A PROJECT REPORT

on

TEXT SUMMARIZATION AND FEEDBACK CLASSIFICATION USING PYTHON

Submitted in partial fulfilment of the requirements for the award of the degree of

Bachelor of Technology

in

Computer Science and Engineering

by

TIRUVIDHULA DEEPIKA-O180156

Under the Guidance of

Mrs. Baby Lakshmi Prasanna M.Tech

Assistant Professor

Department of CSE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES

Ongole campus

2023-2024

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES

ONGOLE CAMPUS

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project entitled "**TEXT SUMMARIZATION AND FEEDBACK CLASSIFICATION USING PYTHON"** submitted by **TIRUVIDHULA DEEPIKA (O180156)** in partial fulfillment of the requirements for the award of the degree of the Bachelor of Technology in Computer Science and Engineering in **RGUKT Ongole** is a record of bona fide work carried out under my guidance and supervision during the academic year 2023-2024.

The results presented in this project have been verified and found to be satisfactory. The results embodied in this project report have not been submitted to any other University for the award of any other degree or diploma.

Mrs. BABY LAKSHMI PRASANNA M. Tech
Assistant Professor,
Department of CSE,
RGUKT ONGOLE.

Mr. B. SAMPATH BABU M. Tech (Ph. D)
Head of Department,
Department of CSE,
RGUKT ONGOLE.

APPROVAL SHEET

The report entitled **TEXT SUMMARIZATION AND FEEDBACK CLASSIFICATION USING PYTHON** by **TIRUVIDHULA DEEPIKA** (O180156) is approved for the degree of the Bachelor of Technology in Computer Science & Engineering.

Examiners: _____

_____

_____

Supervisor(s): _____

_____

_____

Date: _____

Place: _____

ACKNOWLEDGEMENT

It is our privilege to express a profound sense of respect, gratitude and indebtedness to our guide **Mrs. BABY LAKSHMI PRASANNA**, Assistant Professor, Department of Computer Science and Engineering RGUKT Ongole, for her indefatigable inspiration, guidance, cogent, discussion, constructive criticisms and encouragement throughout the dissertation work.

We express our sincere gratitude to **Mr. B. SAMPATH BABU,** Assistant Professor & Head of Department of Computer Science and Engineering, RGUKT Ongole for his suggestion motivations and co-operation for the successful completion of the work.

We extend our sincere thanks **Prof. B. JAYA RAMI REDDY** sir, Director, RGUKT Ongole for his encouragement.

With sincere Regards,

TIRUVIDHULA DEEPIKA

(O180156)

Date:

DECLARATION

We hereby declare that the project work entitles **"TEXT SUMMARIZATION AND FEEDBACK CLASSIFICATION USING PYTHON"** submitted to the **RGUKT Ongole** in partial fulfilment of the requirement for the award of the degree of Bachelor of Technology (B Tech) in Computer Science and Engineering is a record of an original work done by us under the guidance of **Mrs. BABY LAKSHMI PRASANNA,** Assistant Professor and this project work have not been submitted to any university for the award of any other degree or diploma.

Signature:

TIRUVIHDULA DEEPIKA (O180156)

Date: _____

Place: _____

ABSTRACT

In today's digital age, the enormous volume of textual data generated from feedback and reviews across various platforms presents a significant challenge for businesses and users seeking to summarize meaningful insights. This project focuses on leveraging Python for two primary tasks: Feedback Classification and Text summarization. The project begins by implementing a feedback classification system. Using machine learning and natural language processing techniques, it aims to classify feedback into predefined categories (e.g., positive, negative, neutral) to provide a quick understanding of sentiment distribution. Furthermore, the project delves into text summarization, a pivotal aspect for distilling lengthy feedback or reviews into concise, informative summaries. The main concentration is on URLs which contains lot of information to be summarized. Abstractive or summarizeive summarization techniques will be employed, where the former generates summaries by interpreting and paraphrasing the content, while the latter selects essential sentences to construct the summary.

CONTENTS

LIST OF FIGURES

CHAPTER-1
INTRODUCTION

The project with the name " TEXT SUMMARIZATION AND FEEDBACK CLASSIFICATION USING PYTHON" involves two major tasks Feedback Classification and Text summarization. Text summarization involves condensing large amounts of text into shorter yet coherent representations while preserving essential information. The project employs techniques to automatically generate summaries, providing a concise overview of lengthy documents, articles, or textual content. Feedback classification, which determines the opinion expressed in a piece of text, often categorized as positive, negative, or neutral. This aspect involves training a model to automatically categorize and classify feedback or reviews based on the opinion they convey.

Text summarization is beneficial in various fields including journalism, information retrieval, and content analysis. It allows users to obtain quick insights and an overview of lengthy documents, articles, or content, thus saving time and providing access to key information without reading through the entire text. The project involves the development and application of summarization techniques such as summarizeive methods (which select and combine existing sentences) or abstractive methods (which generate new sentences to represent the original text). Feedback classification using Python involves the process of categorizing and analyzing textual feedback or reviews to determine the sentiment or opinion expressed within the text. By utilizing various natural language processing (NLP) techniques, machine learning models, and sentiment analysis tools, the goal is to automatically classify and sort feedback into different sentiment categories, typically positive, negative, or neutral.

1.1 MOTIVATION:

The project for Text summarization and Feedback Classification using Python is motivated by the growing volume of textual data across various domains. The need to efficiently process and summarize key information from large text corpora, along with the desire to comprehend and categorize sentiments expressed in feedback, has led to the inception of this project. In today's digital landscape, vast amounts of textual data, reviews, and feedback are generated daily, making it challenging for individuals and organizations to quickly grasp critical insights. The motivation behind this project is to address these challenges by developing automated tools capable of summarizing extensive text content and categorizing sentiments within feedback.

1.2 PROBLEM DEFINITION

The project confronts two significant challenges in dealing with the escalating volume of textual data. The first challenge revolves around the need to distill key information from extensive text sources efficiently. This task demands automated text summarization techniques to summarize essential content while maintaining its context and significance. The large text in the URLs aren't actually summarized which are very lengthy to summarize and consists of many complexities. The second challenge lies in deciphering and categorizing sentiments within textual feedback or reviews. Interpreting and classifying sentiments expressed in user feedback or reviews, whether positive, negative, or neutral, is crucial for understanding customer perspectives and making informed decisions based on these sentiments

1.3 OBJECTIVE OF THE PROJECT:

The primary objective of this project is to develop robust and efficient tools for automated text summarization and feedback classification. Implement automated techniques to summarize extensive text, generating concise and informative summaries while retaining the most crucial information and context. The large text in the URLs aren't actually summarized which are very lengthy to summarize and consists of many complexities. So the text in URL is summarized into short sentences. Facilitate improved customer service, market analysis, and content understanding by automating the process of summarizeing critical insights from user-generated content. Provide accessible and automated solutions to quickly summarize essential information from text sources, aiding decision-making processes across various industries.

CHAPTER-2
LITERATURE REVIEW

[1] Kleinberg J. M., “Authoritative sources in a hyperlinked environment”. Journal of the ACM, Volume 46 issue 5, pp.604–632, Sep 1999.

[2]Vinodhini, G.; Chandrasekaran, R. Feedback analysis and opinion mining: A survey. Int. J. 2012,2, 282–292.

[3]Yuan, Y.; Zhou, Y. Twitter feedback analysis with recursive neural networks. In CS224D Course Project; Stanford University: Stanford, CA, USA, 2015

[4] Hemalatha, I.; Varma, G.; Govardhan, A. Automated Feedback Analysis System Using Machine Learning Algorithms. IJRCCT 2014,3, 300–303

[5] Deepali K. Gaikwad, C. Namrata Mahender, "A Review Paper on Text Summarization", “International Journal of Advanced Research in Computer and Communication Engineering”. Vol.5, Issue 3, March 2016.

[6] Neelima Bhatia and Arunima Jaiswal, "Automatic Text summarization and its Methods-A Review", 6th International Conference. Cloud System and Big Data Engineering, 2016.

[7] Li, Ailin, et al. "The Mixture of Text rank and Lexrank Techniques of Single Document Automatic Summarization Research in Tibetan." 2016 8th International Conference on Intelligent Human Machine Systems and Cybernetics (IHMSC). Vol. 1. IEEE, 2016.

[8] Pratibha Devihosur, Naseer R. "Automatic Text Summarization Using Natural Language Processing" International Research Journal of Engineering and Technology (IRJET) Volume: 04 Issue: 08, Aug-2017.

[9] Devika, M.; Sunitha, C.; Ganesh, A. Feedback Analysis: A Comparative Study on Different Approaches. Procedia Comput. Sci.2016,87, 44–49.

[10] Yousefpour, A.; Ibrahim, R.; Hamed, H.N.A. Ordinal-based and frequency-based integration of feature selection methods for feedback analysis. Expert Syst. Appl. 2017,75, 80–93.

CHAPTER-3
METHODOLOGY

3.1 EXISTING SYSTEM

In present days, the text summarization techniques aren't actually very accurate. The current system only generates the summary as first few lines of the large text. Abstractive summarization aims to generate new phrases to summarize content, but it might sometimes produce grammatically incorrect or semantically inaccurate sentences, affecting the quality of the summary. Summarization often requires condensing a large amount of text into a much shorter version. As a result, important details might be lost or distorted during the compression process. As well as the feedback data can be noisy, containing spelling mistakes, abbreviations, slang, or grammatical errors. Handling such noisy data can impact the accuracy of analysis. So current systems aren't so favorable to analyze the text and feedback.

3.2 PROPOSED SYSTEM

The proposed system aims to integrate advanced natural language processing techniques using Python for the dual tasks of text summarization and feedback classification. Leveraging Python's rich ecosystem of libraries and machine learning frameworks, the system intends to implement cutting-edge algorithms to distill lengthy textual feedback into concise summaries while categorizing feedback sentiments into predefined classes. Through the application of robust models and methodologies, this system endeavors to offer accurate, efficient, and scalable solutions for text summarization and feedback analysis in diverse feedback datasets. Implementation of Natural Language Processing techniques to preprocess raw text data, including tokenization and using various python libraries and modules to implement these major tasks.

3.3 SOFTWARE REQUIREMENT SPECIFICATION

Our project was created using the Python programming language. The text embedded in the URL is summarized and the feedback is classified. We employ several hardware requirements as well as software languages in this project. They are the following: keyboard, monitor, hard disk, RAM, and ethernet connection. Editors like Jupyter Notebook, Google Colab are used to build all of these through coding, and "chrome/Firefox" is the preferred browser program for visualizing our project.

3.3.1 TECHNOLOGIES USED

PYTHON:

In the "Text summarization and Feedback Classification using Python" project, Python's machine learning capabilities may be employed in various ways to enhance text summarization and feedback analysis. Sentiment Analysis Models: Machine learning models, like classifiers or neural networks, might be utilized for sentiment analysis in feedback. These models can learn to identify sentiment from textual data. Text summarization Models: Machine learning algorithms, such as Natural Language Processing (NLP) models, could be used for automatic text summarization. These models learn to summarize the most essential information from a body of text. Feature Engineering: Machine learning techniques might be employed for feature engineering, which involves transforming text data into suitable formats for analysis or modelling. Model Training: Training machine learning models to recognize patterns, sentiments, or essential information within text data, allowing for the automatic summarization of insights.

VISUAL STUDIO CODE:

Visual Studio Code (VS Code) plays a pivotal role in text summarization and feedback classification projects using Python. As an intuitive and versatile code editor, it provides a robust environment for developers to create, modify, and debug Python scripts efficiently. Its extensive array of extensions empowers users with diverse functionalities, including support for various Python libraries essential for natural language processing tasks. For text summarization, VS Code offers a streamlined interface for writing and organizing code. Its syntax highlighting, IntelliSense, and debugging capabilities enhance coding accuracy and speed. Developers can seamlessly integrate libraries such as NLTK or spaCy to preprocess text, generate summaries, and fine-tune models, all within the editor. In feedback classification projects, VS Code's flexibility enables the implementation and experimentation with machine learning models. Its integration with Jupyter Notebooks and extensions like Jupyter provides a convenient platform for interactive development and visualization, crucial for training and evaluating classification algorithms.

GOOGLE COLAB:

Google Colab, a part of Google's Colaboratory, is an online platform that offers a free, cloud-based development environment for running Python code. It provides an easy and accessible space to execute Python scripts, run Jupyter notebooks, and work with data science and machine learning projects. Google Colab provides a Python environment with a set of pre-installed libraries like Pandas, NumPy, Matplotlib, and even the ability to install additional libraries like NLTK or Plotly for natural language

processing and visualization. Projects can be developed using Jupyter notebooks, allowing code to be executed in cells and combined with explanatory text, making it easier to document and share code with others. Colab offers free access to GPUs and TPUs, which can significantly accelerate machine learning computations.

3.3.2 MODULES

We divide our project into different modules and methodologies. The modules are:

- Beautifulsoup4
- Urllib.request
- re
- collections
- PIL (Python Imaging Library)
- io
- pandas
- matplotlib
- tkinter

MODULE DESCRIPTION

Beautifulsoup4 module:

BeautifulSoup4, commonly imported as bs4, is a Python library used for pulling data out of HTML and XML files. It facilitates the parsing, navigating, and searching of structured data retrieved from web pages. Used for web scraping and parsing HTML or XML content. Provides tools to summarize specific data from web documents or markup files. Transforms raw HTML or XML content into a parse tree of Python objects. Enables navigation through the parsed tree structure to access elements and their attributes. Facilitates locating elements based on attributes, tags, or their hierarchical position. Helps summarize relevant information or content from web pages for analysis or manipulation.

Urllib.request Module:

It is a Python module that facilitates sending HTTP requests to retrieve data from URLs. It offers functionalities to handle various aspects of network communication. The module supports common HTTP request methods like GET, POST, PUT, DELETE, etc., and it can interact with web servers to fetch data. It's primarily utilized to fetch content from web pages, APIs, and other online resources.

urllib.request allows developers to: Open and read URLs. Send HTTP requests with custom headers for better interaction with web servers. Handle URL redirections. Retrieve web content and download files from URLs.

Re Module:

The re module in Python is used for working with regular expressions (regex). It provides a set of functions that allow developers to perform various operations involving pattern matching within strings. Key features and functionalities of the re module include: Enables the creation and utilization of regular expressions to identify specific patterns within strings. Supports various string operations such as search, substitution, splitting, and more. Allows searching for patterns within strings using methods like search() and match().

Collections Module:

The Counter class from the collections module in Python is used extensively in the text summarization and feedback classification project. It serves as a powerful tool for tallying word frequencies in the summarizeed text. By converting a list of words into a dictionary-like structure, it efficiently counts the occurrences of each word. This enables the identification of the most common words within the text, aiding in the summarization process by selecting sentences that contain these prevalent words.

PIL Module:

The Python Imaging Library (PIL) module, now known as Pillow, serves a crucial role in text summarization and feedback classification projects. It facilitates image processing tasks that complement these textual analyses. Although primarily an image-handling library, PIL/Pillow can be employed to visualize text data or convert text-based visualizations into image formats. In text summarization, PIL/Pillow can convert textual outputs into image files, enabling the representation of summarization results through graphs, word clouds, or other visual summaries.

IO Module:

The io module in Python is employed in the text summarization and feedback classification project to manage streaming data, specifically for image handling and manipulation. Within this project, the io.BytesIO class is used to handle image data in memory, facilitating the conversion of images to a format that can be displayed in the graphical user interface. It assists in the process of converting images to a compatible format for presentation within the GUI, enhancing the visualization of feedback analysis results, particularly in creating and displaying the pie chart depicting feedback percentages.

Pandas Module:

The pandas module in Python is a powerful library designed for data manipulation, analysis, and handling structured data. It provides high-performance, easy-to-use data structures and tools for working with tabular or labeled data. Offers the primary data structure known as DataFrame, a two-dimensional, size-mutable, and potentially heterogeneous tabular data structure with labeled axes (rows and columns). Enables operations for data cleaning, reshaping, slicing, indexing, merging, joining, and aggregating datasets.

Matplotlib Module:

The Matplotlib module in Python is instrumental in text summarization and feedback classification projects. It enables developers to create visual representations of data, facilitating a deeper understanding of text analysis results. In text summarization, It allows for graphical representations of word frequency, sentence importance, or summarization evaluation metrics, aiding in the analysis and fine-tuning of summarization models. In feedback classification tasks, Matplotlib helps in visualizing classification results, such as confusion matrices, precision-recall curves, or ROC curves.

Tkinter Module:

The tkinter module in Python serves as the fundamental framework for creating the graphical user interface (GUI) in the text summarization and feedback classification project. It provides a set of tools and widgets for building the application's interface, enabling the creation of windows, buttons, text entry fields, labels, and more. tkinter facilitates user interaction by allowing users to input URLs, trigger functions for summarizing text and analyzing feedback, and display results within the application window.

METHODOLOGIES:

- Web Scraping
- Text Cleaning
- Word Frequency Analysis
- Text summarization
- Feedback Analysis
- GUI Development
- Data Visualization

Web Scraping:

Web scraping methodology is employed to summarize information from web pages for text summarization and feedback analysis. The project utilizes web scraping to summarize content from web pages, specifically targeting text-based information necessary for the text summarization process and feedback analysis. Leveraging the BeautifulSoup library, the project accesses and parses HTML content to navigate and summarize relevant text from web pages. URL Retrieval: Users can input a URL, allowing the program to access the designated webpage for data summarization. Text Content Processing: The summarizeed data undergoes preprocessing steps, including formatting, cleaning, and transforming into a usable text format for further analysis.

Text Cleaning:

In the text summarization and feedback classification project, text cleaning is a crucial methodology employed to refine the summarizeed text from various sources. This process involves using regular expressions (regex) to filter and refine the text data. Specifically, the cleaning methodology focuses on removing non-alphabetic characters and standalone numbers from the text. By utilizing regex patterns, the method systematically eliminates any characters that are not part of the alphabet and isolates individual digits that are not part of a word, streamlining the text content for subsequent analysis.

Word Frequency Analysis:

The word frequency analysis methodology within the "Text summarization and Feedback Classification using Python" project involves analyzing the frequency of words in the summarizeed text data. This analysis assists in the text summarization process and feedback sentiment analysis. Tokenization, the summarized text is tokenized into individual words or tokens. Stopword Removal, commonly occurring and less meaningful words, known as stopwords, are removed to focus on significant terms. Word Frequency Count, the remaining words' occurrences are counted to determine their frequency within the text.

Text summarization:

Text summarization is a methodology employed in the text summarization and feedback classification project to condense and summarize the most significant information from a larger body of text. In this project, the text summarization methodology primarily involves the identification and summarization of crucial information or key points from the provided text, often obtained from a URL. This is achieved by analyzing the frequency of words in the text and selecting sentences that contain the most

common words. By determining the most prevalent words and sentences, the algorithm condenses the content into a shorter, more concise summary. The methodology aims to provide an abridged version of the original text, enabling users to grasp the primary content and meaning without going through the entirety of the text.

Feedback Analysis:

Text summarization and Feedback Classification using Python" project, the feedback analysis methodology involves the evaluation of sentiments expressed within textual feedback or reviews using few analysis techniques. Data Collection, feedback data from various sources is obtained, such as product reviews or customer comments. Feedback Analysis, feedback analysis techniques are applied to assess the sentiment expressed in the feedback. This can involve using tools like NLTK's VADER for feedback scoring. Sentiment Scoring, sentiments within the feedback are quantified, typically as positive, negative, or neutral, by analyzing the tone and language used in the text. Visualization, visual representations, such as pie charts or graphs, may be used to display the sentiment distributions within the feedback. Insight Generation, analysis results help in understanding the overall sentiment or opinions expressed by users.

GUI Development:

GUI (Graphical User Interface) development in the text summarization and feedback classification project involves creating a user-friendly interface to interact with the functionalities of the application. The methodology focuses on utilizing the Tkinter module in Python to build a visual platform where users can input a URL for text summarization, select products for feedback analysis, and view the results. Through the GUI, users can access text summarization features, initiate feedback analysis, and receive summarized text and feedback results in a user-friendly and intuitive environment.

Data Visualization:

In the "Text summarization and Feedback Classification using Python" project, the visualization methodology is utilized to represent and communicate insights obtained from the processed text data and feedback analysis. Plotly Library: The project employs the Plotly library for generating interactive visualizations like pie charts, bar graphs, or other types of plots. Data Representation: Processed data or analysis results, such as sentiment distributions or summarized information, are represented visually. Interactive Graphs: Plotly enables the creation of interactive graphs that can be explored, customized, and used for data presentation. Insight Communication: Visualizations aid in conveying insights

derived from the summarization process or sentiment analysis in a more interpretable and understandable format.

3.4 SAMPLE CODE

```
import requests
from bs4 import BeautifulSoup
import re
from collections import Counter
import pandas as pd
import tkinter as tk
from PIL import Image, ImageTk
import matplotlib.pyplot as plt
from io import BytesIO
def get_text_from_url(url):
try:
response = requests.get(url)
soup = BeautifulSoup(response.content, "html.parser")
text = ' '.join([p.get_text() for p in soup.find_all('p')])
return text
except requests.exceptions.RequestException as e:
return f"Error: {e}"
"""def clean_text(text):
text = re.sub(r'[^a-zA-Z\s]', '', text)
return text"""
def clean_text(text):
text = re.sub(r'[^a-zA-Z\s]', '', text)  # Remove non-alphabetic characters
text = re.sub(r'\b\d+\b', '', text)  # Remove standalone numbers
return text
def word_frequencies(text):
words = text.split()
word_freq = Counter(words)
return word_freq
def summarize_url(url, num_sentences=10):
```

```
text = get_text_from_url(url)
cleaned_text = clean_text(text)
word_freq = word_frequencies(cleaned_text)

most_common_words = word_freq.most_common(20)
most_common_words = [word[0] for word in most_common_words]
sentences = re.split(r'(?<=[.!?])\s+', text)
selected_sentences = []
for sentence in sentences:
for word in most_common_words:
if word in sentence:
selected_sentences.append(sentence)
break
if len(selected_sentences) >= num_sentences:
break
cleaned_summary = [re.sub(r'\[\d+\]', '', s) for s in selected_sentences]
summary = ' '.join(cleaned_summary)
return summary
def simple_sentiment_analysis(text):
positive_words = ["good", "great", "excellent"]  # Add more positive words
negative_words = ["bad", "terrible", "poor"]  # Add more negative words
text_lower = text.lower()
words = text_lower.split()
num_positive = sum(word in positive_words for word in words)
num_negative = sum(word in negative_words for word in words)
if num_positive > num_negative:
return "Positive"
elif num_positive < num_negative:
return "Negative"
else:
return "Neutral"
def display_results():
url = entry_url.get()
text_summary = get_text_from_url(url)
```

```
text_summary_widget.insert(tk.END, f"\nSummary of the text from the provided
URL:\n{text_summary}\n")
def analyze_feedback():
product_name = variable.get()
selected_product = data[data['Product'] == product_name]
if not selected_product.empty:
selected_product["Score"] = selected_product["ReviewBody"].apply(simple_sentiment_analysis)
Mno = selected_product[selected_product.Score == "Positive"]["Score"].count()
Fno = selected_product[selected_product.Score == "Negative"]["Score"].count()
Nno = selected_product[selected_product.Score == "Neutral"]["Score"].count()
total = Mno + Fno + Nno
positive_percent = (Mno / total) * 100
negative_percent = (Fno / total) * 100
neutral_percent = (Nno / total) * 100
feedback_widget.delete(1.0, tk.END)  # Clear previous text
if not total == 0:
feedback_widget.insert(tk.END, f"Positive Comments: {positive_percent:.1f}%\n"
f"Negative Comments: {negative_percent:.1f}%\n"
f"Neutral Comments: {neutral_percent:.1f}%\n")
fig, ax = plt.subplots()
ax.pie([positive_percent, negative_percent, neutral_percent], labels=["Positive", "Negative", "Neutral"],
autopct='%1.1f%%')
ax.axis('equal')
buf = BytesIO()
fig.savefig(buf, format='png')
buf.seek(0)
image = Image.open(buf)
pie_image = ImageTk.PhotoImage(image)
feedback_label = tk.Label(root, image=pie_image)
feedback_label.image = pie_image  # Keep a reference
feedback_label.pack()
else:
feedback_widget.insert(tk.END, "No reviews found for the selected product.\n")
else:
```

```
feedback_widget.insert(tk.END, "Product not found in the dataset. Please enter a valid product
name.\n")
root = tk.Tk()
root.title("Text summarization & Feedback Classification")

label_url = tk.Label(root, text="Enter the URL you'd like to summarize:")
label_url.pack()
entry_url = tk.Entry(root, width=50)
entry_url.pack()
summary_button = tk.Button(root, text="Get Summary", command=display_results)
summary_button.pack()
text_summary_widget = tk.Text(root, height=10, width=50)
text_summary_widget.pack()
data = pd.read_csv(r"C:\Users\varun\Documents\AllProductReviews.csv")  # Replace with the path to
your CSV file
unique_products = data['Product'].unique()
variable = tk.StringVar(root)
variable.set("Select a Product")  # default value
product_label = tk.Label(root, text="Select a product for feedback classification:")
product_label.pack()
product_dropdown = tk.OptionMenu(root, variable, *unique_products)
product_dropdown.pack()
analyze_button = tk.Button(root, text="Analyze Feedback", command=analyze_feedback)
analyze_button.pack()
feedback_widget = tk.Text(root, height=5, width=50)
feedback_widget.pack()
root.mainloop()
```

3.5 UML DIAGRAMS:

The abbreviation UML stands for Unified Modelling Language. In the area of object-oriented software engineering, UML is a general purpose modeling language that has been standardized. The Object Management Group oversees and developed the standard. The creation of a common modeling language for object-oriented software engineering is a key objective of UML. UML now consists of two main parts: a notation and a meta-model. In the future, UML might also be coupled with or added to in the form of a method or process. The Unified Modeling Language is a standard language used for business modeling, non-software systems, and specifying, building, and documenting the artifacts of software systems.

UML, or Unified Modeling Language, is the acronym. UML is a general purpose modeling language that has been standardized in the field of object-oriented software engineering. The standard was created and is supervised by the Object Management Group. Establishing a common modeling language for object-oriented software engineering is one of UML's key objectives. The two core elements of UML today are the notation and the meta-model. A technique or procedure may be added to or improved upon in the future in addition to UML. For business modeling, non-software systems, as well as for describing, producing, and documenting the artifacts of software systems, the Unified Modeling Language is used as a standard language.

The following are the main objectives in the UML's design:

1. Offer users an expressive visual modeling language that is ready to use so they can create and trade meaningful models.
2. Offer methods for specialization and extendibility to expand the fundamental ideas.
3. Be unreliant on specific development methodologies and programming languages.
4. Promote the commercial expansion of OO tools.
5. Encourage the integration of best practices and support higher level development ideas like collaboration, frameworks, patterns, and components.

The goal of a UML Diagram, which is based on the UML (Unified Modeling Language), is to graphically represent a system together with its primary players, roles, actions, objects, or classes in order to better understand, edit, maintain, or document system-related information. Structural and behavioral UML diagrams make up the UML diagrams.

STRUCTURAL UML DIAGRAMS:

Static representations of a system's structure are shown in structural diagrams. It is frequently employed in software architecture documentation. Developers and stakeholders can better understand and convey the system architecture, linkages, and interactions between various classes thanks to these diagrams, which provide a visual depiction of a system's static structure. They are:

- Class Diagram
- Object Diagram
- Component Diagram
- Composite Structure Diagram
- Deployment Diagram
- Package Diagram
- Profile Diagram

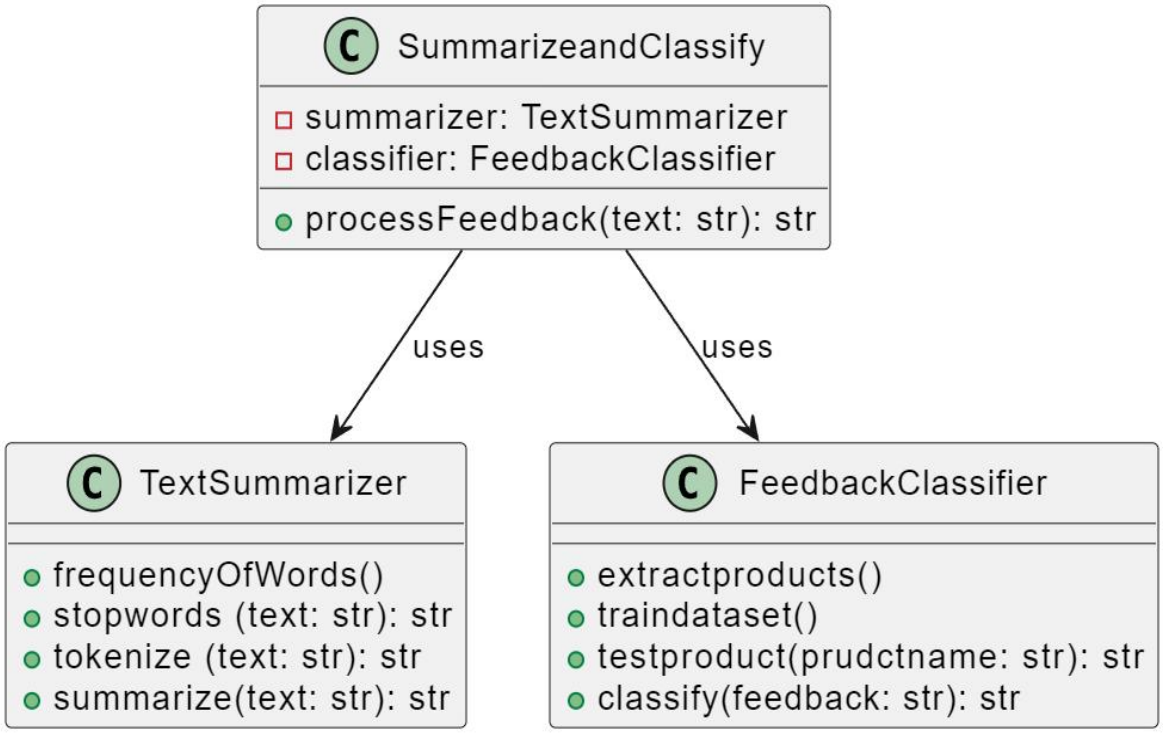
BEHAVIOURAL UML DIAGRAM:

A dynamic picture of a system or the behavior of a system, which explains the operation of the system, is provided by behavioral diagrams. These illustrations aid in explaining how objects interact, react to circumstances, and change states. They support the design, analysis, and validation of the system, ensuring that it operates as intended and satisfies the required specifications. There are seven diagrams. They are:

- Use case Diagram
- Sequence Diagram
- Activity Diagram
- State Machine Diagram
- Interaction Overview Diagram
- Communication Diagram
- Timing Diagram

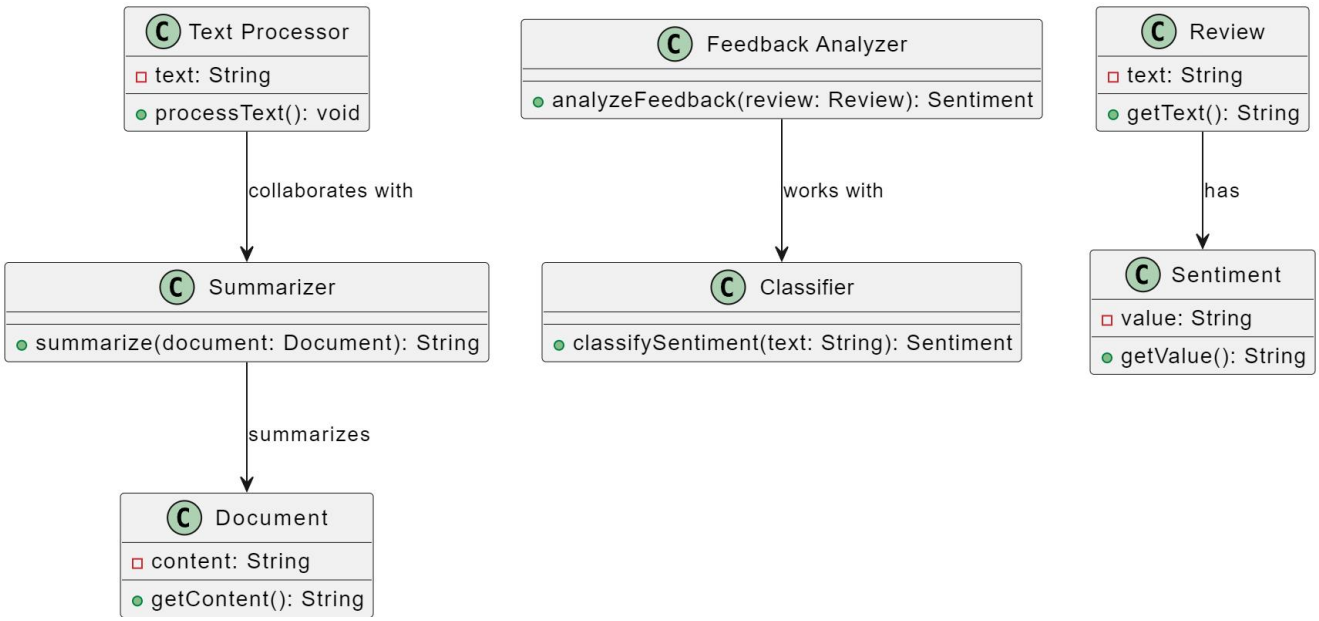
3.5.1 CLASS DIAGRAM

One of the most often used diagrams is the class diagram. All object-oriented software systems are built on it. It shows the system's static structure. It shows the class, properties, and operations of the system. It aids in understanding the relationships between various classes and objects.



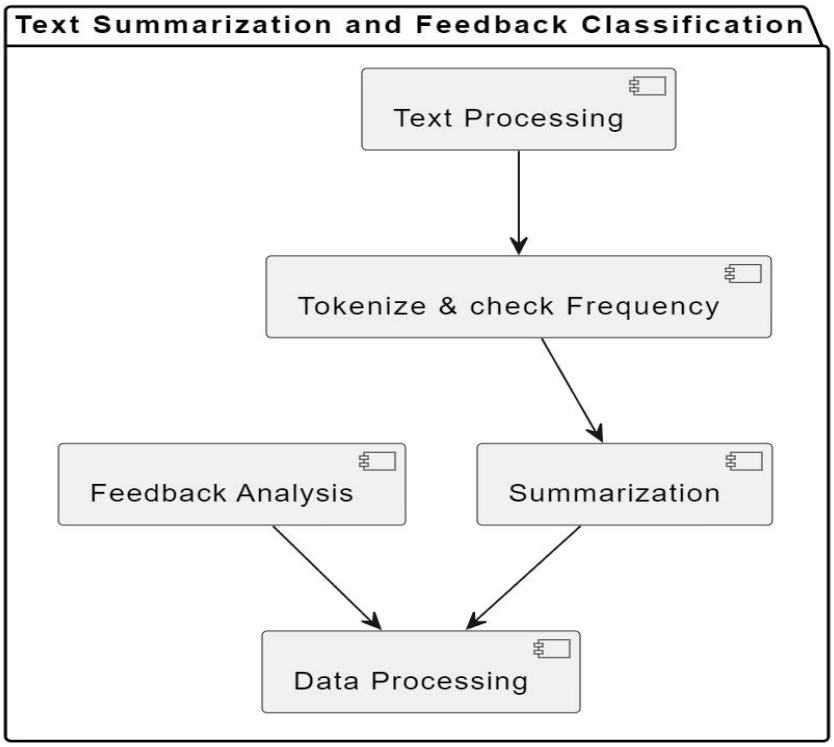
3.5.2 OBJECT DIAGRAM

Class diagrams are exemplified by object diagrams. Both class diagrams and object diagrams have the same fundamental ideas. It describes the system's static structure at a specific period. It can be utilized to gauge how accurate class diagrams are. It displays various instances of classes at once along with their relationships.



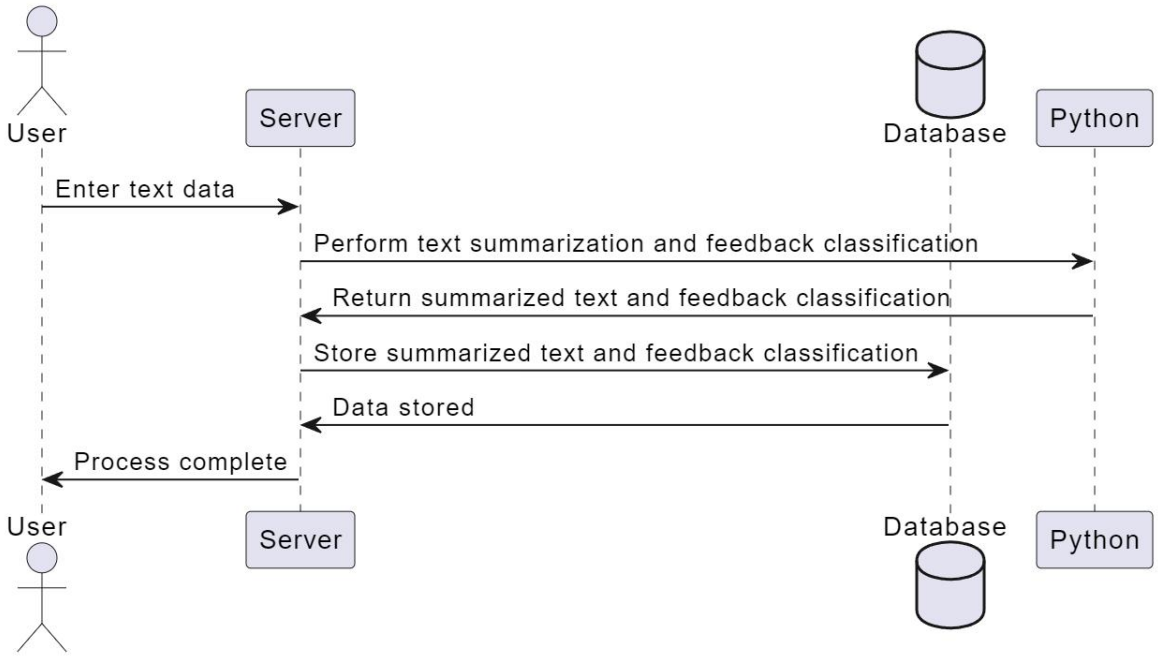
3.5.3 COMPONENT DIAGRAM

It illustrates how the system's physical components are arranged. It is employed for modeling specifics of execution. Because it contains structural linkages between the components of a software system, it can tell whether the planned development has taken into account the desired functional requirements or not.



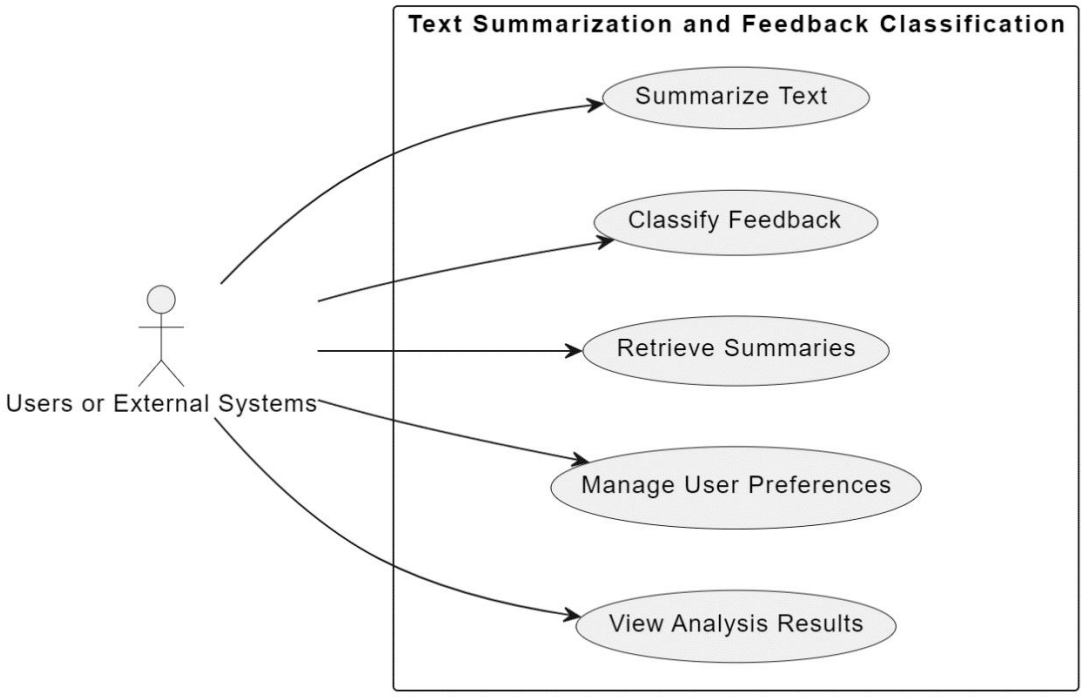
3.5.4 DEPLOYMENT DIAGRAM

By describing the physical components that are now in place and the software components that are executing on them, it shows both the hardware and software of the system. It generates data on system software. Every time software is utilized, disseminated, or installed across a number of machines with various configurations, it is included.



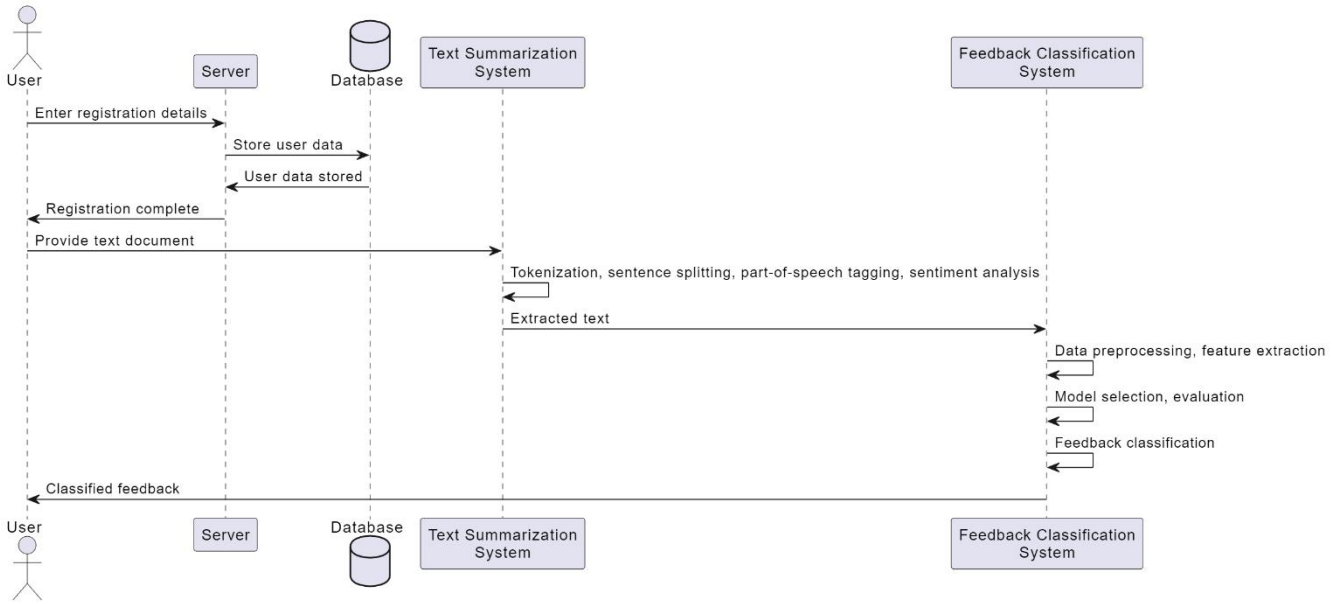
3.5.5 USECASE DIAGRAM

Use-case diagrams aid in capturing system requirements and depict a system's behavior in UML. The scope and high-level functions of a system are described in use-case diagrams. The interactions between the system and its actors are also depicted in these diagrams. utilize-case diagrams show what the system does and how the actors utilize it, but they do not show how the system works within.



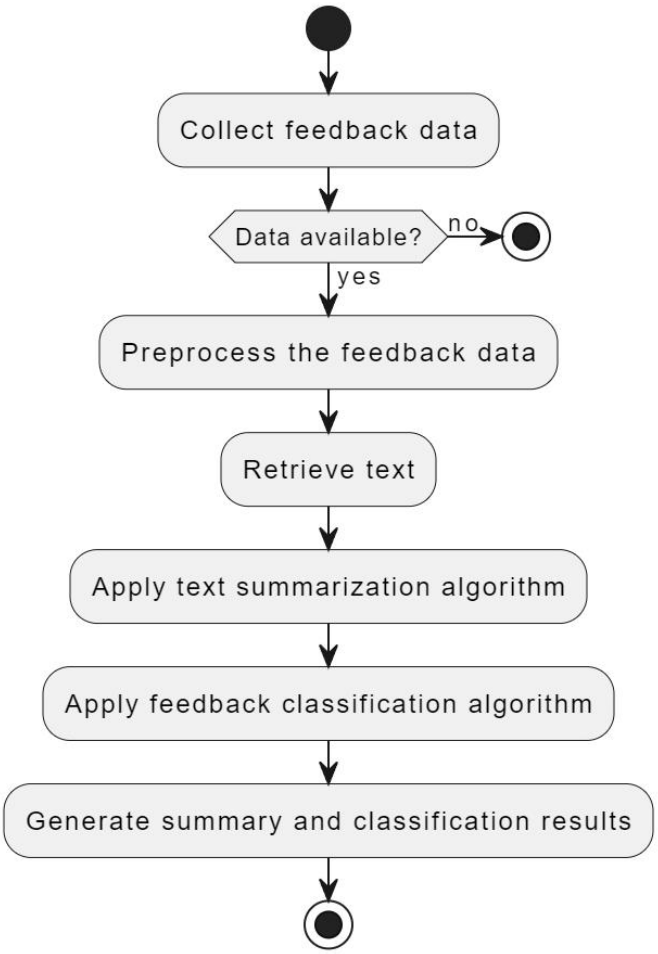
3.5.6 SEQUENCE DIAGRAM

The sequence diagram, which is also known as an event diagram, shows how messages move through the system. It aids in creating a variety of dynamic settings. It depicts communication between any two lifelines as a chronologically ordered series of activities, implying that these lifelines were active at the moment of communication. In UML, the lifeline is represented by a vertical bar, whereas the message flow is represented by a vertical dotted line that extends across the bottom of the page. It incorporates the iterations as well as branching.



3.5.7 ACTIVITY DIAGRAM

The activity diagram in UML is used to show the system's control flow rather than its implementation. Both concurrent and sequential activities are modeled. The workflow from one action to the next can be seen using the activity diagram. It placed emphasis on the existence of flow and the sequence in which it takes place. The flow may be sequential, branching, or concurrent, and the activity diagram has been designed with a fork, join, etc. to deal with these many types of flows. A flowchart that is object-oriented is another name for it. It includes tasks that include applying a series of actions or processes to simulate the behavioral diagram.



CHAPTER-4
RESULTS & DISCUSSION

4.1 TEST CASES

Each test case ensures the integrity and effectiveness of the implemented methodologies in text summarization and feedback sentiment analysis. Successful testing guarantees the project's ability to generate accurate summaries and sentiment insights from diverse textual data sources.

User input: The window which pops up contains the user input bars asking to enter the url to summarize and enter the product to get the feedback analysis. Check whether the url is in accessible format or not and whether the entered product is in the dataset or not.

Web Scraping: Verify that the web scraping mechanism accurately summarizes text content from different web pages, ensuring consistency and completeness. There are few web pages which are secured and unable to perform web scraping. The error forbidden should be displayed while summarizeing info from such pages.

Word Frequency Analysis: Assess the accuracy of word frequency calculations, verifying that high-frequency words are correctly identified and ranked. The summary of the text is summarizeed mainly based on the frequency of different words and the methodology performed here.

Feedback Analysis: Verify the feedback analysis by comparing the results against known reviews, ensuring the accurate identification of positive, negative, and neutral reviews within feedback. The feature summarization, score analysis and all the functionalities should be performed properly for proper output.

Visualization: Validate the visualization output to confirm that the generated graphs or charts accurately represent feedback distributions or summarization results. The visualization results should accurately match with the results given in the dataset.

4.2 SCREENSHOTS

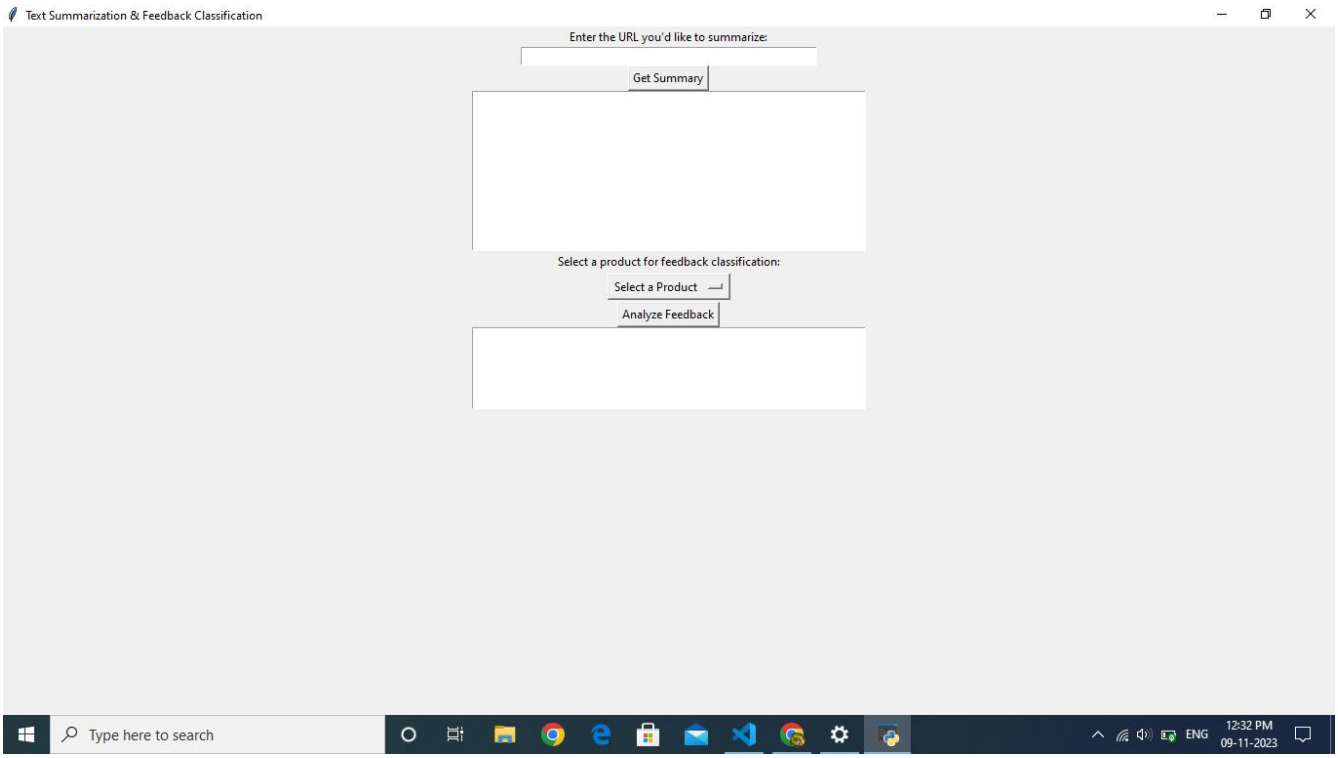


Fig: 4.2.1 User input Window

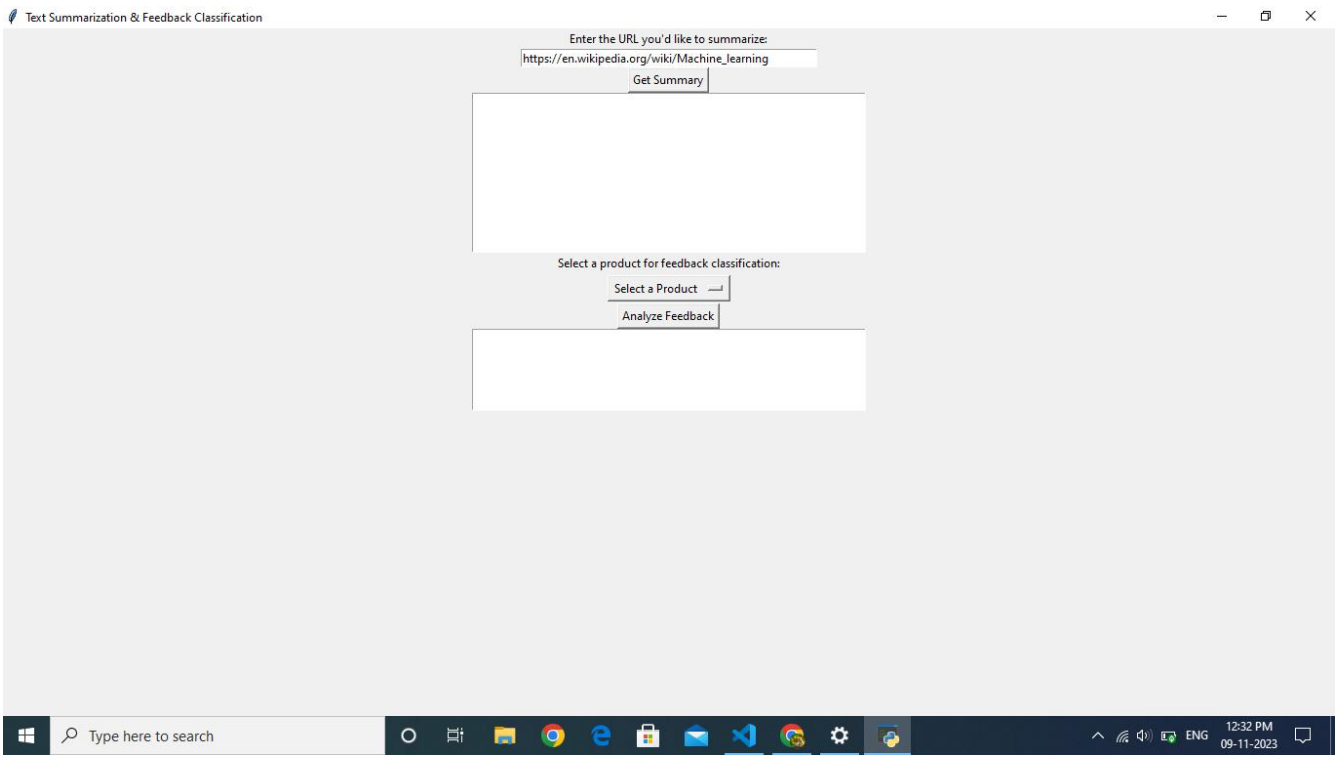


Fig: 4.2.2 URL User Input

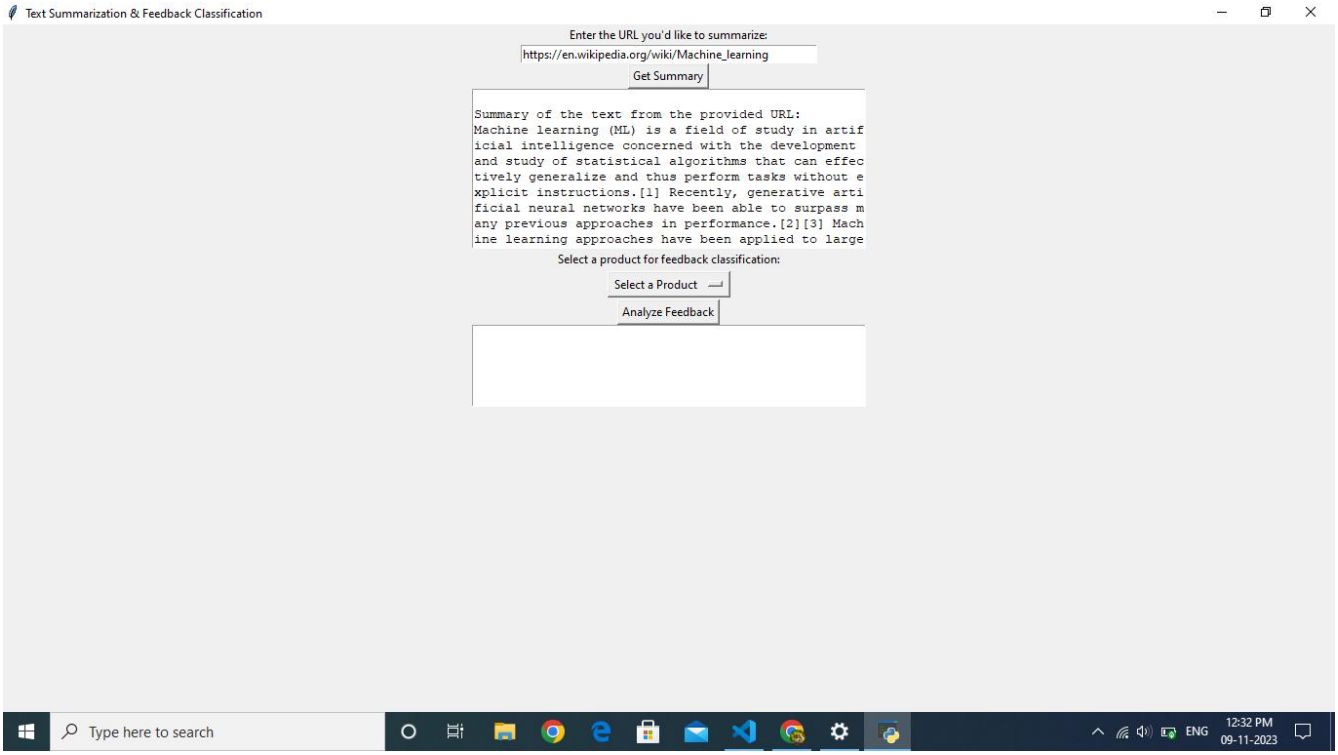


Fig: 4.2.3 Summarized Text Output

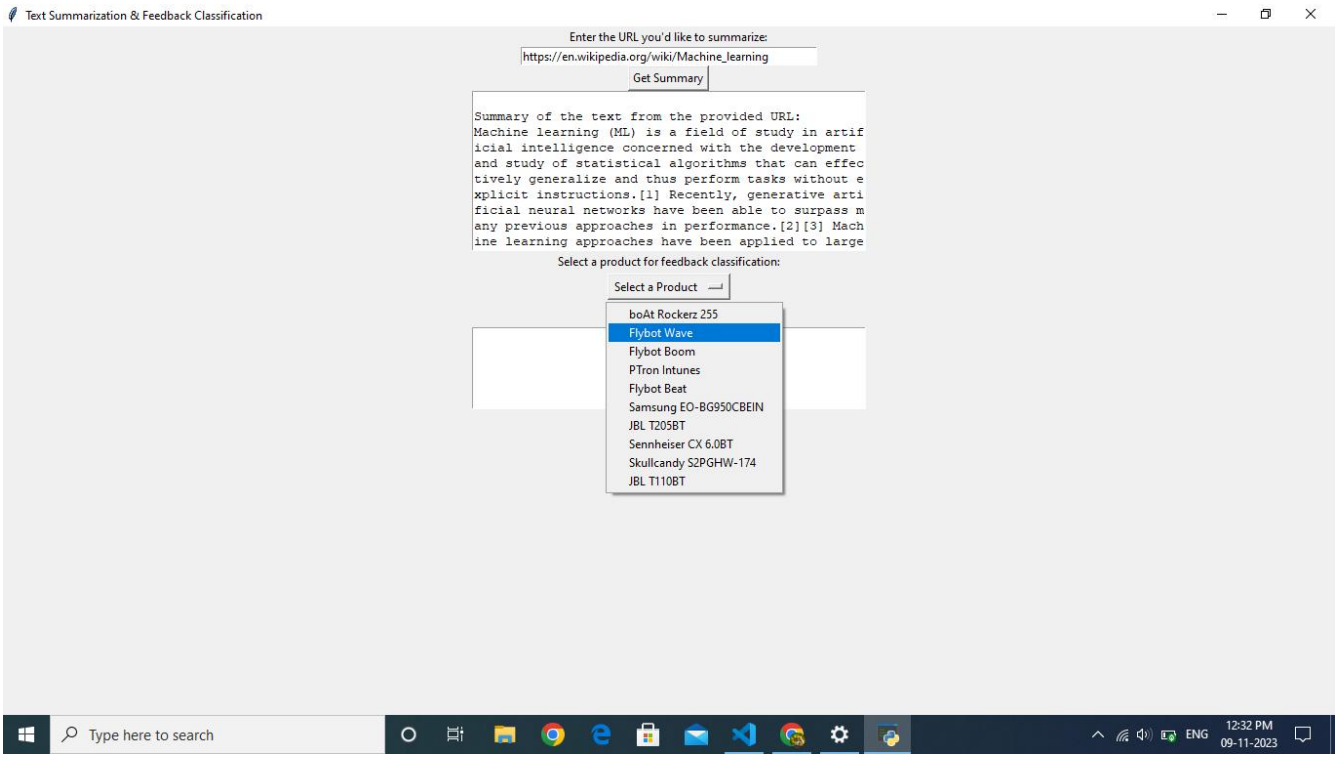


Fig: 4.2.4 Choosing the product to analyze

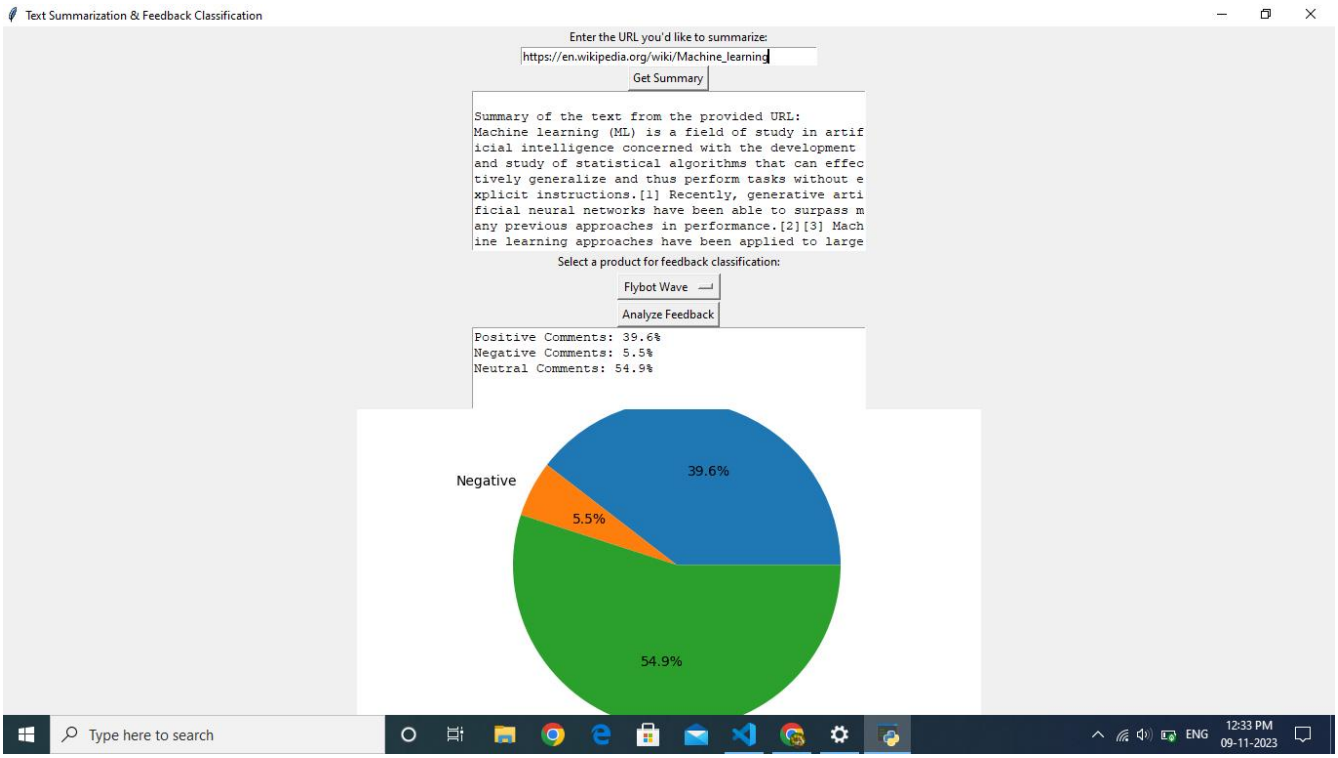
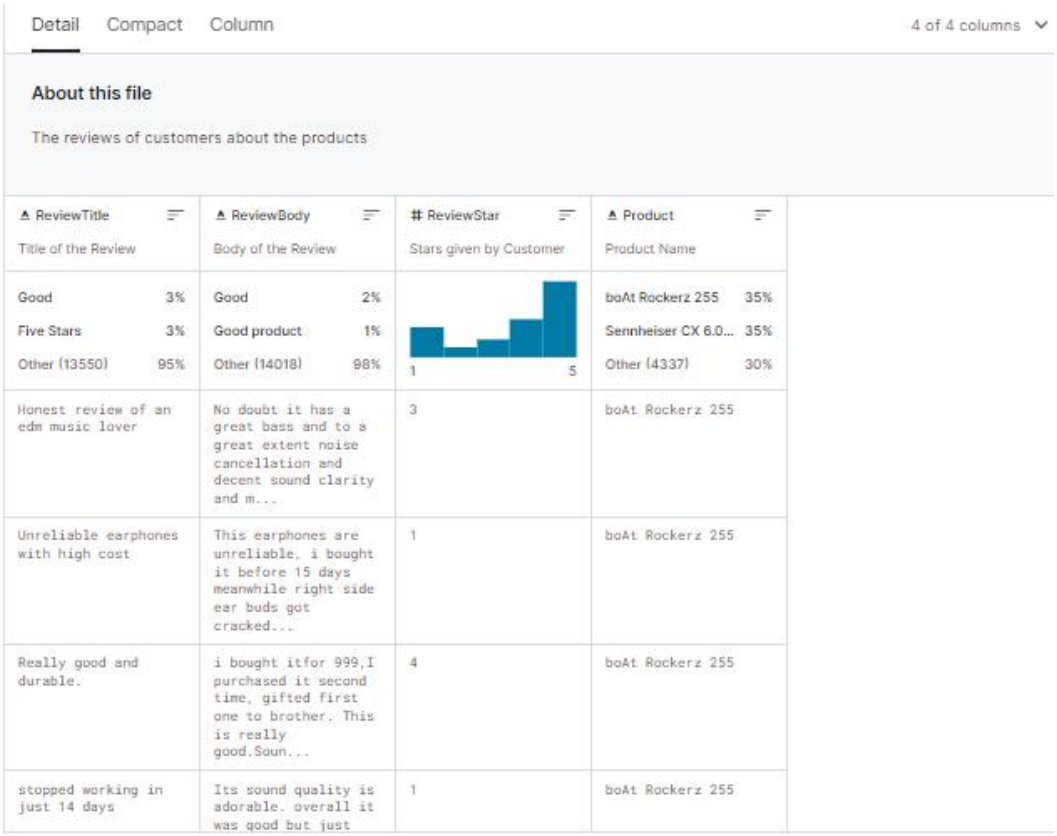


Fig: 4.2.5 Feedback Analysis



Detail Compact Column 4 of 4 columns

About this file

The reviews of customers about the products

ReviewTitle Title of the Review	ReviewBody Body of the Review	ReviewStar Stars given by Customer	Product Product Name
Good 3% Five Stars 3% Other (13550) 95%	Good 2% Good product 1% Other (14018) 98%	1 5	boAt Rockerz 255 35% Sennheiser CX 6.0... 35% Other (4337) 30%
Honest review of an edm music lover	No doubt it has a great bass and to a great extent noise cancellation and decent sound clarity and m...	3	boAt Rockerz 255
Unreliable earphones with high cost	This earphones are unreliable, i bought it before 15 days meanwhile right side ear buds got cracked...	1	boAt Rockerz 255
Really good and durable.	i bought itfor 999,I purchased it second time, gifted first one to brother. This is really good.Soun...	4	boAt Rockerz 255
stopped working in just 14 days	Its sound quality is adorable. overall it was good but just	1	boAt Rockerz 255

Fig 4.2.6 Dataset of Amazon Earphone Reviews

CHAPTER-5
SUMMARY & CONCLUSION

The "Text summarization and Feedback Classification using Python" project demonstrates the effectiveness of utilizing Python's libraries and machine learning techniques for text analysis. Through the implementation of text summarization methodologies and sentiment analysis tools, the project successfully showcases the summarization of meaningful insights from textual data. The combination of web scraping, tokenization, sentiment scoring, and visualization offers a comprehensive approach for summarizing text content and understanding sentiments expressed in feedback. By employing tokenization, stopwords removal, and frequency analysis, the project effectively summarizes key insights from textual data. The integration of sentiment analysis models, visualization tools, and machine learning approaches further enriches the project, allowing for a comprehensive understanding of user feedback sentiment. Overall, this project highlights the potential of Python's versatile capabilities in natural language processing and demonstrates its practical application in summarizing text content and analyzing sentiments from various sources.

5.1 FUTURE SCOPE

The "Text summarization and Feedback Classification using Python" project holds promising avenues for future exploration and enhancement. The potential areas for further development include: Advanced NLP Techniques, exploring advanced natural language processing (NLP) methodologies and algorithms to refine text summarization and sentiment analysis for more accurate and insightful results. Interactive User Interface: Designing an interactive and user-friendly interface to allow users to input URLs for summarization or feedback analysis and receive intuitive visual representations of the processed data. Deep Learning Models, implementing and experimenting with deep learning models for text summarization and sentiment analysis to achieve more sophisticated and context-aware results. \ By delving deeper into these areas, the project can evolve to deliver more accurate, sophisticated, and efficient text summarization and sentiment analysis solutions, catering to a wider range of applications and delivering more valuable insights.

5.2. BIBLIOGRAPHY

[1] Ahmad T. Al-Taani. “,Automatic Text summarization Approaches” International Conference on Infocom Technologies and Unmanned Systems (ICTUS'2017).

[2] Neelima Bhatia, ArunimaJaiswal, “Automatic Text summarization: Single and Multiple Summarizations ”, International Journal of Computer Applications.

[3] Mehdi Allahyari, SeyedaminPouriyeh, Mehdi Assefi, SaeidSafaei, Elizabeth D. Trippe, Juan B. Gutierrez, KrysKochut, “ Text summarization Techniques: A Brief Survey”, (IJACSA) International Journal of Advanced Computer Science and Applications.

[4] Bing Liu. Sentiment analysis: Mining opinions, sentiments, and emotions.Cambridge University Press.

[5] Jayashri Khairnar and Mayura Kinikar. Machine learning algorithms for opinion mining and sentiment classification. International Journal of Scientific and Research Publications, 3(6):1–6.

Reference links:

https://www.activestate.com/blog/how-to-do-text-summarization-with-python/

https://stackabuse.com/text-summarization-with-nltk-in-python/

https://www.researchgate.net/publication/356753421_Text_Summarizing_Using_NLP

https://www.researchgate.net/publication/363181339_Sentiment_Analysis_of_Consumer_Reviews_Using_Deep_Learning

https://rcciit.org/students_projects/projects/cse/2018/GR20.pdf