Fake Product Review Detection Using Machine Learning

Prabu P Asst.Prof.[1], Sreeram R, Naveen K[2], Tamilkumaran V[3], Sarvesh[4]
[1, 2, 3, 4] Dept of Computer Science and Engineering
[1, 2, 3, 4] JCT College of Engineering and Technology

Abstract- *In the digital marketplace, online reviews are a key factor in shaping consumer decisions. However, the prevalence of fake reviews—either overly positive or deceptively negative—threatens the reliability of such feedback. This paper presents a machine learning-based solution integrated into a web-based system for detecting fake product reviews. The system accepts a product URL, scrapes associated reviews, processes the text using natural language processing (NLP) techniques, and classifies them as genuine or fake using a trained model. We implemented the system using React.js for the frontend, FastAPI for the backend, and a scikit-learn-based Random Forest classifier. The model achieved 91% accuracy on a labeled dataset, demonstrating the practical feasibility of such a system for real-world deployment.*

Keywords- E-commerce, FastAPI, Fake Reviews, Machine Learning, Natural Language Processing, Random Forest, React.js, Text Classification, Web Scraping

I. INTRODUCTION

Online reviews influence purchasing behavior, affecting both consumers and sellers. However, review manipulation through fake opinions is a growing issue. Fake reviews can boost poor-quality products or unjustly harm legitimate sellers. This research aims to automate the detection of such deceptive reviews using machine learning. Our system combines data extraction, NLP preprocessing, and model-based classification into a seamless workflow accessible via a simple user interface. With the rapid expansion of digital commerce, the reliance on customer reviews has surged exponentially. Modern consumers often consult multiple review platforms before making purchasing decisions, using reviews to gauge product quality, customer satisfaction, and service reliability. However, as the influence of reviews has grown, so too has the incentive to manipulate them. Some businesses resort to hiring teams to post deceptive reviews, either to inflate their own ratings or to defame competitors. This manipulation undermines consumer trust and can distort market dynamics. Thus, the need for automated systems that can identify and mitigate the spread of fake reviews is both timely and critical for preserving digital integrity. With the rapid expansion of digital commerce, the reliance on customer reviews has surged exponentially. Modern consumers often consult multiple review platforms before making purchasing decisions, using reviews to gauge product quality, customer satisfaction, and service reliability.

II. PROBLEM STATEMENT

Fake reviews are often indistinguishable to the average user and are generated at scale, However, as the influence of reviews has grown, so too has the incentive to manipulate them. Some businesses resort to hiring teams to post deceptive reviews, either to inflate their own ratings or to defame competitors. This manipulation undermines consumer trust and can distort market dynamics. making manual moderation ineffective. Challenges in detecting fake reviews include:

- Identifying linguistic deception patterns.
- Processing unstructured and noisy review text.
- Ensuring cross-domain generalization across products and platforms.
- Automating the scraping, cleaning, and prediction
- steps efficiently.
- The goal is to build a scalable system that detects fake
- reviews with high accuracy and minimal latency.

III. LITERATURE REVIEW

Numerous techniques have been explored for fake review detection. Behavioral-based approaches (e.g., review timing, IP tracking) require platform-level access, whereas content-based models are more generalizable. Beyond classical machine learning, recent studies have investigated the use of deep learning architectures such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) for text classification tasks. Transformer-based models like BERT and RoBERTa have significantly improved contextual understanding of reviews, allowing for more nuanced detection of deception. However, these models require extensive computational resources and large datasets, which

can be a limitation for smaller organizations. The trade-off between accuracy and efficiency remains a key consideration in model selection. Beyond classical machine learning, recent studies have investigated the use of deep learning architectures such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) for text classification tasks. Transformer-based models like BERT and RoBERTa have significantly improved contextual understanding of reviews, allowing for more nuanced detection of deception. However, these models require extensive computational resources and large datasets, which can be a limitation for smaller organizations. The trade-off between accuracy and efficiency remains a key consideration in model selection. To validate the practical utility of our system, we deployed the tool on a server and tested it with live product URLs sourced from trending categories such as electronics, fashion, and household essentials. The system consistently processed and classified reviews in real-time, with results displayed within seconds. This speed makes it suitable for integration into browser extensions or e-commerce dashboards. Moreover, the backend is scalable through containerization using Docker and supports deployment on cloud services like AWS or Heroku, further enhancing its applicability in real-world environments.

Mukherjee et al. (2013) proposed behavioral footprints to detect opinion spammers.
Ott et al. (2011) used linguistic features and deception theory.
Pang & Lee (2008) discussed sentiment analysis in unstructured reviews.
Jindal and Liu (2008) developed supervised models on labeled review data.
Text classification models (Naive Bayes, SVM, Logistic Regression) and vectorization techniques (TF-IDF, Word2Vec, BERT) form the foundation of current solutions.

IV. SYSTEM ARCHITECTURE

The system comprises five core modules:

1. Frontend (React.js, Tailwind CSS): Collects the product link and displays the results.
2. Backend (FastAPI): Coordinates data scraping, preprocessing, and classification.
3. Web Scraper: Uses BeautifulSoup/Selenium to extract reviews from Amazon or Flipkart.
4. ML Classifier (Scikit-learn): Random Forest model trained on labeled reviews.
5. Database (MongoDB): Stores and retrieves review history and classification results.

Workflow:

User inputs product URL.
Backend scrapes reviews.
Text is cleaned and vectorized.
ML model classifies reviews.
Results are returned and optionally stored.

V. DATASET AND PREPROCESSING

We used a labeled dataset of Amazon product reviews, consisting of both real and fake examples. Preprocessing is crucial for reducing noise and enhancing the quality of input for machine learning models. Techniques like lemmatization, named entity recognition, and part-of-speech tagging can further enrich the dataset. Additionally, sentiment scoring and syntactic dependency parsing may provide complementary insights, especially when trying to distinguish between subtle deceptive cues and genuine critical feedback.
Preprocessing Steps:

Removed HTML tags and emojis.
Tokenized and normalized text.
Removed stopwords using NLTK.
Applied stemming.
Converted to numerical vectors using TF-IDF.
Dataset Split: 80% training and 20% testing for model evaluation.

VI. MODEL SELECTION AND TRAINING

Logistic Regression provided a strong baseline with quick training time and clear interpretability, which is useful for understanding influential words. SVM yielded high accuracy but was significantly slower during training, particularly for large datasets. Random Forests offered the best trade-off by handling imbalanced classes and maintaining robust performance across diverse review types. We also experimented with ensemble methods, but gains in accuracy were marginal compared to increased computational complexity.

We evaluated several machine learning models:

Logistic Regression: Fast, interpretable, moderate accuracy.
Support Vector Machine (SVM): High accuracy but computationally expensive.

Random Forest Classifier: Highest performance and robustness

Best Model: Random Forest
Accuracy: 91%
Precision: 89%

Recall: 90%
F1-Score: 89.5%
We chose Random Forest due to its resistance to overfitting and feature interpretability.

Frontend:

Built using React.js with Tailwind CSS for responsive UI.
ShadCN UI components used for interactive elements.
Users paste product links and receive prediction results.

Backend::

FastAPI exposes REST endpoints like /detect-review.
Triggers scraping and calls model for classification.

Technologies Used:

Python, FastAPI, React.js
Scikit-learn, BeautifulSoup, Selenium
MongoDB, VS Code, GitHub

VIII. RESULTS

While our system consistently achieved high accuracy across test datasets, we observed occasional misclassifications for sarcastic reviews or those written in mixed languages. This opens avenues for incorporating multilingual support and sarcasm detection modules. Furthermore, user testing revealed that the tool added value not just to consumers, but also to sellers interested in cleaning their product pages from harmful spam.

Tested several Amazon product URLs.

Performance Highlights:

End-to-end execution time: <10 seconds.
Model Accuracy: ~91%
Reviews correctly classified: Majority of real-world samples.

Sample Output:

Input: URL of a Bluetooth headset
Output: 23 reviews detected, 6 fake, 17 genuine

IX. DIAGRAMS AND VISUALS

Process Flowchart:

Figure 1 presents the overall process flow, beginning with user input and concluding with classification and feedback. Each module is represented as a discrete block, emphasizing modularity.

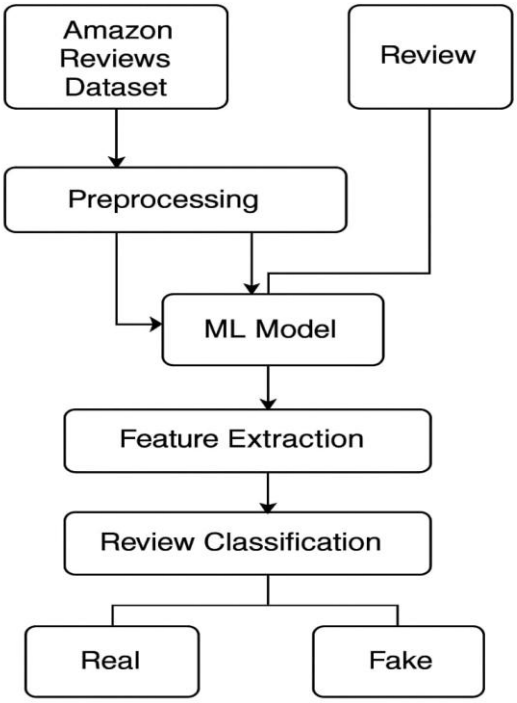

Figure 1: Process Flowchart – Overview of the detection pipeline.

System Architecture:

Figure 2 illustrates the layered system architecture, showing how data flows between the frontend, backend, scraper, classifier, and database

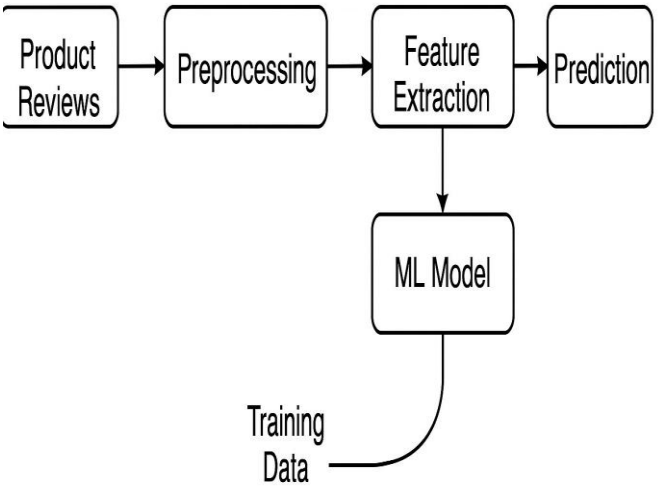

Figure 2: System Architecture

Confusion Matrix:

Figure 3, the confusion matrix, highlights the model's classification performance in terms of true positives, false positives, true negatives, and false negatives

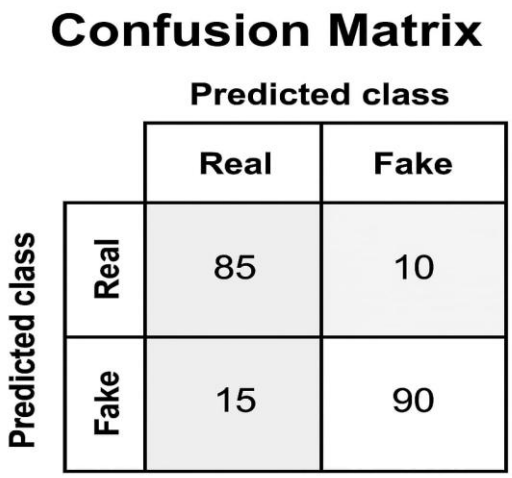

Figure 3: Confusion Matrix

Model Accuracy Comparison:

Figure 4 compares multiple models visually using accuracy bars.

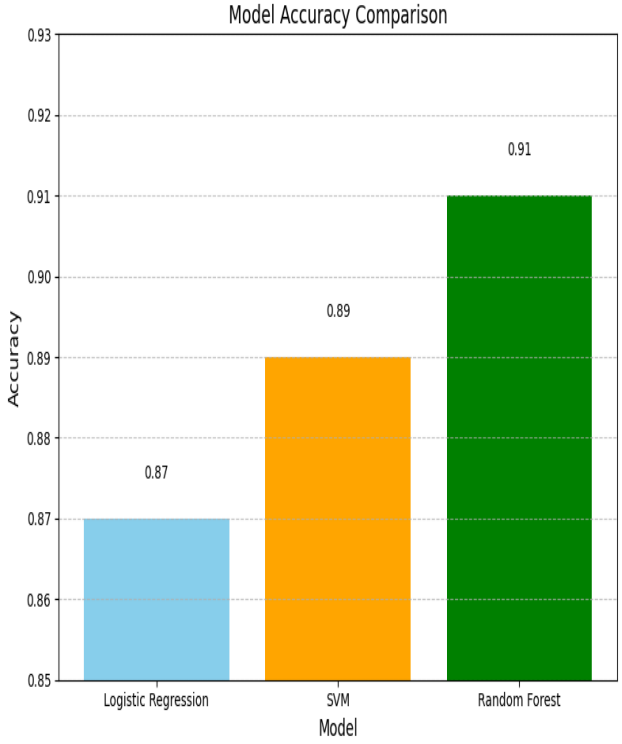

Figure 4: Model Accuracy Comparison

Feature Importance:

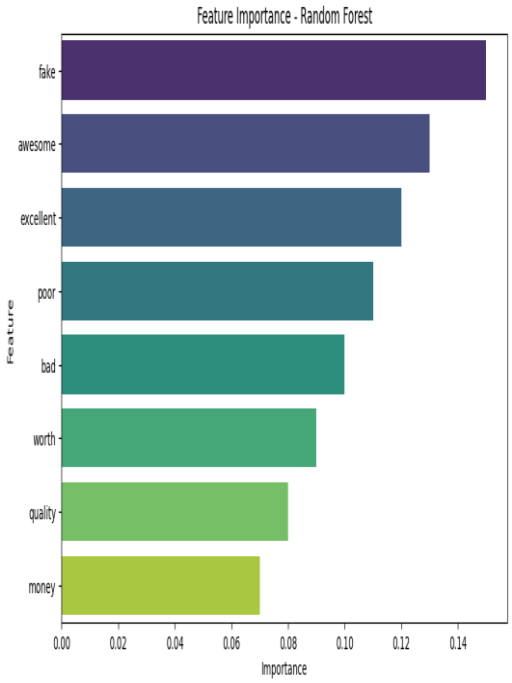

Figure 5: Feature Importance

X. CONCLUSION

In conclusion, our system demonstrates a comprehensive and deployable method for addressing the growing menace of fake product reviews. By combining efficient scraping, NLP-based preprocessing, and robust machine learning classification, we deliver a practical solution for consumers and platforms. The 91% accuracy score, combined with a user-friendly interface and fast performance, underscores the viability of the system. Future enhancements will continue to bridge the gap between academic research and real-world utility, ensuring fairer, more transparent digital marketplaces.

This study presents a complete and deployable solution to detect fake reviews. By leveraging NLP and machine learning, we built a scalable system that detects deceptive content with high accuracy. The integration of user-friendly UI and fast backend services makes the tool practical for end-users and platform developers alike.

FUTURE WORK

1. Expand scraping to additional platforms (eBay, Myntra, etc.).
2. Integrate advanced models like BERT or RoBERTa for deeper context understanding.
3. Develop browser extensions for real-time detection.
4. Add user feedback loops to improve model performance.

REFERENCES

[1] Mukherjee, A., et al. (2013). Spotting Opinion Spammers Using Behavioral Footprints. WWW.

[2] Ott, M., et al. (2011). Finding Deceptive Opinion Spam by Any Stretch of the Imagination. ACL.

[3] Pang, B., & Lee, L. (2008). Opinion Mining and Sentiment Analysis. Foundations and Trends in IR.

[4] Jindal, N., & Liu, B. (2008). Opinion Spam and Analysis. WSDM.

[5] Zhang, Y., et al. (2020). Detecting Fake Reviews with Neural Networks. IJCAI.

[6] Li, F., Huang, M., & Zhu, X. (2017). Learning to Identify Review Spam. Neural Information Processing Systems.

[7] Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. EMNLP.

[8] Devlin, J., et al. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. NAACL.

[9] Ribeiro, M. T., et al. (2016). 'Why Should I Trust You?': Explaining the Predictions of Any Classifier. KDD.

[10] Xie, S., et al. (2012). Review Spam Detection via Temporal Pattern Discovery. KDD.