Determining the Most Accurate Text Classifier Model at Predicting Whether Online Product Reviews are Human or Computer Generated

Abstract

In recent years, the issue of fake online product reviews have become more and more prevalent (McCluskey). This project attempts to address this issue of pervasive fake reviews negatively impacting e-commerce platforms by focusing on the automated detection of computer-generated reviews. The study evaluates various text classifiers, including Logistic Regression, Support Vector classifier, Decision Tree classifier, K-Nearest Neighbors (KNN) classifier, Random Forest classifier, Extra Trees classifier, AdaBoost classifier, Bagging classifier, and Gradient Boosting classifier, and compares their performances in accomplishing these tasks. These classifiers are assessed based on their accuracy and F1 scores in distinguishing between human and computer-generated reviews after being trained on a dataset comprising of approximately 20,000 computer-generated and approximately 20,000 human-written reviews, which has transformed into numerical vectors using TF-IDF vectorization. Results indicate that Logistic Regression consistently outperforms other classifiers, demonstrating robust accuracy and F1 scores across trials. K-Nearest Neighbors classifier shows the poorest performance, likely due to challenges in high-dimensional text data. Ensemble methods, such as Random Forest and Extra Trees, deliver notable success, leveraging multiple decision trees to enhance predictive performance. AdaBoost and Gradient Boosting also demonstrate competitive results, showcasing the capabilities of adaptive boosting. The study concludes that Logistic Regression is the most accurate classifier for detecting computer-generated reviews, offering insights into its simplicity and effectiveness in capturing linear patterns within the data. Further exploration could involve tuning hyperparameters for models with poor performance, and exploration with even more models.

Introduction

In recent years, fake reviews have emerged as a pervasive and concerning issue plaguing e-commerce platforms in the contemporary digital landscape, even influencing “around $152 billion in global spending on lackluster products” in 2021 (McCluskey). As online shopping continues to gain popularity, the reliance on reviews as a decision-making factor for making online purchases has surged, making them an integral part of the consumer experience. However, the rise of fake reviews has tainted the authenticity and reliability of user-generated content on these platforms. A primary challenge posed by fake reviews is the distortion of product perception. Consumers heavily depend on reviews to gauge their quality, functionality, and overall satisfaction associated with a product, and thus, potential buyers are misled into making decisions based on fabricated positive feedback or negative criticism, resulting in an inaccurate representation of the product ("Emplifi Reveals").

Furthermore, studies show that people are only about 60-80% accurate in detecting fake reviews (Walther et al.). Human susceptibility to biases, cognitive limitations, and the sheer volume of reviews on online platforms make it challenging for individuals to consistently identify deceptive content. The reliance on human judgment alone leaves the online review ecosystem vulnerable to manipulation and deceit. Recognizing the shortcomings in human review scrutiny emphasizes the critical need for sophisticated technological interventions. Automated systems, equipped with advanced algorithms and machine learning models, can provide a more objective and effective means of detecting computer-generated reviews. This demonstrates that to address this issue, there is a dire need fo automated solutions aimed at detecting computer-generated reviews. This research project seeks to contribute to ongoing efforts, such as those recently initiated by Amazon (Mehta), in enhancing the integrity of online reviews by focusing on the type of model used to detect these computer-generated reviews.

These reviews, often produced at scale by automated systems, pose a unique challenge due to their ability to mimic authentic user feedback (Salminen et al.). The nuances may range from subtle linguistic patterns to context-specific cues that people and traditional classifiers may struggle to discern. Given this complexity, the choice of classifier becomes crucial in addressing this specific issue. Different classifiers exhibit varying degrees of effectiveness in distinguishing between authentic and computer-generated reviews. This project aims to systematically evaluate and compare the performance of various classifiers to ascertain which type is most effective at accurately solving the challenges posed by computer-generated reviews. By identifying the strengths and weaknesses of different classifiers, the research endeavors to provide valuable insights that can guide the development of more robust and versatile automated systems capable of enhancing the integrity of online review platforms.

Methods

Data Cleaning and Acquisition

The original text dataset (Salminen) utilized for this project consisted of approximately 20,000 computer-generated reviews and an equivalent number of human-written reviews, organized in the format illustrated in figure 1.

Figure 1. First 11 of the approximately 20,000 rows of computer-generated reviews

category	rating	label	text_
Home_and_Kitchen_5	5.0	CG	Love this! Well made, sturdy, and very comfortable. I love it!Very pretty
Home_and_Kitchen_5	5.0	CG	love it, a great upgrade from the original. I've had mine for a couple of years
Home_and_Kitchen_5	5.0	CG	This pillow saved my back. I love the look and feel of this pillow.
Home_and_Kitchen_5	1.0	CG	Missing information on how to use it, but it is a great product for the price! I
Home_and_Kitchen_5	5.0	CG	Very nice set. Good quality. We have had the set for two months now and have not been
Home_and_Kitchen_5	3.0	CG	I WANTED DIFFERENT FLAVORS BUT THEY ARE NOT.
Home_and_Kitchen_5	5.0	CG	They are the perfect touch for me and the only thing I wish they had a little more space.
Home_and_Kitchen_5	3.0	CG	These done fit well and look great. I love the smoothness of the edges and the extra
Home_and_Kitchen_5	5.0	CG	Great big numbers & easy to read, the only thing I didn't like is the size of the
Home_and_Kitchen_5	5.0	CG	My son loves this comforter and it is very well made. We also have a baby
Home_and_Kitchen_5	5.0	CG	As advertised. 5th one I've had. The only problem is that it's not really a

The primary objective of this project was to identify computer-generated reviews without relying on star-rating and product category information. Consequently, both the

star-rating and product category columns were eliminated from the dataset. Subsequently, the dataset underwent a division into training and testing sets, maintaining an 80-20% split ratio. This division was achieved using the train_test_split method from the sk.model_selection module. Following the division, the dataset was transformed into a vector format using a TF-IDF (Term frequency-inverse document frequency) vectorizer.

Figure 2. Twelve datapoints after being vectorized

```
(0, 17743)    0.1638616229486196
(0, 22473)    0.1174867182818726
(0, 32880)    0.06866838674685247
(0, 23238)    0.11224381719100973
(0, 4076)     0.08961306159008037
(0, 36467)    0.09255355402985942
(0, 33047)    0.14271786222210697
(0, 32868)    0.2260601741198411
(0, 19729)    0.13425463771390117
(0, 4997)     0.23500621968219912
(0, 8410)     0.2805758439156411
(0, 21620)    0.10236632090062607
```

This entire process was repeated twice to accommodate for three distinct trials (later labeled as Trial 1, Trial 2, and Trial 3). In each trial, every classifier underwent training and testing on identical datasets to minimize potential areas of variability and enhance comparability. The performance metrics, including accuracy and F1 score, were systematically recorded for each classifier in every trial. This rigorous approach ensures the reliability and consistency of the results obtained across the three trials.

Classifiers Tested

A total of nine distinct text-classification algorithms were tested for their efficacy in the detection of computer-generated product reviews. As mentioned previously, this was done in three different trials, with each trial having the dataset split differently, but consistently between classifiers. Each algorithm that was tested carries a unique set of features and classifying algorithm, allowing us to not only identify the most effective classifiers, but also

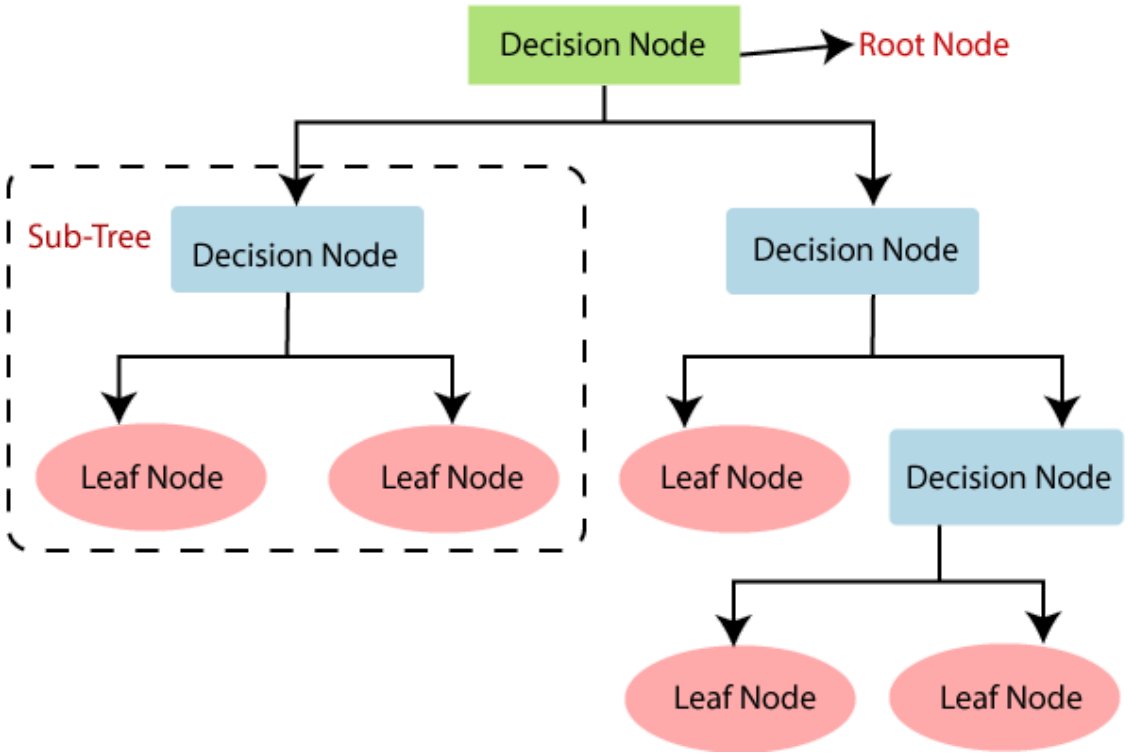
gain insight about patterns within the training data itself. This section serves as a preliminary exposition of these classifiers, explaining their basic mechanisms and shedding light on their respective strengths and limitations.

Logistic Regression is a classification method that generally models the probability of a binary outcome, and thus is useful in problems like this one. It works by essentially fitting a best-curve dividing line to the data it is trained with, effectively creating a decision boundary and outputting a probability of a certain outcome given an inputs features. This makes it well-suited for problems in which the relationship between features can be clearly delineated by some curve ("Logistic Regression"). Similarly, Support Vector classifiers work by finding a hyperplane in a high-dimensional space that most effectively separates datapoints with different labels (effectively a higher-dimension equivalent to a two-dimensional decision boundary). In this way, they plot multidimensional decision boundaries, similarly to Logistic Regression, but are more versatile than the latter in many cases (Saini).

Decision Tree classifiers, on the other hand, function very differently. Instead of creating decision boundaries, Decision Tree classifiers recursively split a labeled dataset based on its features to create a tree-like structure, hence the algorithm's name. Each internal node represents a decision based on a feature, each branch a rule, and each leaf node a class label, as shown in figure 3. This algorithm is very versatile, and can be used in both classification and regression problems ("Decision Tree"). The Random Forest classifier also utilizes this algorithm, essentially functioning as a fusion between a voting classifier and a decision tree classifier, in that it builds multiple decision trees, and then combines their predictions. Because it introduces randomness during both the trees' creations and the resulting voting process, it reduces overfitting and improves its generalization capabilities (Dutta). The Extra Trees classifier is very similar to both the Decision Tree and Random Forest classifiers, but it introduces even more randomness during the tree-building process by

aggregating the results of many decorrelated trees and by utilizing more trees than the Random Forest classifier. This additional randomness enhances the model's robustness, especially in the presence of noisy data (Gupta).

Figure 3. Visual representation of Decision Tree classification algorithm ("Decision Tree Classification")



An AdaBoost classifier is an adaptive boosting ensemble algorithm that begins with a weak classifier, then boosts the weak classifier by weighing misclassified datapoints slightly more heavily, and repeats this process until it arrives at a strong classifier. It then combines all of the classifiers it created in this way, weighting classifiers by their expected accuracy. It is thus less prone to overfitting and performs well in practice (Saura H.). Similarly to an AdaBoost classifier, Gradient Boosting sequentially builds a series of weak learners, with each subsequent learner correcting the errors of its predecessor by minimizing a cost function using gradient descent, resulting in a strong and accurate predictive model (Nikki).

Bagging classifiers are another type of voting classifier that involve training multiple instances of the same model on different bootstrap samples of the dataset, and aggregating their predictions. Compared to classic voting classifiers, this ensemble method helps reduce overfitting and improves stability (Dey). The K-Nearest Neighbors (KNN) classifier classifies data points based on the majority class (and weights) of their K-nearest neighbors in the feature space. While very simple, this algorithm is a non-parametric and instance-based learning algorithm which does not make any assumptions about data distribution, and is thus essential to machine learning (LaViale).

Results and Discussion

Each text classifier performed differently between trials and from each other in the task of detecting computer-generated reviews. Multiple models, including Logistic Regression, Support Vector Machine (SVM), Decision Tree, K-Nearest Neighbors (KNN), Random Forest, Extra Trees, Adaboost, Bagging, and Gradient Boosting, were subjected to standardized evaluation; each model underwent three trials, during which accuracy and F1 scores were recorded to comprehensively assess their effectivenesses (which are reflected in tables 1-11). Furthermore, these effectivity scores were analyzed and interpreted in context of each classifier's typical strengths and weaknesses, and serve as a foundation for proposing potential avenues of improvement for these classifiers.

Logistic Regression consistently exhibited strong performance across all of its trials, as shown in table 1. Several factors contribute to the effectiveness of Logistic Regression in the context of detecting computer-generated reviews– firstly, the simplicity of the model, relying on a linear decision boundary, appears well-suited to capture the underlying patterns in the data. Although this was not previously observed, the results of this model suggest that the relationships between features and the target variable are approximately linear.

Furthermore, Logistic Regression's inherent resistance to irrelevant features allows it to focus on the most discriminative aspects of the reviews, potentially filtering out noise in the dataset. The consistency in performance across trials suggests that the linear model successfully generalizes to different subsets of the data, demonstrating its reliability for the binary classification task.

Table 1. Accuracy and F1 score performance of Logistic Regression

Logistic Regression	Trial 1	Trial 2	Trial 3	Mean
Accuracy	0.8929145542	0.8995919377	0.8986026957	0.8970363959
F1 score	0.8907117617	0.8979129997	0.8980606663	0.8955618093

The Support Vector Machine (SVM) also exhibited strong yet slightly lower performance compared to Logistic Regression, as seen in table 2. This performance can be attributed to SVM's inherent characteristics and the nuances of the dataset, since SVM's strength lies in its effectiveness in high-dimensional spaces, making it well-suited for text classification tasks with numerous features. However, SVM is known to be sensitive to outliers, and the dataset's handling of outliers might have impacted its performance. Further preprocessing techniques, such as outlier removal or robust feature scaling, could be explored to enhance SVM's robustness. Despite these considerations, SVM's competitive performance underscores its utility for text classification, suggesting that addressing specific characteristics of the dataset may further optimize its effectiveness in detecting computer-generated reviews.

Table 2. Accuracy and F1 score performance of Support Vector Classifier

Support Vector	Trial 1	Trial 2	Trial 3	Mean
Accuracy	0.8800544083	0.8780759243	0.8822802028	0.8801368451
F1 score	0.8784461153	0.8772715957	0.8821490468	0.8792889193

The Decision Tree model displayed comparatively lower performance, as shown in table 3, raising insights into its behavior and potential limitations. In general, Decision Trees

are inherently interpretable and capable of capturing non-linear relationships within the data. However, as indicated by Logistic Regression's success, the data was approximately linear, and thus, this functionality was not taken advantage of. Furthermore, Decision Trees' susceptibility to overfitting might have played a role in the observed outcomes, as its ability to create complex decision boundaries comes with a caveat of a higher sensitivity to noise and specifics of the training data, causing it to generalize less effectively to unseen instances. The lower accuracy and F1 scores across trials suggest that the Decision Tree struggled to encapsulate the intricate patterns inherent in distinguishing computer-generated reviews, shortcomings that an ensemble approach, like Random Forest, might be able to account for by exploring to harness the strength of multiple trees for improved predictive performance.

Table 3. Accuracy and F1 score performance of Decision Tree Classifier

Decision Tree	Trial 1	Trial 2	Trial 3	Mean
Accuracy	0.7333992828	0.7270928651	0.7289476938	0.7298132806
F1 score	0.7052228603	0.7009079821	0.706323687	0.7041515098

K-Nearest Neighbors (KNN) demonstrated the lowest performance among the classifiers, as shown in table 4. One primary factor contributing to its suboptimal results may be that the effectiveness of distance-based metrics diminishes as the number of features increases (Thorn). Given the nature of text classification tasks, which typically involve high-dimensional spaces due to the large number of unique words and features, KNN may have struggled to discern meaningful patterns. Furthermore, the default value of K might not have been optimal for the dataset, and exploring a range of values for K could help identify a more suitable configuration. The over-reliance on local information and the simple lack of inherent ability to capture global patterns might have further contributed to the K-Nearest Neighbors classifier's performance challenges.

Table 4. Accuracy and F1 score performance of K-Nearest Neighbors Classifier

K-Nearest Neighbors	Trial 1	Trial 2	Trial 3	Mean

Accuracy	0.5542228268	0.5555830345	0.5620131075	0.5572729896
F1 score	0.6867124359	0.6872062663	0.692961165	0.6889599558

Unlike Decision Tree, Random Forest, a robust ensemble method, demonstrated notable success across trials, which can be seen in table 5. The ensemble nature of Random Forest, comprised of multiple decision trees, played a pivotal role in its comparative effectiveness; this approach allowed the model to mitigate overfitting by aggregating predictions from diverse trees, each trained on a different subset of the data. Random Forest excels in capturing complex relationships within the dataset, leveraging the collective wisdom of individual trees. It is able to assign importance to features and thereby focuses on the most discriminative aspects of the reviews, contributing to its strong predictive performance. Its capacity to handle intricate patterns make it a compelling choice for text classification tasks, and further exploration might involve investigating the impact of the number of trees or other hyperparameter adjustments for potential optimizations.

Table 5. Accuracy and F1 score performance of Random Forest Classifier

Random Forest	Trial 1	Trial 2	Trial 3	Mean
Accuracy	0.8815382713	0.8725114381	0.8832694448	0.8791063847
F1 score	0.8844390832	0.8759475394	0.8875387181	0.8826417803

Like Random Forest, the Extra Trees classifier consistently exhibits robust and competitive performance, as one can see in table 6. This makes intuitive sense– The Extra Trees classifier is essentially just a Random Forest classifier that introduces even more randomness. The model's inherent strength, stemming from its ensemble of decision trees and their randomized nature, allows for it to capture complex relationships within the dataset, leveraging the collective wisdom of individual trees of the model. Its approach of being comprised of multiple decision trees allowed the model to mitigate overfitting by aggregating predictions from many different trees, each trained on a different subset of the dataset.

Further exploration might involve investigating hyperparameter tuning and other potential optimizations to fully leverage the model's capabilities.

Table 6. Accuracy and F1 score performance of Extra Trees Classifier

Extra Trees	Trial 1	Trial 2	Trial 3	Mean
Accuracy	0.8924199332	0.8936564857	0.893780141	0.89328552
F1 score	0.8952564411	0.8959341723	0.8976528059	0.8962811398

Adaboost, another ensemble method, demonstrated moderately competitive performance in the task of detecting computer-generated reviews, as shown in table 7. The model's adaptive boosting strategy, focusing on correcting misclassifications from previous weak learners, may have contributed to its success. Adaboost's consistent accuracy and F1 scores across trials indicate its ability to learn and adapt to complex patterns within the dataset. The lower accuracy compared to some other classifiers might be attributed to the model's sensitivity to outliers or noise in the data, which it may have focused to hard on correcting. Fine-tuning hyperparameters, such as the learning rate and the choice of weak learners, could potentially enhance Adaboost's performance. Additionally, exploring the impact of different weak learners, such as decision stumps or shallow trees, might provide insights into the model's behavior and avenues for improvement.

Table 7. Accuracy and F1 score performance of Adaboost Classifier

Adaboost	Trial 1	Trial 2	Trial 3	Mean
Accuracy	0.8261407197	0.8271299617	0.8308396191	0.8280367668
F1 score	0.825990099	0.8262922465	0.8305672529	0.8276165328

The Bagging classifier demonstrated robust and consistent performance, as seen in table 8. Because bagging involves training multiple models on random subsets of the data and aggregating their predictions, it proved successful in providing stability and reducing variance. This also allows the classifier to capture diverse patterns within the dataset, contributing to its high accuracy and F1 scores. In addition, the positive outcomes suggest

that the combination of models trained on diverse subsets complemented each other, collectively improving predictive performance. Further exploration might involve investigating variations in sampling techniques or adjusting hyperparameters such as the number and types of base models to understand the specific configurations that contribute to Bagging's success.

Table 8. Accuracy and F1 score performance of Bagging Classifier

Bagging	Trial 1	Trial 2	Trial 3	Mean
Accuracy	0.850129838	0.8444416966	0.853592185	0.8493879065
F1 score	0.8525906106	0.8472559495	0.8576238576	0.8524901392

Unsurprisingly, Gradient Boosting, as seen in table 9, exhibited similar results to Adaboost, showcasing its ability to iteratively refine the model and correct errors made by previous weak learners. Like in the case of Adaboost, the moderately competitive accuracy and F1 scores across trials suggest that the boosting strategy effectively contributed to the model's robustness. The iterative nature of boosting allows the model to adapt to complex patterns and nuances within the data, providing a nuanced understanding of distinguishing features in reviews. However, fine-tuning hyperparameters, such as the learning rate, may further optimize Gradient Boosting's performance. Additionally, experimenting with different weak learners or assessing the impact of varying the number of boosting iterations could unveil opportunities for refinement. Overall, the competitive performance of Gradient Boosting indicates its suitability for discerning computer-generated reviews, and further investigations into hyperparameter tuning and model variations may unlock its full potential for text classification tasks.

Table 9. Accuracy and F1 score performance of Gradient Boosting Classifier

Gradient Boosting	Trial 1	Trial 2	Trial 3	Mean
Accuracy	0.8216891307	0.8230493384	0.8299740324	0.8249041672
F1 score	0.8142231384	0.8181933681	0.826848004	0.8197548368

Table 10. Accuracy test performances of each classifier for three trials

Classifier	Trial 1	Trial 2	Trial 3	Mean
Logistic Regression	0.8929145542	0.8995919377	0.8986026957	0.8970363959
Support Vector	0.8800544083	0.8780759243	0.8822802028	0.8801368451
Decision Tree	0.7333992828	0.7270928651	0.7289476938	0.7298132806
K-Nearest Neighbors	0.5542228268	0.5555830345	0.5620131075	0.5572729896
Random Forest	0.8815382713	0.8725114381	0.8832694448	0.8791063847
Extra Trees	0.8924199332	0.8936564857	0.893780141	0.89328552
AdaBoost	0.8261407197	0.8271299617	0.8308396191	0.8280367668
Bagging	0.850129838	0.8444416966	0.853592185	0.8493879065
Gradient Boosting	0.8216891307	0.8230493384	0.8299740324	0.8249041672

Figure 4. Graph of accuracy test performances of each classifier for three trials

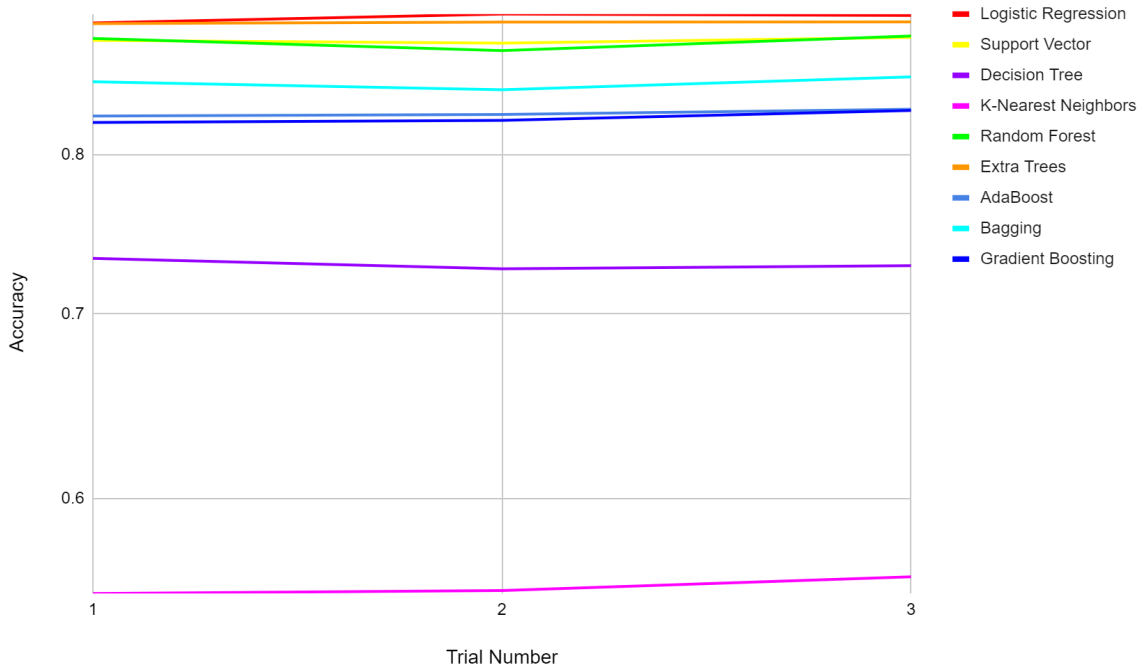
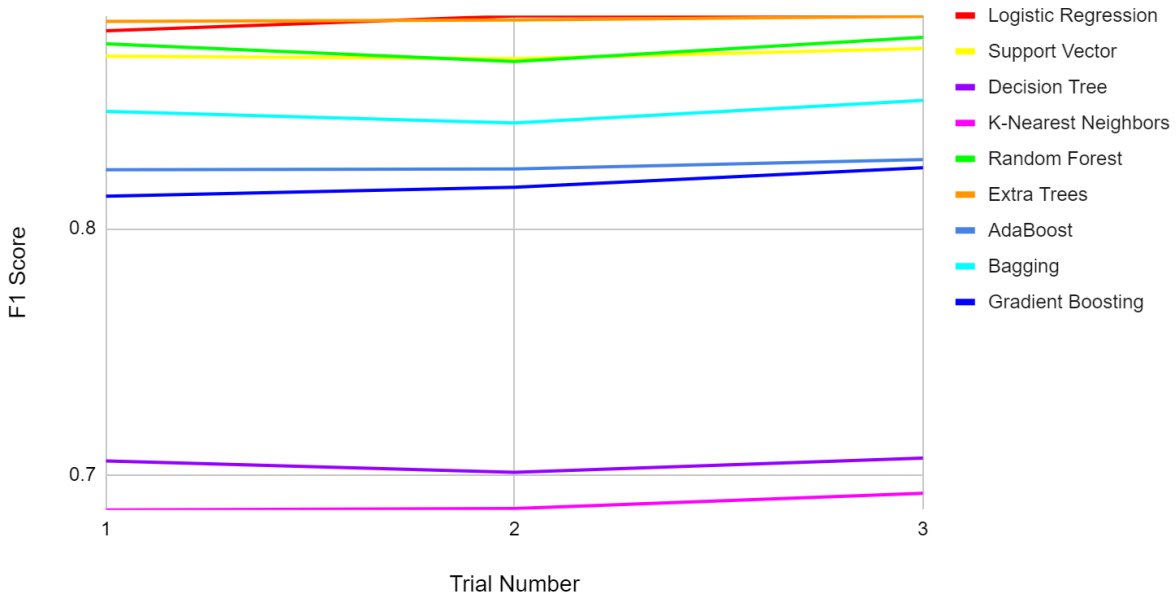


Table 11. F1 score performances of each classifier for three trials

Classifier	Trial 1	Trial 2	Trial 3	Mean
Logistic Regression	0.8907117617	0.8979129997	0.8980606663	0.8955618093
Support Vector	0.8784461153	0.8772715957	0.8821490468	0.8792889193
Decision Tree	0.7052228603	0.7009079821	0.706323687	0.7041515098
K-Nearest Neighbors	0.6867124359	0.6872062663	0.692961165	0.6889599558
Random Forest	0.8844390832	0.8759475394	0.8875387181	0.8826417803
Extra Trees	0.8952564411	0.8959341723	0.8976528059	0.8962811398

AdaBoost	0.825990099	0.8262922465	0.8305672529	0.8276165328
Bagging	0.8525906106	0.8472559495	0.8576238576	0.8524901392
Gradient Boosting	0.8142231384	0.8181933681	0.826848004	0.8197548368

Figure 5. Graph of F1 score performances of each classifier for three trials



In summation, the evaluation of text classifiers for detecting computer-generated reviews clearly revealed that certain classifiers were much more effective and accurate at the task, a disparity that is observable in both tables 10 and 11 and both figures 4 and 5. Logistic Regression consistently excelled, leveraging simplicity and linearity. Support Vector Machine showed potential in high-dimensional spaces but may have fallen short handling outliers. Decision Tree faced challenges, suggesting the need for ensemble approaches, but Random Forest and Extra Trees, both ensemble methods, exhibited robust performance, mitigating overfitting. K-Nearest Neighbors struggled in high-dimensional spaces, emphasizing the importance of exploring optimal parameters. Adaboost and Gradient Boosting demonstrated moderate success, indicating room for improvement through hyperparameter tuning. Bagging displayed consistent and robust performance, highlighting the benefits of model aggregation.

The comparison of accuracy and F1 scores (which have been visualized in figure 4 and figure 5) among the evaluated classifiers serves as a valuable foundation for future investigations aimed at enhancing the performance of text classifiers in the challenging task of reliably detecting computer-generated reviews. The observed variations in accuracy and F1 scores highlight specific strengths and weaknesses inherent in each model, and also suggest that to capitalize on these insights and ameliorate the models' performances, a targeted improvement approach involving a mixture of hyperparameter tuning and closer data analysis is warranted.

Conclusion

As discussed, fake online product reviews have become more and more prevalent, negatively impacting e-commerce platforms. Some of the issues surrounding fake reviews, such as their ability to mimic authentic user feedback and their sheer mass on popular platforms such as Amazon, make dealing with them a very demanding task for consumers. This research sought to address this issue, evaluating various text classifiers and reporting their accuracy and F1 scores. Logistic Regression, Support Vector, and Extra Trees proved to work best, consistently outperforming other classifiers, and demonstrating robust accuracy and F1 scores. K-Nearest Neighbors performed poorly in handling high-dimensional text data, and ensemble methods like Random Forest showed notable success, as did AdaBoost and Gradient Boosting. Logistic Regression emerged as the most accurate classifier for detecting computer-generated reviews, offering insights into the task's unapparent simplicity and proving effective in capturing linear patterns. Further exploration could involve tuning hyperparameters for poorly performing models and exploring additional classifiers.

Works Cited

"Decision Tree." *Geeks for Geeks*, 20 Aug. 2023, www.geeksforgeeks.org/decision-tree/. Accessed 15 Jan. 2024.

Dey, Debomit. "ML | Bagging classifier." *Geeks for Geeks*, 1 Aug. 2023, www.geeksforgeeks.org/ml-bagging-classifier/. Accessed 15 Jan. 2024.

Dutta, Avik. "Random Forest Regression in Python." *Geeks for Geeks*, 6 Dec. 2023, www.geeksforgeeks.org/random-forest-regression-in-python/. Accessed 15 Jan. 2024.

"Emplifi Reveals Nearly 90% of Consumers Say Customer Ratings and Reviews Have the Biggest Impact on Purchasing Decisions." *Emplify*, 28 Feb. 2023, emplifi.io/press/study-reveals-customer-ratings-reviews-impact-on-purchase-decisions. Accessed 11 Jan. 2024.

Gupta, Alind. "ML | Extra Tree Classifier for Feature Selection." *Geeks for Geeks*, 18 May 2023, www.geeksforgeeks.org/ml-extra-tree-classifier-for-feature-selection/. Accessed 15 Jan. 2024.

H., Saura B. "Implementing the AdaBoost Algorithm from Scratch." *Geeks for Geeks*, 23 May 2023, www.geeksforgeeks.org/implementing-the-adaboost-algorithm-from-scratch/. Accessed 15 Jan. 2024.

LaViale, Trevor. "Deep Dive on KNN: Understanding and Implementing the K-Nearest Neighbors Algorithm." *Arize*, 16 Mar. 2023, arize.com/blog-course/knn-algorithm-k-nearest-neighbor/. Accessed 15 Jan. 2024.

"Logistic Regression in Machine Learning." *Geeks for Geeks*, 7 Dec. 2023, www.geeksforgeeks.org/understanding-logistic-regression/. Accessed 15 Jan. 2024.

McCluskey, Megan. "Inside the War on Fake Consumer Reviews." *Time*, 6 July 2022, time.com/6192933/fake-reviews-regulation/.

Mehta, Dharmesh. "A Blueprint for Private and Public Sector Partnership to Stop Fake Reviews." *Amazon*, 12 June 2023, www.aboutamazon.com/news/policy-news-views/how-amazon-is-working-to-stop-fake-reviews. Accessed 11 Jan. 2024.

Nikki. "Gradient Boosting in ML." *Geeks for Geeks*, 31 Mar. 2023, www.geeksforgeeks.org/ml-gradient-boosting/. Accessed 15 Jan. 2024.

Saini, Anshul. "Guide on Support Vector Machine (SVM) Algorithm." *Analytics Vidhya*, 4 Jan. 2024, www.analyticsvidhya.com/blog/2021/10/support-vector-machinessvm-a-complete-guide-for-beginners/. Accessed 15 Jan. 2024.

Salminen, Joni. *Fake Reviews Dataset*. OSF, 24 Oct. 2023, osf.io/tyue9.

Salminen, Joni, et al. "Creating and Detecting Fake Reviews of Online Products." *ScienceDirect*, 10 Aug. 2021, www.sciencedirect.com/science/article/pii/S0969698921003374#sec1. Accessed 11 Jan. 2024.

Thorn, James. "The Surprising Behaviour of Distance Metrics in High Dimensions." *Medium*, 4 Jan. 2021, towardsdatascience.com/the-surprising-behaviour-of-distance-metrics-in-high-dimensions-c2cb72779ea6. Accessed 13 Jan. 2024.

Visual Representation of Decision Tree Classification Algorithm. *Java T Point*, www.javatpoint.com/machine-learning-decision-tree-classification-algorithm. Accessed 15 Jan. 2024. Chart.

Walther, Michelle, et al. "A Systematic Literature Review About the Consumers' Side of Fake Review Detection – Which Cues Do Consumers Use to Determine the Veracity of Online User Reviews?" *ScienceDirect*, May 2023,

www.sciencedirect.com/science/article/pii/S2451958823000118. Accessed 11 Jan. 2024.