DESIGN & DEVELOPMENT OF SENTIMENT ANALYSIS SYSTEM

MAJOR PROJECT-I REPORT

Submitted in partial fulfillment of the requirements

for the degree of

BACHELOR OF TECHNOLOGY

in

CSE-ARTIFICIAL INTELLIGENCE & DATA SCIENCE

By

GROUP NO. 5

DISHA VISHWAKARMA (0187AD211013)
YUVIKA SINHA (0187AD211046)
PRIYAM SAHU (0187AD211032)

Under the guidance of

Prof. Ruchi Jain

(Assistant Professor)

Department of CSE-Artificial Intelligence & Data Science
Sagar Institute of Science & Technology (SISTec) Bhopal (M.P.)

Approved by AICTE, New Delhi & Govt. of M.P.
Affiliated to Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal (M.P.)

DECEMBER-2024

Sagar Institute of Science & Technology (SISTec), Bhopal

Department of CSE-Artificial Intelligence & Data Science

CERTIFICATE

I hereby certify that the work which is being presented in the B.Tech. Major Project-I Report entitled **Design & Development of Sentiment Analysis System,** in partial fulfillment of the requirements for the award of the degree of ***Bachelor of Technology*** in **CSE-*Artificial Intelligence & Data Science*** and submitted to the Department of Computer Science & Engineering, *Sagar Institute of Science & Technology (SISTec)***,** Bhopal (M.P.) is an authentic record of my own work carried out during the period from July-2024 to Dec-2024 under the supervision of **Prof. Ruchi Jain (Assistant Professor).**

The content presented in this project has not been submitted by me for the award of any other degree elsewhere.

Disha Vishwakarma	***Yuvika Sinha***	***Priyam Sahu***
0187AD211013	***0187AD211046***	***0187AD211032***

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date:

Prof. Ruchi Jain	**Dr. Vasima Khan**	**Dr. D.K. Rajoriya**
Project Guide	**HOD, CSE-AI&DS**	**Principal, SISTec**

ACKNOWLEDGEMENT

We would like to express our sincere thanks to **Dr. D.K. Rajoriya**, **Principal, SISTec** and **Dr. Swati Saxena**, **Vice Principal, SISTec Gandhi Nagar**, Bhopal for giving us an opportunity to undertake this project.

We also take this opportunity to express a deep sense of gratitude to **Dr. Vasima Khan, HOD, Department of CSE-Artificial Intelligence & Data Science** for her kindhearted support.

We extend our sincere and heartfelt thanks to our Guide, **Prof. Ruchi Jain (Assistant Professor)** for providing us with the right guidance and advice at the crucial junctures and for showing us the right way.

I am thankful to the **Project Coordinator, Ms. Ruchi Jain** who devoted her precious time in giving us the information about the various aspect and gave support and guidance at every point of time. I am thankful to their kind and supportive nature. His inspiring nature has always made my work easy.

I would like to thank all those people who helped me directly or indirectly to complete my project whenever I found myself in any issue.

TABLE OF CONTENTS

ABSTRACT

This project presents a sentiment analysis system developed to analyze and classify the sentiments expressed in social media posts, specifically targeting airline-related tweets. With the increasing volume of user-generated content on platforms like Twitter, understanding public sentiment has become crucial for businesses to enhance customer satisfaction and improve service quality.

The system utilizes a dataset of airline tweets labeled as positive, neutral, or negative, enabling it to learn the nuances of sentiment within user feedback. The pipeline includes data preprocessing techniques such as text cleaning, stopword removal, and lemmatization. For feature extraction, the Term Frequency-Inverse Document Frequency (TF-IDF) technique is employed, which converts textual data into numerical representations that highlight significant terms.

Two machine learning models, Naive Bayes and Logistic Regression, were evaluated based on their performance in classifying sentiment accurately. Logistic Regression was selected as the final model due to its higher F1-score on both training and validation sets, demonstrating robust classification capabilities.

The final application is implemented as a web-based interface using Flask, allowing users to input tweets and receive real-time sentiment predictions. This system showcases how machine learning and NLP can be effectively combined to extract actionable insights from social media data, offering potential applications in brand reputation management, customer service improvement, and market analysis.

LIST OF ABBREVIATIONS

ACRONYM	FULL FORM
SDLC	Software Development Life Cycle
AI	Artificial Intelligence
DL	Deep Learning
NLP	Natural Language Processing
TF-IDF	Term Frequency-Inverse Document Frequency
IDE	Integrated Development Environment
VS Code	Visual Studio Code
HTML	Hyper Text Markup Language
CSS	Cascading Style Sheet
UML	Unified Modeling Language
RAM	Random Access Memory

LIST OF FIGURES

Chapter 1
Introduction

CHAPTER-1
INTRODUCTION

1.1 ABOUT PROJECT:

This project focuses on **Sentiment Analysis of Airline Tweets** using **Natural Language Processing (NLP)** and **Machine Learning** to classify tweets related to airlines as positive, negative, or neutral. By analyzing tweets, airlines can gain insights into customer opinions and service experiences, allowing them to improve customer service and brand management. The workflow involves data collection, where airline-related tweets are preprocessed to remove noise like punctuation and stop words, and converted into numerical features using TF-IDF or Bag of Words techniques. Next, machine learning models, specifically Naive Bayes and Logistic Regression, are trained on labeled data to classify sentiment accurately. Although this project does not operate in real time, it is designed to provide actionable insights from pre-collected data, making it an effective tool for understanding customer sentiment trends and enhancing customer feedback analysis.

1.2 PURPOSE:

The purpose of this project is to develop a tool that enables airlines to automatically analyze and classify customer sentiments expressed in tweets as positive, negative, or neutral. By leveraging Natural Language Processing (NLP) and machine learning techniques, this project aims to transform unstructured social media data into actionable insights. This sentiment analysis helps airlines understand customer satisfaction, identify common issues, and monitor public perception of their services. Ultimately, the project supports airlines in making data-driven decisions to enhance customer service, improve brand reputation, and proactively address customer concerns.

1.3 PROJECT OBJECTIVE:

The objective of this project is to develop an automated sentiment analysis tool that classifies airline-related tweets as positive, negative, or neutral. By applying NLP and machine learning techniques, the tool aims to provide airlines with insights into customer feedback, enabling them

to identify trends, address concerns proactively, and improve overall customer experience and brand

1.4 INTERFACE:

The interface for this project is a user-friendly web application developed using **Flask**. Users can input text, such as tweets related to airline services, and receive an instant sentiment classification of **positive**, **negative**, or **neutral**. The interface allows users to easily enter or upload multiple tweets for batch analysis and displays results in a clear and organized format.

The interface design is simple and intuitive, providing easy navigation with minimal steps to interact with the model. The output includes both the sentiment prediction and relevant metrics, offering users a straightforward way to analyze customer sentiment effectively.

CHAPTER-2 HARDWARE & SOFTWARE REQUIREMENT

CHAPTER-2
HARDWARE & SOFTWARE REQUIREMENT

2.1 SOFTWARE REQUIREMENTS

2.1.1 FOR DEVELOPERS:

2.1.1.1 IDE –VISUAL STUDIO CODE

Visual Studio Code is a source-code editor made by Microsoft for Windows, Linux and macOS .Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality.

Visual Studio Code was first announced on April 29, 2015, by Microsoft at the 2015 Build conference. A preview build was released shortly thereafter.

2.1.1.2 PROGRAMMING LANGUAGE

- **HTML**

The **Hyper Text Markup Language**, or HTML is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.

Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

- **CSS**

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript. CSS is designed to enable the separation of presentation and content, including layout, colours, and fonts. This separation can improve content accessibility; provide more flexibility and control in the specification of presentation characteristics; enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, which reduces complexity and repetition in the structural content; and enable the .css file to be cached to improve the page load speed between the pages that share the file and its formatting.

2.1.1.3 WEB BROWSER- GOOGLE CHROME (FOR TESTING PURPOSE)

Google Chrome is a cross-platform web browser developed by Google. It was first released in 2008 for Microsoft Windows, built with free software components from Apple WebKit and Mozilla Firefox. It was later ported to Linux, macOS, iOS, and Android, where it is the default browser. The browser is also the main component of Chrome OS, where it serves as the platform for web applications.

2.1.1.4 FLASK FRAMEWORK

Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions.

2.1.2 FOR END-USERS:

- Web Browser like Google chrome (Latest Version) or Microsoft Edge (Latest Version)
- Internet Connectivity

2.2 Hardware Requirement

- **PROCESSOR**: The minimum or recommended CPU requirements, such as a multi-core processor (e.g., Intel i5 or above) for efficient model training and data processing.
- **MEMORY (RAM):** The recommended RAM, such as 8GB or 16GB, to handle the data processing tasks smoothly. For larger datasets, more RAM may be beneficial.
- **STORAGE**: The disk space required, especially if working with a large dataset of tweets or other text data. Generally, at least 10GB of free space is recommended for data storage and model saving.
- **GRAPHICS PROCESSING UNIT (GPU)** : If using deep learning models, a dedicated GPU (such as NVIDIA's GPUs) can significantly speed up model training.
- **INTERNET CONNECTIVITY:** The project involves real-time tweet analysis, internet connectivity is required for data collection and for deploying the web application online.

CHAPTER-3
PROBLEM DESCRIPTION

CHAPTER-3
PROBLEM DESCRIPTION

3.1 BACKGROUND

In today's digital age, social media platforms such as Twitter serve as public forums where customers frequently share their experiences, opinions, and grievances. For the airline industry, this user-generated content on social media is a valuable resource for understanding passenger sentiments about various aspects of service, including delays, in-flight experiences, and customer support. With the high volume of daily tweets, there is a need for automated sentiment analysis to help airlines gain real-time insights into customer satisfaction and service expectations, allowing them to adapt quickly to customer feedback.

3.2 PROBLEM STATEMENT

The primary problem addressed by this project is the need for an automated solution to analyze and classify the sentiment of airline-related tweets. Specifically, this project aims to categorize tweets as positive, negative, or neutral in sentiment using machine learning models. The challenge lies in processing,

3.3 CHALLENGES IN SENTIMENT ANALYSIS

- **DATA VOLUME**: Airlines receive massive volumes of tweets each day. Manually interpreting this data is unmanageable.
- **LANGUAGE VARIABILITY**: Tweets may contain informal language, abbreviations, or slang, making sentiment classification complex.
- **CONTEXT AND TONE**: Tweets often contain sarcasm, humor, or strong emotions, which can be challenging for sentiment analysis models to interpret accurately.
- **TIME SENSITIVITY**: Real-time insights into customer sentiment would help airlines respond quickly to emerging issues, although the current model does not operate in real time.

3.4 OBJECTIVES OF THE SOLUTION

- **AUTOMATED SENTIMENT CLASSIFICATION**: Develop a model to automatically classify tweets into positive, negative, or neutral sentiment, minimizing manual effort.
- **EFFICIENT DATA PROCESSING**: Use machine learning algorithms to analyze and process large datasets accurately.
- **INSIGHT GENERATION**: Provide airlines with data-driven insights to identify trends, address complaints proactively, and improve service quality.

3. 5 IMPACT OF THE SOLUTION

With automated sentiment analysis, airlines can better understand customer expectations and respond proactively to feedback. This project enables airlines to make data-driven decisions that enhance customer experience, strengthen brand loyalty, and improve operational strategies.

CHAPTER-4
LITERATURE SURVEY

CHAPTER-4
LITERATURE SURVEY

There is a vigorous improvement in the micro blogging websites as well as social networks. One of the major web destinations to the users is micro blogging websites, which are helpful for expressing the user's attitudes, opinions, and thoughts regarding various contexts. The most used social networking services and the micro blogging platform is twitter, which provides more data. At present, for the sentiment analysis of the user's opinions on the product, event, or context, researchers make use of social data. Moreover, the other name for sentiment analysis is opinion mining, which is the significant NLP task. This sentiment analysis defines orientation of sentiments related to text as either neutral, positive, or negative. Moreover, sentiment analysis represents the text analytics, computational linguistics, and NLP implementations for recognizing and categorizing the opinions of the user.

In general, the main intention of the sentiment analysis is to define the author's point of view concerning the similar context or the entire document's contextual polarity. The view can be either a user's judgment or assessment, affective state or the deliberated communication of emotion. In general, the classification of text expressions in source materials into facts and opinions is done by the sentiment analysis. Facts are the objective expressions regarding the events and their attributes as well as entities. The opinions are the subjective expressions of sentiments, emotions, feelings, events and attributes, and attitudes. This must be specified that not all the objective sentences include no opinions and not all subjective sentences include opinions. Thus, for sentiment analysis, it is significant for recognizing and extracting the facts and opinions from source materials. However, this seems to be quite complex for attaining precisely.

In recent times, important approaches are related to machine learning, rule-based, and the combination of both techniques. Machine learning models consist of conventional approaches like deep learning and conditional random field approaches, whereas the rule-based models consist of lexicon-based approach. Object detection, network optimization, image recognition, system security, sensor networks, and transportation are based on deep learning methods, which are mostly utilized in different fields. Several researchers have combined deep learning as well as machine learning

algorithms into text sentiment analysis by sentiment lexicon formulation and best results are obtained. The main aim of the sentiment lexicon-based model is to develop a sentiment lexicon, which is done by choosing suitable negative words, sentimental words, and degree adverbs. For the constructed sentiment lexicon, sentimental polarity and intensity are marked. Once the text is given as input, the words are matched with the sentiment words present in the sentiment lexicon, and those words are weighted and added for acquiring the input text's sentiment value, thus the determination of sentimental polarity is done as per the sentiment value. However, there are few approaches for acquiring the features of word vector related to the text like Glove, Word2Vec, and Fast Text automatically. However, the conventional machine learning models still require the emotional feature extraction of the structured information from the input text by text vectorization, human intervention, and later that algorithms are utilized for categorizing the sentiment of the text features.

The main contributions of this paper are portrayed as follows.

- To undergo a critical review of sentiment analysis under different applications.
- To carry out the detailed review of various sentiment analysis models based on the machine learning algorithms, types of data, tools, and different performance measures.
- To formulate the valuable research gaps and challenges based on the existing contributions under sentiment analysis.

CHAPTER-5
SOFTWARE REQUIREMENTS SPECIFICATION

CHAPTER-5
SOFTWARE REQUIREMENTS SPECIFICATION

5.1 FUNCTIONAL REQUIREMENT

Functional requirements define the basic system behaviour. Essentially, they are what the system does or must not do, and can be thought of in terms of how the system responds to inputs.
Functional requirements usually define if/then behaviours and include calculations, data input, and business processes.

5.1.1 DATA INPUT

- The system should allow users to input airline-related tweets either manually or through batch upload.
- The system should be able to handle pre-collected tweet data, supporting various file formats (e.g., CSV).

5.1.2 DATA PREPROCESSING

- The system should clean the input text data by removing punctuation, special characters, URLs, and stop words.
- It should tokenize the text data, convert it into numerical feature vectors using methods like TF-IDF or Bag of Words, and prepare it for model training and prediction.

5.1.3 SENTIMENT CLASSIFICATION

- The tool should classify each tweet as **positive**, **negative**, or **neutral** based on the trained machine learning models (Naive Bayes and Logistic Regression).
- It should display the predicted sentiment for each tweet in an organized manner.

5.1.4 MODEL TRAINING AND EVALUATION

- The system should allow for the training of Naive Bayes and Logistic Regression models on a labeled dataset.
- It should provide evaluation metrics (e.g., accuracy, precision, recall) to assess the model's performance.

5.1.5 USER INTERFACE

- A web interface (built with Flask) should be provided for users to interact with the system.
- The interface should enable users to enter text data, view sentiment results, and analyze multiple tweets simultaneously.

5.1.6 SYSTEM PERFORMANCE

- The tool should process and classify tweets efficiently, ensuring a smooth user experience.
- The system should handle large datasets without significant performance issues, although it is not required to operate in real time.

5.2 NON-FUNCTIONAL REQUIREMENT

Non-functional requirements outline essential characteristics and qualities of a system that contribute to its overall performance, usability, security, and reliability beyond core functionalities. In a sentiment analysis system, these requirements focus on the quality and effectiveness of the system's text analysis, aiming to provide accurate and efficient sentiment predictions while supporting a seamless user experience.

5.2.1 PERFORMANCE :

- The system should efficiently process text inputs and deliver sentiment predictions within a reasonable time frame, ensuring minimal delay for users.
- Response times for sentiment predictions should be optimized to handle concurrent requests, even during peak load conditions, without degrading performance.

5.2.2 ACCURACY :

- The sentiment analysis model should demonstrate high accuracy in classifying sentiments (e.g., positive, negative, neutral) from diverse text data, minimizing false positives and negatives.
- The Extensive testing and validation should be conducted across varied datasets and real-world scenarios to ensure consistent and reliable performance.

5.2.3 SCALABILITY :

- The system should support scaling, accommodating increased data volume and user requests without compromising response time or accuracy.
- Scalability should be considered in both the model training and inference stages, as well as in backend infrastructure and user interface handling.

5.2.4 USABILITY :

- The user interface should be straightforward, accessible, and intuitive, catering to users with diverse levels of technical expertise.
- The system should provide clear instructions, feedback, and error messages, helping users understand sentiment results and ensuring a smooth user experience.

5.2.5 RELIABILITY :

- The system should exhibit high reliability, minimizing downtime through redundancy and fault tolerance.
- Error handling and recovery strategies should ensure that critical functionalities remain available, with minimal interruptions in case of unexpected failures.

5.2.6 COMPATIBILITY:

- The system should be compatible with a wide range of devices, browsers, and operating systems, ensuring seamless access and functionality for users across different platforms.
- Compatibility with existing software systems and APIs should be considered to facilitate integration and insteroperability with other applications and services.

CHAPTER-6
SOFTWARE DESIGN

CHAPTER-6
SOFTWARE DESIGN

The software design for the *Sentiment Analysis Using NLP* project is structured to efficiently process and analyze user input, classify sentiment, and deliver results. It consists of a user interface for data input and result viewing, along with an admin interface for managing and retraining the model. The design includes a preprocessing module to clean and tokenize input text, a sentiment classification model for analyzing data, and a result module to store and display outcomes. This modular approach enables easy updates, allowing for future improvements in model performance and data handling. Together, these components create a user-friendly, adaptable, and scalable sentiment analysis system.

6.1 USE CASE DIAGRAM

The Use Case Diagram for the *Sentiment Analysis Using NLP* project illustrates the primary interactions between users, administrators, and the system. It defines two main actors: the **User**, who inputs data for sentiment analysis and views the results, and the **Admin**, who manages model training and updates.

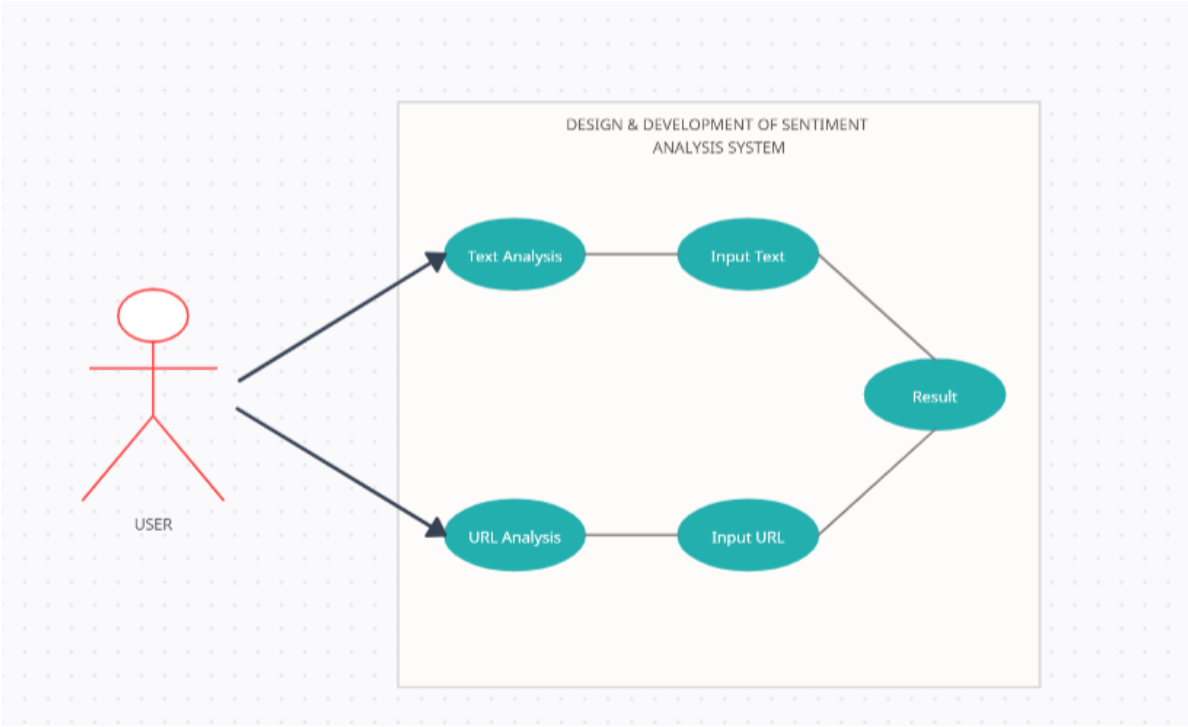


Figure 6.1: Use Case Diagram

The core use cases include **Input Data** for user-provided text, **Preprocess Data** to prepare input for the model, and **Classify Sentiment** for sentiment categorization (positive, neutral, or negative). The **View Results** use case allows users to see the sentiment output, while the **Train Model** and **Update Model** cases let admins enhance the model's accuracy and relevance. This diagram captures key functionalities and responsibilities, supporting a clear understanding of the system's user interactions and administrative capabilities.

6.2 PROJECT FLOW DIAGRAM

The Flow Diagram for the *Sentiment Analysis Using NLP* project begins with the **User** entering text, which is processed by the **Preprocessing Module**. The cleaned text is analyzed by the **Sentiment Analysis Model**, which classifies the sentiment. The result is stored and displayed to the user. The **Admin** can update and retrain the model to improve accuracy.

Figure 6.2: Flow Diagram

CHAPTER-7
MACHINE LEARNING MODULE

CHAPTER-7
MACHINE LEARNING MODULE

This section outlines the main components of the machine learning workflow used in the sentiment analysis project, including dataset details, preprocessing techniques, visualization methods, model description, and result analysis.

7.1 DATA SET DESCRIPTION

The dataset used in this project consists of **tweets related to airline services**, labeled by sentiment as **positive, negative, or neutral**. Each record in the dataset includes a tweet and its corresponding sentiment label, making it suitable for supervised machine learning tasks. The dataset was likely sourced from Kaggle, containing a significant number of samples to ensure the model can learn diverse patterns across different sentiments.

7.2 PRE-PROCESSING STEPS

Preprocessing is a crucial part of preparing the text data for analysis in any Natural Language Processing (NLP) task. For this sentiment analysis project, the preprocessing pipeline transforms raw tweet data into a cleaner, structured format that machine learning models can effectively analyze. Each step contributes to reducing noise and standardizing the input data, helping the model focus on meaningful information for sentiment classification.

7.2.1 TEXT CLEANING

- **Objective**: To remove irrelevant elements from the text that do not contribute to sentiment analysis
- **Actions**:
    - **Remove Punctuation and Special Characters**: These include symbols such as @, # , ! , etc., which do not hold significant meaning for sentiment but add to data complexity.
    - **Remove URLs**: Tweets often contain links, which are irrelevant to the sentiment expressed and can be removed to simplify the text.

- **Remove Numbers**: Numbers generally don't add value in sentiment analysis of tweets and are removed to focus on the words that convey emotion.

- **Impact**:

  This cleaning step reduces noise in the text, making it easier for the model to focus on relevant words and improving classification performance.

7.2.2 LOWERCASING

- **Objective**: To standardize the text by making it case-insensitive, as words in different cases (e.g., "Happy" vs. "happy") would otherwise be treated as separate words.
- **Actions**:
  - Convert all text to lowercase, ensuring consistency across the dataset.
- **Impact**:

  This step reduces the vocabulary size and prevents the model from treating words with the same meaning as different tokens, enhancing the model's ability to generalize.

7.2.3 TOKENIZATION

- **Objective**: To break down each tweet into individual words or tokens, making it easier to analyze.
- **Actions**
  - Split each tweet into tokens (words), allowing the model to understand each word's contribution to the sentiment of the text.
- **Impact**:

  Tokenization provides a structured format for the text, enabling feature extraction and further processing steps. Each token represents a unit of meaning that the model can use to identify sentiment patterns.

7.2.4 STOP WORDS REMOVAL

- **Objective**: To remove common words (such as "and," "the," "is") that do not carry significant meaning in terms of sentiment.
- **Actions**:
  - Use a predefined list of stop words (words that frequently appear but add little value to sentiment analysis) and remove them from the text.

- **Impact**:
  This step helps reduce the vocabulary size and focuses the model on sentiment-carrying words, leading to more accurate feature representations and improved model performance.

7.2.5 FEATURE EXTRACTION

- **Objective**: To convert the processed text data into numerical representations that machine learning models can interpret.
- **Actions**:
  - **TF-IDF (Term Frequency-Inverse Document Frequency)**: A technique that assigns weights to words based on their frequency in the document relative to their frequency across the entire dataset. This method helps the model focus on words that are more unique to each tweet
  - **Bag of Words (BOW)**: A simpler method that counts the occurrence of each word in a tweet, disregarding grammar and word order. BOW creates a vector of word counts for each tweet.
- **Impact**:
  Feature extraction transforms text into a format that models can analyze, enabling them to detect sentiment patterns. TF-IDF and BOW are common techniques that help identify the importance of specific words in sentiment expression.

7.3 DATA VISUALIZATION

Data visualization techniques were applied to explore the dataset and gain insights into the distribution and patterns within the data:

- **Sentiment Distribution:** A bar chart was used to show the proportion of positive, negative, and neutral tweets in the dataset, giving a quick overview of the sentiment balance.
- **Word Cloud:** Word clouds were generated for each sentiment category, highlighting the most frequently used words in positive, negative, and neutral tweets.
- **Length Distribution:** Visualizations were used to analyze the length of tweets across different sentiments, providing insights into the text's complexity and variety

7.3.1 ROC CURVE

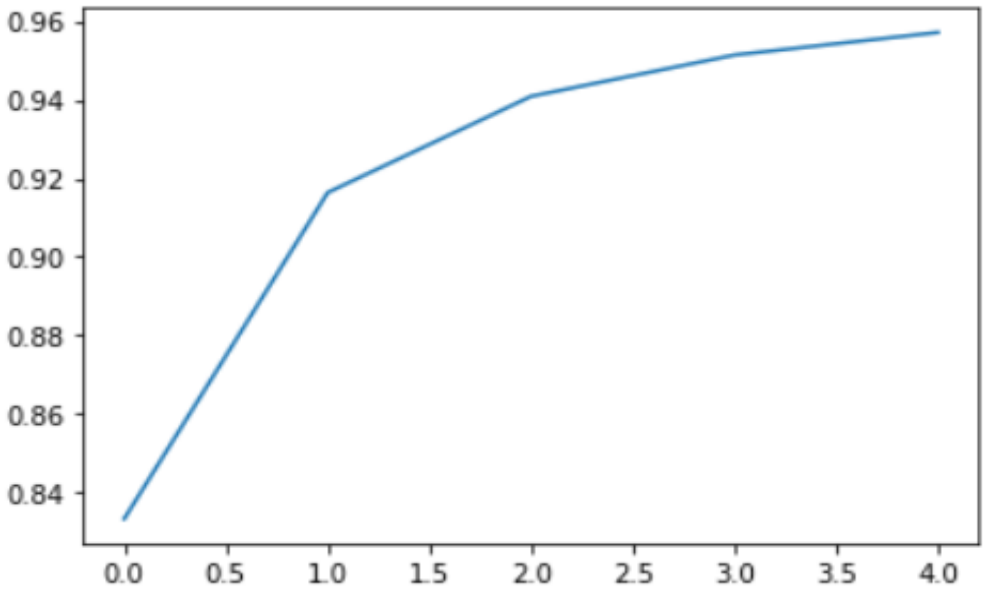


Figure 7.3.1: Accuracy

7.3.2 Confusion Matrix

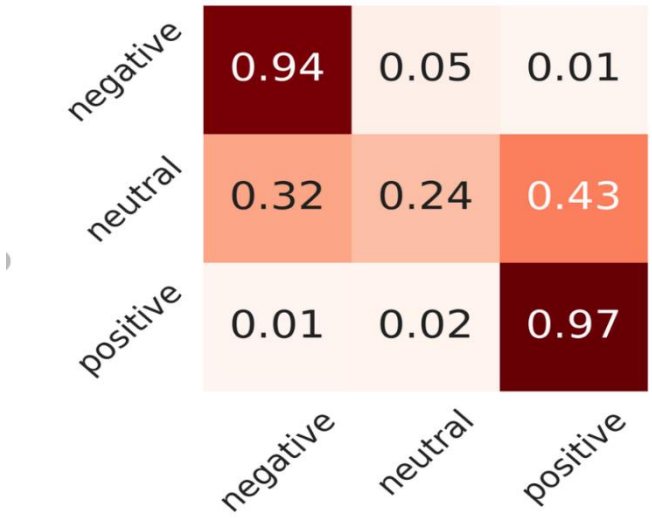


Figure 7.3.2: Confusion Matrix

7.4 ML MODEL DESCRIPTION

The machine learning models used in this project include Naive Bayes and Logistic Regression. Here's a brief description of each:

- **Naive Bayes**: A probabilistic classifier based on Bayes' Theorem, particularly effective for text classification tasks. It assumes that each word contributes independently to the probability of a tweet belonging to a particular sentiment class.
- **Logistic Regression:** A linear model used for binary and multiclass classification tasks. It calculates the probability of a tweet belonging to each sentiment class and assigns the label with the highest probability.

Each model was trained on the preprocessed dataset, using feature vectors created from the text data. Hyperparameters were optimized to maximize performance, and the models were evaluated on a test set to assess their ability to classify new data accurately.

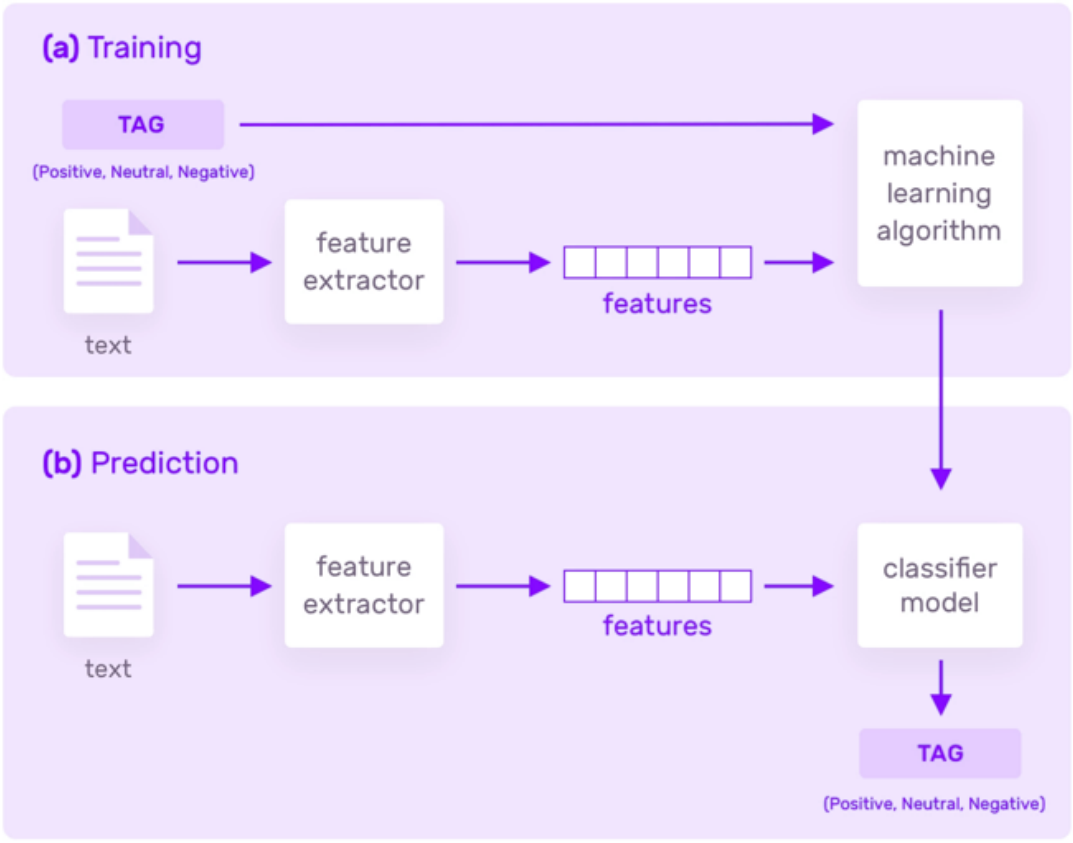


Figure 7.4: Model Workflow

7.5 RESULT ANALYSIS

The performance of the models was assessed based on various evaluation metrics:

- **Accuracy**: The proportion of correct predictions out of all predictions made, indicating the overall performance of the model.
- **Precision, Recall, and F1-Score**: These metrics provided insights into the model's ability to identify each sentiment accurately. Precision measures how many positive predictions were correct, recall measures how well the model captures actual positives, and F1-score balances precision and recall.
- **Confusion Matrix**: A confusion matrix was used to visualize the model’s performance across different sentiment classes, showing how many instances of each sentiment were correctly or incorrectly classified.

The results of each model were compared to determine the best-performing classifier for the task. Logistic Regression and Naive Bayes both showed promising results, though the best choice may vary based on specific airline sentiment analysis goals.

CHAPTER-8
FRONT END CONNECTIVITY

CHAPTER-8
FRONT END CONNECTIVITY

This chapter details the direct integration of the front-end interface with the sentiment analysis model, allowing users to input text for sentiment predictions seamlessly.

8.1 OVERVIEW

A user-friendly front end captures text inputs from users and displays sentiment results in real-time, ensuring ease of interaction with the sentiment analysis system.

8.2 TECHNOLOGIES USED

Technologies such as **HTML**, **CSS**, and **JavaScript** were utilized to create an intuitive interface. **Python** and a web framework like **Flask** (if applicable) serve to connect the front end with the backend logic directly.

8.3 DATA FLOW

The system processes data by capturing text input directly from the user interface, passing it to the backend model, and rendering the results on the front end.

8.4 USER INTERFACE ELEMENTS

The interface includes essential components such as text input fields, a “Submit” button, and an output area where sentiment analysis results (positive, negative, or neutral) are displayed.

8.5 INTEGRATION WITH MODEL

The sentiment analysis model is embedded within the application, directly processing inputs from the front end. Functions in the backend are invoked when users submit inputs, and results are then displayed on the same page.

8.6 ERROR HANDLING

Error messages are provided to users for invalid inputs or internal errors to enhance user experience and guide successful interactions.

8.7 SECURITY AND ACCESSIBILITY

Input sanitization is implemented to protect against potential security issues. Accessibility features are incorporated to ensure that all users can effectively interact with the system.

CHAPTER-9
PROJECT CODE

CHAPTER-9
PROJECT CODE

7.1 MODEL (model.ipynb)

```
import pandas as pd
import numpy as np
import os
import matplotlib.pyplot as plt
import cv2
from keras.models import Sequential, Model
from keras.layers import Conv2D, MaxPooling2D, Dense, Dropout, Flatten, Input,
BatchNormalization
from keras.optimizers import Adam
from keras.preprocessing.image import ImageDataGenerator
from sklearn.model_selection import train_test_split

# Load the data
path = r'C:\Users\Disha\Desktop\Design & Development of GenAge Detection System\Design &
Development of GenAge Detection System\crop_part1'
images = []
age = []
gender = []
target_size = (100, 100)  # Choose a smaller size

for img_file in os.listdir(path):
    ages = int(img_file.split("_")[0])
    genders = int(img_file.split("_")[1])
    img = cv2.imread(os.path.join(path, img_file))

    # Check for corrupt images
    if img is None:
        print(f"Corrupt image: {img_file}")
        continue

    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    img = cv2.resize(img, target_size)
    images.append(np.array(img))
    age.append(ages)
    gender.append(genders)

# Convert lists to numpy arrays
age = np.array(age, dtype=np.int64)
images = np.array(images) / 255.0  # Scale images
gender = np.array(gender, np.uint64)
```

```
# Split the data into training and testing sets
x_train, x_test, y_train_age, y_test_age, y_train_gender, y_test_gender = train_test_split(images,
age, gender, random_state=42)

# Data Augmentation
datagen = ImageDataGenerator(
    rotation_range=20,
    width_shift_range=0.1,
    height_shift_range=0.1,
    horizontal_flip=True,
    vertical_flip=True
)
datagen.fit(x_train)

# Define the shared convolutional base
input_shape = (100, 100, 3)
input_layer = Input(shape=input_shape)
conv1 = Conv2D(128, kernel_size=3, activation='relu')(input_layer)
conv1_bn = BatchNormalization()(conv1)
pool1 = MaxPooling2D(pool_size=3, strides=2)(conv1_bn)
conv2 = Conv2D(128, kernel_size=3, activation='relu')(pool1)
conv2_bn = BatchNormalization()(conv2)
pool2 = MaxPooling2D(pool_size=3, strides=2)(conv2_bn)
conv3 = Conv2D(256, kernel_size=3, activation='relu')(pool2)
conv3_bn = BatchNormalization()(conv3)
pool3 = MaxPooling2D(pool_size=3, strides=2)(conv3_bn)
conv4 = Conv2D(512, kernel_size=3, activation='relu')(pool3)
conv4_bn = BatchNormalization()(conv4)
pool4 = MaxPooling2D(pool_size=3, strides=2)(conv4_bn)
flatten = Flatten()(pool4)
dropout = Dropout(0.5)(flatten)
dense1 = Dense(512, activation='relu')(dropout)

# Define separate output layers for age and gender
output_age = Dense(1, activation='linear', name='age')(dense1)
output_gender = Dense(1, activation='sigmoid', name='gender')(dense1)

# Combine input and output layers into a single model
model = Model(inputs=input_layer, outputs=[output_age, output_gender])

# Compile the model with appropriate losses and metrics
optimizer = Adam(learning_rate=0.001)
model.compile(optimizer=optimizer,
          loss={'age': 'mse', 'gender': 'binary_crossentropy'},
          metrics={'age': 'mae', 'gender': 'accuracy'})

print(model.summary()
```

```
# Train the model
history = model.fit(datagen.flow(x_train, {'age': y_train_age, 'gender': y_train_gender},
batch_size=32),
                    steps_per_epoch=len(x_train) // 32,  # specify steps_per_epoch
                    validation_data=(x_test, {'age': y_test_age, 'gender': y_test_gender}),
                    epochs=5)

# Plot accuracy for age
plt.plot(history.history['age_mae'], label='Training Age MAE')
plt.plot(history.history['val_age_mae'], label='Validation Age MAE')
plt.xlabel('Epochs')
plt.ylabel('Mean Absolute Error')
plt.title('Training and Validation Age MAE')
plt.legend()
plt.savefig('age_accuracy.png')
plt.show()

# Plot accuracy for gender
plt.plot(history.history['gender_accuracy'], label='Training Gender Accuracy')
plt.plot(history.history['val_gender_accuracy'], label='Validation Gender Accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.title('Training and Validation Gender Accuracy')
plt.legend()
plt.savefig('gender_accuracy.png')
plt.show()

# Save the model
model.save('age_gender_model.h5')
```

7.2 FLASK INTEGRATION (app.py)

```
from flask import Flask, request, render_template
import re
import spacy
import joblib
import requests
from bs4 import BeautifulSoup

# Load the saved models
lr_model = joblib.load('lr_model.joblib')
le = joblib.load('label_encoder.joblib')
word_vectorizer = joblib.load('tfidf_vectorizer.joblib')

# Initialize the Flask app
app = Flask(__name__)

# Load spaCy model
```

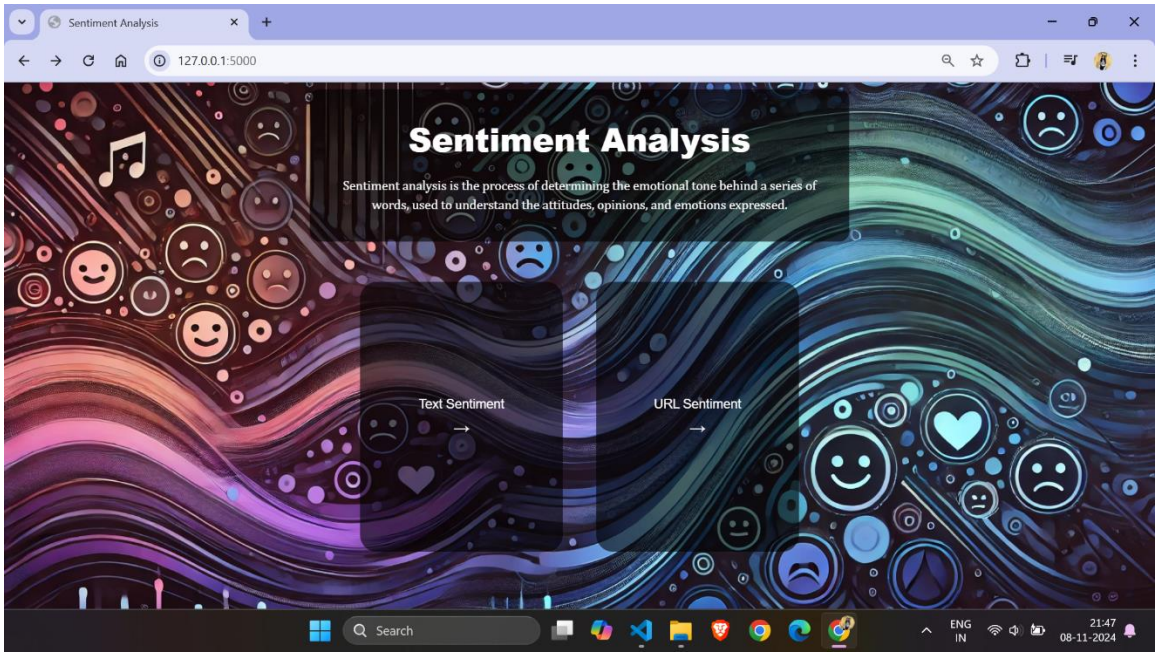
```
nlp = spacy.load('en_core_web_sm', disable=["tagger", "parser", "ner"])

# Define text cleaning function
def text_cleaner(text):
    text = re.sub(r'@[A-Za-z0-9]+', '', text)
    text = re.sub(r'http\S+', '', text)
    text = text.lower()
    text = re.sub("[^a-z]+", " ", text)
    text = re.sub("[\s]+", " ", text)
    doc = nlp(text)
    tokens = [token.lemma_ for token in doc if not token.is_stop]
    return " ".join(tokens)

# Define sentiment analysis function for individual comments
def sentiment_analyzer(comment):
    cleaned_comment = text_cleaner(comment)
    comment_vector = word_vectorizer.transform([cleaned_comment])
    label = lr_model.predict(comment_vector)
    return le.inverse_transform(label)[0]

# Scrape and analyze function for URL
def scrape_and_analyze(url):
    response = requests.get(url)
    soup = BeautifulSoup(response.text, 'html.parser')

    # Extract text from <p> tags
    paragraphs = soup.find_all('p')
    comments = [p.get_text() for p in paragraphs]

    # Count sentiments
    sentiment_counts = {'positive': 0, 'negative': 0, 'neutral': 0}
    for comment in comments:
        sentiment = sentiment_analyzer(comment)
        sentiment_counts[sentiment] += 1

    return sentiment_counts

# Home page route with tabs
@app.route('/description')
def description():
    return render_template('description.html')

@app.route('/')
def home():
    return render_template('home.html')

# Route for individual comment prediction
@app.route('/predict_text', methods=['GET', 'POST'])
```

```
def predict_text():
    if request.method == 'POST':
        tweet = request.form.get('tweet')
        if tweet:
            sentiment = sentiment_analyzer(tweet)
            return render_template('result.html', mode="text", input_text=tweet, sentiment=senti-
ment)
    return render_template('predict_text.html')

# Route for URL-based prediction
@app.route('/predict_url', methods=['GET', 'POST'])
def predict_url():
    if request.method == 'POST':
        url = request.form.get('url')
        if url:
            sentiment_counts = scrape_and_analyze(url)
            return render_template('result.html', mode="url", input_text=url, sentiment_counts=senti-
ment_counts)
    return render_template('predict_url.html')

if __name__ == '__main__':
    app.run(debug=True)
```

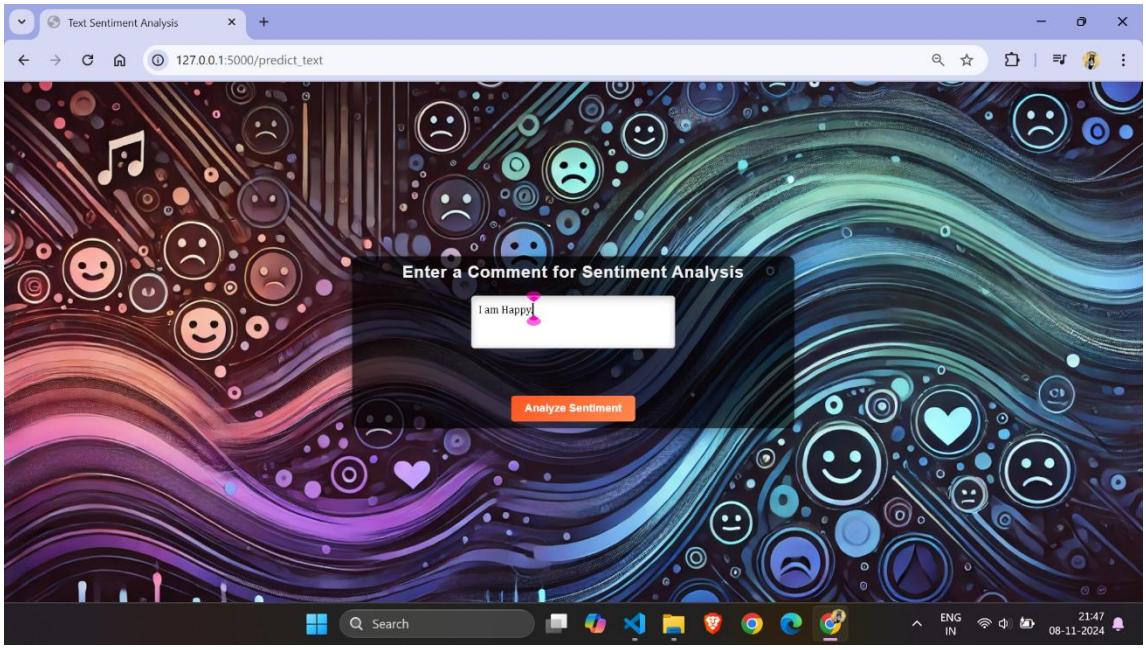
CHAPTER-10
OUTPUT SCREEN

CHAPTER-10
OUTPUT SCREEN

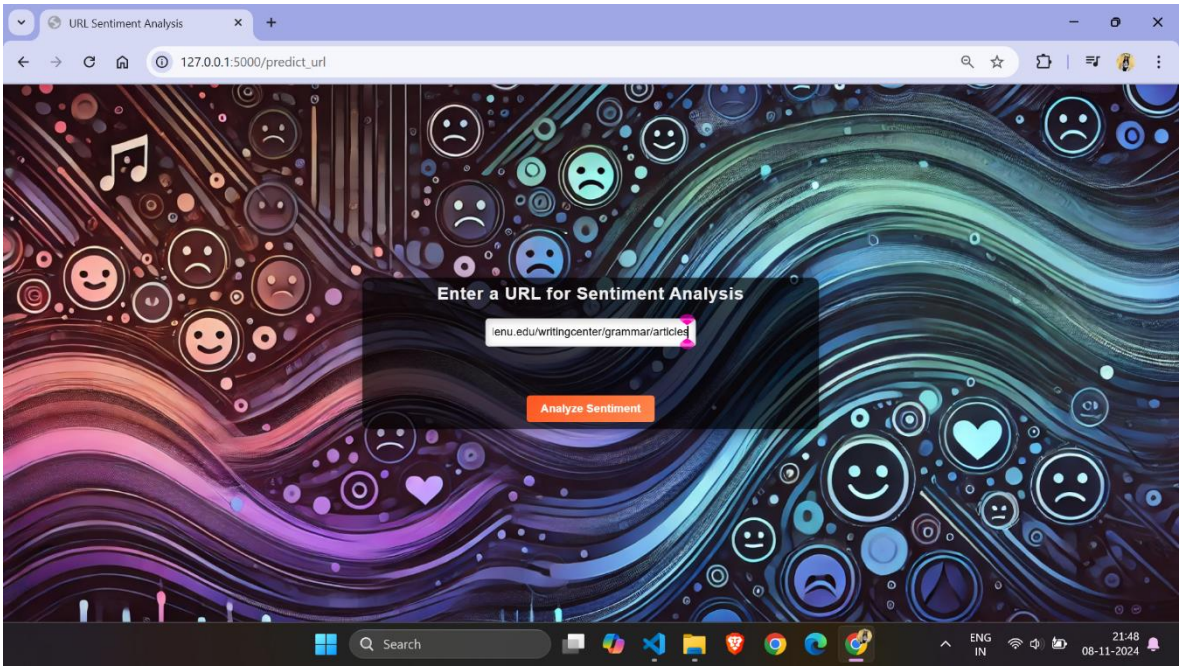
10.1 HOME PAGE:



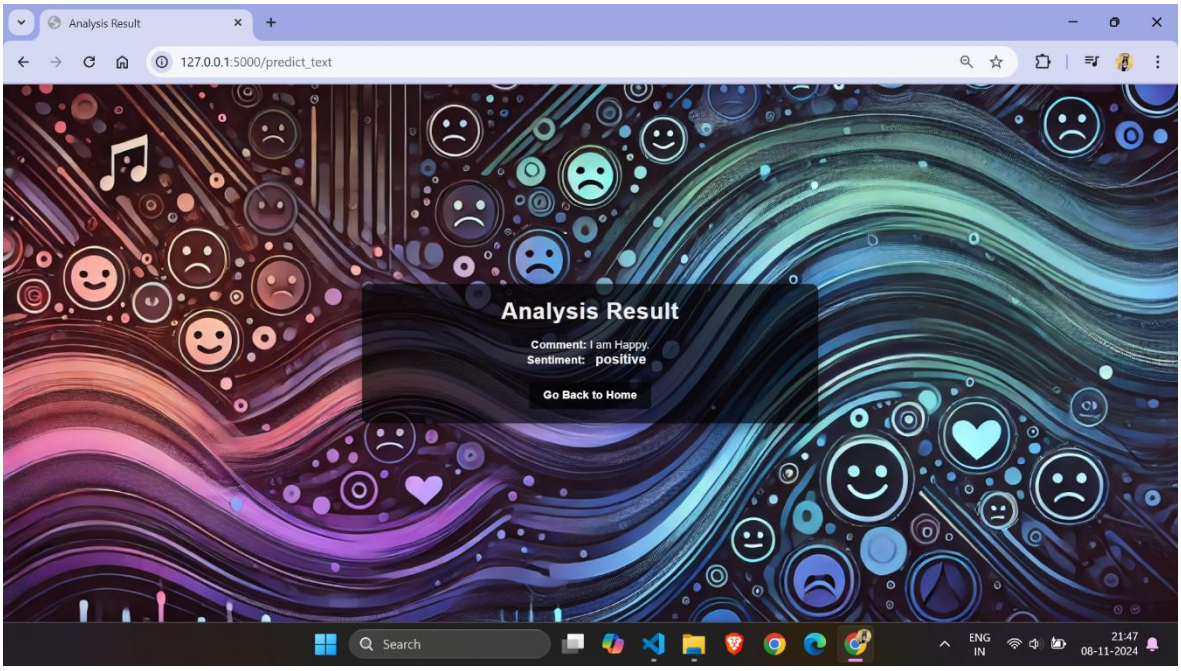
10.2 TEXT ANALYSIS PAGE:



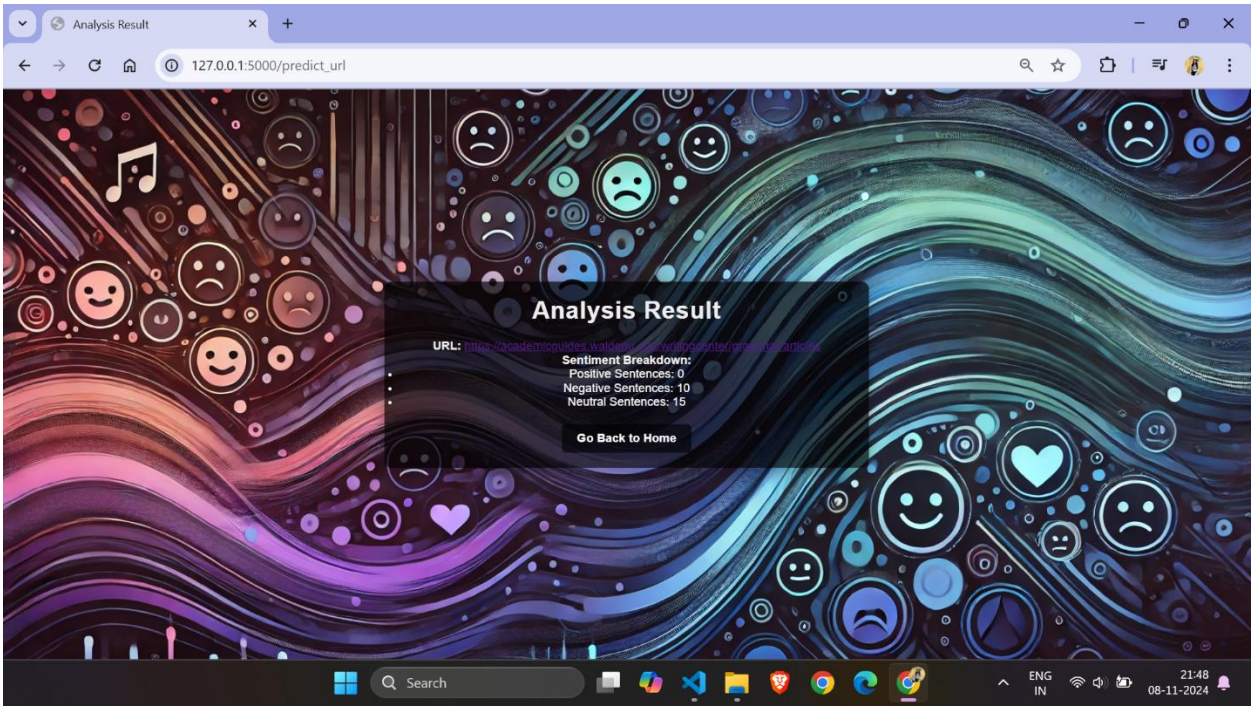
10.3 URL ANALYSIS PAGE:



10.4 RESULT PAGE OF TEXT ANALYSIS:



10.5 RESULT PAGE OF URL ANALYSIS:



REFERENCES

JOURNALS / RESEARCH PAPERS

1. G. Ravi Kumar, K. Venkata Sheshanna & G. Anjan Babu initials. (2020) *Sentiment Analysis Using Deep Learning Techniques*, ICMCSI. https://link.springer.com/chapter/10.1007/978-3-030-49795-8_75

WEBSITES (with exact URL up to page)

1. Dataset : https://www.kaggle.com/code/serkanp/sentiment-analysis-of-airline-tweets-and-comments
2. Natural Language Processing: https://spacy.io/api/doc

APPENDIX-1 GLOSSARY OF TERMS

(In alphabetical order)

B

Bag of Words (BOW)

A text representation method that converts a document into a set of word counts, disregarding grammar and word order but keeping word frequency**.**

C

CSS

Cascading Style Sheets is a style sheet language used for describing the presentation of a document written in a markup language such as HTML. CSS is a corner stone technology of the World Wide Web, alongside HTML.

D

Data Cleaning

The process of removing irrelevant or noisy elements from raw text data, such as punctuation, special characters, and stop words.

F

Flask

Flask is a lightweight Python web framework used in our sentiment analysis project to create a user-friendly interface. It allows users to input tweets or upload datasets and receive sentiment classification results (positive, negative, or neutral) directly on the platform.

H

HTML

HTML is the standard markup language used for creating web pages. In this project, HTML is used to build the structure of the web interface, enabling users to input tweets, upload datasets, and view sentiment classification results. HTML works

alongside Flask to display the sentiment analysis output in an organized and user-friendly format, making it easy to interact with the model.

L

Logistic Regression

A supervised learning algorithm used for binary and multiclass classification tasks. In this project, it is used to classify the sentiment of tweets.

M

MACHINE LEARNING

A field of artificial intelligence that uses statistical techniques to enable computers to learn from and make predictions based on data.

N

Natural Language Processing (NLP)

A field of artificial intelligence focused on the interaction between computers and human languages. NLP is used here to process and analyze text data from tweets

P

Precision

A model evaluation metric that measures the accuracy of positive predictions. It indicates the percentage of relevant positive instances among the predicted positive instances.

Preprocessing

The steps taken to prepare raw text data for analysis, including tokenization, removing stop words, and feature extraction**.**

R

Recall

A model evaluation metric that measures the ability to capture relevant instances. It indicates the percentage of relevant positive instances that were correctly classified.

S

Sentiment Analysis

The process of determining the emotional tone behind a series of words. In this project, sentiment analysis classifies tweets as positive, negative, or neutral.

T

Term Frequency-Inverse Document Frequency (TF-IDF)

A statistical method used to transform text into numerical features, reflecting the importance of a word in a document relative to its occurrence across multiple documents.

U

User Interface (UI)

The part of the application that allows users to interact with the system, view results, and input data.

V

Visual Studio Code

Visual Studio Code is a free and open-source code editor developed by Microsoft, widely used for its support of various programming languages and powerful features. In our sentiment analysis project, Visual Studio Code provides an efficient environment for coding, with helpful tools like syntax highlighting, code completion, debugging, and version control integration. Its extensive customization options through extensions make it ideal for developing and testing our project, enhancing productivity and flexibility throughout the development process.

PROJECT SUMMARY

About Project

Title of the project	**Sentiment Analysis System**
Semester	**7th**
Members	**1. Disha Vishwakarma** **2. Yuvika Sinha** **3. Priyam Sahu**
Team Leader	**Disha Vishwakarma**
Describe role of every member in the project	**Disha Vishwakarma: Frontend + Backend Developer** **Yuvika Sinha: Model Trainer** **Priyam Sahu: Model Trainer**
What is the motivation for selecting this project?	**We chose this project because it helps solve a problem, it's interesting, and it can be useful to people.**
Project Type (Desktop Application, Web Application, Mobile App, Web)	**Web Application**

Tools &Technologies

Programming language used	**Python**
Compiler used (withversion)	**NA**
IDE used (with version)	**Microsoft Visual Studios Code (1.51.1)**
Front End Technologies (with version, wherever Applicable)	**HTML & CSS**
Back End Technologies (with version, wherever applicable)	**Flask**
Database used (with version)	**NA**

Software Design& Coding

Is prototype of the software developed?	**NA**

SDLC model followed (Waterfall, Agile, Spiral etc.)	**Agile Methodology**
Why above SDLC model is followed?	**Agile model has a set of guidelines that are: small, highly motivated project team and supports changing requirements. We need both guidelines to develop our project.**
Justify that the SDLC model mentioned above is followed in the project.	**Since the demand (functionalities) of the website kept on changing every now and then therefore we used the Agile model, so that we could make desired changes whenever needed.**
Software Design approach followed (Functional or Object Oriented)	**Functional oriented approach**
Name the diagrams developed (according to the Design approach followed)	**Class diagram, Use Case diagram & Data Flow Diagram**
In case Object Oriented approach is followed, which of the OOPS principles are covered in design?	**NA**
No. of Tiers (example 3-tier)	**NA**
Total no. of front-end pages	**5**
Total no. of tables in database	**NA**
Database is in which Normal Form?	**NA**
Are the entries in database encrypted?	**NA**
Front end validations applied (Yes / No)	**Yes**
Session management done (in case of web applications)	**No**
Is application browser compatible (in case of web applications)	**Yes**
Exception handling done (Yes / No)	**No**
Commenting done in code (Yes / No)	**No**
Naming convention followed (Yes / No)	**Yes**
What difficulties faced during deployment of project?	**No**

Total no. of Use-cases	**5**
Give titles of Use-cases	**1. Home Page** **2. Text Analysis Page** **3. User Analysis Page** **4. Result Page of Text Analysis** **5. Result Page of URL Analysis**

Project Requirements

MVC architecture followed **(Yes / No)**	**No**
If yes, write the name of MVC architecture followed **(MVC-1, MVC-2)**	**NA**
Design Pattern used **(Yes / No)**	**No**
If yes, write the name of Design Pattern used	**NA**
Interface type **(CLI / GUI)**	**GUI**
No. of Actors	**1**
Name of Actors	**User**
Total no. of Functional Requirements	**6**
List few important non-Functional Requirements	**Accuracy, Maintainability, Performance, User-friendly**

Testing

Which testing is performed? **(Manual or Automation)**	**Manual**
Is Beta testing done for this project?	**No**

Write project narrative covering above mentioned points

Our project is a deep learning-based sentiment analysis system designed to evaluate text inputs and determine the underlying sentiment—positive, negative, or neutral. By leveraging advanced natural language processing and deep learning algorithms, the system efficiently preprocesses and extracts features from text data, enabling accurate sentiment predictions. This solution is adaptable to various applications in customer feedback analysis, social media monitoring, and market research. Focused on user-friendly interfaces and high accuracy, our project aims to enhance text analysis capabilities across diverse domains.

0187AD211013 - Disha Vishwakarma
0187AD211046 - Yuvika Sinha
0187AD211032 – Priyam Sahu

Guide Signature
Prof. Ruchi Jain
(Assistant Professor)