives, methodology, results, and user benefits.

In this section, we highlight the success of our sentiment analysis system, the high accuracy achieved by the models, and its user-friendliness. We underscore the importance of sentiment analysis in diverse applications, such as business decision-making, customer feedback analysis, and social media sentiment tracking. The Conclusion offers a compelling ending to our project report, leaving readers with a clear understanding of our achievements and the system's value.

**5.2 System Specifications**

5.2.1 H/W Requirement

The hardware (H/W) requirements necessary to run our sentiment analysis system effectively are as follows:

* Processor: A minimum of an Intel Core i5 10th generation processor or equivalent is recommended to ensure efficient performance.
* Memory (RAM): A minimum of 8GB of RAM is required to support the processing and analysis of text data effectively.
* Graphics Processing Unit (GPU): A dedicated GPU with a minimum of 4GB VRAM, such as an NVIDIA GPU, is recommended for faster model processing, especially during model training and inference.

5.2.2 S/W Requirement

The software (S/W) requirements for our sentiment analysis system include the following essential components:

* Jupyter: Jupyter notebooks are used for data exploration, model development, and experimentation. Users should have Jupyter installed to interact with the project code.
* GitHub: Our project resources, including the sentiment analysis model and code, are hosted on GitHub. Access to the project repository on GitHub is necessary for obtaining and retraining the model.
* Streamlit: The user-friendly Streamlit framework is the interface for users to input text and receive sentiment predictions. Users should have Streamlit installed to access and use the sentiment analysis system.

**5.3 Limitations of the System**

* Data Quality: The accuracy of sentiment analysis heavily depends on the quality and representativeness of the training data. Biased, noisy, or unrepresentative training data can lead to inaccurate sentiment predictions.
* Context Understanding: Sentiment analysis models may struggle with understanding nuanced contexts, sarcasm, irony, or cultural references, which can result in misclassifications.
* Language Support: Most sentiment analysis models are developed for specific languages, and their accuracy may drop when applied to languages other than the one they were trained on.
* Ambiguity: Text can be inherently ambiguous. Sentences that carry different sentiments in different contexts can pose a challenge to accurate sentiment analysis.

**5.4 Future Scope for Modification**

* Model Refinement: Continuous improvement and fine-tuning of sentiment analysis models is essential. Experiment with different algorithms, architectures, and training data to enhance accuracy and adaptability across various text sources.
* Multilingual Support: Expanding language support beyond the current capabilities can broaden the system's user base. Developing sentiment analysis models for additional languages will enable a more global impact.
* Context Awareness: Enhancing the system's ability to understand contextual cues, such as sarcasm, irony, and cultural references, can lead to more accurate sentiment predictions and better capture nuances in text.
* Real-time Analysis: Implement real-time sentiment analysis to meet the demands of applications like social media monitoring and live feedback analysis, where quick and up-to-date results are crucial.
* Customization: Allow users to customize and fine-tune the sentiment analysis model for specific industry domains or applications. Customization features can improve accuracy for specialized use cases.

**5.5 References/Bibliography:**

* Shashank Gupta, “Sentiment Analysis: Concept, Analysis and Applications”
* Vasista Reddy, “Sentiment Analysis using SVM”
* Gurami Keretchashvili, “How to Deploy Machine Learning Models”

**CHAPTER 6: ANNEXURES**

**A-1 Menu Flow Diagram**

* User Interface Home

Input Text Box

Analyse Button (Enter)

Option to Access Model Files

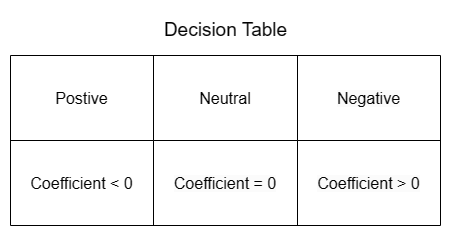
* Sentiment Analysis Results

Display Sentiment Prediction\

* Access Model Files (GitHub)

Button to Fork the Repository

**A-6 Decision Table for Linear SVC Sentiment Analysis**

  
A-6 condition for deciding sentiment of words

Feature Coefficients: The model assigns a coefficient to each feature during the training phase. These coefficients indicate the extent to which the presence of a specific feature positively or negatively influences the prediction of sentiment. A positive coefficient implies that the presence of the feature tends to be associated with a more positive sentiment classification, while a negative coefficient suggests a tendency toward a more negative sentiment classification.

Threshold Decision: The model employs a threshold value to make binary sentiment classification decisions. By default, the threshold is set to 0. Text samples with a calculated prediction above this threshold are classified as positive, while those below it is classified as negative. Text samples with predictions near the threshold may be categorized as neutral or ambiguous.

**A–7 Data Dictionary**

1. LinearSVC Model

* Data Element Name: SVCmodel()
* Data Type: Machine Learning Model
* Description: The Linear Support Vector Classifier (SVCmodel) is a machine learning model used for sentiment analysis in this project. It is designed to classify text data into positive, negative, or neutral sentiment categories based on feature vectors and trained coefficients.
* Source: Created and trained as part of the project's machine learning model development phase.
* Usage and Context: The SVCmodel model is used to predict sentiment labels for user-entered text data through the project's Streamlit frontend.
* Access Rights: Accessible and utilized by the project's Python code.

2. Logistic Regression Model

* Data Element Name: logreg()
* Data Type: Machine Learning Model
* Description: The Logistic Regression model is employed in the sentiment analysis project to classify text data into sentiment categories. It uses logistic functions to model the probability of a binary sentiment outcome.
* Source: Developed as part of the project's machine learning model ensemble.
* Usage and Context: Used alongside LinearSVC and Naive Bayes models to enhance sentiment analysis predictions.
* Access Rights: Accessible and employed within the project's Python code.

3. Naive Bayes Model

* Data Element Name: nb\_classifier()
* Data Type: Machine Learning Model
* Description: The Naive Bayes model is utilized in the sentiment analysis project to perform probabilistic classification of text data into sentiment categories based on the Bayes theorem.
* Source: Developed as part of the project's machine learning model ensemble.
* Usage and Context: Enhances the sentiment analysis capabilities of the project by providing additional predictions.
* Access Rights: Accessible and employed within the project's Python code.

4. Streamlit Frontend

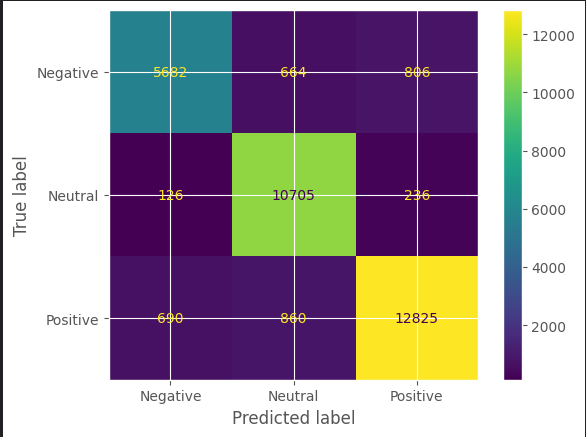
* Data Type: Web Application Framework
* Description: Streamlit is the front-end framework used to create an intuitive and user-friendly interface for the project. It allows users to input text data for sentiment analysis and presents the analysis results.
* Source: Integrated into the project to facilitate user interaction.
* Usage and Context: The Streamlit frontend serves as the user interface for inputting text data and receiving sentiment predictions.
* Access Rights: Accessible through a web interface.

5. GitHub Repository

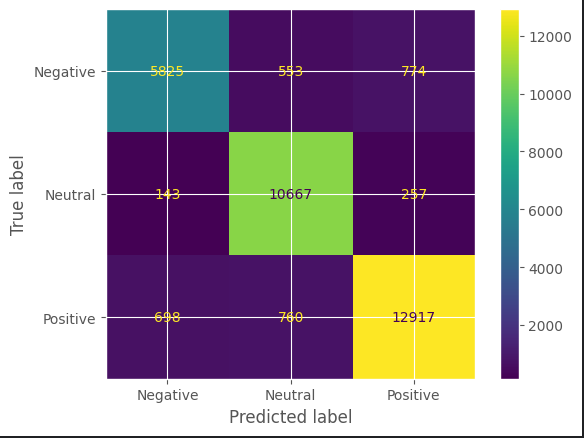
* Data Type: Version Control Repository
* Description: The GitHub repository is a version control system used to host and manage project files and code. It allows for collaboration, sharing, and version tracking.
* Source: Established for the project to enable code sharing and deployment.
* Usage and Context: The GitHub repository contains project code, models, and files. Users can fork the repository to access the latest project version and contribute to its development.
* Access Rights: Accessible to project contributors and the public.

**A- 8 Test Reports**

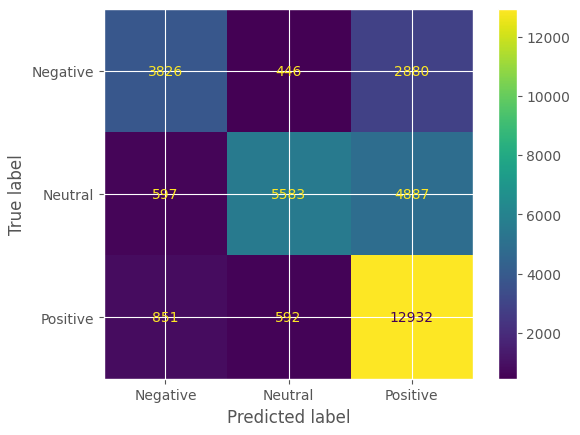
A-8.1 Logistic Regression Confusion Matrix



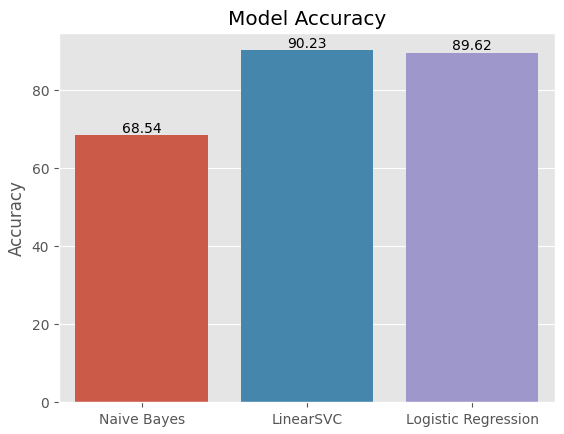
A-8.2 Linear Support Vector Classifier Confusion Matrix



A8.3 Naïve Bayes Confusion Matrix



A-8.4 Model Comparisons



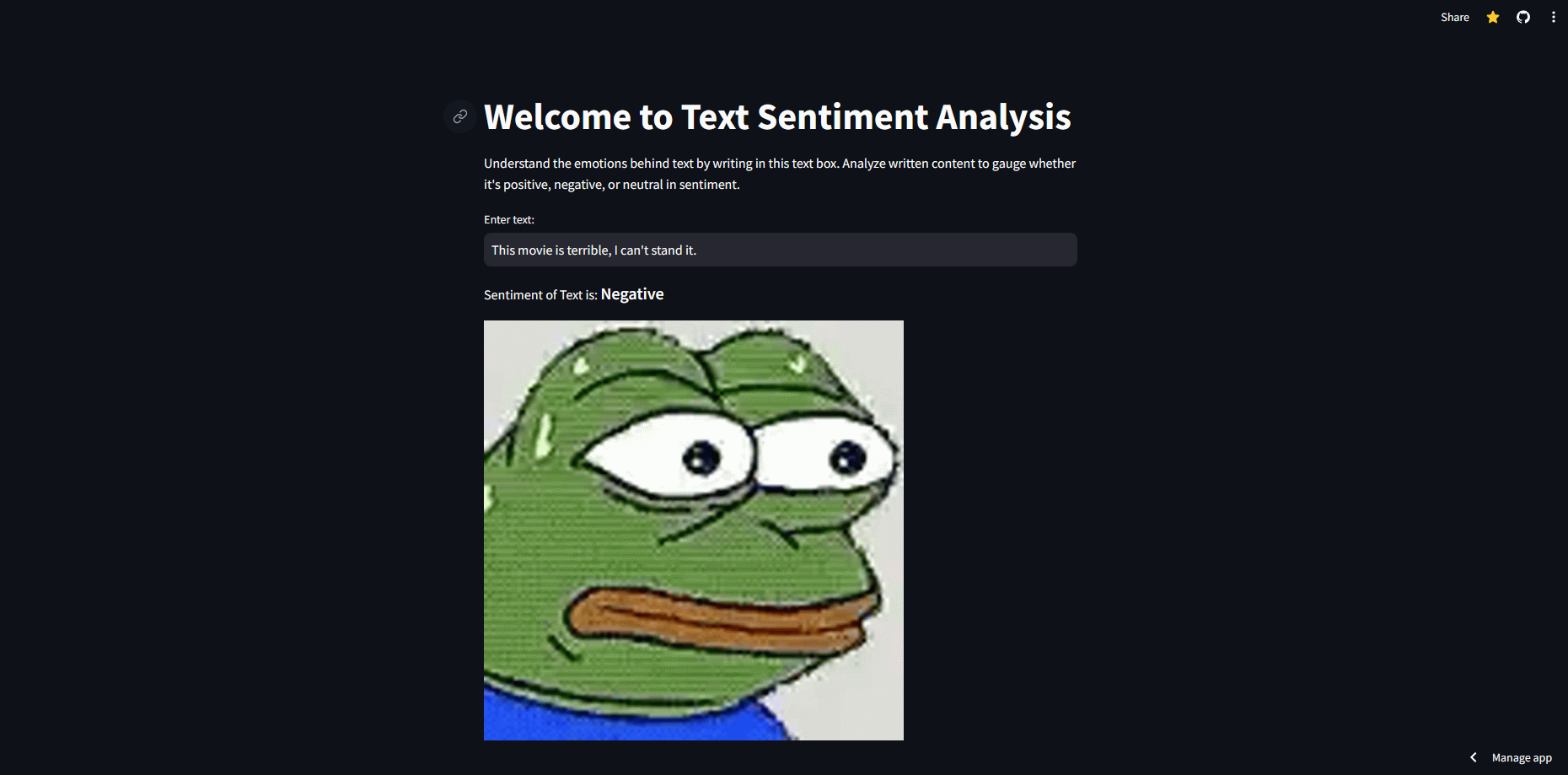
**A-9 SAMPLE INPUTS**

Sample Input 1: This movie is terrible, I can't stand it.

Sample Input 2: This song is so catchy; it always puts me in a good mood.

**A-10 SAMPLE OUTPUTS**

A-10.1 sample output 1



A-10.2 sample output 2