QUALITY ASSESSMENT USING LARGE LANGUAGE MODEL AND PROMPT ENGINEERING

A PROJECT REPORT

Submitted by

MUKESH KUMAR A (311621243014)

GOKULNATH G (311621243007)

in partial fulfilment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

MISRIMAL NAVAJEE MUNOTH JAIN ENGINEERING COLLEGE

THORAIPAKKAM, CHENNAI – 600097

ANNA UNIVERSITY: CHENNAI 600 025

MAY 2025

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report **"Quality Assessment using Large Language Model and Prompt Engineering"** is the bonafide work of **"Mukesh Kumar A (311621243014) & Gokulnath G (311621243007) "** who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
MRS. A. JEYANTHI M.E, (Ph.D.)	MRS. A. JEYANTHI M.E, (Ph.D.)
HEAD OF THE DEPARTMENT	**SUPERVISOR**
	ASSOCIATE PROFESSOR
Department of Artificial	Department of Artificial
Intelligence And Data Science	Intelligence And Data Science
Misrimal Navajee Munoth Jain	Misrimal Navajee Munoth Jain
Engineering College,	Engineering College,
Thoraipakkam, Chennai-600097	Thoraipakkam, Chennai-600097

Submitted for the Project Viva –voce examination held on ____________

INTERNAL EXAMINER **EXTERNAL EXAMINER**

MISRIMAL NAVAJEE MUNOTH JAIN ENGINEERING COLLEGE

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

- **VISION**

To produce high quality, creative and ethical engineers and technologists contributing effectively to the ever-advancing Artificial Intelligence and Data Science field.

- **MISSION**

To educate future software engineers with strong fundamentals by continuously improving the teaching-learning methodologies using contemporary AI & DS. To produce ethical engineers/researchers by instilling the values of humility, humaneness, honesty and courage to serve society. To create a knowledge hub of Artificial Intelligence and Data Science with everlasting urge to learn by developing, maintaining and continuously improving the resource Artificial Intelligence / Data Science.

ACKNOWLEDGEMENT

We express our sincere gratitude to our honourable **Secretary** (Administration) **Dr. Harish L Metha** and Secretary (Academic**) Shri. L. Jaswant Munoth** for providing the infrastructure facilities to do this project during our course period.

We thank our Principal, **Dr. C. Chandrasekar Christopher**, for providing the infrastructure and resources required for the completion of this project.

We thank our Head of department & supervisor Mrs. **A.Jeyanthi M.E., (Ph.D.)** for her valuable suggestions for the development and completion of this project. We thank her for her wholehearted guidance right from the inception to the completion of our project.

Finally, we thank all the Teaching and Non-Teaching Staff members of Artificial Intelligence and Data Science department who helped us to complete our project.

Above all we thank the Almighty, our parents for their constant support and encouragement for completing this project.

ABSTRACT

We present an intelligent framework that combines web scraping, domain-based sentiment classification, and Large Language Models (LLMs) to assess product quality from customer reviews in the e-commerce domain. The system uses automated data extraction techniques to gather real-time reviews from platforms like Flipkart, followed by contextual categorization into predefined domains such as durability, comfort, and performance. Sentiment analysis is performed using **DeepSeek-R1**, a state-of-the-art LLM **hosted and accelerated by the Groq cloud service provider**, enabling fast and context-aware interpretation of sentiment across varied product categories. The pipeline is highly automated, transforming unstructured review text into structured insights, which are then visualized via interactive dashboards developed in PyQt5 and Matplotlib. These dashboards facilitate intuitive exploration of sentiment trends and product-specific insights.

Our approach addresses the scalability, accuracy, and contextual sensitivity challenges faced by traditional sentiment analysis techniques. A comprehensive evaluation was conducted using a manually labeled dataset of 50 reviews across three products. The proposed system achieved an **accuracy of 98%**, **precision of 0.99**, **recall of 0.97**, **F1 score of 0.98**, and an outstanding **Cohen's Kappa score of 0.953**, signifying near-perfect agreement with human-reviewed labels and validating the reliability of the LLM-based classification approach.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

S.NO	ABBREVIATIONS	EXPANSION
1.	NLP	Natural Language Processing
2.	LLM	Large Language Model
3.	API	Application Programming Interface
4.	JSON	Java Script Object Notation
5.	GPU	Graphics Processing Unit
6.	CSV	Comma-Separated Values
7.	SSD	Solid State Drive

CHAPTER 1

INTRODUCTION

Traditional review analysis methods often rely on rule-based Natural Language Processing (NLP) techniques or basic machine learning models. While these methods have been widely adopted for processing customer reviews, they are inherently limited by their inability to capture the full spectrum of linguistic nuance. Standard NLP techniques frequently struggle with complex sentence structures, idiomatic expressions, sarcasm, and contextual subtleties, leading to misinterpretations of the actual sentiment. Additionally, many existing models offer broad sentiment ratings that overlook crucial domain-specific factors such as durability, usability, and comfort, which are essential for extracting precise insights. As the volume of online reviews continues to grow exponentially, these traditional approaches face significant challenges in processing data in real time, and they often fail to deliver dynamic visualizations that facilitate rapid and informed decision-making.

Our research introduces a comprehensive hybrid approach that addresses these shortcomings by integrating advanced deep learning algorithms with state-of-theart NLP techniques. By leveraging powerful Large Language Models, our method discerns subtle contextual cues and complex semantic patterns within extensive datasets, thereby enabling a more refined sentiment classification.

This refined analysis allows us to segment reviews into granular, predefined domains that reflect specific product attributes, ensuring that sentiment interpretation is both detailed and context-aware. Beyond enhancing classification accuracy, our solution automates the entire analytical pipeline—from web scraping and data extraction to real-time processing and interactive visualization—thus ensuring both scalability and operational efficiency for businesses managing vast amounts of feedback.

Furthermore, our approach delves deeper into the layers of sentiment by not only identifying explicit emotional indicators but also uncovering implicit sentiments that are often conveyed indirectly. The integration of advanced natural language processing techniques enables our system to detect nuanced expressions and subtle opinions that traditional models might miss. Complementing this sophisticated analysis are interactive dashboards that provide stakeholders with clear, actionable insights through dynamic visual representations. These dashboards offer real-time monitoring of sentiment trends, enable comparative analysis across different product domains, and utilize predictive analytics to forecast future consumer behaviors. Such insights empower businesses to make data-driven decisions that refine product offerings, enhance customer service, and maintain a competitive edge in an increasingly data-centric marketplace.

In summary, our hybrid framework marks a significant advancement over conventional review analysis methods. It overcomes the inherent limitations of traditional approaches by offering a nuanced, scalable, and automated solution for sentiment analysis in the e-commerce domain. As the digital landscape continues to evolve, our approach provides a robust tool for businesses aiming to fully harness the potential of customer feedback, ultimately driving enhanced product quality, improved user experience, and sustained market competitiveness.

\

CHAPTER 2

LITERATURE SURVEY

Falatouri. T et al (2024) [1] This paper delves into the application of large language models for assessing service quality using user-generated content such as reviews and social media feedback. It presents a framework that uses LLMs to extract service attributes, evaluate sentiment, and generate actionable quality scores. The approach improves upon traditional keyword-based sentiment methods by leveraging the contextual awareness and generative capabilities of models like GPT. The paper demonstrates its framework on datasets from hospitality and healthcare, showing improved precision in detecting service-related concerns. It positions LLMs as transformative tools for customer experience management and real-time feedback analysis. The study concludes by emphasizing the value of LLMs in domains where customer opinions are diverse, unstructured, and sentimentally complex.

Ishtiaq. A et al (2024) [2] This research introduces a novel approach for detecting helpful product reviews by combining BERT-based embeddings with class probability features. Unlike conventional sentiment analysis, it focuses on the *helpfulness* of a review, assessing whether it provides informative content that aids customer decision-making. The authors propose a hybrid model that integrates semantic representations from BERT with statistical learning features to improve classification accuracy. Experimental results on Amazon and Yelp datasets reveal superior performance compared to traditional models and even some standalone transformers. The paper demonstrates the importance of feature fusion for review quality detection and opens new pathways for enhancing content filtering and recommendation systems. It offers a practical solution for platforms aiming to surface the most valuable user-generated content.

Konstantinos I. Roumeliotis et al (2024) [3] This study investigates how two advanced large language models—GPT and LLaMA—perform in evaluating product reviews within the e-commerce domain. By using a large dataset of real-world customer reviews, the paper compares both models in aspects such as sentiment detection, opinion summarization, and fake review identification. It finds that GPT tends to be more effective at capturing subtle sentiment cues and generating coherent summaries, while LLaMA demonstrates competitive performance with faster inference and lower computational costs. The paper underscores the growing role of LLMs in automating review moderation and improving user trust in online platforms. It concludes that model selection should be based on use-case needs, balancing accuracy, speed, and infrastructure constraints.

Lossio-Ventura JA et al (2024) [4] This paper explores the effectiveness of large language models like ChatGPT and OPT in performing sentiment analysis on COVID-19 survey data, comparing their performance against traditional tools such as VADER and TextBlob. The study highlights the superior accuracy and contextual understanding of LLMs, particularly in handling nuanced or ambiguous sentiments found in open-ended survey responses. It emphasizes how fine-tuning OPT further enhances its classification precision. The paper provides a clear methodology for benchmarking LLMs and concludes that transformer-based models outperform rule-based tools in both precision and recall. The research offers valuable insights into the transition from traditional sentiment tools to modern LLM-based approaches, especially in public health and social research contexts where language complexity is high.

Rashid, Aamir & Huang et al (2024) [5] This paper performs sentiment analysis on consumer reviews of Amazon products using traditional machine learning techniques. It primarily relies on preprocessing methods like tokenization and stop-word removal, followed by classifiers such as Naive Bayes and Support Vector Machines. The study evaluates sentiment polarity—positive, negative, or neutral—across multiple product categories, with an aim to help customers make informed decisions and businesses improve products. Though it predates modern LLMs, the research provides a foundational understanding of sentiment analysis workflows. It highlights the limitations of classic models in handling sarcasm, ambiguity, and context sensitivity. Overall, the paper serves as a baseline for understanding the evolution of sentiment analysis techniques in the e-commerce domain.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

3.1.1 Overview

The existing system for e-commerce review analysis primarily relies on traditional sentiment analysis models, keyword-based classifications, and manual review processing. These conventional methods depend on lexiconbased approaches, statistical models, and early machine learning techniques such as Naïve Bayes and Support Vector Machines (SVM). While these models provide basic sentiment classification, they struggle with interpreting contextual nuances, sarcasm, and domain-specific language. Additionally, rule-based sentiment analysis often fails to adapt to evolving language patterns and usergenerated content, resulting in poor generalization across different e-commerce domains.

Furthermore, many businesses still use sentiment analysis tools that operate with fixed rule sets and static keyword lists, leading to low adaptability and inconsistent results. The absence of deep contextual understanding in these models results in frequent misclassification, especially in ambiguous or mixedsentiment reviews. Another major challenge is the reliance on manual review processing, where human intervention is required to validate sentiment predictions, making the system time-consuming, labor-intensive, and prone to subjectivity.

3.1.2 Sentiment Classification Methods

3.1.2.1. Rule-Based Methods

- Lexicon-Based Approach

   - Uses sentiment lexicons containing pre-labeled words (positive, negative, neutral).
   - Example: SentiWordNet, VADER, AFINN.

- Rule-Based System

   - Implements manually crafted rules based on keywords, negation handling, and syntactic patterns.
   - Example: Assigning negative sentiment if "not good" appears in a review.

3.1.2.2 Machine Learning Methods

- Traditional Machine Learning Methods

   - Naïve Bayes: Uses probability-based classification based on word frequencies.

   - Support Vector Machines (SVM): Finds optimal decision boundaries for sentiment classes.

    - Random Forest: Uses multiple decision trees for robust sentiment classification.
    - Logistic Regression: A statistical model for binary sentiment classification.

- Deep Learning Methods

    - Recurrent Neural Networks (RNN): Processes sequential data for better text understanding.
    - Long Short-Term Memory (LSTM): Captures long-range dependencies in sentiment text.
    - Transformers: Advanced models like BERT and GPT that use attention mechanisms.
    - Large Language Models (LLM): AI-driven models that understand context and emotions in text.

3.1.2.3 Hybrid Methods

- Lexicon + ML: Enhances ML models with predefined sentiment lexicons.
- Lexicon + Deep Learning: Combines lexicon-based methods with deep learning models for better context understanding.
- ML + DL: Uses both machine learning and deep learning for enhanced classification.

3.1.2.4 Advanced Methods

- Aspect-Based Sentiment Analysis: Analyzes sentiment for specific product features (e.g., quality, durability).
- Multi-Modal Sentiment Analysis: Uses text, images, and audio for a comprehensive sentiment evaluation.
- Fine-Grained Sentiment Analysis: Classifies emotions beyond positive/negative, such as joy, anger, and sadness.
- Language-Specific Sentiment Analysis: Tailors sentiment models to different languages, considering cultural nuances.

3.1.2 Drawbacks of Existing System

1. Limited Accuracy: Traditional rule-based models and keyword matching often fail to interpret nuanced meanings, leading to misclassification of sentiments.
2. Scalability Issues: Handling vast datasets manually or through outdated models results in slow processing speeds and inefficiency.
3. Lack of Automation: Many traditional systems require human intervention for classification, making the process time-consuming and expensive.

4. Hardware Constraints: Running advanced sentiment analysis models locally requires high-end computational resources, which may not be feasible for small businesses.
5. Inconsistency in Sentiment Analysis: Variability among different sentiment analysis tools leads to unreliable results, as highlighted in comparative studies.
6. Domain-Specific Challenges: Standard NLP models struggle to classify domain-based sentiments accurately, failing to recognize productspecific terminology and user intent.

3.2 PROPOSED SYSTEM

3.2.1 Overview

The proposed system introduces a hybrid framework for e-commerce review analysis, designed to overcome the inefficiencies of traditional sentiment analysis techniques. This system leverages advanced machine learning, deep learning, and large language models (LLMs) to improve accuracy, efficiency, and scalability in analyzing e-commerce product reviews. By integrating stateof-the-art natural language processing (NLP), web scraping, and domainspecific sentiment classification, the system ensures a context-aware approach to sentiment analysis.

One of the key innovations of the proposed system is the incorporation of DeepSeek-R1, a high-performance large language model (LLM) optimized for sentiment analysis. Unlike conventional NLP techniques, DeepSeek-R1 understands contextual sentiment, identifies subtle language cues, and adapts to domain-specific terminology, thereby improving the precision of sentiment classification.

The system also employs batch processing techniques to efficiently handle large datasets, reducing computational overhead and processing delays.

Additionally, the system is designed to be highly automated, leveraging web scraping techniques to extract real-time product reviews from e-commerce platforms. The sentiment analysis results are then structured into JSON-based intermediate processing for seamless data handling. Finally, interactive dashboards provide a user-friendly interface for businesses to gain actionable insights, enabling data-driven decision-making for product improvements and marketing strategies.

3.2.2 Advantages of Proposed System

1. Improved Accuracy: The DeepSeek-R1 model enhances contextual sentiment understanding and domain classification.
2. Automation & Scalability: Web scraping automates data extraction, reducing manual effort, while batch processing optimizes large-scale data handling.
3. Optimized LLM Integration: By leveraging cloud-based Groq LPU, the system achieves faster processing times, overcoming local hardware limitations.
4. Enhanced Output Formatting: Intermediate JSON-based processing ensures structured data handling and compatibility with CSV workflows.
5. Real-Time Insights: The final processed data is visualized using interactive dashboards, providing actionable business intelligence for decision-makers.

3.3 REQUIREMENT ANALYSIS

Requirement analysis is an important step in software development that helps to define the scope of the project and identify the key features required for the system to meet the needs of its users.

3.3.1 Hardware Requirements

The system requires a combination of local and cloud-based infrastructure to ensure efficient performance and scalability.

Local Development:

1. Processor: Intel Core i5/i7 or AMD Ryzen 5/7 (or higher)
2. RAM: Minimum 8GB (16GB recommended)
3. Storage: Minimum 256GB SSD (512GB SSD recommended)
4. GPU: GTX 1660S (Used for local LLM inference)

Cloud Deployment:

- Groq LPU Service: Used for remote LLM processing to improve efficiency and overcome local hardware limitations.

3.3.2 Hardware Specification

To ensure smooth execution, the system utilizes a hybrid computing setup:

- Local System: Handles web scraping, data preprocessing, and result visualization.
- Cloud Processing: Offloads LLM-based sentiment analysis to Groq Cloud for faster inference and scalability.

3.3.3 Software Requirements

The system is built using a variety of software tools and frameworks to facilitate automation, analysis, and visualization.

1. Operating System: Windows 10/11, Linux (Ubuntu 20.04+), or macOS
2. Programming Languages: Python (Primary)
3. Frameworks & Libraries:
    - Web Scraping: BeautifulSoup, Scrapy
    - Sentiment Analysis: Transformers (Hugging Face), DeepSeek-R1,
    - CLOUD API Integration: Groq Cloud API for remote LLM processing
    - File Handling: Pandas for CSV to JSON conversion
    - Dashboard Development: Pyqt5
    - Hosting Services: ngrok (for exposing local LLM if needed)

3.4 TOOLS USED

3.4.1 Python Programming

1. Programming Language – Python:

Python serves as the backbone of this E-Commerce Review Analysis System due to its versatility, simplicity, and extensive library support. It provides an efficient environment for web scraping, machine learning, sentiment analysis, GUI development, and real-time data visualization. Additionally, Python facilitates seamless API integration, making it the ideal choice for connecting with cloud-based services.

2. User Interface Development - PyQt5:

The system's graphical user interface (GUI) is developed using PyQt5, a powerful Python binding for the Qt framework. PyQt5 offers pre-built UI components such as buttons, dropdown menus, and input fields, enabling a user-friendly design. The QStackedWidget feature allows smooth navigation between multiple pages, enhancing the overall user experience. Moreover, PyQt5 provides extensive styling options, ensuring an aesthetically pleasing and intuitive interface.

3. Data Handling – Pandas:

Structured data storage and retrieval are handled using JSON (JavaScript Object Notation). JSON is lightweight, making it highly efficient for storing review data, product trends, and sentiment scores. Its flexibility allows for seamless data interchange between system components and external sources. The ability to store nested data structures makes JSON particularly useful for handling complex datasets without requiring a traditional relational database.

4. Data Visualization – Matplotlib:

The system uses Matplotlib for data visualization, allowing users to analyze e-commerce trends through graphical representations. It supports various chart types, including bar charts, line graphs, scatter plots, pie charts, and histograms, making it easier to interpret sentiment trends and sales patterns. Integration with PyQt5 enables the embedding of interactive visual elements within the GUI, improving the accessibility of insights for end users.

5. Web Scraping - BeautifulSoup & Scrapy (Planned Integration):

Automating data collection from e-commerce platforms is made possible using web scraping tools such as BeautifulSoup and Scrapy. BeautifulSoup is effective for parsing HTML content, extracting product reviews, ratings, and descriptions, while Scrapy is designed for large-scale automated scraping. These tools significantly reduce manual effort by gathering realtime data directly from online marketplaces, ensuring that sentiment analysis is based on up-to-date information.

6. Cloud-Based LLM Integration - Groq Cloud API:

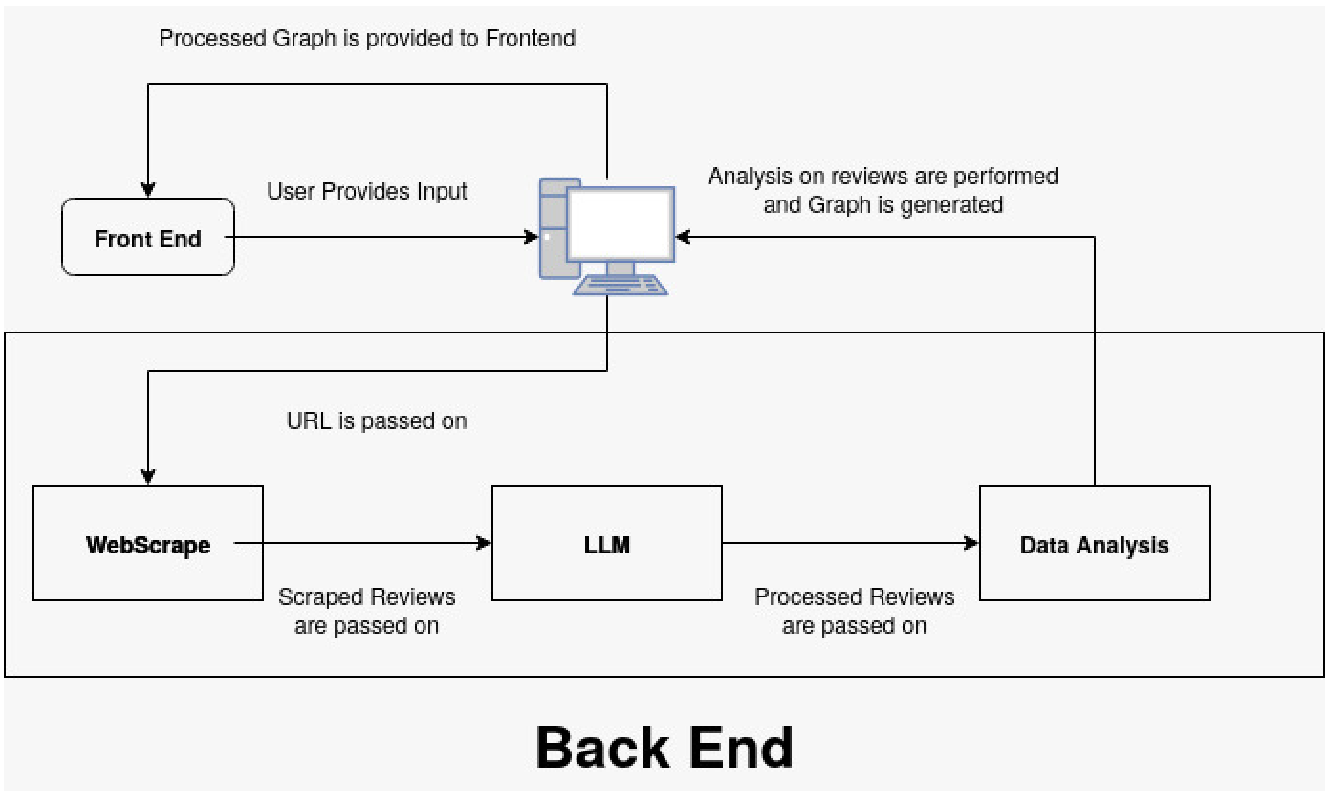
To enhance sentiment analysis, the system integrates the Groq Cloud API, enabling access to advanced Large Language Models (LLMs) like DeepSeek-R1. This cloud-based solution improves sentiment classification accuracy by understanding context and nuances within product reviews. Offloading computational processing to the cloud also ensures faster execution and reduces the reliance on local hardware resources, making it more scalable.

7. Data Processing – Pandas:

Efficient data handling and transformation are managed using Pandas, a Python library designed for structured data processing. Pandas simplifies operations such as reading JSON and CSV files, handling large datasets, and converting raw data into structured insights. Its powerful data manipulation capabilities ensure that the system processes user reviews effectively, maintaining accuracy and consistency in sentiment analysis.

CHAPTER 4

SYSTEM ARCHITECTURE



Processed Graph is provided to Frontend
User Provides Input
Front End
Analysis on reviews are performed and Graph is generated
URL is passed on
WebScrape
LLM
Data Analysis
Scraped Reviews are passed on
Processed Reviews are passed on
Back End

Figure 4.1 System Architecture

4.1 USER INTERACTION

As illustrated in Figure 4.1, The system architecture of our project is designed to ensure a modular, scalable, and interactive flow of data from user input to meaningful output. At the core of the system is the User, who interacts with the application through a graphical user interface. The user begins the process by entering a product link or selecting an e-commerce category of interest. This interaction serves as the entry point into the system's data processing pipeline. Once the input is received, the user interface manages the seamless flow of data across various backend modules and eventually returns visual insights for the user to interpret.

4.2 WEB SCRAPING

The first major backend component is the Web Scraping Module, which is responsible for collecting raw review data directly from e-commerce platforms like Flipkart. This module uses the requests library to send HTTP requests to the desired product pages and fetch the underlying HTML content. The HTML content is then parsed using BeautifulSoup (BS4), which is effective in navigating and extracting specific elements such as review titles, ratings, and full review texts. This scraped data forms the basis for further natural language analysis and is structured into a tabular format for easy processing.

4.3 LLM PROCESSING

Following data collection, the LLM Processing Module takes over. This module interfaces with a Large Language Model, such as DeepSeek, through the Groq API. The scraped reviews are fed in batches to the LLM, which is tasked with extracting relevant quality indicators, feature-specific keywords, and the sentiment (positive or negative) associated with each feature. This process transforms unstructured natural language reviews into structured insights. The use of the dotEnv library ensures secure handling of API keys and other sensitive credentials, keeping the system's configuration isolated and secure.

4.4 DATA ANALYSIS

Once the LLM processing is completed, the Data Analysis and Visualization Module comes into play. The structured output from the LLM is loaded into data structures using the Pandas library. With the help of Pandas, we can filter, group, and summarize the information to extract meaningful patterns. The Matplotlib library is then used to generate visual representations of these patterns, such as bar charts that compare the number of positive versus negative comments for each quality domain (e.g., durability, comfort, aesthetics). These visualizations are saved and dynamically displayed within the frontend, giving users a clear and concise overview of the product's quality based on aggregated customer feedback.

4.5 ANALYTICS DASHBOARD

To enable a user-friendly experience, the system includes a robust Frontend Module, built using the PyQt5 library. This desktop-based graphical interface allows users to initiate scraping, monitor processing stages, and view the final output without needing to interact directly with the code. Supporting libraries like os and sys assist in handling file system paths, launching subprocesses, and managing various environment settings required by the application. The frontend module also integrates dynamic screen updates, allowing for a smooth user experience as the backend completes different phases of the pipeline.

CHAPTER 5

IMPLEMENTATION

5.1 LLM SETUP AND INTEGRATION

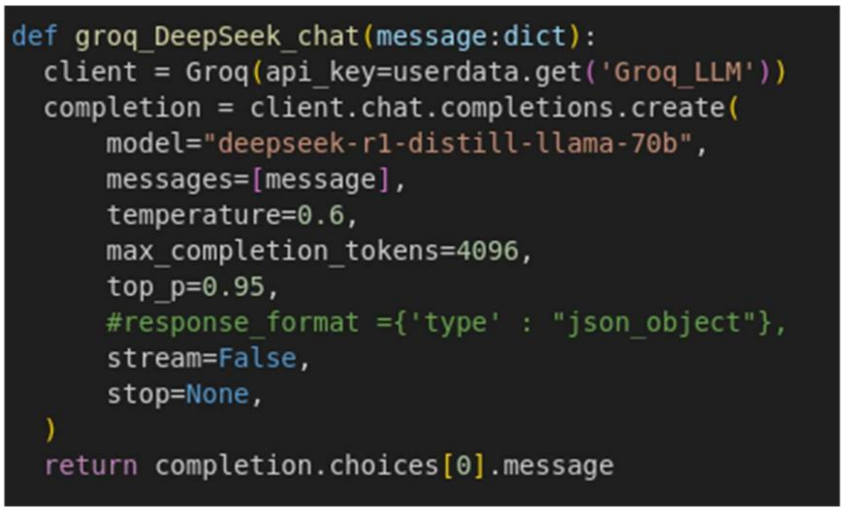
The core of the system's intelligence lies in the setup and integration of a large language model. Initially, DeepSeek-R1 was deployed locally using the Ollama platform, leveraging the GTX 1660S GPU. The model was run through the Ollama CLI, providing local inference capabilities. To expand accessibility, the system also incorporated OpenWEB-UI for user-friendly interaction and ngrok to expose the model as a remotely accessible service. This configuration allowed flexibility in running the model locally while testing deployment possibilities for broader scalability.

5.1.1 Input Module

The input module begins by scraping product reviews from e-commerce websites. These reviews are then saved into a structured CSV file. The collected data is processed in batches, where each batch contains 10–20 reviews. Using the DeepSeek-R1 model integrated through Ollama, each batch undergoes analysis for domain classification and sentiment identification. The module ensures a clean pipeline from raw review collection to structured input for the LLM.

5.1.2 Initial Plan

The original deployment strategy focused on local inference. Reviews were scraped and stored in a CSV format, then analyzed locally using DeepSeek-R1. This involved running the Ollama CLI to process batches of reviews. The OpenWEB-UI was employed for managing LLM operations visually, while ngrok was used to make the local model accessible externally. Reviews were processed in batches, typically 10–20, allowing a manageable workload for the LLM to deliver domain classification and sentiment results.



```python
def groq_DeepSeek_chat(message:dict):
  client = Groq(api_key=userdata.get('Groq_LLM'))
  completion = client.chat.completions.create(
      model="deepseek-r1-distill-llama-70b",
      messages=[message],
      temperature=0.6,
      max_completion_tokens=4096,
      top_p=0.95,
      #response_format ={'type' : "json_object"},
      stream=False,
      stop=None,
  )
  return completion.choices[0].message
```

Figure.5.1 LLM Module Code

As illustrated in Figure.5.1, This function connects to the DeepSeek LLM using the Groq API key to send a chat message for processing. It uses the model deepseek-r1-distill-llama-70b with customized parameters like temperature and token limit.

5.1.3 Identified Issues

The first issue encountered was slow performance. Processing a small batch of 10 reviews took approximately 30 to 45 seconds , which translated into over 5 minutes for 100 reviews. The GTX 1660S GPU struggled with memory and compute requirements, making the process inefficient for large-scale analysis.

The second issue was related to data handling. DeepSeek-R1, during local inference, could not directly edit or output data in CSV format. This limitation complicated the workflow, requiring an intermediate format to bridge the gap.

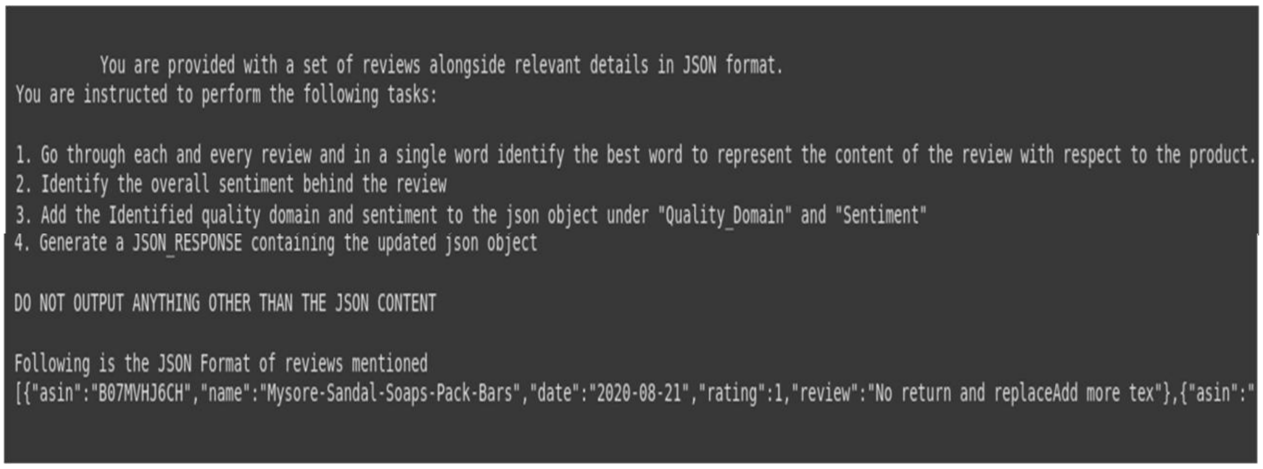
5.1.4 Optimized Solution

To overcome performance bottlenecks, the LLM processing was shifted from local infrastructure to Groq Cloud. This move significantly improved inference time, reducing it to approximately 10 seconds per batch. Additionally, batch sizes were optimized to 12 reviews per set, balancing processing speed and maintaining rate limits imposed by the cloud service. To resolve data handling issues, the CSV files were converted into JSON format before being sent to the LLM. After processing, the output was converted back into CSV using Pandas. This ensured a smooth and structured data transformation process.

5.1.5 Prompt Design

A key part of the optimization was the design of an effective prompt to ensure consistent and useful output from the LLM. The final prompt used is as follows:

1. Identify a single word representing the product quality domain for each review.
2. Determine the overall sentiment (positive, neutral, negative).
3. Append 'Quality_Domain' and 'Sentiment' fields to the JSON object.
4. Return the modified JSON object as plain text, without markdown or extra formatting."



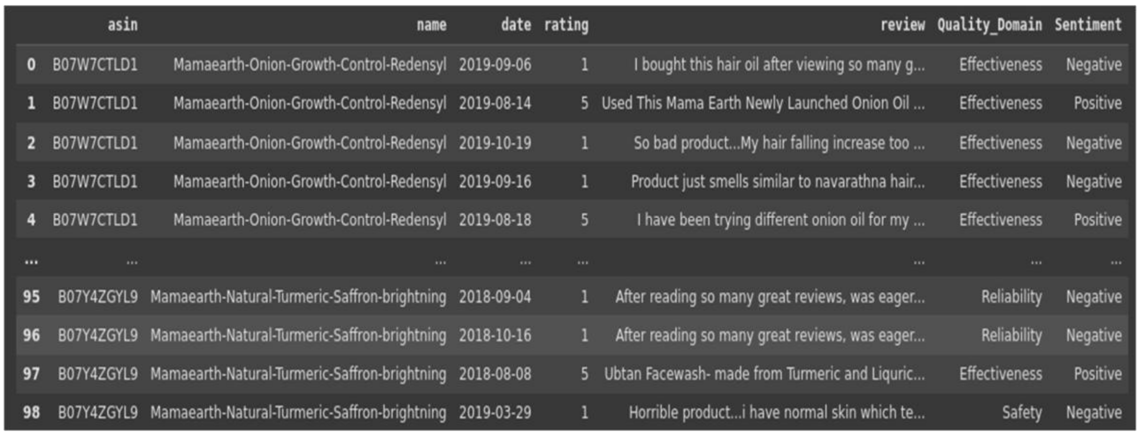
```
        You are provided with a set of reviews alongside relevant details in JSON format.
You are instructed to perform the following tasks:

1. Go through each and every review and in a single word identify the best word to represent the content of the review with respect to the product.
2. Identify the overall sentiment behind the review
3. Add the Identified quality domain and sentiment to the json object under "Quality_Domain" and "Sentiment"
4. Generate a JSON_RESPONSE containing the updated json object

DO NOT OUTPUT ANYTHING OTHER THAN THE JSON CONTENT

Following is the JSON Format of reviews mentioned
[{"asin":"B07MVHJ6CH","name":"Mysore-Sandal-Soaps-Pack-Bars","date":"2020-08-21","rating":1,"review":"No return and replaceAdd more tex"},{"asin":"
```

Figure. 5.2. Sample Prompt To LLM

As illustrated in Figure.5.2 , This prompt instructs the LLM to analyze each review, assign a relevant quality domain, and determine the sentiment. It then appends this information to the existing JSON under "Quality_Domain" and "Sentiment". The model must return only the updated JSON format.

5.1.6 Outcome of the Module

As illustrated in Figure.5.3 , The final processed output consists of a structured CSV file, where each review is annotated with a "Quality_Domain" and "Sentiment". This enriched data is then used for further analysis. It is visualized through interactive dashboards, providing stakeholders with insightful analytics. These insights empower decision-makers by offering clarity on product performance, customer satisfaction, and domain-specific feedback trends.



```
          asin                                       name        date  rating                                          review  Quality_Domain  Sentiment
0   B07W7CTLD1    Mamaearth-Onion-Growth-Control-Redensyl  2019-09-06       1   I bought this hair oil after viewing so many g...   Effectiveness   Negative
1   B07W7CTLD1    Mamaearth-Onion-Growth-Control-Redensyl  2019-08-14       5  Used This Mama Earth Newly Launched Onion Oil ...   Effectiveness   Positive
2   B07W7CTLD1    Mamaearth-Onion-Growth-Control-Redensyl  2019-10-19       1     So bad product...My hair falling increase too ...   Effectiveness   Negative
3   B07W7CTLD1    Mamaearth-Onion-Growth-Control-Redensyl  2019-09-16       1    Product just smells similar to navarathna hair...   Effectiveness   Negative
4   B07W7CTLD1    Mamaearth-Onion-Growth-Control-Redensyl  2019-08-18       5     I have been trying different onion oil for my ...   Effectiveness   Positive
...        ...                                        ...         ...     ...                                                 ...             ...        ...
95  B07Y4ZGYL9  Mamaearth-Natural-Turmeric-Saffron-brightning  2018-09-04       1  After reading so many great reviews, was eager...     Reliability   Negative
96  B07Y4ZGYL9  Mamaearth-Natural-Turmeric-Saffron-brightning  2018-10-16       1  After reading so many great reviews, was eager...     Reliability   Negative
97  B07Y4ZGYL9  Mamaearth-Natural-Turmeric-Saffron-brightning  2018-08-08       5  Ubtan Facewash- made from Turmeric and Liquric...   Effectiveness   Positive
98  B07Y4ZGYL9  Mamaearth-Natural-Turmeric-Saffron-brightning  2019-03-29       1        Horrible product...i have normal skin which te...          Safety   Negative
```

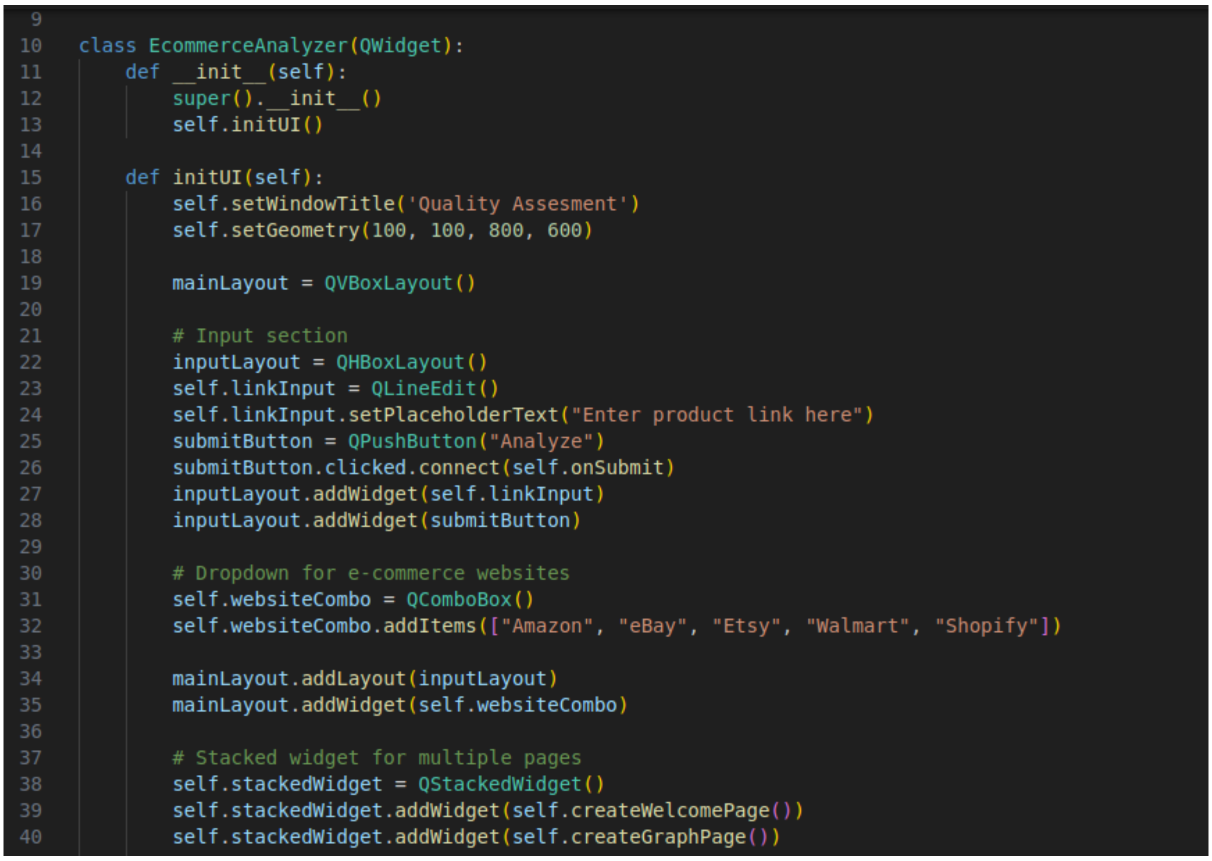
Figure. 5.3.Sample Module Output

5.2 DASHBOARD SETUP AND INTERACTION

The implementation of the proposed system is carried out using Python, primarily leveraging the PyQt5 library for GUI development and Matplotlib for data visualization. The goal is to create an interactive and user-friendly Ecommerce Review Analysis Dashboard that allows users to input product links, select the source website, and visualize sales trends and customer insights through dynamically generated graphs.

5.2.1. Application Structure:

The entire application is structured within a class called EcommerceAnalyzer, inheriting from QWidget. This class initializes the UI and manages all the visual components, including the navigation bar, user input fields, website selection dropdown, and graphical outputs.



```
class EcommerceAnalyzer(QWidget):
    def __init__(self):
        super().__init__()
        self.initUI()

    def initUI(self):
        self.setWindowTitle('Quality Assesment')
        self.setGeometry(100, 100, 800, 600)

        mainLayout = QVBoxLayout()

        # Input section
        inputLayout = QHBoxLayout()
        self.linkInput = QLineEdit()
        self.linkInput.setPlaceholderText("Enter product link here")
        submitButton = QPushButton("Analyze")
        submitButton.clicked.connect(self.onSubmit)
        inputLayout.addWidget(self.linkInput)
        inputLayout.addWidget(submitButton)

        # Dropdown for e-commerce websites
        self.websiteCombo = QComboBox()
        self.websiteCombo.addItems(["Amazon", "eBay", "Etsy", "Walmart", "Shopify"])

        mainLayout.addLayout(inputLayout)
        mainLayout.addWidget(self.websiteCombo)

        # Stacked widget for multiple pages
        self.stackedWidget = QStackedWidget()
        self.stackedWidget.addWidget(self.createWelcomePage())
        self.stackedWidget.addWidget(self.createGraphPage())
```

Figure.5.4 Sample Frontend Dashboard Code

As illustrated in Figure.5.4 ,This PyQt5 code defines a GUI class EcommerceAnalyzer for product quality assessment. It sets up a main window with input fields for a product link, an "Analyze" button, and a dropdown to choose an e-commerce site (e.g., Amazon, Etsy). The interface uses stacked widgets to manage multiple pages like a welcome and graph view, forming the foundation for a dynamic and interactive sentiment analysis dashboard.

5.2.2 Navigation and Layout:

The GUI employs vertical layout (QVBoxLayout) that includes a navigation bar created using horizontal layouts (QHBoxLayout) and styled using CSS for a modern look. Navigation buttons such as Home and Info provide basic page switching functionalities.

5.2.3. Input Section:

Users can enter a product link using QLineEdit, select the platform (Flipkart, Amazon, eBay, etc.) using a QComboBox, and trigger analysis with a Submit button (QPushButton). The data input by users is collected to display relevant graphical information from a pre-existing dataset.

5.2.4. Page Management:

A QStackedWidget is used to manage multiple views within the application. Two main pages are created:

- A welcome page (createHomePage) for initial navigation.
- A graph display page (createGraphPage) that showcases five types of charts.

5.2.5. Dynamic Graph Generation:

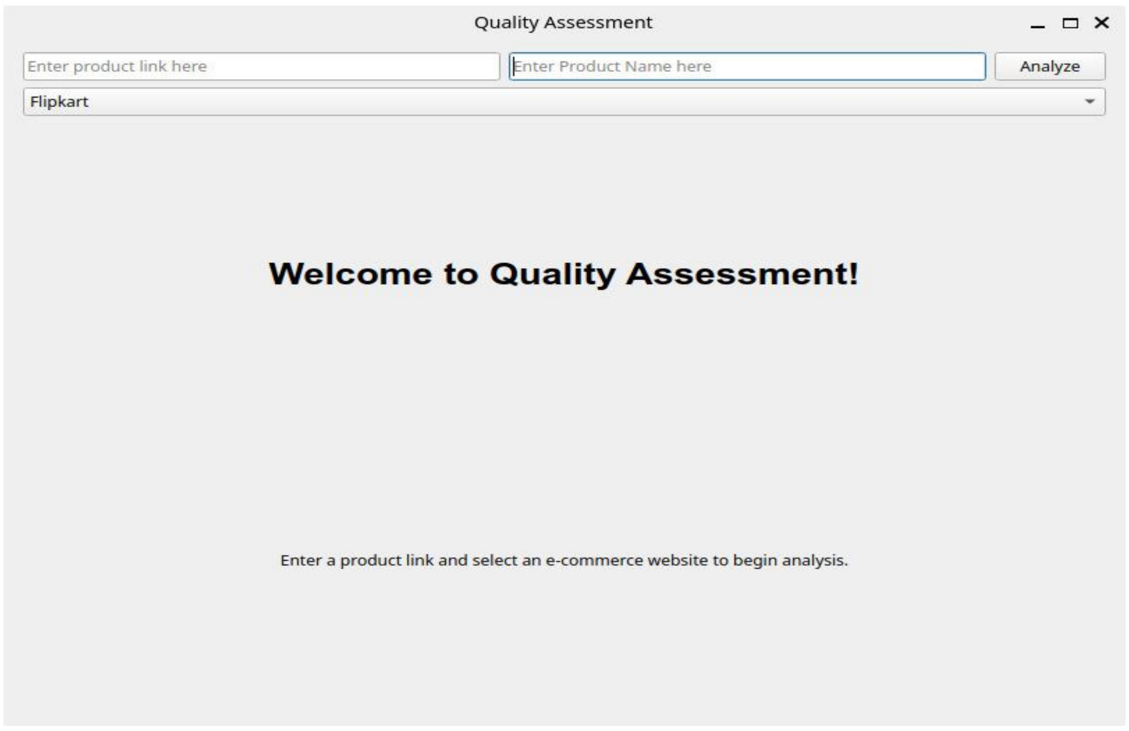
The system utilizes matplotlib.pyplot and Figure Canvas to embed five chart types: bar, line, scatter, pie, and histogram. These charts are dynamically updated based on the JSON data corresponding to the selected e-commerce website.

5.2.6. Data Handling:

The function updateGraphs reads from a static JSON file containing structured data (sample.json). It extracts sales trends under the selected platform and uses the values to render visual insights. Proper error handling is implemented to manage missing or malformed files.

5.2.7. Execution:

The application is executed using a PyQt5 event loop, and the main window is displayed through Qapplication.



Welcome to Quality Assessment!

Enter a product link and select an e-commerce website to begin analysis.

Figure.5.5. Sample Frontend Dashboard

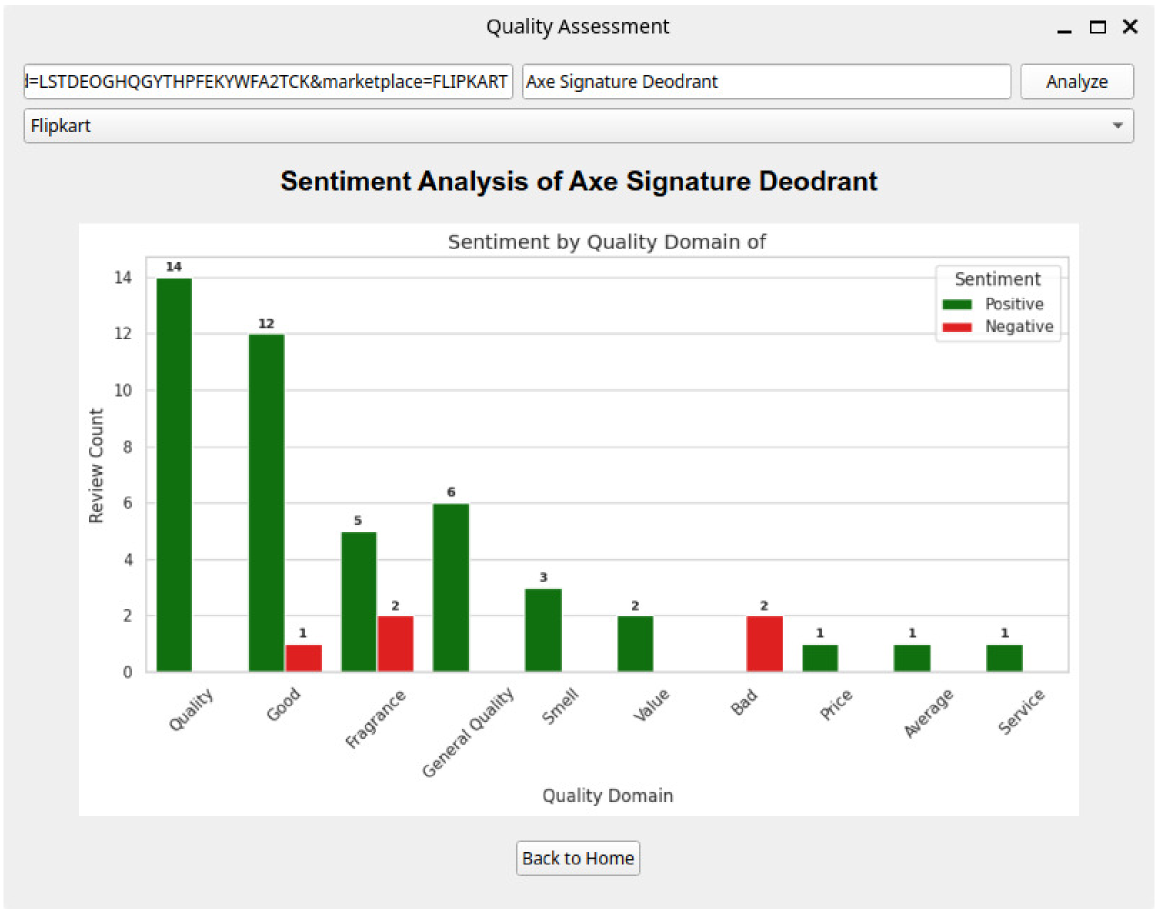


Figure.5.6 Sample Infographic Results Dashboard

As Illustrated in Figure.5.5 and Figure.5.6, The dashboard includes two main screens: a welcome screen for entering the product link, name, and selecting the e-commerce platform; and a result screen that displays sentiment analysis in a bar chart format. The chart shows sentiment distribution across quality domains using colour codes—green for positive and red for negative. This setup ensures a smooth, interactive, and user-friendly analysis experience.

5.3 WEB-SCRAPING REVIEWS

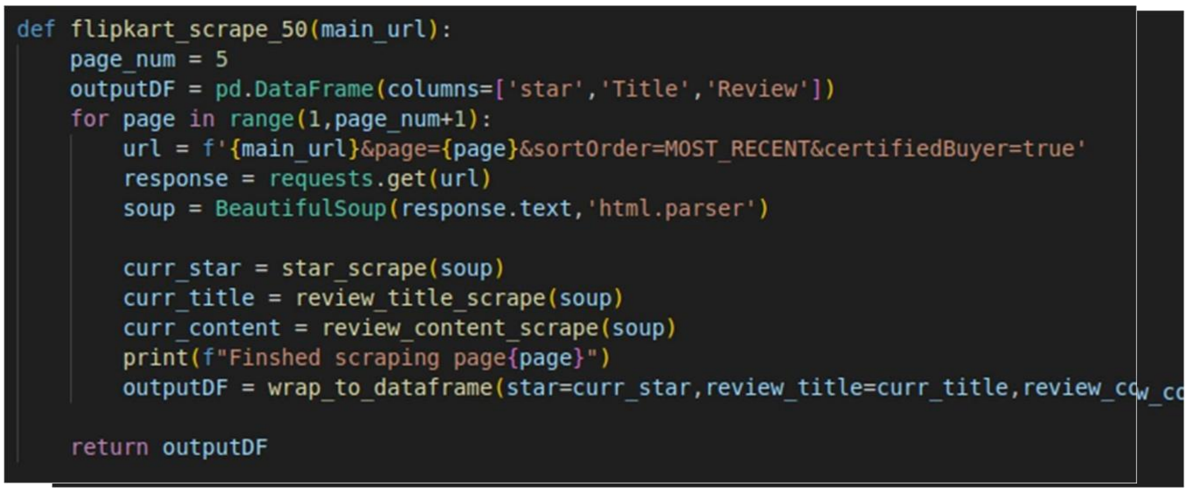
As part of our review analysis pipeline, we implemented a custom web scraping module to extract structured review data from product pages on Flipkart. The scraping script was developed in Python using the requests, BeautifulSoup, and pandas libraries.

The implemented program focuses on collecting user review data for a specified product. The process is encapsulated in a main function, flipkart_scrape_50, which is designed to retrieve up to 50 reviews. The function constructs paginated URLs dynamically and sends HTTP requests to fetch the HTML content of each page.

Three core scraping functions are defined:

- star_scrape(soup): Extracts the star ratings of each review using the appropriate CSS class identifiers.
- review_title_scrape(soup): Captures the title or headline of each user review.
- review_content_scrape(soup): Extracts the main body text of the review. Since each review ends with metadata such as the date and helpfulness, the script trims unnecessary trailing characters to isolate clean review content.

After data extraction, the wrap_to_dataframe function is used to aggregate the star rating, title, and review content into a pandas DataFrame. This structure allows for easy downstream processing, including sentiment analysis, aspectbased classification, and data visualization.



```
def flipkart_scrape_50(main_url):
    page_num = 5
    outputDF = pd.DataFrame(columns=['star','Title','Review'])
    for page in range(1,page_num+1):
        url = f'{main_url}&page={page}&sortOrder=MOST_RECENT&certifiedBuyer=true'
        response = requests.get(url)
        soup = BeautifulSoup(response.text,'html.parser')

        curr_star = star_scrape(soup)
        curr_title = review_title_scrape(soup)
        curr_content = review_content_scrape(soup)
        print(f"Finshed scraping page{page}")
        outputDF = wrap_to_dataframe(star=curr_star,review_title=curr_title,review_cow_co

    return outputDF
```

Figure.5.6 Example Flipkart Web Scrape Code

As illustrated in Figure.5.6 ,This Python function flipkart_scrape_50 scrapes up to 5 pages of product reviews from a Flipkart URL. It collects star ratings, review titles, and content using BeautifulSoup, processes each page in a loop, and compiles the data into a structured DataFrame outputDF for downstream sentiment analysis

5.4. DATA ANALYSIS

The data analysis phase in our project is designed to transform raw, scraped product reviews into structured insights that assess product quality based on customer feedback. This involves a multi-step pipeline integrating preprocessing, categorization, sentiment analysis, and visualization.

- Data Acquisition
  - Product reviews were scraped from Amazon using automated tools.
  - Each review includes product name, review text, and metadata.

▪ Data Preprocessing

o Cleaning operations included removing HTML tags, emojis, and special characters.

o Reviews were tokenized and normalized for further processing.

▪ LLM-Based Annotation

o A Large Language Model (LLM) was used to classify each review into:

▪ Quality Domain (e.g., packaging, usability, performance) o Sentiment (Positive, Negative, Neutral)

o This enriched the dataset with meaningful qualitative metadata.

▪ Exploratory Data Analysis (EDA)

o Distribution of sentiments across products and quality domains was visualized using bar plots.

o Count plots revealed which quality aspects were most praised or criticized per product.

o Neutral sentiments were also tracked to identify ambiguous or mixed reviews.

▪ Visualization

o Matplotlib and Seaborn were used for plotting: o Sentiment distribution per product.

▪ Sentiment distribution across quality domains.

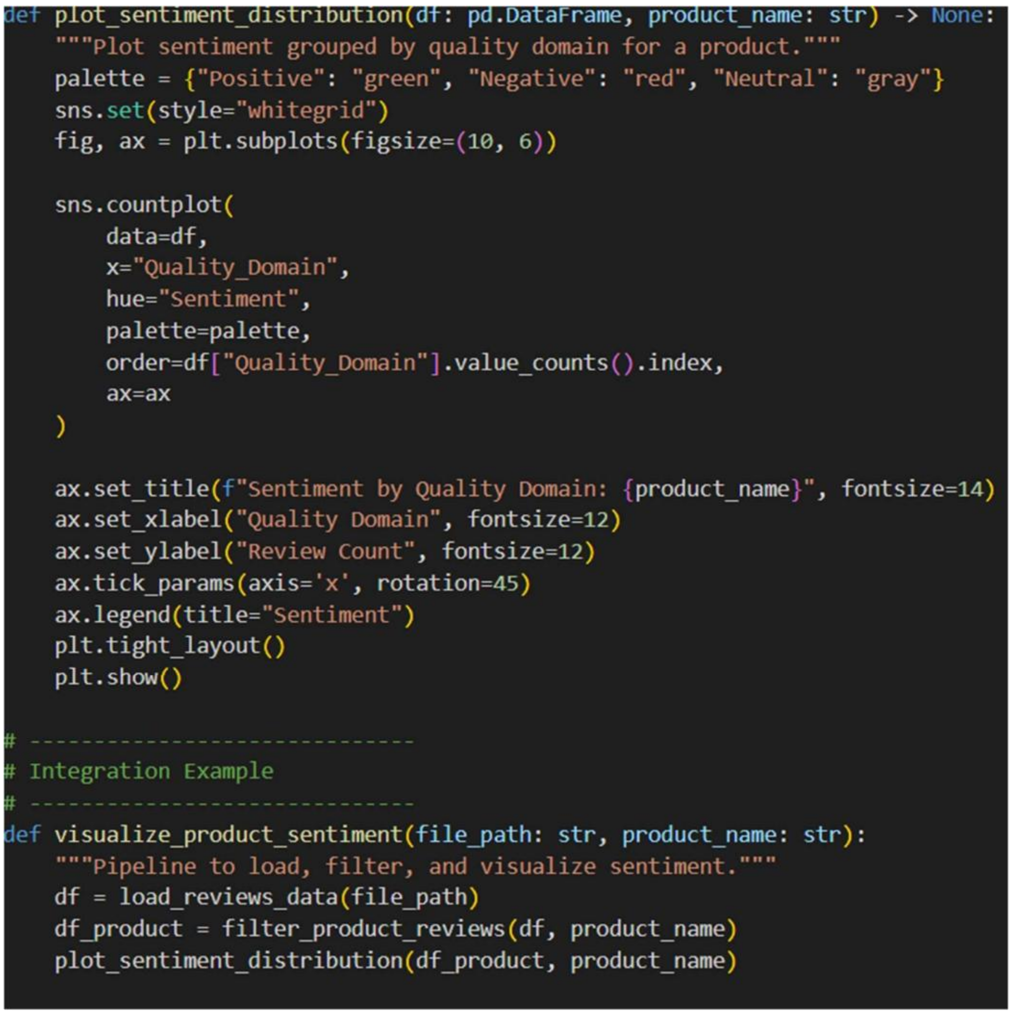
o Visual outputs helped interpret which product features drive customer satisfaction or complaints.

▪ Key Insights

o Specific quality domains consistently received negative sentiment for some products, signalling improvement areas.

o Products with balanced sentiment showed opportunity for better positioning or targeted enhancements.

The script concludes by analysing the given data and visualizing the sentiment analysis in graphical inference.



```python
def plot_sentiment_distribution(df: pd.DataFrame, product_name: str) -> None:
    """Plot sentiment grouped by quality domain for a product."""
    palette = {"Positive": "green", "Negative": "red", "Neutral": "gray"}
    sns.set(style="whitegrid")
    fig, ax = plt.subplots(figsize=(10, 6))

    sns.countplot(
        data=df,
        x="Quality_Domain",
        hue="Sentiment",
        palette=palette,
        order=df["Quality_Domain"].value_counts().index,
        ax=ax
    )

    ax.set_title(f"Sentiment by Quality Domain: {product_name}", fontsize=14)
    ax.set_xlabel("Quality Domain", fontsize=12)
    ax.set_ylabel("Review Count", fontsize=12)
    ax.tick_params(axis='x', rotation=45)
    ax.legend(title="Sentiment")
    plt.tight_layout()
    plt.show()

# -----------------------------
# Integration Example
# -----------------------------
def visualize_product_sentiment(file_path: str, product_name: str):
    """Pipeline to load, filter, and visualize sentiment."""
    df = load_reviews_data(file_path)
    df_product = filter_product_reviews(df, product_name)
    plot_sentiment_distribution(df_product, product_name)
```

Figure.5.7 Sample Data Analysis code

As illustrated in Figure.5.4 , This code snippet defines a sentiment visualization pipeline for product reviews. The plot_sentiment_distribution function uses seaborn to display a countplot of sentiment (Positive, Negative, Neutral) grouped by quality domains. The visualize_product_sentiment function loads the data, filters reviews by product, and invokes the plotting function to generate a sentiment distribution chart for a given product.

PSEUDO CODE

5.1 Main Application Structure

```
Class EcommerceAnalyzer:
    Initialize UI components:
        - Create window with title "Quality Assessment"
        - Create input fields for product link and product name
        - Create dropdown for selecting e-commerce website (Flipkart,
Amazon, etc.)
        - Create "Analyze" button
        - Create stacked widget with welcome page and loading screen

    Function createWelcomePage:
        Create welcome widget with:
            - Title label "Welcome to Quality Assessment!"
            - Instruction label for user guidance
        Return welcome widget

    Function onSubmit:
        Get link from input field
        Get selected website from dropdown
        Get product name from input field
        Display loading screen
        Create and start worker thread with link, website, and product name

    Function processingDone:
        Create graph window with analysis results
        Display graph window

    Function show_graph_window:
        Create widget to display graph image
        Add title label
        Load and display image from file path
        Add back button to return to welcome page
        Return graph widget
```

5.2 Web Scraping Module (draft_code.py)

```
Class Webscrape:
   Function star_scrape(soup):
      Initialize empty array for star ratings
      Find all elements with class 'XQDdHH'
      For each element:
         Extract text and strip whitespace
         Add to star ratings array
      Return star ratings array

   Function review_title_scrape(soup):
      Initialize empty array for review titles
      Find all 'p' elements with class 'z9E0IG'
      For each element:
         Extract text and strip whitespace
         Add to review titles array
      Return review titles array

   Function review_content_scrape(soup):
      Initialize empty array for review content
      Find all elements with class 'ZmyHeo'
      For each element:
         Extract text, strip whitespace, and remove last 9 characters
         Add to review content array
      Return review content array

    Function wrap_to_dataframe(star, review_title, review_content,
output_Df):
      For each triplet of star, title, and review_text:

         Append new row to output_Df with these values
      Return updated dataframe
    Function flipkart_scrape_50(main_url):
      Set page_num to 5
    Create empty dataframe with columns 'star', 'Title', 'Review'

    For page from 1 to page_num:
        Construct URL with page number and filters
        Send HTTP request to URL
        Parse HTML response with BeautifulSoup

        Call star_scrape to extract star ratings
        Call review_title_scrape to extract review titles
        Call review_content_scrape to extract review content

        Call wrap_to_dataframe to add extracted data to dataframe

    Return completed dataframe with all reviews
```

5.3 LLM Processing Module (LLM_Module.py)

```
Function groq_DeepSeek_chat(message):
    Initialize Groq client with API key
    Create chat completion with:
        - Model: "deepseek-r1-distill-llama-70b"
        - Messages: the input message
        - Temperature: 0.6
        - Max tokens: 4096
    Return completion message

Function response_Cleanup(response):
    Find position of "json" or "</think>" in response content

    Extract JSON content from response
    Return extracted JSON string

Function json_to_dataframe(json):
    Convert JSON string to pandas dataframe
    Return dataframe
```

```
Function run_deepseekLLM_50_batch(dataFrame):
   Set chunk_size to 10


   Split dataFrame into chunks of size chunk_size
   Initialize empty array for responses

   For each chunk:
      Convert chunk to JSON format
      Create prompt asking LLM to:
         1. Identify quality domain for each review
         2. Classify sentiment as "Positive" or "Negative"
         3. Add these classifications to the JSON

      Call groq_DeepSeek_chat with the prompt
      Clean up the response with response_Cleanup
      Add cleaned response to responses array

   Initialize empty output dataframe

   For each response:
      Convert JSON response to dataframe
      Concatenate with output dataframe

   Return completed output dataframe with quality domains and
sentiments
```

5.4 Main Execution

```
If this is the main program:
    Initialize Qt application
    Create EcommerceAnalyzer instance
    Show application window
    Execute application
```

CHAPTER 6

RESULT AND ANALYSES

6.1 PERFORMANCE METRICS

In order to evaluate the effectiveness and reliability of the implemented model, several standard classification performance metrics have been utilized. These metrics provide a quantitative understanding of how well the model performs in terms of prediction accuracy, error minimization, and class-wise prediction capability. The metrics used in this project include Accuracy, F1 Score, Precision, Recall, and the Confusion Matrix. A brief explanation of each metric is provided below:

6.1.1. Accuracy

Accuracy is one of the most straightforward and widely used metrics for evaluating classification models. It is defined as the ratio of correctly predicted instances to the total number of predictions made. In simpler terms, it measures how often the model makes correct predictions.

$$\text{Accuracy} = (TP + TN) / (TP + TN + FP + FN)$$

While accuracy provides a quick overview of model performance, it may not be a reliable indicator when dealing with imbalanced datasets, where one class dominates the others.

6.1.2 Precision

Precision refers to the proportion of correctly predicted positive observations to the total predicted positive observations. It is particularly useful in situations where the cost of false positives is high.

$$\text{Precision} = TP / (TP + FP)$$

High precision indicates a low false positive rate, meaning the model is conservative in labeling instances as positive unless it is very sure.

6.1.3. Recall

Also known as Sensitivity or True Positive Rate, recall measures the proportion of actual positive cases that were correctly identified by the model. This metric is crucial when the cost of false negatives is high.

$$\text{Recall} = TP / (TP + FN)$$

A high recall indicates that the model successfully captures most of the relevant positive instances, even at the risk of including some false positives.

6.1.4. F1 Score

The F1 Score is the harmonic mean of Precision and Recall. It is a balanced metric that considers both false positives and false negatives and is especially useful when the dataset has an uneven class distribution.

F1 Score = 2 × (Precision × Recall) / (Precision + Recall)

A high F1 Score indicates that the model has a good balance between precision and recall, and it is a more reliable metric than accuracy in the case of imbalanced data.

6.1.5. Confusion Matrix

The Confusion Matrix is a tabular representation of the actual versus predicted classifications made by the model. It provides a detailed breakdown of true positives, true negatives, false positives, and false negatives.

	Predicted Positive	**Predicted Negative**
Actual Positive	True Positive (TP)	False Negative (FN)
Actual Negative	False Positive (FP)	True Negative (TN)

Analyzing the confusion matrix allows for deeper insight into the types of errors the model is making and can help in fine-tuning the model's performance for better generalization.

6.1.6. Cohen's Kappa Score

Cohen's Kappa Score is a statistical measure used to evaluate the level of agreement between two raters or classifiers when categorizing items into discrete classes. Unlike accuracy, which simply computes the proportion of correct predictions, Kappa adjusts for the agreement that could happen by chance, offering a more robust assessment of classification performance—especially when dealing with imbalanced datasets.

$$\text{Kappa } (\kappa) = (Po - Pe) / (1 - Pe)$$

Where:

Po is the observed agreement (i.e., the proportion of instances where the model and the human annotator agree),

Pe is the expected agreement by random chance.

The Kappa value ranges from -1 to 1:

$\kappa=1\kappa=1$: Perfect agreement,

$\kappa=0\kappa=0$: Agreement equivalent to chance,

$\kappa<0\kappa<0$: Worse than chance agreement.

A Kappa score above 0.80 generally indicates strong agreement, and values above 0.90 are considered almost perfect agreement. In this project, the LLM-based classifier achieved a Kappa score of 0.953, which signifies an exceptionally high level of consistency with human-labeled sentiment annotations. This confirms the model's reliability in correctly interpreting the sentiment of customer reviews, even in subtle or ambiguous cases.

6.2. PERFORMANCE SCORE OF LLM BASED SENTIMENTAL ANALYSIS

A Large Language Model (LLM), specifically **DeepSeek-R1**, was employed to perform sentiment analysis on customer reviews obtained through web scraping. A total of **50 reviews** spanning **three different products** were collected and manually annotated by a human reviewer to serve as ground truth for performance evaluation. The LLM was tasked with classifying each review as either *Positive* or *Negative*. The classification results were then compared with the human-labeled data to evaluate performance using standard metrics.

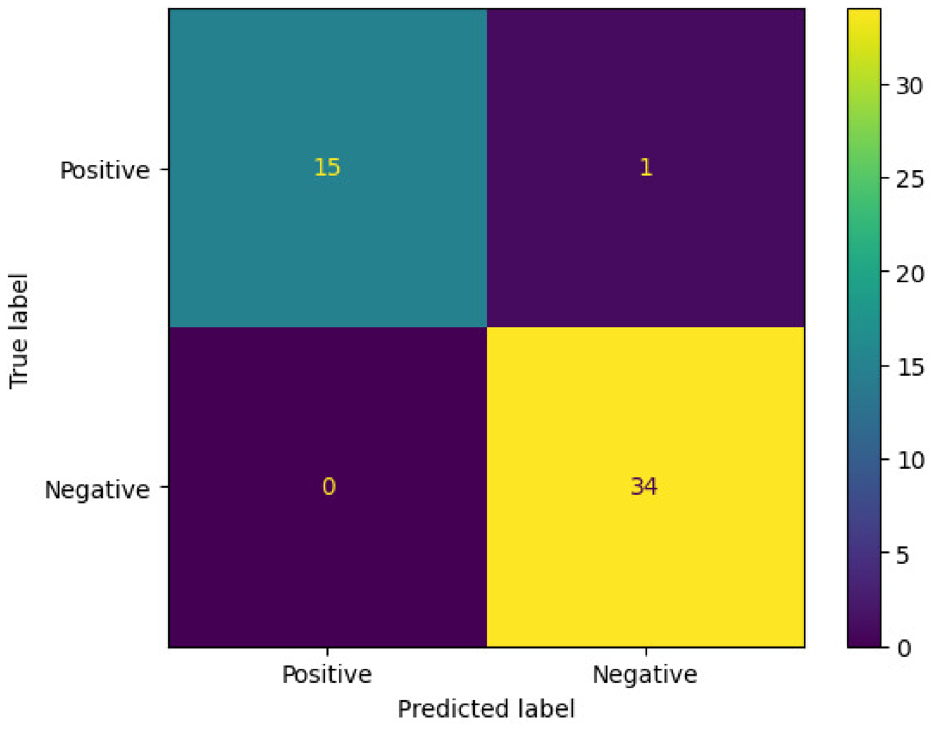
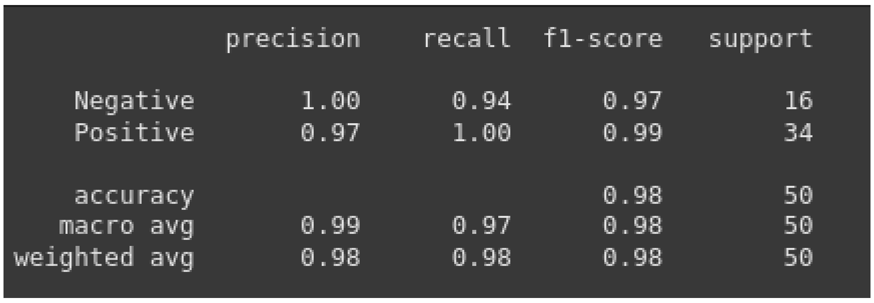


Figure.6.1 Confusion Matrix

As illustrated in Figure.6.1, the confusion matrix, the model correctly classified 15 out of 16 Positive reviews and all 34 Negative reviews, demonstrating high reliability.



```
              precision    recall  f1-score   support

    Negative       1.00      0.94      0.97        16
    Positive       0.97      1.00      0.99        34

    accuracy                           0.98        50
   macro avg       0.99      0.97      0.98        50
weighted avg       0.98      0.98      0.98        50
```

Figure 6.2 Classification Report of Sentiment Analysis

As illustrated in Figure 6.2 ,The model achieved an overall accuracy of 98%, precision of 0.99, recall of 0.97, and an F1 score of 0.98 (macro average). These results suggest that DeepSeek-R1 effectively captures sentiment cues in natural language and performs robustly in binary classification tasks, even in limited datasets.

6.3. COMPARATIVE ANALYSIS OF SENTIMENTAL ANALYSIS WITH CURRENT APPROACHES

To validate the effectiveness of the implemented sentiment analysis model using DeepSeek-R1, the model's agreement with human-annotated reviews was measured using Cohen's Kappa, a statistical metric used to assess inter-rater reliability while accounting for chance agreement. The proposed model achieved a kappa score of 0.953, indicating *almost perfect agreement* between the LLM's sentiment predictions and human-reviewed ground truth. In comparison, earlier methods reported in [1] such as ChatGPT, Claude 3, and Transformer-based models attained kappa scores of only 0.67, 0.59, and 0.42, respectively. This substantial performance gap demonstrates the superior consistency and reliability of the DeepSeek-R1 model in aligning with human judgment.

Table 6.1 Comparison Table of Sentiment Analysis Models with Current Approaches

Model/Approach	**Kappa Score**	**Accuracy**	**Precision**	**Recall**	**F1 Score**
Proposed DeepSeek-R1 Sentiment Model	**0.953**	**0.98**	**0.99**	**0.97**	**0.98**
ChatGPT [1]	0.67	-	-	-	-
Claude 3 [1]	0.59	-	-	-	-
ft:gpt-3.5-turbo-1106 (100%) [3]	-	0.642	0.641	0.642	0.6409
ft:roberta-adam (100%) [3]	-	0.606	0.615	0.606	0.6017
ft:bert-adam (100%) [3]	-	0.588	0.588	0.588	0.5871
ft:bert-sgd (100%) [3]	-	0.222	0.070	0.222	0.0888

As illustrated in Table 6.1 , In addition to the kappa score comparison, a further benchmark was drawn using traditional performance metrics like **Accuracy, Precision, Recall**, and **F1 Score**, referencing values published in [3]. While the proposed DeepSeek-R1 sentiment model achieved an **accuracy of 0.98** and **F1 score of 0.98**, the top-performing models in [3] such as **ft:gpt-3.5-turbo-1106 (100%)** and **ft:roberta-adam (100%)** achieved F1 scores of only **0.6409** and **0.6017**, respectively. Some models, including **ft:bert-sgd** and **ft:roberta-sgd**, recorded F1 scores below **0.1**, indicating comparatively poor classification capability. These results clearly emphasize the enhanced sentiment analysis capability of the DeepSeek-R1 model over widely studied LLMs across recent literature

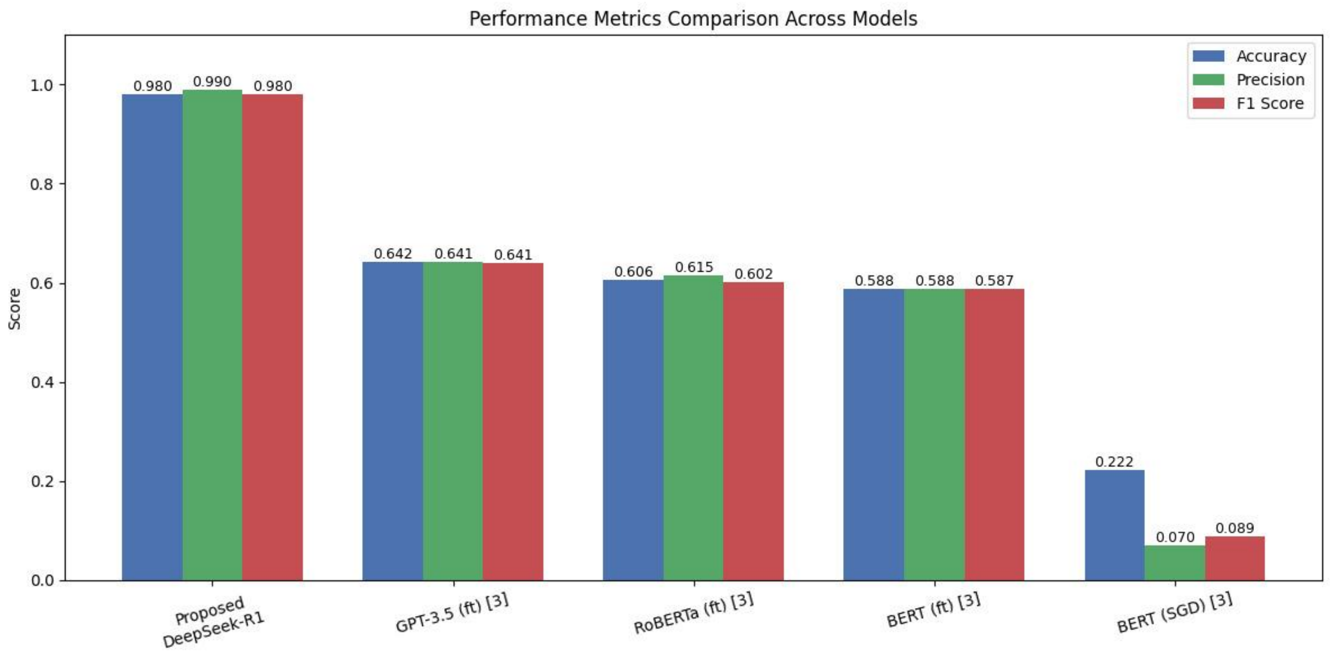


Figure.6.3 Comparison of Performance Metrics Using Inference Graphs

As Illustrated in Figure 6.3, This bar chart compares the performance metrics—Accuracy, Precision, and F1 Score—of various models. The proposed DeepSeek-R1 significantly outperforms all others with scores near 0.98–0.99. GPT-3.5, RoBERTa, and BERT (fine-tuned) show moderate performance (~0.58–0.64), while BERT (SGD) performs poorly across all metrics, highlighting DeepSeek-R1's superiority in review sentiment classification.

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENT

7.1 CONCLUSION

The proposed system successfully integrates multiple components—web scraping, domain-specific sentiment analysis using Large Language Models (LLMs), and an interactive frontend dashboard—to provide a comprehensive and intelligent e-commerce review analysis solution. The system enables users to input product links from popular e-commerce platforms such as Amazon and Flipkart, extract customer reviews, analyze sentiments, and visualize sales trends and customer opinions through various charts.

The frontend, developed using PyQt5 and Matplotlib, offers a userfriendly and responsive interface that dynamically updates based on the website and data selected. The LLM-based sentiment analysis ensures contextual and domain-aware interpretation of customer feedback, offering more accurate and meaningful insights than traditional sentiment classifiers.

Through effective system integration, real-time analysis, and a modular design, the application meets its primary objectives of assisting users in making data-driven decisions about e-commerce products. The robust architecture also ensures scalability and maintainability, making the solution adaptable to future enhancements and a broader range of ecommerce domains.

7.2 FUTURE ENHANCEMENT

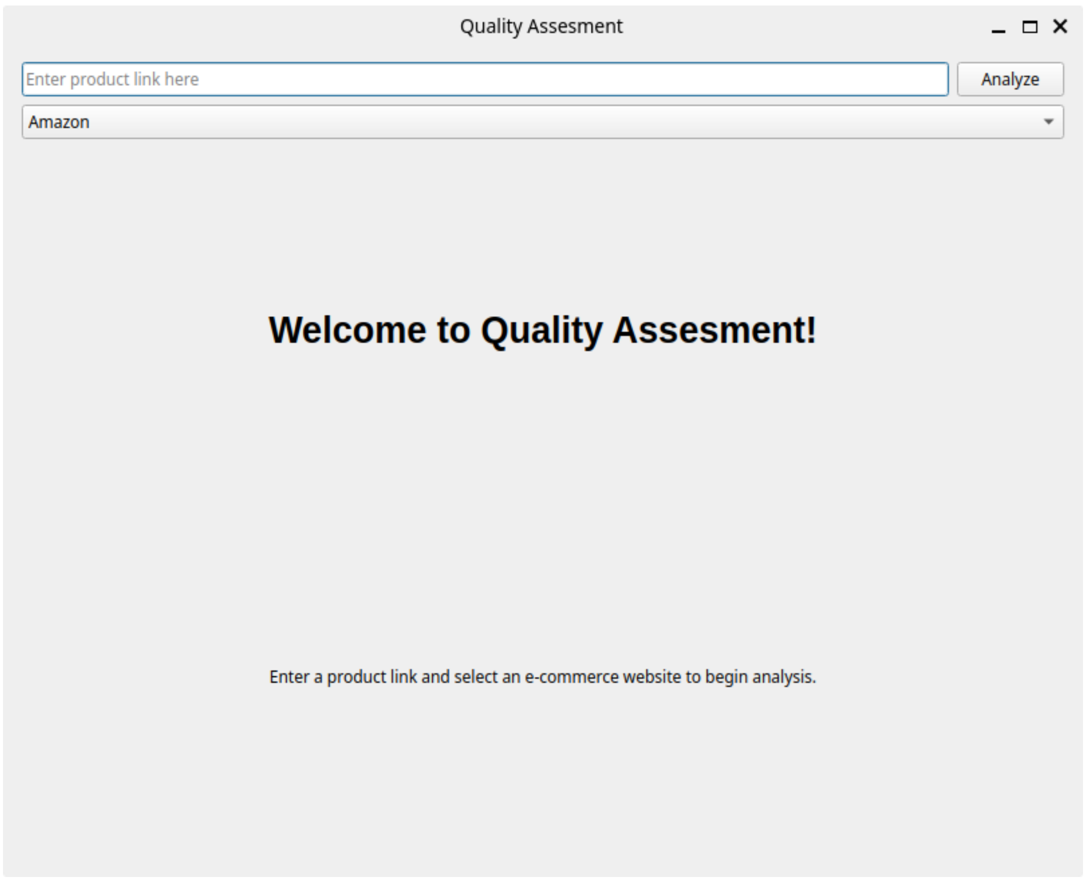
- Multi-language Support: Extend the sentiment analysis capability to support customer reviews written in regional or international languages, increasing accessibility and global reach.
- Real-Time Web Scraping: Incorporate headless browser automation (e.g., Selenium or Playwright) to handle dynamic websites and real-time scraping of more complex product pages.
- Enhanced LLM Integration: Fine-tune or train domain-specific LLMs for even higher accuracy in sentiment classification and aspect-based analysis (e.g., analyzing sentiments by price, delivery, product quality, etc.).
- Mobile & Web Deployment: Deploy the solution as a web application or mobile app using frameworks like Flask, Django, or React Native, making it accessible beyond desktop environments.
- Automated Review Summarization: Integrate text summarization to present a concise, AI-generated summary of all reviews for quick insights.
- User Authentication & History Tracking: Add login features and allow users to save and review past analyses for multiple products.
- Real-time Alerts & Recommendations: Implement a recommendation engine that suggests better alternatives based on sentiment scores and trending data across platforms.

CHAPTER 8

APPENDIX 1

SAMPLE SCREEN SHOTS

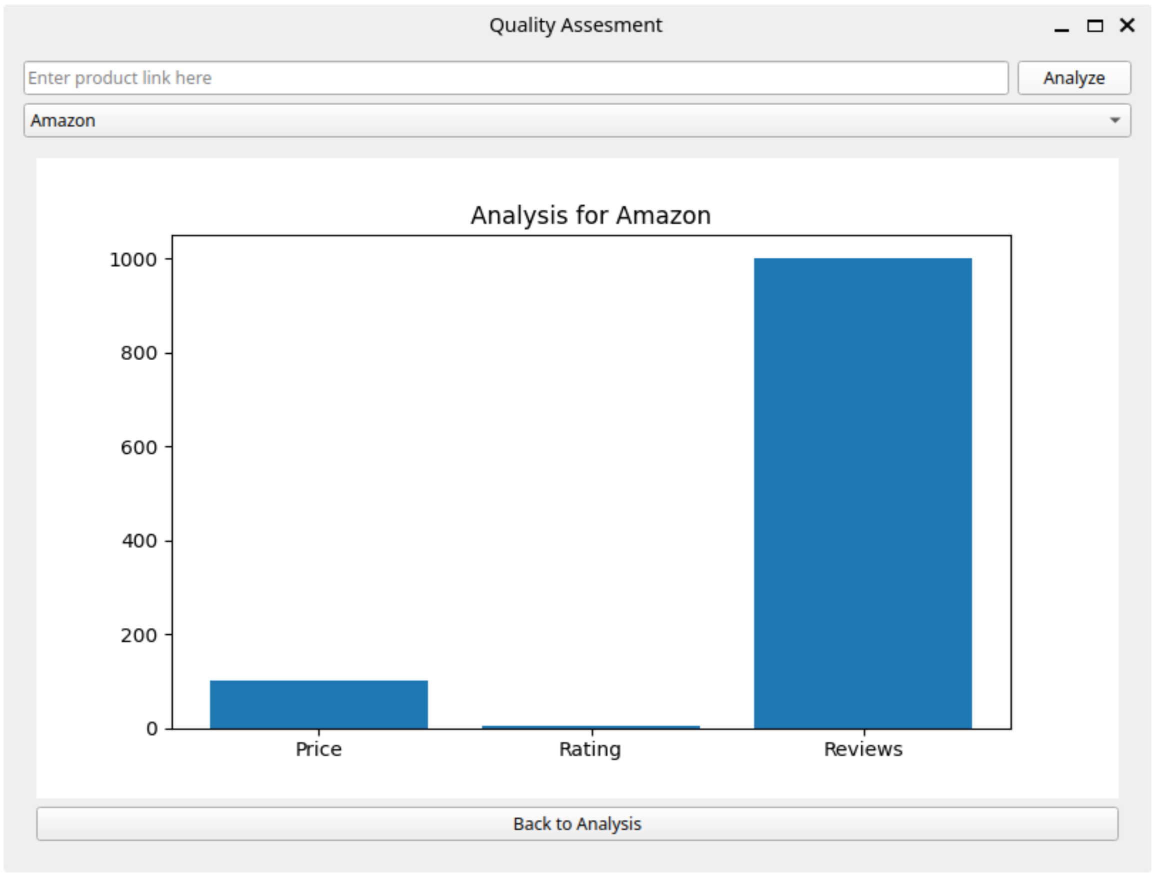
USER INTERFACE



Welcome to Quality Assesment!

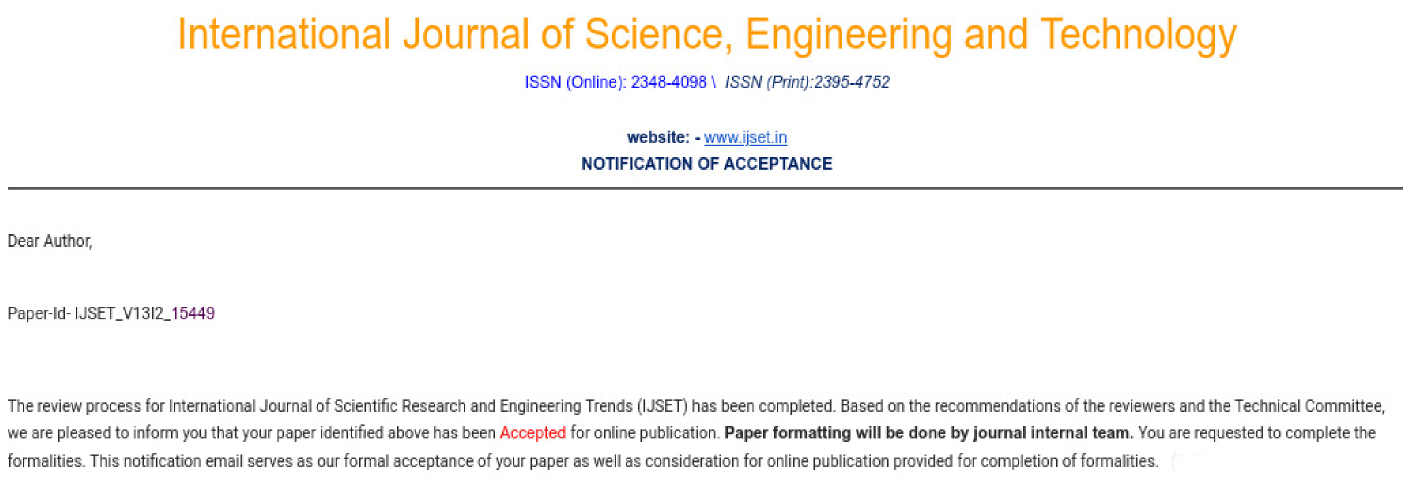
Enter a product link and select an e-commerce website to begin analysis.

ANALYSIS RESULT DASHBOARD



APPENDIX 2

PUBLICATION OF PAPER



International Journal of Science, Engineering and Technology

ISSN (Online): 2348-4098 \ *ISSN (Print):2395-4752*

website: - www.ijset.in

NOTIFICATION OF ACCEPTANCE

Dear Author,

Paper-Id- IJSET_V13I2_15449

The review process for International Journal of Scientific Research and Engineering Trends (IJSET) has been completed. Based on the recommendations of the reviewers and the Technical Committee, we are pleased to inform you that your paper identified above has been Accepted for online publication. **Paper formatting will be done by journal internal team.** You are requested to complete the formalities. This notification email serves as our formal acceptance of your paper as well as consideration for online publication provided for completion of formalities.

CHAPTER 8

REFERENCES

1. Harnessing the Power of LLMs for Service Quality Assessment From User-Generated Content - July 2024 - https://ieeexplore.ieee.org/abstract/document/10599371/citations

2.Product Helpfulness Detection With Novel Transformer Based BERT Embedding and Class Probability Features - April 2024 https://ieeexplore.ieee.org/abstract/document/10504273/citations

3.LLM s In E-Commerce: A Comparative Analysis Of GPT And Llama Models In Product Review Evaluation - March 2024 - https://www.sciencedirect.com/science/article/pii/S2949719124000049

4.A Comparison of ChatGPT and Fine-Tuned Open Pre-Trained Transformers (OPT) Against Widely Used Sentiment Analysis Tools: Sentiment Analysis of COVID-19 Survey Data - January 2024 - https://mental.jmir.org/2024/1/e50150/citations

5.Sentiment Analysis on Consumer Reviews of Amazon Products - January 2021 - https://www.researchgate.net/publication/355706880_Sentiment_Analysis_on_Consumer_Reviews_of_Amazon_Products