GEN AI FEEDBACK REPORT GENERATOR

MINI PROJECT

Submitted by

Rajesh S	**412522243125**
Pradeep S	**412522243117**

in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

SRI SAIRAM ENGINEERING COLLEGE

(An Autonomous Institution; Affiliated to Anna University, Chennai - 600 025)

ANNA UNIVERSITY: CHENNAI 600 025

NOVEMBER 2024

SRI SAIRAM ENGINEERING COLLEGE

(An Autonomous Institution; Affiliated to Anna University)

BONAFIDE CERTIFICATE

Certified that this project report **"GEN AI FEEDBACK REPORT GENERATOR"** is the Bonafide work of **Rajesh S (412522243125), and Pradeep S (412522243117)** who carried out the **MINI PROJECT** under my supervision.

SIGNATURE	**SIGNATURE**
Staff in Charge	HOD
Ms. Jeena A Thankachan	**Dr. Swagata Sarkar**
Assistant Professor	**Head of the Department**

Submitted for project Viva – Voce Examination held on ________________

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

"Let our advance worrying become advance thinking and planning." -Winston Churchill.

Such a successful personality is our beloved founder Chairman, **Thiru. MJF. Ln. LEO MUTHU**. At first, we express our sincere gratitude to our beloved chairman through prayers, who in the form of a guiding star has spread his wings of external support with immortal blessings.

We express our gratitude to our **CEO Mr. J. Sai Prakash Leomuthu** for having given us spontaneous and whole-hearted encouragement for completing this project.

We express our sincere thanks to our beloved **Principal, Dr. J. Raja** for having given us spontaneous and whole-hearted encouragement for completing this project.

We are indebted to our **Head of the Department Dr. Swagata Sarkar** for her support during the entire course of this project work.

We express our gratitude and sincere thanks to our **subject** (**20AIPC503, Natural Language Processing and Chatbot) handling staff, Assistant Professor, Ms. Jeena A Thankachan,** for her valuable suggestions and constant encouragement for successful completion of this project.

We thank all the teaching and non-teaching staff members of the **Department of Artificial Intelligence and Data Science** and all others who contributed directly or indirectly for the successful completion of the project.

TABLE OF CONTENTS

ABSTRACT

The project is a feedback analysis tool designed to simplify and automate the process of interpreting large volumes of feedback data. Users can upload feedback sheets directly into the app, which processes the data to extract key parameters like "hands-on work," satisfaction levels, and other essential metrics. The results are visually represented through intuitive charts, including pie charts and bar graphs, offering a clear overview of trends.

A standout feature of the tool is the generation of automated PDF reports. These reports provide a quick summary of all responses on the first page, giving users an immediate snapshot of overall feedback. Subsequent pages offer detailed insights with visually appealing donut charts, presenting percentage breakdowns for critical parameters such as hands-on experience and satisfaction levels.

This project is an invaluable solution for teams managing extensive feedback data. By automating analysis and delivering easy-to-read reports, it saves significant time and effort while ensuring that the most important insights are highlighted effectively. The tool enables fast, data-driven

CHAPTER 1

INTRODUCTION

1.1. OBJECTIVE

1. Automate Data Processing: Enable users to upload feedback sheets and extract key metrics such as satisfaction levels, "hands-on work," and other essential parameters without manual effort.

2. Visualize Insights Clearly: Present the analyzed data using intuitive visual formats like pie charts, bar graphs, and donut charts to highlight trends and patterns effectively.

3. Generate Comprehensive Reports: Create automated PDF reports that summarize feedback data on the first page and provide detailed insights in subsequent sections, ensuring clarity and accessibility.

4. Save Time and Effort: Minimize the time and effort required for manual feedback analysis, offering a fast, accurate, and user-friendly solution for teams.

5. Support Decision-Making: Provide actionable insights to help organizations make informed decisions based on feedback trends and metrics.

1.2. OUTLINE OF THE REPORT

1. Introduction

- Overview of the project and its significance.
- Challenges in manual feedback analysis and the need for automation.
- Objectives of the project.

2. System Requirements

- Hardware Requirements: Specifications for hosting and running the application.
- Software Requirements:
    - Programming languages and frameworks (e.g., Python, JavaScript, etc.).
    - Libraries for data processing, visualization, and PDF generation.
    - Database for storing feedback data.

3. Methodology

- Input Collection:
    - Uploading feedback sheets in supported formats (e.g., CSV, Excel).
- Data Preprocessing:
    - Cleaning and formatting the data for analysis.
    - Handling missing or inconsistent entries.
- Data Analysis:
    - Extracting key parameters like satisfaction levels, hands-on experience, etc.
    - Aggregating responses for trend identification.
- Visualization:
    - Generating pie charts, bar graphs, and donut charts.
- Report Generation:
    - Compiling insights into automated PDF reports.
    - Summary on the first page; detailed breakdowns on subsequent pages.

4. System Design

- Architecture: Overview of the system architecture (e.g., client-server model).
- Flow Diagram: Visual representation of the data flow through the system.
- User Interface Design:
    - Upload interface.
    - Visual dashboard for trends and insights.

5. Implementation

- Backend Development: Data processing and analysis modules.
- Frontend Development: Interactive dashboards and upload functionality.
- Integration: Merging backend and frontend for seamless functionality.

6. **Testing and Validation**

- Functional testing to ensure accurate data processing and reporting.
- User testing for interface usability and report clarity.
- Performance testing to handle large datasets efficiently.

7. Results and Discussion

- Presentation of sample outputs and insights.
- Evaluation of time saved and accuracy improvements.
- Feedback from initial users or stakeholders.

8. Conclusion

- Summary of project achievements.
- Potential impact on organizations handling feedback.
- Limitations and future enhancements.

9. Future Scope

- Supporting additional file formats and larger datasets.
- Incorporating machine learning for predictive analysis or sentiment evaluation.
- Enabling real-time analysis through cloud-based implementation.

10. References

- Tools, frameworks, and libraries used.
- Research papers and resources for methodologies.

1.3. SCOPE OF THE PROJECT

The scope of this project is to develop a feedback analysis tool that simplifies and automates the process of extracting and interpreting insights from large datasets. It supports uploading feedback sheets in various formats, processes the data to identify key parameters such as satisfaction levels and engagement metrics, and presents the results using intuitive visualizations like pie charts, bar graphs, and donut charts. The tool generates automated PDF reports, providing both summary and detailed insights to help users make informed decisions. Designed for organizations, educational institutions, and event planners, it aims to save time, reduce manual effort, and deliver actionable insights. Future enhancements include integrating real-time feedback, sentiment analysis, and compatibility with other platforms to provide a scalable and versatile solution.

CHAPTER 2

SDG JUSTIFICATION

1. SDG 4: Quality Education

Target 4.3 & 4.4: The tool can be utilized by educational institutions to analyze student feedback, evaluate course effectiveness, and improve teaching methods. By automating feedback processing, it ensures actionable insights that enhance learning experiences and teaching quality.

2. SDG 8: Decent Work and Economic Growth

Target 8.2 & 8.3: By improving organizational processes, this tool supports innovation and productivity, enabling businesses and teams to address employee and customer feedback more effectively. This contributes to better workplace environments and customer satisfaction, driving economic growth and fostering sustainable employment.

3. SDG 9: Industry, Innovation, and Infrastructure

Target 9.5 & 9.c: The project promotes the use of advanced technologies like AI and data analytics, fostering innovation in feedback analysis and reporting. It supports organizations in adopting modern, scalable solutions for data-driven decision-making, strengthening their infrastructure for future challenges.

CHAPTER 3

PROPOSED SOLUTION

The proposed solution is an **AI-powered Feedback Analysis Tool** designed to automate the extraction, analysis, and reporting of insights from feedback data. This tool addresses the challenges of manually processing large volumes of feedback, ensuring that organizations can efficiently derive actionable insights.

3.1. SOURCE CODE:

```
import pandas as pd
import torch
from transformers import pipeline
import gradio as gr
import matplotlib.pyplot as plt
from fpdf import FPDF
import google.generativeai as genai
import tempfile
import os
import ast

sentiment = pipeline("text-classification",
model="distilbert/distilbert-base-uncased-finetuned-
sst-2-english",
                torch_dtype=torch.bfloat16)

def analysis(response):
    output = sentiment(response)
    return output[0]['label']

def read_excel(file_path):
    df = pd.read_excel(file_path)
    return df, list(df.columns)

def sentiment_pie_chart(df):
    sentiment_counts =
df['Sentiments'].value_counts()
    fig, ax = plt.subplots()
    sentiment_counts.plot(kind='pie', ax=ax,
autopct='%1.1f%%', color=['#8DA7E2', '#a3d2ca'])
    ax.set_title('Review Sentiment Distribution')
    return fig

def sentiment_donot_chart(df):
    percentage = float(df[1])
    sizes = [percentage, (100 - percentage)]
    colors = ['#8DA7E2', '#a3d2ca']
    fig, ax = plt.subplots()
    wedges = ax.pie(sizes, colors=colors,
startangle=90)
    centre_circle = plt.Circle((0, 0), 0.70, fc='white')
    fig.gca().add_artist(centre_circle)
    ax.axis('equal')
    total = sum(sizes)
    percentage = sizes[0] / total * 100
    ax.text(0, 0, f'{percentage:.1f}%', ha='center',
va='center', fontsize=20)
    ax.set_title(str(df[0]))
    return fig

def sentiment_bar_chart(df):
    sentiment_counts =
df['Sentiments'].value_counts()
    fig, ax = plt.subplots()
    bars = sentiment_counts.plot(kind='bar', ax=ax,
color=['#8DA7E2', '#a3d2ca'])
    ax.set_title('Review Sentiment Counts')
    ax.set_xlabel('Sentiment')
    ax.set_ylabel('Count')
    for bar in bars.patches:
        ax.annotate(f'{bar.get_height()}', (bar.get_x() +
bar.get_width() / 2, bar.get_height()),
                    ha='center', va='bottom')
    return fig

def process_sentiment(file_path, selected_column):
    df, _ = read_excel(file_path)
    if selected_column not in df.columns:
        raise ValueError(f"No Column named
'{selected_column}'")
    df['Sentiments'] =
df[selected_column].apply(analysis)
    pie_chart = sentiment_pie_chart(df)
    bar_chart = sentiment_bar_chart(df)
    return df, pie_chart, bar_chart

def generate_pdf(file_path, selected_column):
    df, _ = read_excel(file_path)
    df['Sentiments'] =
df[selected_column].apply(analysis)
    llm =
genai.GenerativeModel(model_name='gemini-pro')

genai.configure(api_key='AIzaSyCr393tpEJumNQ
YoT3oYXcfl5WPCnSRe3Q')
    responses_para =
list(df[selected_column].to_numpy())
    responses_para = " , ".join(responses_para)
    res = llm.generate_content(
        '''
        Responses''' +
        str(responses_para) +
        '''
        what are the key parameters we can measure
from these feedback responses. minimum - 10,
maximum - 15. give the answer in a python list
nested with list with parameters and their positive
percentages(scale is 100, so give values from 0 to
100 only) from the text, i don't need any other extra
words or letters in the answer. The final list example
format is given [
            ["Parameter1", value],
            ["Parameter2", value],
            ["Parameter3", value],
        ]
        '''
    )
    response_list = ast.literal_eval(res.text)
    response_description = llm.generate_content(
        '''
        give a short description that can be understood
from the below content in 100 words without any
title and give only text for the responses given as
feedbacks.
        Responses
        ''' + str(responses_para)
    )
    pdf = FPDF()
    pdf.add_page()
    pdf.set_font("Arial", size=14)
    pdf.cell(190, 10, txt="Feedback Responses
Report", ln=True, align='C', border=True)
    pie_chart = sentiment_pie_chart(df)
    pie_temp_file =
tempfile.NamedTemporaryFile(delete=False,
suffix=".png")
    pie_chart.savefig(pie_temp_file.name)
    pie_temp_file.close()
    plt.close(pie_chart)
    x_position = 10
    y_position = 540
    pdf.image(pie_temp_file.name, x=x_position,
y=y_position, w=80)
    os.remove(pie_temp_file.name)
    pie_chart = sentiment_bar_chart(df)
    pie_temp_file =
tempfile.NamedTemporaryFile(delete=False,
suffix=".png")
    pie_chart.savefig(pie_temp_file.name)
    pie_temp_file.close()
    plt.close(pie_chart)
    x_position = 110
    y_position = 540
    pdf.image(pie_temp_file.name, x=x_position,
y=y_position, w=80)
    os.remove(pie_temp_file.name)
    pdf.ln(10)
    pdf.set_font("Arial", size=12)
    response_title = 'Summary\n'
    response_description = response_title +
str(response_description.text).strip()
    pdf.multi_cell(170, 10, txt=response_description,
align='L')
    pdf.ln(10)
    pdf.add_page()
    pdf.set_font("Arial", size=14)
    pdf.cell(190, 10, txt="Feedback Responses
Charts", ln=True, align='C', border=True)
    chart_count = 0
    for i in range(len(response_list)):
        pie_chart =
sentiment_donot_chart(response_list[i])
        pie_temp_file =
tempfile.NamedTemporaryFile(delete=False,
suffix=".png")
        pie_chart.savefig(pie_temp_file.name)
        pie_temp_file.close()
        plt.close(pie_chart)
        x_position = 10 if chart_count % 2 == 0 else
        y_position = 40 + (chart_count // 2) * 70
        pdf.image(pie_temp_file.name, x=x_position,
y=y_position, w=80)
        if chart_count == 5:
            pdf.add_page()
            chart_count = 0
        else:
            chart_count += 1
        os.remove(pie_temp_file.name)
    pdf_temp_file =
tempfile.NamedTemporaryFile(delete=False,
suffix=".pdf")
    pdf.output(pdf_temp_file.name)
    return pdf_temp_file.name

def get_columns(file_path):
    _, columns = read_excel(file_path)
    return gr.Dropdown(choices=columns)

with gr.Blocks() as demo:
    file_input = gr.File(file_types=["xlsx"],
file_count='single', label="Upload your .xlsx file")
    column_dropdown = gr.Dropdown(label="Select
the Responses Column after uploading file...",
choices=[],
                                  interactive=True)
    output_df = gr.Dataframe(label="Sentiments
Predicted...")
    with gr.Row():
        output_pie_chart = gr.Plot(label="Pie Chart")
        output_bar_chart = gr.Plot(label="Bar Chart")
    file_input.upload(fn=get_columns,
inputs=file_input, outputs=column_dropdown)
    analyze_button = gr.Button("Analyze
Sentiments")
    analyze_button.click(fn=process_sentiment,
inputs=[file_input, column_dropdown],
                        outputs=[output_df,
output_pie_chart, output_bar_chart])
    generate_pdf_button = gr.Button("Generate PDF
Report")
    generate_pdf_button.click(fn=generate_pdf,
inputs=[file_input, column_dropdown],
outputs=gr.File())
demo.launch()
```

CHAPTER 4
REQUIREMENT SPECIFICATION

1. Backend

- Python: Version 3.6 or higher for developing the application.
- Flask: Web framework for building the server-side application and handling file uploads, analysis, and report generation.

2. Libraries and Frameworks

- Transformers: For sentiment analysis using pre-trained models (e.g., DistilBERT).
- Pandas: For handling and processing Excel data.
- Matplotlib: For generating visualizations (pie charts, bar charts, donut charts).
- FPDF: For generating PDF reports with visual charts and summaries.
- Gradio: For building the interactive web interface.
- google-generativeai: For using Google Generative AI to extract insights from the feedback.

3. Frontend

- HTML: For structuring the web page and user interface.
- CSS: For styling the web interface, optionally using Bootstrap for responsive design.

4. Development Environment

- IDE/Text Editor: Any Python-supported editor, such as VS Code, PyCharm, or Jupyter Notebooks.
- Web Browser: For testing and accessing the application,

compatible with browsers such as Chrome, Firefox, or Edge.

5. **Operating System**

- The application is compatible with **Windows**, **macOS**, or **Linux** operating systems.

CHAPTER 5

IMPLEMENTATION

3.2. IMPLEMENTATION TECHNOLOGY

1. Programming Languages and Frameworks

- **Python:**
    - The primary language for developing the application, providing a versatile environment for backend processing, sentiment analysis, and report generation.
- **Flask:**
    - A lightweight Python web framework used to build the server-side application, handle HTTP requests, manage file uploads, and serve the frontend. It enables a clean and straightforward architecture for web-based applications.
    -

2. AI and Machine Learning

- **Transformers (Hugging Face):**
    - A deep learning library used to implement pre-trained models like DistilBERT for sentiment analysis, classifying feedback as positive or negative.
- **Google Generative AI:**
    - Provides advanced NLP capabilities for extracting key insights from feedback data, generating summaries, and identifying relevant parameters with high accuracy.

3. Data Handling and Processing

- **Pandas**:
    - Used for data manipulation and analysis, allowing the application to load, process, and manage data from **.xlsx** files (feedback data) efficiently.
- **NumPy**:
    - May be used for additional data operations (e.g., handling numerical computations or array manipulation) if needed.

4. Data Visualization

- **Matplotlib**:
    - A popular Python library for creating visualizations, used to generate pie charts, bar charts, and donut charts for sentiment distribution and feedback analysis.
- **Seaborn**:
    - Can be optionally used in combination with Matplotlib for more advanced visualizations and styling of charts.

5. Report Generation

- **FPDF**:
    - A Python library for creating PDF documents. It allows the tool to generate professional PDF reports, embedding sentiment analysis results, charts, and insights from feedback data.

6. Frontend Development

- **HTML/CSS**:
    - Used for structuring and styling the user interface, allowing users to upload files, select columns, view results, and download reports.
- **JavaScript (optional)**:
    - For added interactivity or frontend enhancements, JavaScript can be used, though Flask and Gradio manage the main user interaction.

7. User Interface

- **Gradio**:
    - An open-source Python library that provides an easy-to-use interface for users to interact with the application. It simplifies integrating file uploads, displaying visual outputs (charts), and generating downloadable reports.

8. Operating System Compatibility

- The application is compatible with **Windows**, **macOS**, and **Linux** operating systems, ensuring a broad reach for deployment across different environments.

9. Development and Deployment Tools

- **IDE/Text Editor**:
  - **VS Code** or **PyCharm** are commonly used for Python development, with support for Flask, data science libraries, and version control.
- **Version Control**:
  - **Git** is used to manage source code and track changes during development, with **GitHub** or **GitLab** for repository hosting.

10. Deployment (Optional)

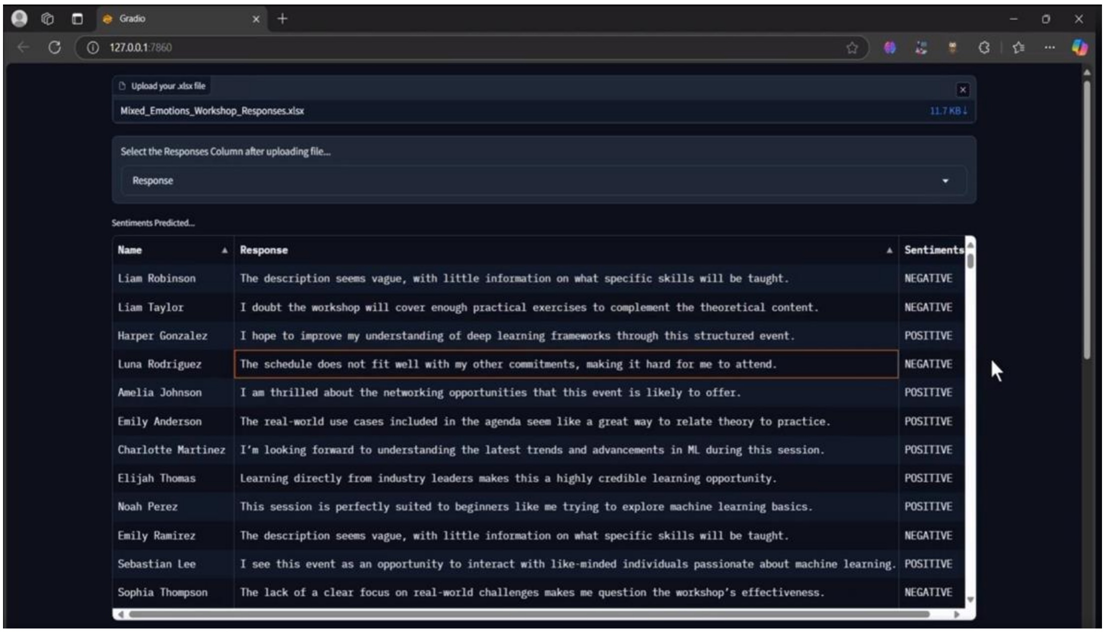
- **Heroku** or **AWS**:
  - These cloud platforms are options for deploying the web application to production. Heroku offers an easy way to deploy Python web applications, while AWS provides more control and scalability.

CHAPTER 6

RESULT AND CONCLUSION

6.1 RESULTS

The Feedback Analysis Tool processes the uploaded feedback data by classifying each response into positive or negative sentiment using the DistilBERT model. The results are visually represented through pie charts and bar charts, showing the overall sentiment distribution and count of positive and negative feedback. Additionally, the tool identifies key parameters from the feedback (e.g., "Customer Service," "Ease of Use") and calculates their positive feedback percentages, displayed in donut charts. A detailed PDF report is generated, summarizing the analysis, visual insights, and key parameters, providing users with actionable insights and an easy-to-understand overview of the feedback trends.



Upload your .xlsx file

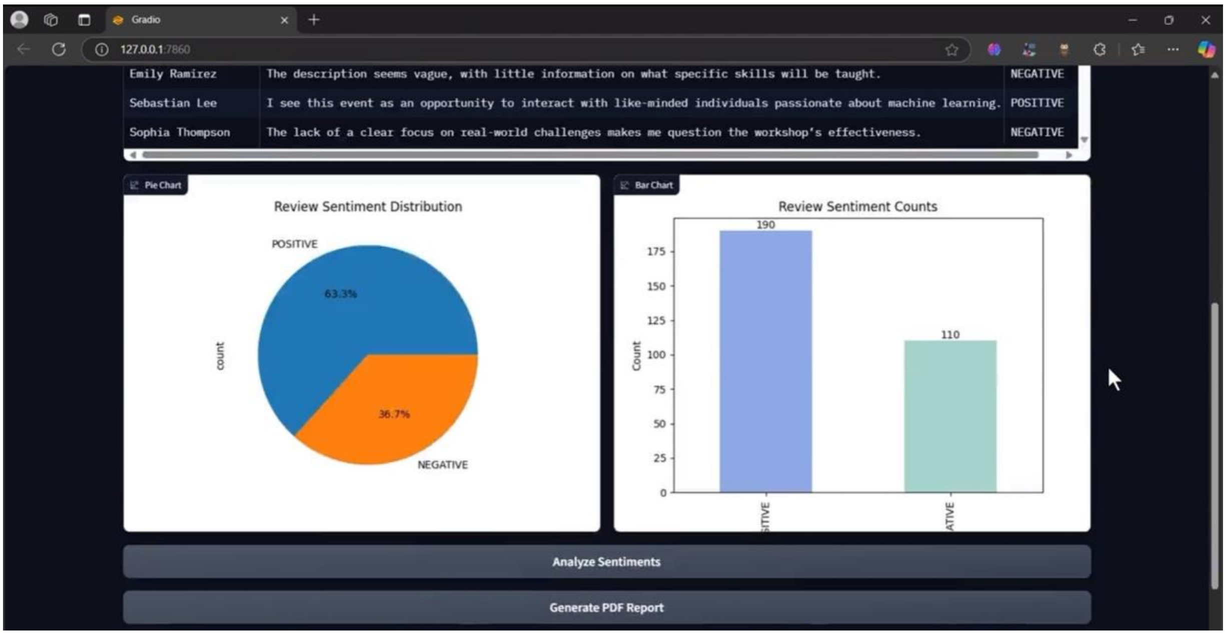
Mixed_Emotions_Workshop_Responses.xlsx 11.7 KB

Select the Responses Column after uploading file...

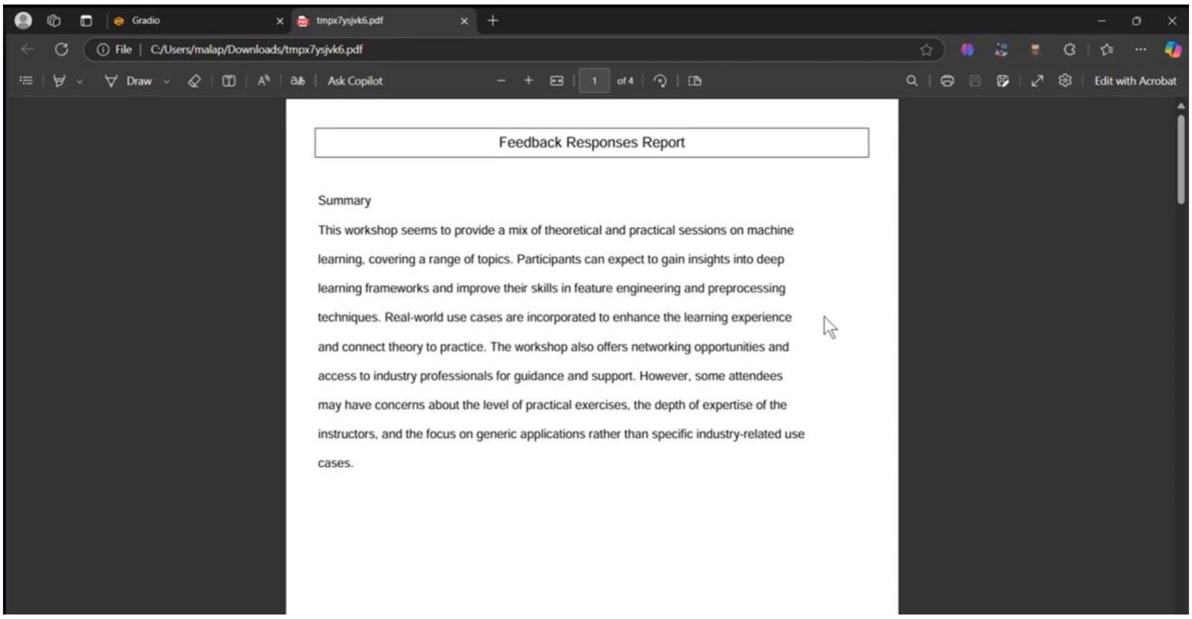
Response

Sentiments Predicted...

Name	Response	Sentiments
Liam Robinson	The description seems vague, with little information on what specific skills will be taught.	NEGATIVE
Liam Taylor	I doubt the workshop will cover enough practical exercises to complement the theoretical content.	NEGATIVE
Harper Gonzalez	I hope to improve my understanding of deep learning frameworks through this structured event.	POSITIVE
Luna Rodriguez	The schedule does not fit well with my other commitments, making it hard for me to attend.	NEGATIVE
Amelia Johnson	I am thrilled about the networking opportunities that this event is likely to offer.	POSITIVE
Emily Anderson	The real-world use cases included in the agenda seem like a great way to relate theory to practice.	POSITIVE
Charlotte Martinez	I'm looking forward to understanding the latest trends and advancements in ML during this session.	POSITIVE
Elijah Thomas	Learning directly from industry leaders makes this a highly credible learning opportunity.	POSITIVE
Noah Perez	This session is perfectly suited to beginners like me trying to explore machine learning basics.	POSITIVE
Emily Ramirez	The description seems vague, with little information on what specific skills will be taught.	NEGATIVE
Sebastian Lee	I see this event as an opportunity to interact with like-minded individuals passionate about machine learning.	POSITIVE
Sophia Thompson	The lack of a clear focus on real-world challenges makes me question the workshop's effectiveness.	NEGATIVE



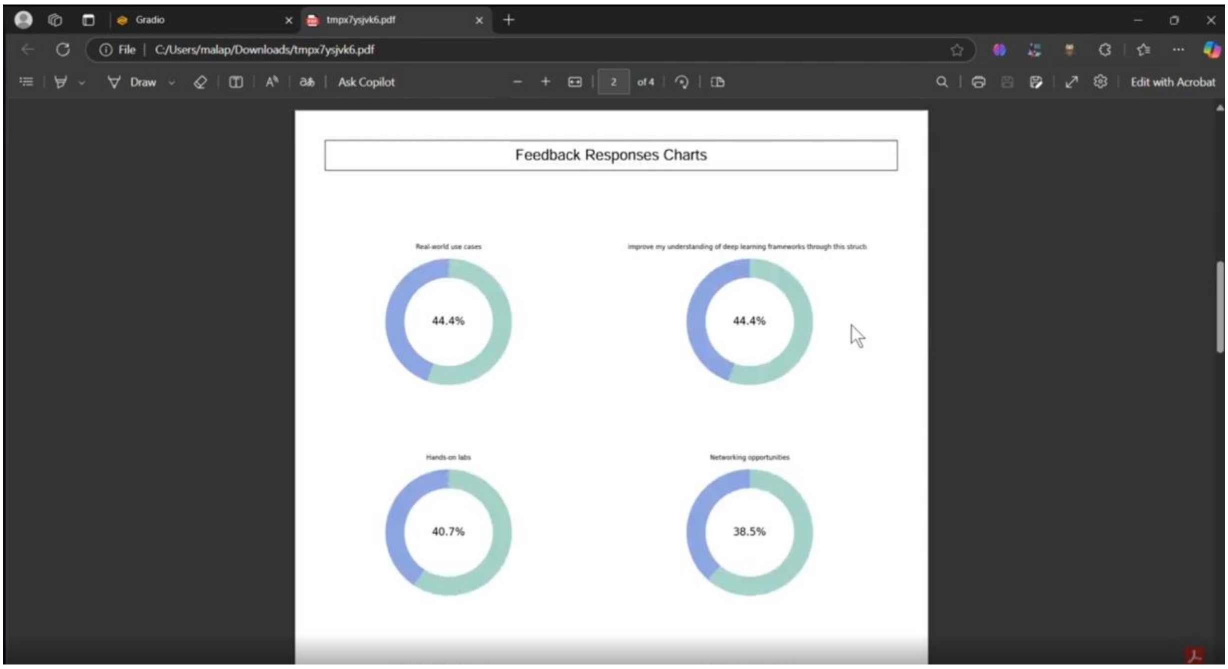
Emily Ramirez	The description seems vague, with little information on what specific skills will be taught.	NEGATIVE
Sebastian Lee	I see this event as an opportunity to interact with like-minded individuals passionate about machine learning.	POSITIVE
Sophia Thompson	The lack of a clear focus on real-world challenges makes me question the workshop's effectiveness.	NEGATIVE



Feedback Responses Report

Summary

This workshop seems to provide a mix of theoretical and practical sessions on machine learning, covering a range of topics. Participants can expect to gain insights into deep learning frameworks and improve their skills in feature engineering and preprocessing techniques. Real-world use cases are incorporated to enhance the learning experience and connect theory to practice. The workshop also offers networking opportunities and access to industry professionals for guidance and support. However, some attendees may have concerns about the level of practical exercises, the depth of expertise of the instructors, and the focus on generic applications rather than specific industry-related use cases.



Gradio
tmpx7ysjvk6.pdf
File | C:/Users/malap/Downloads/tmpx7ysjvk6.pdf
Draw
Ask Copilot
2
of 4
Edit with Acrobat
Feedback Responses Charts
Real-world use cases
44.4%
improve my understanding of deep learning frameworks through this structi
44.4%
Hands-on labs
40.7%
Networking opportunities
38.5%

6.2. CONCLUSION

The Feedback Analysis Tool provides an efficient and automated solution for processing and analyzing large volumes of feedback data. By leveraging advanced sentiment analysis and data visualization techniques, the tool transforms raw feedback into actionable insights. It not only classifies feedback into positive and negative sentiments but also identifies key parameters and visualizes them through intuitive charts. The automated generation of PDF reports further streamlines the process, making it easy to share and interpret the results. This tool significantly reduces the time and effort required for feedback analysis, enabling teams to quickly identify trends and areas for improvement.

CHAPTER 7

REFERENCES

Hugging Face Transformers Documentation:

Hugging Face provides a wide range of pre-trained models, including **DistilBERT**, which is used for sentiment analysis in this tool.

https://huggingface.co/docs/transformers

Pandas Documentation:

Pandas is a powerful data manipulation library used to handle and process feedback data in Excel format.

https://pandas.pydata.org/pandas-docs/stable/

🎬 **Matplotlib Documentation**:

Matplotlib is used for data visualization, creating pie, bar, and donut charts to represent feedback insights visually.

https://matplotlib.org/stable/contents.html

🎬 **FPDF Documentation**:

FPDF is used for generating PDF reports, embedding charts, and summarizing the analysis results.

https://pyfpdf.readthedocs.io/en/latest/

🎬 **Gradio Documentation**:

Gradio simplifies the creation of interactive user interfaces for Python applications, which is used in this project for user interaction.

https://gradio.app/

🎬 **Google Generative AI Documentation**:

Google’s generative AI tools assist in extracting key insights and generating content based on feedback data.

https://developers.google.com/ai

🎬 **DistilBERT Model**:

The **DistilBERT** model is a smaller version of BERT used for fast and efficient text classification tasks like sentiment analysis.

https://huggingface.co/transformers/model_doc/distilbert.html

APPENDIX

LINK :

https://github.com/SEC-NLP-2026-C/nlp-mini-project-gform-response-summarizer

This project utilizes key tools and technologies such as Hugging Face Transformers, Pandas, Matplotlib, FPDF, and Gradio to streamline feedback analysis. The tool supports .xlsx files as input and generates comprehensive PDF reports as output. Abbreviations used include AI (Artificial Intelligence), NLP (Natural Language Processing), and PDF (Portable Document Format). It requires Python 3.6 or higher, Flask, and Gradio, and is compatible with Windows, macOS, and Linux. While the tool delivers efficient analysis, its reliance on internet-based APIs and model accuracy may present limitations.