Deducing Synthetic Reviews Using Gradient-Boosting Technique

Vijet Hegde[1], Yashwanth S[2], Vinay Kumar B K[3], Veeresh Mulimani[4], Dr. Nirmala S[5]

[1,2,3,4]Student, Amc Engineering College
[5]Professor, Amc Engineering College

ABSTRACT
This research explains a comprehensive system for detecting fake, computer-generated reviews on e-commerce platforms like Amazon and Flipkart. The proposed solution integrates a robust machine learning model, a scraping mechanism for collecting reviews, and a user-friendly frontend built with React. Using a novel ensemble method combining LightGBM, CatBoost, and XGBoost, alongside natural language processing (NLP) techniques, the system achieves high accuracy in separating genuine opinion from fake ones. This study aims to improve customer trust and maintain the integrity of online marketplaces.

Keywords: LightGBM, CatBoost, XGBoost, Puppeteer, spaCy, TextBlob, Docker, API.

1. INTRODUCTION

The growth in which e-commerce has revolutionized the way people shop, offering unprecedented convenience and access to a wide range of products. However, alongside this growth, the reliance on online reviews has become a critical factor influencing consumer purchasing decisions. Positive feedback can increase sales, while negative feedback can affect buyers. This reliance on reviews has, unfortunately, given rise to a growing concern: the prevalence of fake, computer-generated reviews. These deceptive reviews undermine trust in e-commerce platforms, mislead consumers, and create unfair advantages or disadvantages for sellers.

In order to solve this raising problem, our project aims to develop a comprehensive solution that identifies and mitigates the impact of fake reviews on e-commerce platforms. The project focuses on three key objectives:

Developing a Machine Learning Model to Classify Reviews. The key to this project is to develop an advanced machine learning model to distinguish between Computer and generated reviews. Using NLP techniques and supervised learning, the model will analyze patterns in the language, sentiments, and metadata to confirm whether it is good or bad reviews. This will then be trained on a very large labeled dataset of reviews for getting high precision and recall of fraudulent content.

Review Scraping Using Puppeteer to train and test our model effectively, we need a diverse and representative dataset. This will be accomplished by scraping reviews from leading e-commerce sites such as Amazon. Using Puppeteer, a powerful Node.js library for web scraping, we will extract review data efficiently and securely. It supports dynamic interaction with web pages and thus allows us to collect visi-

ble as well as hidden elements for a comprehensive analysis.
Providing Insights via a React-Based Frontend, we will create an intuitive, visually appealing interface using React. This frontend would display insights obtained from the machine learning model, including a percentage of fake reviews about a specific product or a seller, and even call out flagged reviews. The frontend will enable consumers, sellers, and platform administrators to take informed decisions based on correct data by providing a seamless user experience.
These components are going to create a robust system that will not perform only in identifying fake revie-ws but also help rebuild trust in e-commerce platforms. In this regard, this project is aligned with the bigger objective of promoting ethical practices in the digital marketplace.

1.1 Dataset and Preprocessing

1.1.1 Dataset

There are advantages of using the labeled reviews from the available dataset. Reviews were labeled either computer generated (CG) or human generated (HG). Reviews contained text content plus rating. Labeled reviews become a crucial base where appropriate data to train as well as evaluate the efficacy of the machine learning model could be set. It contains plenty of diversified examples to tackle every different kind of reviewing styles and formats.

1.1.2 Preprocessing

Content Cleaning first of all, the preparation of data involves cleaning up the content. This includes converting everything to lowercase for uniformity, removing URLs to prevent unnecessary links, and eliminating non-alphanumeric characters so that only meaningful text is focused upon. These will ensure that the data fed into the model is standardized and free of noise.
Feature Engineering t o improve the model's ability to classify reviews correctly, we use multiple feature extraction techniques:
TF-IDF Vectorization: This technique captures textual patterns by generating term frequency-inverse document frequency (TF-IDF) vectors. We use unigrams, bigrams, and trigrams to represent the textual data comprehensively.
Linguistic Features: We extract other linguistic features using spaCy and TextBlob. These features are sentiment polarity, word count, and POS ratios which gives greater insight in the structure and tone of the text.
Rating Analysis: The numerical rating of the review is also included as another feature because it often correlates with the sentiment and authenticity of the review.
These preprocessing steps transform raw review data into a structured format, optimized for effective analysis by the machine learning model. By combining textual, linguistic, and rating-based features, our approach ensures a comprehensive understanding of review characteristics, paving the way for accurate classification and actionable insights

2. LITERATURE SURVEY

Fake reviews, also known as "comment spam," appear on many e-commerce websites, review sites, and social networks. The practice deceives consumers into making poor purchases, causing them financial losses and further eroding confidence in online platforms. Finding fake reviews is therefore imperative to help consumers make intelligent decisions about products based on accurate feedback.
Methods for the Detection of Fake Reviews

Machine Learning Algorithms:
This Supervised Learning Algorithms, Support Vectored Machine, Decision Trees, or Random Forests is utilized in separating the existence of fake and real reviews, trained on a given labeled dataset. It functions on a set of derived features from both textual data and behavioral data.
Deep Learning Models: Techniques like BERT, a transformer-based model, use more complex mechanisms such as self-attention to capture contextual meanings in reviews and have been reported to classify accuracy when fine-tuned on labeled data.
XGBoost: An advanced gradient boosting algorithm, It is widely used for fake review detection due to its high predictive accuracy.
NLP Techniques:
Tokenization: Text is split into smaller units like words or phrases for detailed analysis.
Stop-word Removal: Removing common, non-informative words helps focus on more meaningful terms.
Sentiment Analysis: Identifying the sentiment (positive, negative, or neutral) expressed in a review can help detect emotional exaggeration, which is often present in fake reviews.
TF-IDF: This technique evaluates the importance of words within a document, helping identify terms that indicate unusual behavior in fake reviews.
Behavioral and Contextual Features: The frequency of reviews, review timestamps, and rating format are assessed for patterns that would point toward fraudulent activity.
Fake review detection model performance is usually measured against the following metrics:
Accuracy: Percentage of reviews that are correctly classified.
Precision: Proportion of actual correct positive predictions.
Recall: This model is used to identify all actual false positives.
F1 Score: The harmonic mean of precision and recall, providing a balanced evaluation.
ROC-AUC Score: It measures the model's ability to differentiate between fake and real reviews.
Logistic Regression with TF-IDF: A study combining logistic regression with TF-IDF feature extraction achieved an accuracy of 87%. This model focused on classifying reviews based solely on textual content, and its simplicity made it effective, though further feature additions could improve accuracy.
Ensemble Model with XGBoost, CatBoost, and LightGBM: This method achieved 91% accuracy by using an ensemble of three strong classifiers. This method used both linguistic features like word count and sentiment polarity and behavioral data such as ratings for better quality prediction.
Difficulties and Ethical Issues
Adaptive Strategies: Fake review strategies are always in flux, which makes static models not very effective.
Imbalanced Datasets: There is a natural class imbalance as genuine reviews are more abundant than fake ones.
Generalization: Models trained on particular datasets might not perform well on unseen data from different domains.
Bias in Detection: Machine learning models are going to introduce bias, misclassifying legitimate reviews as fake.
Risk of Censorship: Overly aggressive detection systems may suppress truth.

3. METHODOLOGY

3.1 Ensemble Model

In order to classify reviews as either real or fake with high accuracy, we use a robust ensemble classification framework that brings together the power of several advanced models. The ensemble approach is constructed on top of a voting-based system consisting of three state-of-the-art machine learning algorithms:

LightGBM A gradient-boosting framework famous for its speed and efficiency, LightGBM handles big datasets and high-dimensional data pretty efficiently. It optimizes for faster training times and lesser memory usage, so it does the job really well here.

CatBoostCatBoost is specifically designed to handle categorical features and provides great performance with minimal parameter tuning. Its built-in mechanisms prevent overfitting and guarantee stable predictions even in the most complex data.

XGBoost is one of the very powerful gradient-boosting algorithms that ensure great accuracy with robust performance. Fine-tuning parameters for model generalization using regularization parameters

Each of the models was finely tuned for parameters like learning rate, tree depth, and regularization terms to obtain optimal performance individually. In this ensemble method, a soft voting mechanism is employed for combination of the predictions by taking the probabilities of individual models, computing their weights, and then averaging the resultant values for final classification.

This voting-based ensemble framework enhances the overall robustness and accuracy of the system through the use of unique strengths from each model. By using diverse predictive capabilities, the ensemble decreases the chances of misclassification and ensures consistency in performance across different review datasets. This methodology forms the foundation of our efforts toward developing a reliable and scalable solution for deducing fake reviews.

3.2 Scraping with Puppeteer

3.2.1 Overview of Using Puppeteer

We used Puppeteer, which is a powerful Node.js library that enables automated web scraping and browser automation to develop a good-quality dataset to train and test the machine learning model. Puppeteer allows effective extraction of data through dynamic interaction with them while simulating real-user behavior to collect comprehensive and structured information for the creation of an efficient dataset.

3.3 Key Functionalities

Automated Navigation Puppeteer automates navigation through web pages to locate and access specific sections containing user reviews. For platforms like Amazon, this involves identification of review sections, dynamic loading of additional reviews in case they are needed, and avoiding potential roadblocks such as CAPTCHAs and rate-limiting mechanisms.

3.4 Data Extraction

The primary goal of using Puppeteer is to extract essential review data, including:

Review Content: The textual content of each review, which serves as the foundation for textual analysis.

Ratings: Each review will carry numerical ratings that will quantify user sentiment.

Metadata: These include dates of review, reviewer IDs, and product identifiers. These make the dataset more comprehensive and enable further analysis.

3.5 Advantages of Puppeteer

Dynamic Interactions: With Puppeteer, it is possible to interact with content rendered using JavaScript, ensuring that data otherwise inaccessible to traditional scraping tools will be captured.
Customizable Scraping: The library offers flexibility for tailoring the workflows of scraping to adapt with changes in the structure of web pages.
Robust Error Handling: Puppeteer contains mechanisms that handle error and retries smoothly, preventing interruption of data collection work.
By using the advanced capabilities of Puppeteer, we were able to create a rich and diverse dataset that is a must for training a reliable machine learning model. The dataset created is the foundation of our project, helping in the accurate classification of reviews and contributing to the bigger goal of improving transparency in e-commerce platforms.

4. IMPLMENTATION

The frontend of the audit genuineness verification application built with React provides an intuitive user interface for checking the audit data. The main features include real-time visualizations of the audit genuineness scores through charts.
Scraper API and AI evaluation API are hosted in server using Docker. The frontend interacts with the API's to give real-time data to verify the genuineness of the reviews.
Relational databases such as MySQL or PostgreSQL were used to maintain datasets of user interactions, reviews, and system logs. Tools like PyCharm, VSCode , and Jupyter Notebooks were also there for development, coupled with Git for versioning hosted on GitHub. Cloud platforms such as AWS or GCP were used for hosting, providing scalable resources for model and database services. Docker was used to containerize the application for easier deployment.

5. TESTING AND PERFORMANCE

The performance of the classification model was evaluated on a labeled dataset and results summarized in form of classification report, which gives insight in how well a model performs in classifying data in two separate categories referred to as Course 0 (HG) and Course 1 (CG). The
following metrics are used to evaluate the effectiveness of classification model: Precision, Recall, and F1-Score. Each of these metrics gives another view of the ability of this model to correctly classify an instance, and the final result gives a comprehensive view of it.

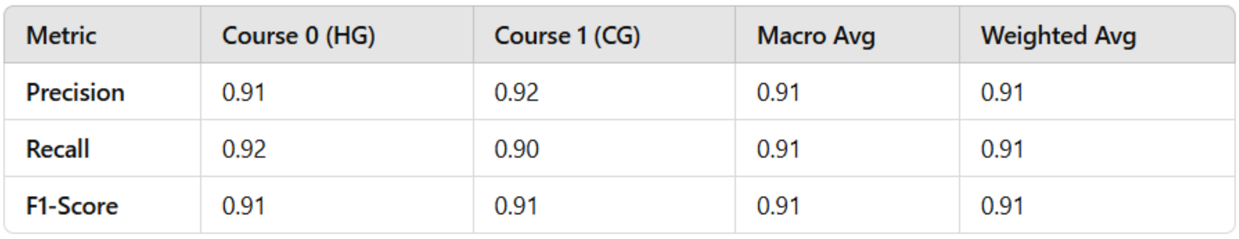
5.1 Classification Report

It includes the precision, recall, and F1-score for both Course 0 (HG) and Course 1 (CG), as well as macro-average and weighted-average values across all classes.
Precision: The precision measures how many of the instances predicted as a particular class (either Course 0 (HG) or Course 1 (CG)) are actually correct. It is defined as the ratio of true positives to the sum of true positives and false positives. A precision of 0.91 for both Course 0 (HG) and Course 1 (CG) indicates that 91% of the instances predicted as either class were correct. This means the model has a high level of accuracy in positive predictions whether it was for Course 0 (HG) or Course 1 (CG).
Recall: Recall measures to what extent the model was able to correctly identify all the instances of a given class. It is defined as true positives divided by the sum of true positives and false negatives. In this report, Course 0 (HG) has a recall of 0.92, meaning that 92% of all actual instances of Course 0 (HG) were correctly identified. Course 1 (CG) has a recall of 0.90, meaning that 90% of actual Course 1 (CG)

instances were correctly recognized by the model. This indicates that the model is very effective at detecting both classes' instances, though it prefers Course 0 (HG) slightly in terms of recall performance. F1-Score: The F1-score is actually the harmonic mean of the precision and the recall values. It presents an average assessment of a model when it takes into account class-imbalance issues. So in case of Classes 0: HG with F1 = 0.91 as well as Class 1: CG with F1 = 0.91 also a superb figure, ensuring a good ratio of precision relative to recall for both, thus providing optimal results:.

5.2 Macro Average and Weighted Average

Macro Average is the unweighted average of precision, recall, and F1-score over all classes. The Macro Average gives equal weightage to all classes, irrespective of the number of instances in each class.



Metric	Course 0 (HG)	Course 1 (CG)	Macro Avg	Weighted Avg
Precision	0.91	0.92	0.91	0.91
Recall	0.92	0.90	0.91	0.91
F1-Score	0.91	0.91	0.91	0.91

Fig 1.1 Classification Report

The **ROC-AUC Score** is a crucial metric used to evaluate the model's ability to differentiate between two classes. A high ROC-AUC score indicates whether model is highly effective at distinguishing between the two categories, in this case, **Course 0 (HG)** and **Course 1 (CG)**. The model's performance in this regard is exceptional, showcasing its strong **two-class separation ability**. This means that the model can reliably predict which instances belong to **Course 0 (HG)** and which belong to **Course 1 (CG)**, with minimal overlap between the two classes. A high ROC-AUC score is particularly valuable in situations where it is important to differentiate between classes, as it reflects the model's ability to avoid both false positives and false negatives across a wide range of decision thresholds.

5.2.1 Confusion Matrix

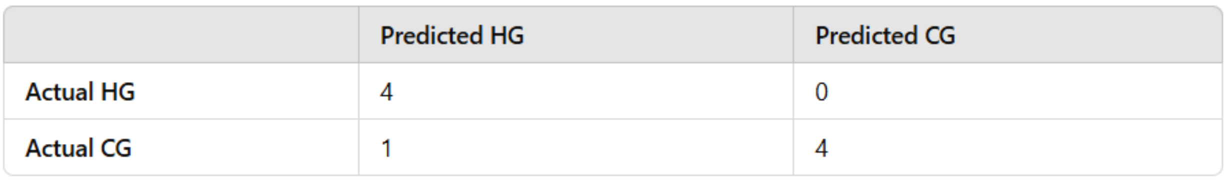
The confusion matrix provides deep insight in the working of the classification model by showing which classes has many instances were wrongly and correctly predicted. This is shown in following confusion matrix:

True Positives (TP): The model correctly predicted 4 instances of HG (Course 0) as HG and 4 instances of CG (Course 1) as CG.

False Positives (FP): It mistakenly predicted 0 cases of HG as CG and 1 case of CG as HG.

False Negatives (FN): The model missed 1 instance of HG and predicted it as CG.

True Negatives (TN): There were 4 cases of CG that were actually classified as CG.



	Predicted HG	Predicted CG
Actual HG	4	0
Actual CG	1	4

Fig 1.2 Confusion Matrix

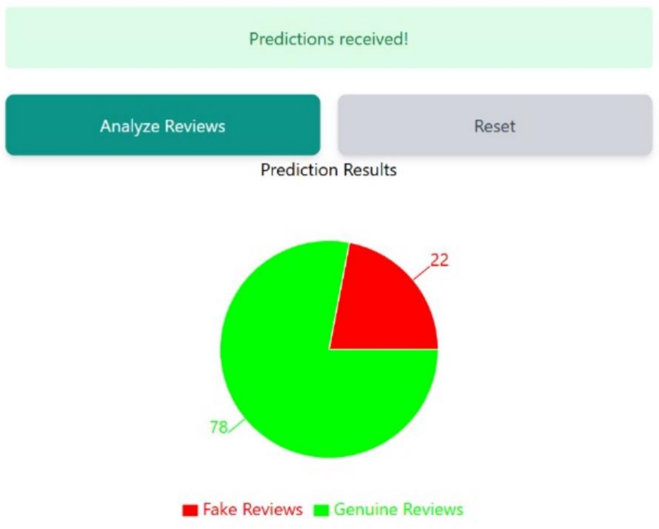


Fig. 1.3 Visualization of result

An additional important metric is the measure of the overall proportion of correct predictions, termed as accuracy. The model has achieved 91% accuracy. That is, the model predicts whether a review is fake or genuine 91% of the time. Overall, it is showing high performance.
This is a react frontend shown in the pie chart above, it is implemented using a react library chart.js. This visualizes the result from the analysis of reviews.

6. CONCLUSION

In conclusion, the proposed framework in this work is a very effective solution to issue of fake audits on e-commerce platforms. Combining machine learning classification techniques with linguistic analysis and automation results in a remarkable accuracy and adaptability in detecting fraudulent audit reports. This framework represents a giant leap forward in automating the audit verification process, which ensures only genuine and reliable data used in decision-making.
The results, as shown by the classification report, ROC-AUC score, and confusion matrix, indicate the model is capable to distinguish in between audit categories, HG and CG, and does so well in all the evaluation metrics, such as precision, recall, F1-score, and area under the curve. This strong performance will be helpful for stakeholders who are looking to automate and improve their audit verification processes in an efficient and scalable manner.
Future tasks with this framework will be devoted to the following areas:
The dataset should be expanded. The generalization of model can achieve by collecting more data. In other words, by training the system on a wider scope of audit reports, it can be developed to tackle more complex or varying cases of fraudulent behavior that make the system more robust.
The forward step involves using transformer-based models, such as BERT. The models have the ability to grasp language nuances at a deeper level and can significantly enhance the content understanding capability of the framework, thus enabling more accurate classifications.