International Journal of Information Technology & Computer Engineering

ISSN 2347–3657

Volume 14, Issue 1, 2026

Sentiment Analysis On Product Review

Sameera Begum[1],Ayesha Zareen[2],K.Devisri[3],V. Madhumitha[4]

[1]Assistant Professor; Department Of Computer Science And Engineering(AI & ML), Bhoj Reddy Engineering College For Women, Hyderabad, India.

[2,3,4]B.Tech Students; Department Of Computer Science And Engineering(AI & ML), Bhoj Reddy Engineering College For Women, Hyderabad, India.

Mail Id; zareen4ug@gmail.com[2],Kalakotladevisri5@gmail.com[3],vasala.madhumitha@gmail.com[4]

Abstract

Sentiment Analysis on Product Reviews represents an advanced application of Natural Language Processing (NLP) that automatically interprets and categorizes customer opinions into positive, negative, and neutral sentiments. The rapid expansion of e-commerce platforms has led to an overwhelming volume of user-generated reviews, making manual evaluation inefficient and time-consuming. To address this challenge, the proposed system integrates multiple NLP techniques, including text preprocessing steps such as tokenization, stop-word elimination, and stemming or lemmatization. These processes are followed by feature extraction methods like Term Frequency–Inverse Document Frequency (TF-IDF) and word embeddings to transform textual data into meaningful numerical representations.The model further employs machine learning and deep learning algorithms, including Logistic Regression, Support Vector Machines, Long Short-Term Memory (LSTM) networks, and Transformer-based architectures, to detect sentiment patterns and underlying user emotions. The system generates structured outputs along with confidence scores, enabling organizations to assess customer satisfaction levels and identify potential product-related concerns. Visualization tools such as sentiment distribution charts and bar graphs are incorporated to improve interpretability and support data-driven decision-making.By automating sentiment classification, the framework reduces manual effort, limits subjective bias, and ensures scalability for large-scale datasets. This study highlights how NLP and artificial intelligence can convert unstructured textual feedback into actionable insights. The proposed approach supports various applications, including customer experience management, product enhancement strategies, competitive market analysis, and intelligent recommendation systems. Ultimately, sentiment analysis enables businesses to gain deeper understanding of consumer behavior and align their services with evolving market demands.

Keywords— *Sentiment Analysis, Natural Language Processing (NLP), Product Reviews, Machine Learning, Deep Learning, Text Classification*

Introduction

In the modern digital marketplace, online product reviews play a crucial role in shaping consumer purchasing decisions. Customers increasingly rely on feedback available on e-commerce platforms, social media, and online forums to evaluate the quality, reliability, and performance of products and services. As a result, these platforms generate an enormous volume of textual data on a daily basis, reflecting user experiences, preferences, and expectations. However, manually analyzing such large-scale data is impractical, time-consuming, and often subject to human bias and inconsistency.

Sentiment analysis, a key application of Natural Language Processing (NLP), provides an automated solution to extract meaningful insights from textual data by classifying opinions into categories such as positive, negative, and neutral. By leveraging machine learning and deep learning techniques, sentiment analysis systems can accurately interpret user emotions and opinions embedded within reviews. This enables organizations to monitor customer satisfaction, identify product-related issues, and make informed business decisions. Furthermore, sentiment analysis supports competitive analysis by allowing companies to compare consumer perception across different brands and products. The proposed system demonstrates how advanced NLP techniques can transform unstructured review data into actionable insights, making it an essential tool for data-driven decision-making in customer-centric industries.

Existing System

Traditional approaches to analyzing product reviews primarily rely on manual evaluation methods such as customer surveys, human analysis, and basic rating systems. These methods often focus on numerical ratings, which provide limited insight into the underlying reasons behind customer opinions. Additionally, manual evaluation processes are not scalable and struggle to handle the rapidly growing volume of online reviews. The reliance on human judgment introduces subjectivity, leading to inconsistent and less reliable outcomes.

Moreover, conventional systems lack the capability to process complex linguistic patterns, sarcasm, or contextual meaning present in textual data. As a result, valuable insights hidden within reviews often remain unexploited. The absence of automation further increases processing time and operational

costs, making traditional systems inefficient for large-scale applications.

Proposed System

The proposed system introduces an automated sentiment analysis framework that utilizes Natural Language Processing and machine learning techniques to classify product reviews effectively. The system begins by preprocessing the input text through steps such as tokenization, stop-word removal, and stemming or lemmatization. These steps help in cleaning and standardizing the data, ensuring that only meaningful information is retained for analysis.

Following preprocessing, feature extraction techniques such as TF-IDF and word embeddings are applied to convert textual data into numerical representations suitable for machine learning models. The system then employs classification algorithms, including Logistic Regression, Support Vector Machines, and deep learning models such as LSTM, to categorize reviews into positive, negative, or neutral sentiments.

The proposed framework is designed to handle large datasets efficiently while minimizing manual intervention. It also supports near real-time analysis, enabling organizations to respond promptly to customer feedback and improve service quality. By automating the sentiment classification process, the system ensures faster, more consistent, and reliable results.

Advantages of Proposed System

The proposed system offers several advantages over traditional approaches. It significantly reduces the need for manual analysis by automating sentiment classification, thereby saving time and effort. The use of machine learning algorithms ensures consistent and accurate results, minimizing human bias. Additionally, the system is capable of processing large volumes of data efficiently, making it suitable for real-world applications involving massive datasets. The ability to perform near real-time analysis further enhances its usefulness in dynamic business environments.

Requirement Analysis

Requirement analysis plays a vital role in defining both functional and non-functional aspects of the system, ensuring that it meets user expectations and operational constraints effectively.

The **functional requirements** of the system include several modules designed to enhance user interaction and system performance. The authentication module provides secure user registration and login mechanisms, ensuring controlled access to the system. The course or video module delivers instructional content in a structured manner, allowing users to learn sequentially. The quiz module generates multiple-choice questions automatically and evaluates user responses, providing instant feedback.

The engagement monitoring module incorporates advanced features such as face detection and audio monitoring to track user attention and participation. It also includes a rewind mechanism that ensures users revisit content if disengagement is detected. The progress tracking module maintains records of user activities, quiz performance, and engagement metrics, enabling continuous assessment. Additionally, the explanation module provides automated feedback, helping users understand correct and incorrect responses in detail.

Non-Functional Requirements

The system is designed with several non-functional requirements to ensure optimal performance and usability. Scalability is a key consideration, allowing the system to handle large datasets and increasing user interactions without performance degradation. Usability is enhanced through a simple and intuitive interface, making the system accessible to users with varying levels of technical expertise.

Security and privacy are prioritized by implementing robust authentication mechanisms and protecting user data from unauthorized access. Performance is optimized to ensure fast response times, even when processing large volumes of data. Maintainability is achieved through a modular design, allowing easy updates and improvements. Compatibility is also ensured, enabling the system to function as a web-based application accessible across different platforms and devices.

Computational Resource Requirements

The system requires specific software and hardware resources for efficient operation. The software requirements include frontend technologies such as HTML, CSS, and JavaScript for building the user interface, along with Flask (Python) for backend development. MySQL is used as the database for storing user data and system information, while development tools such as Visual Studio Code are utilized for implementation.

On the hardware side, the system requires a client device equipped with a webcam and microphone for engagement monitoring. A processor equivalent to Intel i5, a minimum of 8 GB RAM, and at least 50 GB of storage are recommended for smooth operation. The system can be deployed on a local machine or a basic server configuration, and GPU support is not mandatory.

Lifecycle Model

The development of the system follows the Agile lifecycle model, which emphasizes iterative development and continuous improvement. Agile methodology allows the system to be developed in incremental stages, where each iteration includes requirement analysis, design, implementation,

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 14, Issue 1, 2026

testing, and feedback integration. This approach enhances flexibility, enabling the system to adapt to changing requirements and user feedback efficiently. Continuous testing and refinement ensure that the final product meets quality standards and user expectations, resulting in a robust and reliable sentiment analysis system.

System Design

In software engineering, system design involves defining the structure and operational workflow of an application before implementation. The design phase translates user requirements into a structured framework that ensures both functional and non-functional specifications are satisfied.For the proposed sentiment analysis system, the design outlines how product reviews are collected, preprocessed, analyzed, and categorized into sentiment classes. The objective is to create an efficient architecture capable of handling large volumes of textual data while providing accurate sentiment classification. The design emphasizes automation, scalability, and ease of integration with machine learning models.

Software Architecture

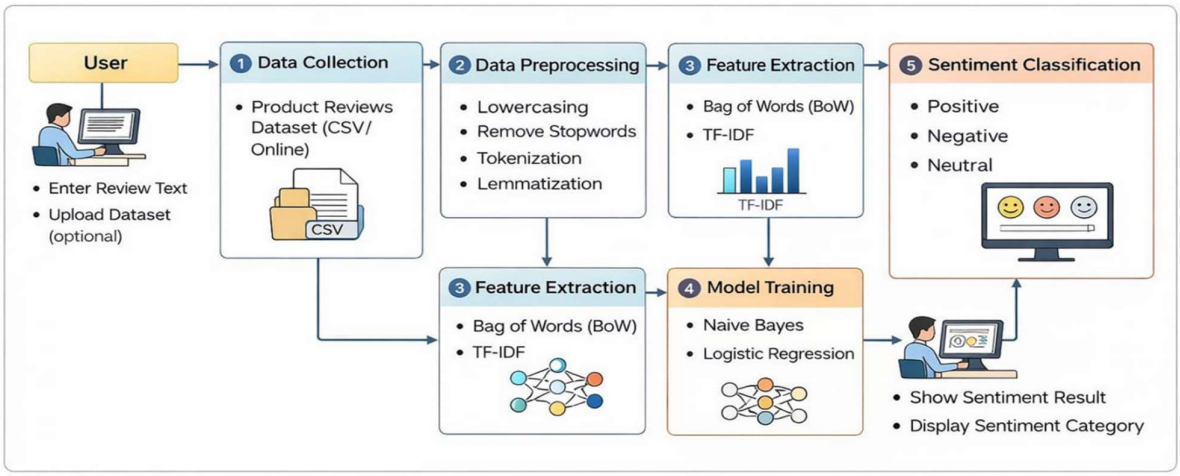


Fig. 1 Software Architecture

The software architecture describes the interaction between system components. The proposed model follows a layered structure consisting of user interface, application logic, and data processing layers. Users submit product reviews through the interface, which are then processed by the backend system. The backend performs preprocessing, feature extraction, and sentiment classification. Finally, the results are stored and visualized for interpretation.

Technical Architecture

The technical architecture highlights the technology stack used for system implementation. The frontend layer is responsible for user interaction and visualization. The backend layer handles request processing, NLP operations, and model execution. The database stores user data, review text, and sentiment results. The machine learning module integrates classification algorithms for sentiment detection. Communication between components occurs through API calls, ensuring modularity and maintainability.

Use Case Diagram

Use case diagrams illustrate system functionality from the user's perspective. They define interactions between actors and system operations. In the proposed system, actors interact with functionalities such as uploading reviews, performing sentiment analysis, viewing results, and managing data.

Class Diagram

The class diagram presents the structural design of the system by identifying classes, attributes, methods, and relationships. It provides a high-level overview of system components such as user, review handler, preprocessing module, classifier, and result visualization. This diagram supports object-oriented design and simplifies implementation.

Sequence Diagram

Sequence diagrams describe the order of interactions between system components. They illustrate message flow and execution sequence during system operation.

Sequence Diagram for Admin

This diagram represents the interactions performed by the administrator, including managing datasets, monitoring system performance, and reviewing sentiment analysis results.

Sequence Diagram for User

This diagram illustrates user interaction with the system, such as submitting reviews, initiating analysis, and viewing sentiment outputs.

Data Flow Diagram

A Data Flow Diagram (DFD) illustrates how data moves through the system. It shows data input, processing stages, storage components, and output generation. In the proposed system, product reviews are provided as input, processed through preprocessing and classification modules, stored in the database, and presented as sentiment results. The DFD helps visualize system functionality and serves as a reference for implementation.

Implementation

Technologies Used

The implementation of the Sentiment Analysis on Product Reviews system is carried out using a combination of web development technologies, database management tools, and machine learning techniques. The system is designed to provide automated sentiment classification while ensuring efficiency, scalability, and ease of use. The architecture follows a client–server model in which the frontend handles user interaction, the backend processes requests, and the machine learning module performs sentiment prediction.The frontend of the application is developed using HTML, CSS, and JavaScript. HTML is responsible for defining the structure of the web pages, including login interfaces, review submission forms, and sentiment result displays. Semantic HTML elements are used to improve code readability and maintainability. CSS is applied to enhance the visual appearance of the application by styling forms, buttons, and layout components. Responsive design techniques ensure that the system functions correctly across multiple devices such as desktops, tablets, and smartphones. JavaScript is utilized to implement dynamic features such as form validation, asynchronous communication with the server, and real-time display of sentiment results, thereby improving overall user experience.The backend is implemented using the Flask framework in Python. Flask is lightweight and flexible, allowing seamless integration with machine learning models. The backend is responsible for handling HTTP requests, processing user input, invoking the sentiment analysis module, and communicating with the database. When a user submits a review, the backend receives the text, forwards it to the preprocessing module, and then sends the processed data to the classification model. The predicted sentiment is returned to the user interface and stored for future analysis. RESTful APIs are created to enable smooth communication between frontend and backend components.Natural Language Processing techniques are employed to process unstructured textual data. Since product reviews vary in format and vocabulary, preprocessing steps are applied to standardize the input. These steps include tokenization, stop-word removal, conversion to lowercase, and stemming or lemmatization. Tokenization divides text into individual words, stop-word removal eliminates frequently occurring words that do not contribute to sentiment, and stemming or lemmatization reduces words to their base forms. These preprocessing operations improve model accuracy and reduce noise in the dataset.Several Python libraries are used to support implementation. Scikit-learn is utilized for building and training machine learning models. NLTK is used for text preprocessing tasks such as tokenization and stop-word removal. Pandas assists in data manipulation and dataset handling, while NumPy supports numerical computations required for machine learning operations. The development process is carried out using Visual Studio Code, which provides syntax highlighting, debugging tools, extension support, and an integrated terminal, thereby improving developer productivity.

Algorithm Description

The implementation follows a structured workflow. Initially, the user submits a product review through the web interface. The system receives the input and performs preprocessing to clean and standardize the text. The processed data is transformed into feature vectors using TF-IDF. These vectors are then passed to the trained machine learning model, which predicts the sentiment category. The predicted result is stored in the database and displayed to the user. Additionally, administrators can analyze stored reviews and generate reports summarizing sentiment distribution.

Implementation Workflow

When a review is submitted, the backend stores the review along with associated metadata such as product identification. The preprocessing module converts the text to lowercase, removes stop words, and tokenizes the content. The sentiment prediction module applies the trained classifier to determine whether the review expresses positive, negative, or neutral sentiment. The resulting label is stored in the database and returned to the user interface. The admin module retrieves stored reviews and sentiment labels to generate analytical reports and visualizations.The system also includes monitoring mechanisms for enhanced engagement. Face detection is implemented using browser-based APIs to ensure user presence during interaction. Audio monitoring detects background noise levels, and if excessive noise is identified, the system triggers alerts. These additional features help maintain interaction quality during system usage.

Overall, the implementation integrates web technologies, machine learning algorithms, and database management into a unified platform capable of performing automated sentiment analysis efficiently. The modular design allows easy maintenance, scalability, and future enhancements, making the system suitable for real-world deployment.

Testing

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 14, Issue 1, 2026

Testing is an essential phase in software development that ensures the reliability, correctness, and usability of the system before deployment. The primary objective of testing is to identify defects, validate functionality, and confirm that the application meets specified requirements. Various testing approaches are applied to evaluate different aspects of the system. Unit testing focuses on verifying individual modules, while integration testing ensures proper interaction between components. System testing evaluates the complete application as a whole, and acceptance testing validates the software against user requirements. Automated testing tools can further enhance efficiency by executing repetitive test cases quickly and accurately. A comprehensive testing strategy reduces potential risks, improves software quality, and increases confidence in system performance.

Dimensions of Testing

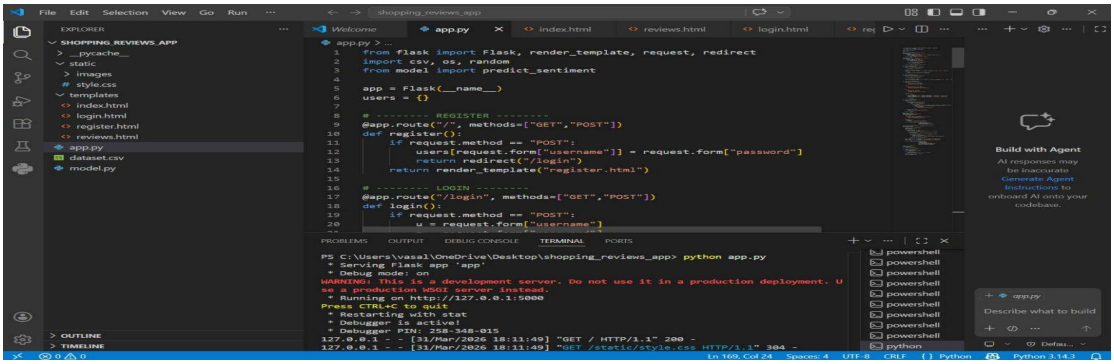
Testing can be categorized based on several dimensions that define the scope and methodology of evaluation. From a purpose perspective, testing may be functional, which verifies system features, or non-functional, which evaluates performance and usability. In terms of scope, testing includes unit, integration, and system-level assessments. Testing levels may involve manual execution, automated scripts, or regression validation. Timing-based classification includes static testing, which evaluates code without execution, and dynamic testing, which involves running the application. Techniques such as black box, white box, and gray box testing are used depending on access to internal code. Additionally, testing domains may include web applications, mobile platforms, APIs, or cloud-based environments.

Stages of Testing

The testing process is performed in multiple stages to ensure thorough validation. Unit testing is conducted to verify individual modules independently and confirm that each component functions according to its design. This stage is typically performed by developers during implementation. Integration testing follows, where multiple modules are combined and tested to ensure that interactions between them function correctly. This stage helps identify interface-related issues. System testing evaluates the entire application in an integrated environment to verify end-to-end functionality and performance. **Test Cases**

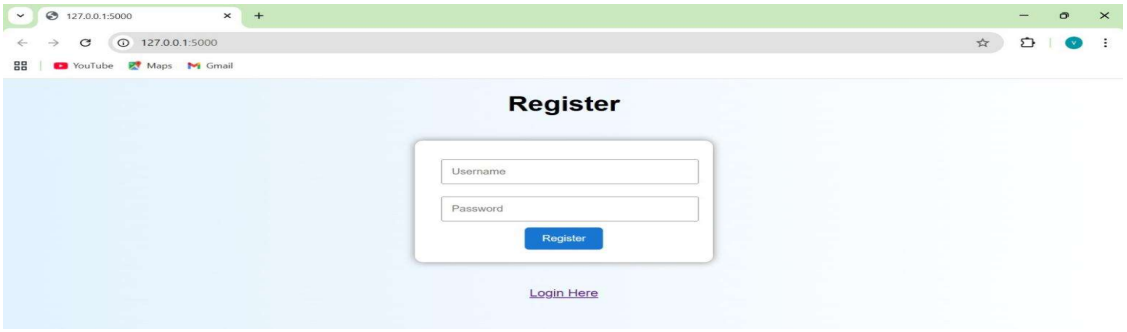
Test cases are prepared to validate critical functionalities of the application. Each test case includes input conditions, expected outcomes, and actual results obtained after execution. For example, the user registration test verifies whether valid credentials result in successful account creation. Similarly, the login test confirms that authorized users can access the system using correct credentials. Additional test cases evaluate review submission, sentiment prediction, and result display. Successful execution of these test cases confirms that the system operates according to design specifications. The results demonstrate that major modules such as authentication, review processing, and sentiment classification function correctly without errors.

Screenshots

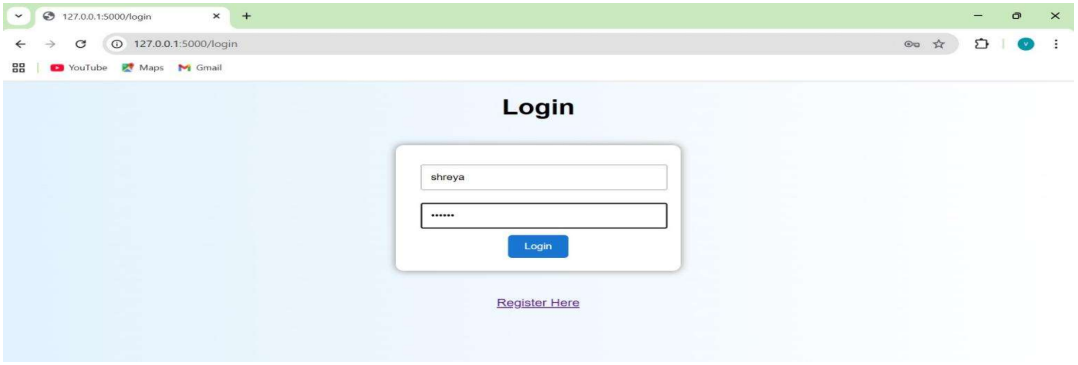


Backend Execution

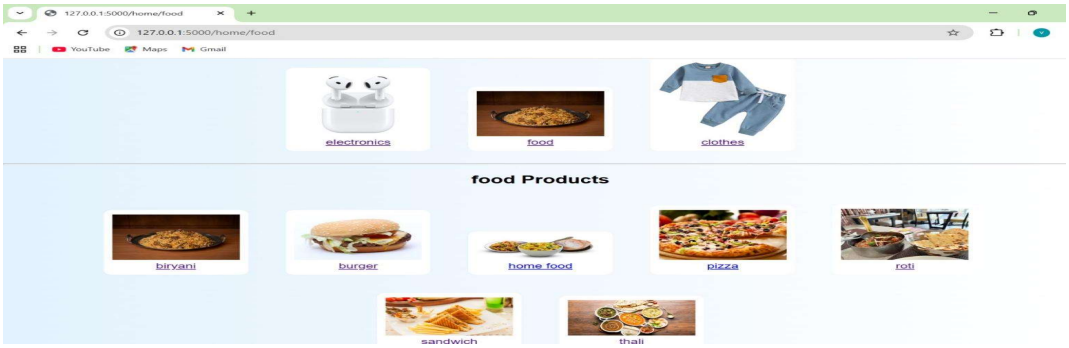
International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 14, Issue 1, 2026



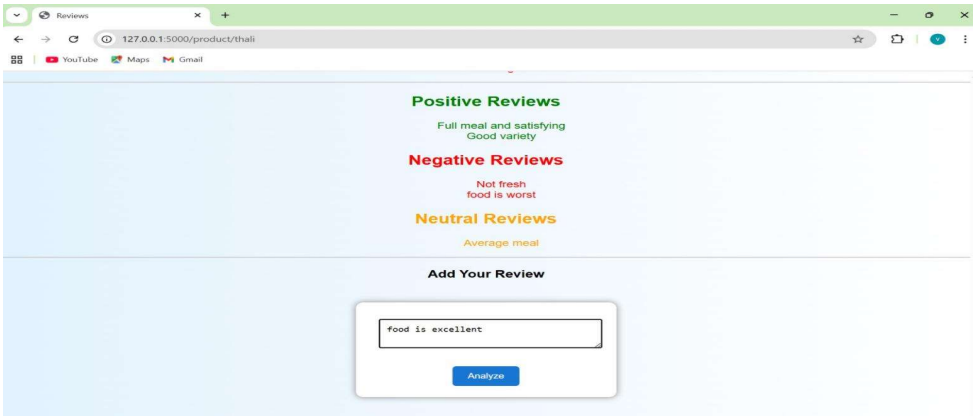
Home Page



Login Page



Categories of products and food



Positive , Neutral & Negative Reviews

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 14, Issue 1, 2026

Result Chart

Conclusion

The Sentiment Analysis on Product Reviews system presents an effective and automated solution for extracting customer opinions from textual feedback. By integrating Natural Language Processing techniques with machine learning algorithms, the proposed system classifies reviews into positive, negative, and neutral categories with improved efficiency. This automation eliminates the limitations associated with manual review analysis, such as time consumption and subjective bias.The developed system enables organizations to monitor customer satisfaction, identify product strengths and weaknesses, and make informed business decisions based on data-driven insights. The architecture is designed to be scalable and capable of handling large volumes of reviews without compromising performance. Additionally, the user-friendly interface simplifies interaction and ensures accessibility for different types of users.

Future Scope

The proposed system can be further enhanced by incorporating advanced deep learning models such as transformer-based architectures to improve sentiment prediction accuracy. Future improvements may include support for multilingual sentiment analysis, enabling the system to process reviews written in different languages. Real-time streaming analysis can also be integrated to monitor customer opinions as they are generated.

Another potential extension involves aspect-based sentiment analysis, which identifies sentiment related to specific product features such as price, quality, or usability. Detection of fake or spam reviews can also be implemented using anomaly detection techniques. Furthermore, integration with recommendation systems can allow automatic product suggestions based on customer sentiment trends. These enhancements would make the system more intelligent, robust, and suitable for large-scale commercial deployment.

References

1) B. Pang and L. Lee, “Opinion Mining and Sentiment Analysis,” *Foundations and Trends in Information Retrieval*, vol. 2, no. 1–2, pp. 1–135, 2008.
2) M. Hu and B. Liu, “Mining and Summarizing Customer Reviews,” *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2004.
3) A. Sharma and R. Mehta, “Sentiment Analysis of Product Reviews Using Machine Learning Techniques,” *IEEE International Conference on Computing and Communication Technologies*, 2023.
4) S. Kumar and P. Singh, “Aspect-Based Sentiment Analysis on Online Product Reviews,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 6, 2022.
5) J. Liu and Y. Zhang, “Deep Learning Approaches for Sentiment Analysis in E-Commerce Reviews,” *IEEE Access*, 2024.