**<PROJECT TITLE>**

**Submitted for**

**Statistical Machine Learning CSET211**

Submitted by:

**(E23CSEU1964) Anak Vyas**

**(E23CSEU1966) Syed Shaaz Abdullah**

**(E23CSEU1952) Divyansh Kartik**

Submitted to

**DR. Sudhanshu Gupta**

**July-Dec 2024**

**SCHOOL OF COMPUTER SCIENCE AND ENGINEERING**

A close-up of a logo

Description automatically generated

**INDEX**

|  |  |  |
| --- | --- | --- |
| Sr.No | Content | Page No |
| 1 | Abstract | 3 |
| 2 | Introduction | 4-6 |
| 3 | Methodology | 5-6 |
| 4 | Hardware/Software Required | 7-8 |
| 5 | Experiment Results | 9 |
| 6 | Conclusion | 10 |
| 7 | Future Scope | 11 |
| 8 | Github Link of the Project | 12 |

**ABSTRACT**

In today’s world, online reviews are a big deal – they can make or break a product, a business, or even a whole brand. But with so many fake reviews out there, it’s getting harder to tell what’s real and what’s not. That’s where this project comes in. The goal here was simple: to build an AI tool that can spot fake reviews and help people trust what they’re reading online.

This project aims to create a web-based application that leverages AI to detect fake reviews for products. Users can submit reviews, and the product author can immediately know whether the review is real or fake. The system uses machine learning models, primarily based on Natural Language Processing (NLP), to analyze and classify the reviews. The website is developed using Node.js, Express, Python Flask, and MongoDB for the backend and database management. The project seeks to improve the reliability of online reviews and reduce misinformation, benefiting both consumers and businesses. The AI model’s performance is evaluated using common classification metrics such as accuracy, precision, recall, and F1-score, with Random Forest providing the best results.

Preliminary testing has shown that the model is on the right track, achieving decent accuracy, though improvements are ongoing. The results suggest that with more data and fine-tuning, this tool could help reduce the influence of fake reviews, restoring some honesty to online ratings. Overall, this project highlights how AI and machine learning can address real-world challenges by making the online review ecosystem more transparent and trustworthy.

**INTROCDUCTION**

With the explosion of online shopping and digital platforms, reviews have become a powerful force in shaping consumer behavior. From deciding on a restaurant to buying tech gadgets, people rely heavily on reviews to make informed choices. But the downside? Not all reviews are genuine. Fake reviews – whether generated by bots, paid reviewers, or biased sources – have started flooding platforms, leading people to question the authenticity of what they read.

The impact of fake reviews goes beyond just misleading consumers; they also distort the market by boosting low-quality products or services while penalizing genuine ones. Businesses can suffer unjustly, and consumers lose trust in the entire online review system. Addressing this issue is crucial, as a trustworthy review system is the backbone of a healthy digital marketplace.

In this project, we explore the use of Artificial Intelligence (AI) to detect fake reviews automatically. The approach combines Natural Language Processing (NLP) techniques to analyze review text with machine learning models trained to classify reviews as fake or genuine. Built using Node.js and Express for the server-side and Python Flask for the AI model integration, along with MongoDB for database management, the platform offers a seamless and interactive user experience. This solution aims to promote transparency and trust in online marketplaces. By focusing on key language cues, sentiment patterns, and unique markers found in deceptive reviews, this AI tool aims to restore some transparency to online reviews. Our ultimate goal is to help both consumers and platforms by offering a way to filter out unreliable reviews, making digital spaces more trustworthy and user-friendly

**METHODOLOGY**

Data Collection :

First, we needed a data set of reviews, both genuine and fake, to train the model effectively. We sourced this data from publicly available review datasets, which often label reviews as real or fake based on known patterns or verified user feedback. This step ensured we had a balanced data set for training and evaluation.

Data Pre processing :

After collecting the data, we pre-processed it to remove any irrelevant or noisy information. This included:

Tokenization: Splitting review text into individual words or tokens for easier analysis.

Lowercasing: Converting all words to lowercase to ensure consistency.

Stopword Removal: Removing common words (like "the" or "is") that don’t carry significant meaning.

Stemming/Lemmatization: Reducing words to their root forms (e.g., “running” to “run”) to improve model accuracy.

Feature Extraction :

To enable the model to learn from review text, we transformed the text data into numerical features using:

TF-IDF (Term Frequency-Inverse Document Frequency): This approach gives weight to words that are more informative and relevant for each review.

Sentiment Analysis: We also analyzed the sentiment of each review, which helped capture the tone (positive, negative, or neutral) as an additional feature for the model.

N-Grams: We extracted bigrams and trigrams (combinations of two or three consecutive words) to detect phrases commonly associated with fake or real reviews.

Model Selection and Training :

After feature extraction, we experimented with different machine learning models to classify the reviews:

Naive Bayes: A probabilistic classifier that works well with text data and performs well in classifying fake vs. real reviews based on word patterns.

Support Vector Machine (SVM): A robust classifier that finds an optimal hyperplane to separate fake and genuine reviews based on feature vectors.

Random Forest: An ensemble method that combines multiple decision trees to improve accuracy and reduce overfitting.

Each model was trained on a portion of the dataset and tuned for better performance using techniques like grid search to find optimal hyperparameters.

Backend Development :

The website was built using Node.js with the Express framework for handling HTTP requests. Python Flask was used to integrate the trained machine learning model, allowing the website to send and receive data for real-time predictions.

Database Management :

MongoDB was used for storing user-submitted reviews and associated metadata. The database allows for scalable storage of review data and easy retrieval for further analysis.

Evaluation :

To measure the effectiveness of each model, we evaluated their performance using metrics like:

Accuracy: The percentage of correctly classified reviews.

Precision and Recall: Metrics that assess the model’s ability to detect fake reviews specifically, ensuring a good balance between false positives and false negatives.

F1-Score: A balanced metric that combines precision and recall, giving an overall measure of the model’s reliability.

**HARDWARE/SOFTWARE REQUIRED**

Hardware Requirements :

Computer with Minimum Specs

Processor: Intel i5 or AMD Ryzen 5 (or higher)

RAM: At least 8GB

Storage: 500GB HDD or 256GB SSD

Graphics Card: Not strictly necessary, but a basic GPU.

Software Requirements :

Programming Language:

Python 3.x – Python is widely used for machine learning projects due to its libraries and community support.

Development Environment:

Jupyter Notebook or Google Colab – Interactive notebooks are ideal for exploring data, training models, and visualizing results.

Key Python Libraries:

Numpy – For handling numerical data and matrix operations.

Pandas – Essential for data manipulation and analysis.

Scikit-learn – Contains most of the machine learning algorithms (like Naive Bayes, SVM, Random Forest) and metrics needed for evaluation.

NLTK or SpaCy – For natural language processing, tokenization, and stopword removal.

TF-IDF Vectorizer (from Scikit-learn) – For transforming text data into numerical features.

Matplotlib & Seaborn – Useful for visualizing data and results.

Node.js and Express - for the web application backend

Python Flask for integrating AI model predictions into the website

MongoDB for storing review data.

Version Control :

Git/GitHub – For version control and collaboration.

**EXPERIMENTAL RESULTS**

The model was evaluated on the classification task of detecting fake reviews. Below are the performance metrics achieved during the evaluation:

Accuracy: ~87.27%

Precision: ~87%

Recall: ~87%

F1-Score: ~87%

These results highlight the model's ability to effectively differentiate between real and fake reviews, ensuring accurate predictions for both positive and negative classes. The overall performance indicates that the model is well-suited for real-world applications in detecting fake reviews.

**CONCLUSION**

In this project, we developed a web-based application that integrates artificial intelligence to detect fake reviews for products. Users can submit reviews, and the product author can instantly determine whether the review is real or fake. The website was built using Node.js, Express, Python Flask, and MongoDB, which allowed for efficient backend management and real-time review processing. The AI model, using Natural Language Processing and machine learning algorithms like Random Forest, provided accurate predictions. This project demonstrates the potential of AI in enhancing online platforms, helping to ensure that reviews are trustworthy.

Our experimental results suggest that machine learning can play a crucial role in maintaining the integrity of online reviews, benefiting both consumers and businesses by reducing misinformation. However, we also identified areas for improvement, especially when dealing with highly nuanced or longer reviews, where simpler models like Naive Bayes tend to fall short. Future work could explore more advanced NLP methods, such as deep learning approaches, to handle such cases more effectively.

Overall, this project shows promising steps toward creating a more transparent online environment. By detecting fake reviews with increasing accuracy, we can help restore consumer trust and ensure that genuine feedback stands out in an increasingly crowded digital space.

**FUTURE SCOPE**

While the current model has shown strong performance, there are several areas where improvements could be made:

Advanced NLP Techniques:

Future work could explore deep learning models such as BERT or GPT-based models, which have demonstrated better contextual understanding of text. These models could enhance the accuracy of fake review detection, especially for more complex or subtle fake reviews.

Improved Data Collection:

To further refine the model, it would be beneficial to collect a larger, more diverse dataset of reviews. This would ensure that the model generalizes better to various types of products and writing styles.

Real-Time Sentiment Analysis:

Implementing real-time sentiment analysis could help in understanding not just whether a review is fake or real, but also the sentiment behind it, adding an extra layer of insight for users and product authors.

User Feedback Loop:

Integrating a feedback mechanism where users can flag or confirm whether a review is fake or not could help improve the model's performance over time through continuous learning.

Scalability:

As the application gains users, optimizing the system for performance and scalability will be crucial. This could involve using more efficient algorithms or distributed systems for handling large datasets.

**GitHub LINK**

GitHub link - https://github.com/Anakvyas/ai-project/blob/main/python/flask\_api.py