International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 11 Issue: 05 | May 2024 www.irjet.net p-ISSN: 2395-0072

Echoes of Sentiment: Unveiling Amazon Alexa Product Reviews with Decision Trees, Random Forests, and XGBoost

Arkajit Banerjee

Department of Statistics , Acharya Prafulla Chandra College , Kolkata , West Bengal , India

---

Abstract - *In this study, we conducted an in-depth investigation into sentiment analysis of Amazon Alexa product reviews using a publicly available dataset sourced from Kaggle. Our research journey commenced with a meticulous Exploratory Data Analysis (EDA), unearthing nuanced insights and patterns inherent within the dataset. Leveraging visualization techniques, we illuminated key trends, sentiments, and correlations within the Amazon Alexa reviews landscape. A pivotal aspect of our analysis involved the generation of word clouds, distinguishing between positive and negative sentiments. Through these visual representations, we meticulously showcased prevalent words that drove specific sentiments, enhancing the interpretability of our findings. Moving forward, we undertook rigorous data pre-processing steps, encompassing stemming using the Porter Stemmer algorithm and uniform conversion of all characters to lowercase. Subsequently, we ventured into the realm of sentiment analysis model development, employing three of the most renowned machine learning algorithms: Decision Trees, Random Forests, and XGBoost. Finally, we arrived at a pivotal stage of our study, wherein we evaluated the performance of our sentiment analysis models. Utilizing confusion matrices, we conducted a comprehensive comparative analysis, scrutinizing the accuracy and efficacy of the Decision Trees, Random Forests, and XGBoost models. Our findings shed light on the strengths and limitations of each model, providing invaluable insights for future sentiment analysis endeavors. In summary, our research not only contributes to the burgeoning field of sentiment analysis but also offers actionable insights for businesses and stakeholders seeking to comprehend and leverage consumer sentiments within the Amazon Alexa product ecosystem.*

Key Words: Natural Language Processing , Bag of Words , Porter Stemmer , Decision Tree , XGBoost , Random Forest Classifier **.**

1.INTRODUCTION

In today's digital age, where vast volumes of textual data are generated daily across various online platforms, the need to extract valuable insights from this data has become paramount. In the ever-evolving landscape of natural language processing (NLP), sentiment analysis emerges as a critical facet, unraveling the intricate layers of human expression embedded in textual data. . Sentiment analysis, a subfield of NLP, involves the automated identification and classification of sentiments expressed in text, enabling organizations and individuals to understand public opinion, consumer sentiment, and market trends. By leveraging NLP techniques, sentiment analysis algorithms analyze text at scale, uncovering valuable insights that can inform decision-making processes across a myriad of domains. From market research and brand sentiment analysis to social media monitoring and customer feedback analysis, sentiment analysis finds applications in diverse fields, including marketing, finance, customer service, politics, and healthcare. Understanding the nuances of sentiment analysis and harnessing the power of NLP techniques are essential for unlocking actionable insights from textual data and gaining a competitive edge in today's data-driven landscape.

This paper aims to shed light on the imperative role of NLP techniques in sentiment analysis and its far-reaching impact on customer reviews of Amazon Alexa Reviews . Sentiment analysis plays a crucial role in understanding user interactions and feedback within voice-activated virtual assistants like Amazon Alexa. As one of the most popular virtual assistants on the market, Amazon Alexa interacts with millions of users daily, processing voice commands and providing responses to queries across a diverse range of tasks, including setting reminders, playing music, controlling smart home devices, and providing weather updates. Sentiment analysis of user interactions with Amazon Alexa can provide valuable insights into user satisfaction, preferences, and areas for improvement, enabling Amazon and third-party developers to enhance the user experience and tailor responses to better meet user needs. By integrating robust machine learning techniques like Decision Trees, Random Forests, and XGBoost into the analysis of user sentiment interactions with Amazon Alexa, stakeholders can gain deeper insights into user sentiment and behavior in the rapidly evolving landscape of voice-activated technology.

The key take-aways from the paper will be the rigorous exploratory data analysis (EDA) as well as the modelling approach . The rest of the paper's structure is delineated as follows: Section 2 delves into the pertinent background and literature survey. In Section 3, the experimental methodology is expounded, encompassing the elucidation of theories, methods, approaches employed. Section 4 elucidates the results and graphs derived from the experimental procedures. Section 5 provides insights into the Conclusion and Future Plans. The last section is about references .

2. LITERATURE REVIEW

Sentiment analysis is useful for businesses to make well-informed decisions because it helps to understand customers' opinions and sentiments towards products. Amine et al [1] provide a thorough comparison of sentiment analysis methods used on Amazon product reviews in this article. In particular, Amine et al use Random Forest, Support Vector Machines (SVM), and Logistic Regression—three well-liked machine learning algorithms. Their research focuses on assessing these algorithms' performance for sentiment classification in terms of F1 score, accuracy, precision, and recall. They make use of a meticulously selected dataset of Amazon product reviews that encompasses a wide variety of products and customer opinions and evaluate the benefits and drawbacks of each algorithm for extracting sentiment from the reviews through in-depth testing and analysis. The study's conclusions shed important light on how well Random Forest, SVM, and Logistic Regression work when analyzing the sentiment of Amazon product reviews.

The quick growth of e-commerce websites has forced consumers to shop in new ways. In addition to convenience, online shopping offers users "suggestions." Additionally, a lot of customer reviews are constantly available on shopping websites, which aids in decision-making and provides additional information about the product. Xu et al [2] used three models in this paper to represent the sentiment analysis of Amazon reviews: Random Forest, Naive Bayes, and SVM. To improve comparisons, these models are trained using token counts and term frequency-inverse document frequency (TF-IDF) features. The dataset containing information about Amazon reviews is explored, and classification performances are assessed using precision, recall, and F-1 scores. The findings demonstrate that while SVM and Random Forest models perform well on positive-labeled data, they perform poorly on negative- and neutral-labeled data. For all three categories, Naive Bayes performs the best overall. Classifications, however, could be skewed in the analysis. As a result, additional advancements in this field of study are anticipated in order to produce optimal and more accurate outcomes, as well as the implementation of more machine learning models.

In order to determine the effectiveness of ChatGPT, reviews from individuals of diverse ages and educational backgrounds are analysed by Bharati et al [3] to determine people's sentiment. Four distinct machine learning models and two hybrid models that combined SVM + Decision Tree and Random Forest + Logistic Regression were used in a comparative analysis. The poll also examined the different impacts of ChatGPT use, including how it makes people feel more drowsy and impairs their capacity for reasoned thought and problem-solving.

Numerous strategies have attempted to use the feature ensemble method to enhance the performance of tweet sentiment analysis techniques. The majority of earlier approaches, however, focused on modeling word syntactic information rather than taking into account the sentiment context of the words. Furthermore, few studies have discussed the effects of phrases with ambiguous sentiment or the placement of words.In order to address tweets with fuzzy sentiment, Huyen et al [4] suggested a novel method based on a feature ensemble model that takes into consideration word elements like lexical, word type, semantic, position, and sentiment polarity. Experiments using real data show that the suggested method is effective in enhancing tweet sentiment analysis performance in terms of the F1score.

In this project, Praveen Kumar et al [5] shed light on the fact that data produced by textual comments in addition to ratings help with effective data classification rather than just ratings alone, and they also get a variety of insights from the analysis. When someone wants to purchase a product, they typically look up reviews first before deciding whether or not to buy. To put it another way, the user only looks at the rating before choosing. Since we are aware that ratings can occasionally mislead consumers and should not be the only factor considered when making a decision, the authors are comparing ratings with comments to improve the effectiveness and veracity of product reviews.

In this paper, Ayesha et al.[6] employed machine learning algorithms such as SVM, random forest, and naive bayes to solve the sentiment analysis problem on any Amazon product. Following preprocessing, the qualifying datasets are classified using the SVM and naive bayes algorithms to remove superfluous data points, such as conjunctions, verbs, and punctuation. The accuracy offered by these existing algorithms was insufficient. The SVM, random forest, and naive bayes algorithms were applied in this work. The quantity of reviews a product receives indicates how accurate it is.

The dataset used in this paper was gathered by Sadhasivam et al [7] from the official product websites. These reviews must first be pre-processed to eliminate unnecessary information like conjunctions, stop words, verbs, and punctuations. Following pre-processing, the trained dataset is classified using the SVM and Naive Bayes algorithms. The accuracy that these current algorithms produced was insufficiently valuable. As a result, an ensemble approach has been used to improve the reviews' accuracy. An ensemble is a classification strategy that combines two or more algorithms and determines the mode value for each algorithm by using the vote reference. This paper combines the ensemble algorithm, SVM, and Naive Bayes. Compared to the current algorithm, the authors proposed an Ensemble method that helps provide better accuracy. The user is

advised to purchase the specific product after the accuracy has been determined based on the reviews.

3. EXPERIMENTAL METHODOLOGY

3.1 Data Analysis

Data Description : Nestled within the vast expanse of digital repositories lies the Amazon Alexa customer reviews dataset, a treasure trove of invaluable insights awaiting discovery. Sourced from Kaggle, this dataset is a testament to the rich tapestry of consumer experiences within the realm of voice-activated technology. Comprising five columns, each imbued with unique significance, it offers a panoramic view of customer sentiments and interactions. The "rating" column stands as a beacon, illuminating the spectrum of customer satisfaction with numerical precision, ranging from a humble 1 to a resplendent 5. Meanwhile, the "date" column serves as a temporal compass, guiding us through the ebb and flow of consumer sentiment over time. As for "variation," it unveils the diverse array of Amazon Alexa products that have captured the imagination of consumers worldwide, each variation a testament to the boundless innovation within the realm of smart devices. The "verified_reviews" column, a treasure trove of descriptive prose, unveils the heartfelt musings and candid critiques penned by customers, offering a glimpse into their experiences and perceptions. However, the narrative does not end there; a stroke of ingenuity graces this dataset with the addition of an "length" column. Crafted by the author, this column bears witness to the power of computational analysis, quantifying the essence of each review in the form of character counts. Through the lens of the “len” function, each review's length is meticulously recorded, providing a nuanced perspective on the depth and breadth of consumer feedback. In essence, this dataset transcends mere rows and columns, embodying the essence of consumer sentiment and technological innovation in the digital age.

Exploratory Data Analysis :

- A bar graph displays the distribution of ratings, showcasing the frequency of each rating(y axis) to ratings(x axis) category to provide insight into the distribution of customer sentiments .

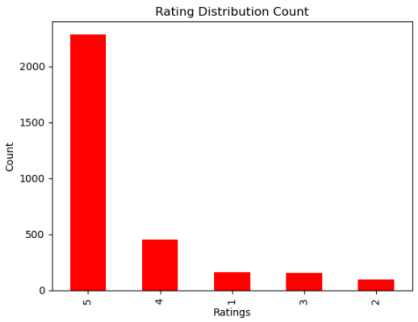

Fig 1 : Rating Distribution Count

- A feedback vs. count bar graph was crafted to unveil the distribution of feedback within the dataset, showcasing the frequency of each feedback. Here, feedback 1 signifies positive reviews, while feedback 0 denotes negative sentiments.

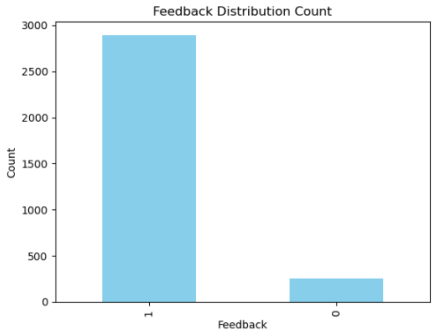

Fig 2 : Feedback Distribution Count

- A bar graph was created to illustrate the mean rating across different types of Alexa products, with variations depicted on the x-axis.

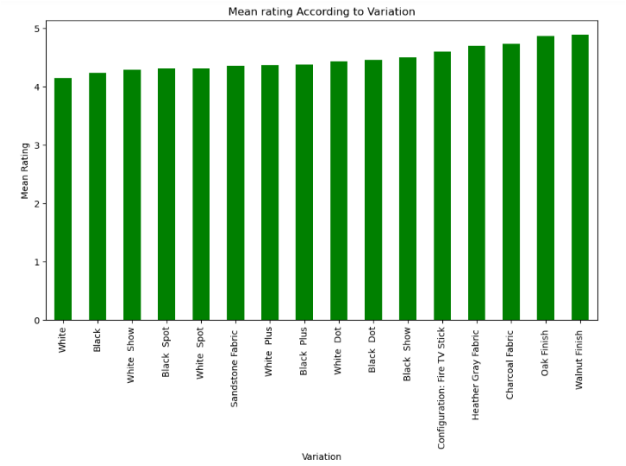

Fig 3 : Alexa Product vs Mean rating graph

- Distribution of length of reviews (both positive and negative) have also been plotted .

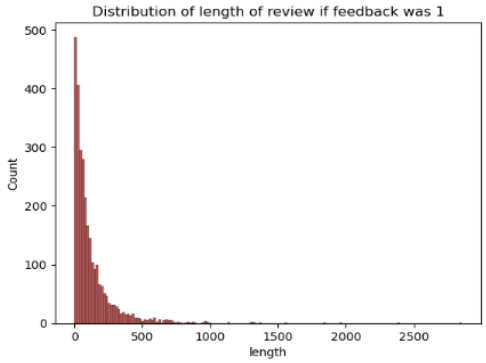

Fig 4 : Distribution of length of reviews when feedback was 1 (positive)

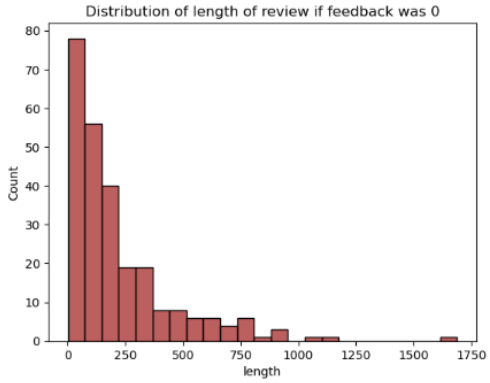

Fig 5 : Distribution of length of reviews when feedback was 0 (negative)

A wordcloud was drawn for all reviews with maximum words of 150 . Some of the highlighted words were “Alexa” , “Love” , “Use” , “Echo”

Fig 6 : Wordcloud for all reviews

3.2 Data Preprocessing

In preparation for our sentiment analysis, we begin by constructing a corpus from the "verified reviews" column of our dataset. A corpus, in the context of natural language processing, refers to a collection of text documents or data used for linguistic analysis. To ensure uniformity and consistency in our textual data, we perform a series of preprocessing steps. Firstly, we replace any non-alphabet characters with a space, ensuring that only alphabetic characters remain in our corpus. Subsequently, we convert all text to lowercase to standardize the text representation and mitigate the impact of case sensitivity. Following this, we tokenize the text by splitting it into individual words, facilitating further analysis at the word level. To refine our corpus and streamline subsequent analyses, we employ the Porter Stemmer algorithm—a widely used stemming algorithm in NLP. The Porter Stemmer algorithm reduces words to their base or root form, effectively removing inflections and suffixes. This process aids in consolidating semantically similar words, thereby enhancing the interpretability and efficiency of our analyses. By iteratively applying the Porter Stemmer algorithm to each word in our corpus and excluding stopwords—commonly occurring words with minimal semantic value—we curate a refined and standardized corpus poised for comprehensive analysis.

Following the application of the Porter Stemmer algorithm to refine our corpus, we proceed to employ the CountVectorizer technique for feature extraction and representation. CountVectorizer, a pivotal component in natural language processing, transforms our textual data into a numerical format suitable for machine learning algorithms. The CountVectorizer technique operates by constructing a "bag of words" representation—a fundamental concept in NLP. The bag of words model represents each document as a vector, where each dimension corresponds to a unique word in the vocabulary, and the value of each dimension represents the frequency of that word in the document. CountVectorizer accomplishes this by first constructing a vocabulary of all unique words in the corpus and then converting each document into a numerical vector based on the count of words in the vocabulary. This process yields a document-term matrix, where rows correspond to documents and columns correspond to words, with each cell containing the count of the corresponding word in the document. Subsequently, to facilitate future analyses and ensure reproducibility, we serialize the CountVectorizer object using the Python pickle module. Pickle enables us to save Python objects, such as CountVectorizer, to disk in a binary format, preserving their state and structure. By storing the serialized CountVectorizer object in "models/countVectorizer.pkl," we can easily reload and reuse it in subsequent analyses or deploy it in production environments, streamlining the process of transforming new text data into a suitable format for machine learning tasks. 30% data will be used for testing.

We now move to the model description and building stage .

3.3 Random Forest Classifier

Among the supervised learning methods is the well-known machine learning algorithm Random Forest [8]. It can be applied to ML problems involving both classification and regression. Its foundation is the idea of ensemble learning, which is the process of merging several classifiers to solve a challenging issue and enhance the model's functionality. If there are more trees in the forest, accuracy is higher and overfitting is avoided.

Two **assumptions** required for better Random Forest Classifier are : **(i)** for the classifier to forecast accurate results instead of a guess, the feature variable of the dataset needs to contain some real values and **(ii)** each tree's predictions need to have extremely low correlations.

Lets understand how the Random Forest classifier works:

Dataset Preparation: Begin with a labeled dataset containing features (independent variables) and corresponding target labels (dependent variable). **Random**

Sampling: Randomly select a subset of the dataset with replacement (bootstrapping) to create multiple training datasets. **Feature Selection**: Randomly select a subset of features from the dataset to train each decision tree in the forest. This promotes diversity among the trees and reduces correlation between them. **Decision Tree Construction**: For each training dataset, construct a decision tree using a subset of features. At each node of the tree, choose the best split based on a criterion such as Gini impurity or information gain. **Ensemble Creation**: Repeat steps 2-4 to create a specified number of decision trees (ensemble).

Prediction: To make predictions for new data, pass the data through each decision tree in the ensemble. For classification tasks, each tree casts a "vote" for the predicted class, and the majority class is selected as the final prediction. For regression tasks, the predictions from all trees are averaged to obtain the final prediction. **Evaluation**: Evaluate the performance of the Random Forest model using appropriate evaluation metrics such as accuracy, precision, recall, F1-score, or mean squared error, depending on the task. **Feature Importance**: Assess the importance of features in making predictions by analyzing how much each feature contributes to the model's accuracy. This can be done by calculating feature importance scores based on measures such as mean decrease in impurity or mean decrease in accuracy. **Tuning Parameters**: Fine-tune the parameters of the Random Forest model, such as the number of trees (n_estimators), maximum depth of trees (max_depth), and minimum number of samples required to split a node (min_samples_split), through techniques like cross-validation to optimize performance. **Deployment**: Once the model is trained and evaluated satisfactorily, deploy it to make predictions on new, unseen data.

3.4 XGBoost

XGBoost [9], short for Extreme Gradient Boosting, is a powerful and efficient machine learning algorithm known for its exceptional performance in classification, regression, and ranking tasks. It employs a gradient boosting framework, where weak learners (typically decision trees) are sequentially added to the ensemble, each correcting the errors made by the previous learners. XGBoost's key features include regularization techniques to prevent overfitting, parallel and distributed computing capabilities for scalability, and support for custom loss functions to handle diverse objectives. Its optimization algorithm efficiently finds the optimal tree structure, and its flexibility allows for fine-tuning of hyperparameters to optimize performance. XGBoost has garnered widespread adoption in various domains due to its speed, accuracy, and versatility, making it a go-to choice for data scientists and practitioners seeking state-of-the-art machine learning solutions.

Lets understand how the XGBoost Classifier works:

Dataset Preparation: Begin with a labeled dataset containing features (independent variables) and corresponding target labels (dependent variable). **Initialize Base Learner**: Start with an initial base learner, typically a decision tree, and fit it to the training data.Compute Residuals: Compute the residuals (the difference between the actual and predicted values) for each sample in the training data. **Fit a New Tree**: Fit a new decision tree to the residuals, aiming to minimize the residuals' error. This tree is added to the ensemble, incrementally improving the model's predictive power. **Update Predictions**: Update the model's predictions by adding the predictions of the new tree to the previous predictions. The model learns from the mistakes of the previous trees, gradually reducing the overall error. **Regularization**: Apply regularization techniques, such as tree pruning and feature subsampling, to prevent overfitting and enhance the model's generalization ability. **Repeat Steps 3-6**: Iterate the process by computing residuals, fitting new trees, and updating predictions until a predefined stopping criterion is met, such as reaching a specified number of trees or achieving satisfactory performance on a validation set. **Predictions:** To make predictions for new data, pass the data through each tree in the ensemble and aggregate the predictions to obtain the final prediction. For regression tasks, predictions are typically averaged across all trees, while for classification tasks, a voting mechanism is used to determine the predicted class. **Evaluation**: Evaluate the performance of the XGBoost model using appropriate evaluation metrics such as accuracy, precision, recall, F1-score, or mean squared error, depending on the task. **Hyperparameter Tuning**: Fine-tune the hyperparameters of the XGBoost model, such as the learning rate, maximum depth of trees, and regularization parameters, through techniques like cross-validation to optimize performance.

3.5 Decision Tree Classifier

The Decision Tree [10] classifier is a versatile and interpretable machine learning algorithm used for both classification and regression tasks. It operates by recursively partitioning the feature space into distinct regions based on the values of input features, with each partition representing a decision rule or node. At each node, the algorithm selects the feature that best splits the data, typically based on criteria such as Gini impurity or information gain, to maximize the homogeneity of the resulting subsets. This process continues until a stopping criterion is met, such as reaching a maximum tree depth or achieving purity within the leaf nodes. Decision Trees offer transparency in model interpretation, as the resulting tree structure provides insight into the decision-making process. However, they are prone to overfitting with complex datasets, necessitating techniques like pruning and regularization to mitigate this risk. Despite their simplicity, Decision Trees serve as

foundational models in machine learning, forming the basis for more complex ensemble methods like Random Forest and Gradient Boosting.

Lets understand how the Decision Tree Classifier works:

Dataset Preparation: Begin with a labeled dataset containing features (independent variables) and corresponding target labels (dependent variable). **Feature Selection**: Identify the feature that best splits the dataset into homogeneous subsets based on a chosen criterion, such as Gini impurity or information gain. **Node Creation**: Create a node representing the chosen feature and its split criterion, partitioning the dataset into two or more child nodes based on the feature's values. **Recursive Splitting**: Repeat the feature selection and node creation process recursively for each child node until a stopping criterion is met, such as reaching a maximum tree depth, achieving purity within the leaf nodes, or exceeding a minimum number of samples per leaf. **Leaf Node Assignment**: Assign a class label or regression value to each leaf node based on the majority class or average target value of the samples within the node.

Prediction: To make predictions for new data, traverse the decision tree from the root node to a leaf node based on the values of the input features. The prediction is then based on the class label or regression value assigned to the leaf node reached. **Evaluation**: Evaluate the performance of the Decision Tree model using appropriate evaluation metrics such as accuracy, precision, recall, F1-score, or mean squared error, depending on the task. **Hyperparameter Tuning**: Fine-tune the hyperparameters of the Decision Tree model, such as the maximum tree depth, minimum samples per leaf, and criterion for node splitting, through techniques like cross-validation to optimize performance.

4. RESULTS AND DISCUSSION

4.1 Evaluation Metrics

A confusion matrix is a powerful tool used to evaluate the performance of a classification model by presenting a comprehensive summary of the model's predictions compared to the actual class labels in the dataset. It provides a clear and concise breakdown of the model's classification results, enabling the calculation of various performance metrics.

Here's how a confusion matrix helps evaluate a model's performance:

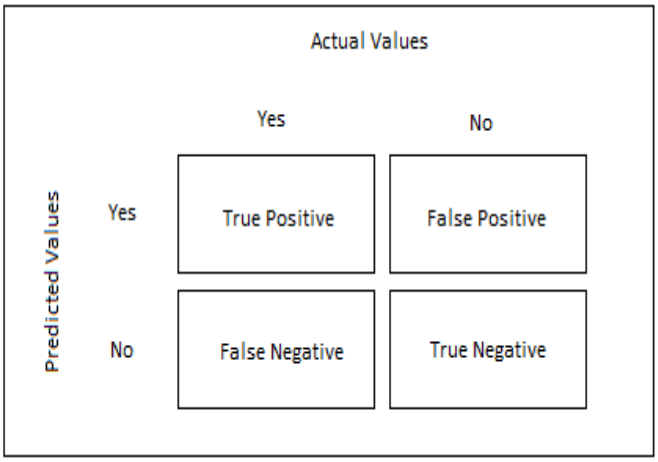

Fig 7 : Structure of a Confusion Matrix

True Positives (TP): The cases where the model correctly predicts the positive class.

False Positives (FP): The cases where the model incorrectly predicts the positive class when it is actually negative (Type I error).

True Negatives (TN): The cases where the model correctly predicts the negative class.

False Negatives (FN): The cases where the model incorrectly predicts the negative class when it is actually positive (Type II error).

By organizing these four outcomes into a matrix, the confusion matrix provides a clear visualization of the model's performance across different classes. From this matrix, various evaluation metrics can be derived, including:

Accuracy: The proportion of correctly classified instances among all instances (TP + TN / Total).

Precision: The proportion of true positives among all instances predicted as positive (TP / (TP + FP)).

Recall (Sensitivity): The proportion of true positives among all actual positive instances (TP / (TP + FN)).

F1-score: The harmonic mean of precision and recall, providing a balanced measure of a model's performance.

Specificity: The proportion of true negatives among all actual negative instances (TN / (TN + FP)).

False Positive Rate (FPR): The proportion of false positives among all actual negative instances (FP / (FP + TN)).

The confusion matrix corresponding to each machine learning algorithm is stated below :

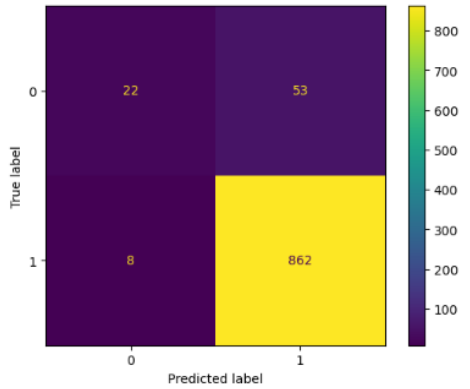

Fig 8 : Confusion Matrix for Random Forest Classifier

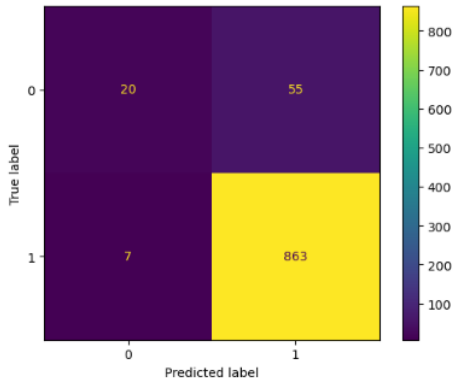

Fig 9 : Confusion Matrix for XGBoost Classifier

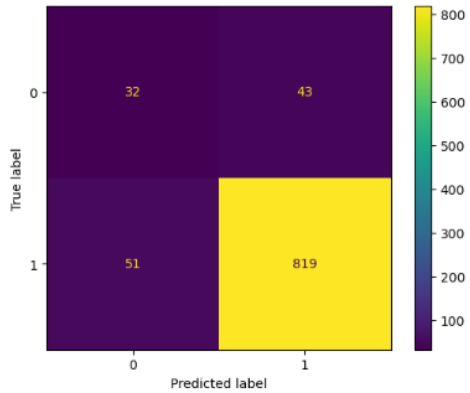

Fig 10 : Confusion Matrix for Decision Tree Classifier

The accuracy values corresponding to Train and Test Dataset are shown below for each corresponding algorithm :

Table -1: Comparison of Train Data Accuracy for the different algorithms

Algorithm	Accuracy
Random Forest Classifier	0.9941043083900227
XGBoost Classifier	0.9700680272108844
Decision Tree Classifier	0.9941043083900227

Interpretation :

The Random Forest and Decision Tree classifiers achieved similar high levels of accuracy 99.41 % each, indicating that they performed exceptionally well on the train dataset.

The XGBoost classifier achieved a slightly lower accuracy of 97.01 % compared to the Random Forest and Decision Tree classifiers, but still demonstrated strong predictive performance.

Table -2: Comparison of Test Data Accuracy for the different algorithms

Algorithm	Accuracy
Random Forest Classifier	0.9354497354497354
XGBoost Classifier	0.9343915343915344
Decision Tree Classifier	0.9037037037037037

Interpretation :

The Random Forest and XGBoost classifiers achieved similar levels of accuracy on the test dataset, with accuracies of approximately 93.54% and 93.44%, respectively.

The Decision Tree classifier achieved a slightly lower accuracy compared to the Random Forest and XGBoost classifiers, with an accuracy of approximately 90.37%.

Both the Random Forest and XGBoost classifiers outperformed the Decision Tree classifier on the test dataset, demonstrating their superior predictive performance.

Based on these accuracy scores, both the Random Forest and XGBoost classifiers perform similarly well on the test dataset. However, the Random Forest classifier has a slightly higher accuracy compared to the XGBoost classifier, indicating that it may perform slightly better in this particular scenario .

5. CONCLUSIONS AND FUTURE SCOPE

In conclusion, this study delved into sentiment analysis of Amazon Alexa customer reviews, employing three distinct machine learning algorithms: Random Forest, XGBoost, and Decision Tree classifiers. The evaluation of these models on the test dataset revealed promising accuracies, with the Random Forest and XGBoost classifiers demonstrating particularly robust performance, achieving accuracies of approximately 93.54% and 93.44%, respectively. These findings underscore the efficacy of machine learning techniques in discerning sentiment from customer reviews, offering valuable insights into consumer perceptions of Amazon Alexa products.

Moving forward, there exist intriguing avenues for further exploration and enhancement of this work. Future research endeavors could explore the integration of advanced natural language processing techniques, such as deep learning-based approaches, to capture more nuanced sentiments and improve model performance. Additionally, incorporating sentiment analysis of multimedia content, such as images and audio reviews, could provide a comprehensive understanding of customer sentiment across diverse modalities, enriching the insights gleaned from this analysis. By embracing these future directions, researchers can continue to advance the field of sentiment analysis and contribute to a deeper understanding of consumer sentiments in the context of emerging technologies like Amazon Alexa.

REFERENCES

[1] Amine, El & Zouhair, Abdelhamid. (2024). Sentiment Analysis Based on Machine Learning Algorithms: Application to Amazon Product Reviews. 10.1007/978-3-031-48573-2_38.

[2] Xu, Beiyu & Gan, Hongwu & Sun, Xinyue & Shao, Xiaoying. (2023). Sentiment analysis of Amazon product reviews. Applied and Computational Engineering. 6. 1673-1681. 10.54254/2755-2721/6/20230831.

[3] Bharati, Alokam & Bhargavi, Motamarri & Harshith, K.V.N.D. & K, Srinivasa. (2023). A Comparative Sentiment Analysis on ChatGPT Reviews using Machine Learning Models1-6 10.1109/ICCCNT56998.2023.10306609.

[4] Huyen Trang Phan, Dosam Hwang "Improving the Performance of sentiment analysis of Tweets Contaning Fuzzy Sentiments Using the feature Ensemble Model" IEEE Access, Volume 8, Feb 2020.

[5] Praveen Kumar, T. & Pamulaparty, Mr. (2023). Sentiment Analysis of Online Products using Ratings and Reviews , 14th International Conference on Advances in Computing, Control, and Telecommunication Technologies, Hyderabad

[6] Naureen, Ayesha & Siddiqa, Ayesha & Devi, Pothereddypally. (2022). Amazon Product Alexa's Sentiment Analysis Using Machine Learning Algorithms. In book: Innovations in Electronics and Communication Engineering 10.1007/978-981-16-8512-5_57.

[7] Sadhasivam, Jayakumar & Babu, Ramesh. (2019). Sentiment Analysis of Amazon Products Using Ensemble Machine Learning Algorithm. International Journal of Mathematical, Engineering and Management Sciences. 4. 508-520. 10.33889/IJMEMS.2019.4.2-041.

[8] Khaled Fawagreh, Mohamed Medhat Gaber & Eyad Elyan (2014) Random forests: from early developments to recent advancements, Systems Science & Control Engineering, 2:1, 602-609, DOI: 10.1080/21642583.2014.956265

[9] Chen, Tianqi & Guestrin, Carlos. (2016). XGBoost: A Scalable Tree Boosting System.22nd ACM SIGKDD International Conference 785-794. 10.1145/2939672.2939785.

[10] Quinlan, J.R. Induction of decision trees. *Mach Learn* **1**, 81–106 (1986). ttps://doi.org/10.1007/BF00116251

[11] Darshan, K & Samuel, Jerusha & Swamy, Manjunatha & Koparde, Prashant & Shivashankara, N. (2024). NLP - Powered Sentiment Analysis on the Twitter. Saudi Journal of Engineering and Technology. 9. 1-11. 10.36348/sjet.2024.v09i01.001.

BIOGRAPHY

Arkajit Banerjee is an Undergraduate student at the Department of Statistics , Acharya Prafulla Chandra College , Kolkata . He has been associated with several research projects under supervision of professors from premier institutes like IIT Indore , IIM Indore , IIIT Ranchi , IIM Ranchi . He is an active member of Jadavpur University Mathematical Society also . His research interests include Copula Theory , GARCH Models , Recommendation Systems , and real life application of ML techniques in various business domains .