Sentiment Analysis on Online Product Reviews using Naïve Bayes Classifier (August 2023)

Rijan Ghimire[1] and Sujan Bhattrai[1]

[1]Department of Electronics and Computer Engineering, IOE, Thapathali Campus, Kathmandu 44600, Nepal
Corresponding author: Rijan Ghimire (rijanghimire96@gmail.com),Sujan Bhattarai (sjnbhttr@gmail.com)

ABSTRACT One important activity in natural language processing is sentiment analysis, which looks for sentiments represented in textual data, including customer reviews and product reviews. The study starts with exploratory data analysis, which uses visualizations like word clouds and histograms to provide insights into the distribution of terms in the text data and which words are most commonly occurring. Textual input is transformed into numerical vectors for efficient machine learning through feature transformation, which is carried out using word- and character-level TF-IDF representations. For sentiment prediction, the Naive Bayes classifier is used, and its performance is evaluated using a range of metrics and visualizations. The results of the experiment show how effective the suggested methodology is; it can accurately classify sentiments into ”Happy” and ”Unhappy” categories. With ”Ok” attitudes, the classifier has difficulties, indicating areas that should be improved in the future.The study contributes to sentiment analysis by providing a robust framework and valuable insights for businesses to make data-driven decisions and improve customer satisfaction based on sentiment analysis. Future research directions include exploring sentiment analysis in multiclass settings and other languages for a more comprehensive understanding of customer sentiments.

INDEX TERMS Sentiment Analysis, Naive Bayes Classifier, TF-IDF Representation, Text Data Visualization, Machine Learning

I INTRODUCTION

This report presents a comprehensive exploration of sentiment analysis using a hybrid Naive Bayes classifier and TF-IDF (Term Frequency-Inverse Document Frequency) representations. Sentiment analysis, also known as opinion mining, is a crucial task in natural language processing that aims to determine the sentiment or emotion expressed in textual data, such as product reviews, social media posts, and customer feedback. Understanding customer sentiments is of great importance for businesses to make informed decisions, enhance customer satisfaction, and improve products and services.

This study begins with exploratory data analysis (EDA) to gain insights into the textual data. Visualizations such as histograms and Word Clouds provide a deeper understanding of text data distribution and frequently occurring words. The feature transformation is performed using TF-IDF representations to convert the textual data into numerical vectors, enabling machine learning algorithms to process and analyze the reviews for sentiment prediction effectively.

The Naive Bayes classifier is employed as the chosen supervised learning algorithm for sentiment analysis. It is widely used in text classification tasks and known for its simplicity and effectiveness. By making the assumption of conditional independence among features given the class label, the Naive Bayes classifier learns patterns from the data and predicts sentiment labels for new, unseen reviews.

To evaluate the model’s performance, various metrics and visualization techniques are utilized. The accuracy curve with increasing training set size allows us to examine the classifier’s performance with different amounts of training data. Also the precision-recall curves and ROC curves are constructed to assess the model’s ability to correctly predict positive instances and to discriminate between classes, respectively.

The experimental results demonstrate the effectiveness of the proposed hybrid Naive Bayes classifier with word- and character-level TF-IDF representations. The visualizations and performance metrics provide valuable insights into the model’s strengths and weaknesses. Through observation it is found out that the classifier performs well in correctly classifying ”Happy” and ”Unhappy” sentiments but struggles with ”Ok” sentiments, indicating potential areas for improvement and further analysis.

The contribution of this study lies in providing a robust sentiment analysis methodology with practical implications for businesses. By accurately predicting sentiment labels, companies can gain valuable insights into customer preferences, identify areas of improvement, and make data-driven decisions to enhance customer satisfaction. Furthermore, the incorporation of visualizations and comprehensive evaluations enables a thorough understanding of the classifier’s performance and highlights future research directions for sentiment analysis in multiclass settings and other languages.

II RELATED WORK

Related works in sentiment analysis have explored various approaches to text categorization and classification. Frasconi et al. [1] proposed a hybrid Naive Bayes Hidden Markov Model (HMM) approach for text categorization in multi-page documents. The method combined the simplicity and efficiency of the Naive Bayes algorithm with the

ability of the HMM to model sequential data. Kibriya et al. [2] revisited the Multinomial Naive Bayes classifier for text categorization, investigating different smoothing techniques to improve its performance. Their study focused on enhancing the accuracy and robustness of the classifier.

TF-IDF (Term Frequency-Inverse Document Frequency) has been extensively used in text mining tasks. Qaiser and Ali [3] explored the use of TF-IDF to examine word relevance in documents. They highlighted its effectiveness in identifying significant words that contribute to the meaning of documents. Ramdhani et al. [4] applied the Naive Bayes algorithm to perform sentiment analysis on product reviews. Their case study demonstrated the classifier's efficacy in categorizing sentiments expressed in reviews.

Some researchers have also shared their implementations for sentiment analysis. Shreyaswankhede [5] presented a GitHub repository with code for sentiment analysis on online product reviews. This open-source resource provides valuable insights into sentiment analysis techniques and serves as a helpful reference for practitioners.

Zhang and Li [6] explored the Naive Bayes text classifier for general text classification tasks. Their work discussed the application of the Naive Bayes algorithm and its potential use in various domains, including sentiment analysis.

Overall, the referenced works have made significant contributions to sentiment analysis, exploring different variations of Naive Bayes classifiers, TF-IDF representations, and text mining techniques. These studies provide valuable knowledge and inspiration for the development of the proposed sentiment analysis methodology and offer directions for future research in this field.

III METHODOLOGY

A DATASET DESCRIPTION

The dataset provided for the experiment consists of product reviews with various attributes. The dataset is stored in a CSV file named product.csv and contains the following columns:

1. Brand: This column represents the brand name of the product.

2. Categories: It lists the categories to which the product belongs, including "Movies, Music & Books," "Music," "R&b," "Movies & TV," "Movie Bundles & Collections," "CDs & Vinyl," "Rap & Hip-Hop," "Bass," "Music on CD or Vinyl," "Rap," "Hip-Hop," "Mainstream Rap," and "Pop Rap."

3. Manufacturer: The `manufacturer` column indicates the manufacturer or company responsible for producing the product.

4. Reviews.rating: This column contains the sentiment rating given by users for the product, represented as numeric values (e.g., 1, 2, 3, 4, or 5).

5. Reviews.text: The reviews.text column contains the textual content of the product reviews provided by users, expressing their opinions and experiences with the product.

6. Label: The label column represents the sentiment label associated with each review, indicating whether the sentiment is positive or negative.

7. Reviews.text: This column contains a reduced version of the Reviews.text, presumably processed and simplified for specific natural language processing (NLP) tasks or analysis.

The primary objective of this experiment is to perform sentiment analysis on the product reviews. By utilizing the textual content and sentiment ratings provided by users, the experiment aims to predict the sentiment of each review, thereby understanding the overall positive or negative sentiment towards products from different brands and manufacturers. Sentiment analysis can be a valuable tool for businesses to gain insights into customer opinions and preferences, enabling them to make informed decisions and improve their products and services.

B PROPOSED METHODOLOGY

To obtain an understanding of the textual data, the experiment begins with exploratory data analysis (EDA) in the suggested technique. Text Length (the number of characters in the review) and Word Count (the number of words in the review) are just a few of the text-related metrics that are calculated for every review. Histograms are used for visualization in order to gain a better understanding of the distribution of these features in overall reviews. Furthermore, a Word Cloud visualization is created to show the words that appear most frequently in the evaluations. The Word Cloud facilitates the identification of the most frequently occurring words within the dataset. The size of each word corresponds to its frequency in the reviews, offering a visual depiction of the salient features found in the textual data.

Feature Transformation:

In the feature transformation step, the textual data is prepared for machine learning algorithms using the TF-IDF (Term Frequency-Inverse Document Frequency) vectorizer. Two types of TF-IDF representations are created: Word-level TF-IDF and Character-level TF-IDF.

Word-level TF-IDF: The Word-level TF-IDF representation is created using the TfidfVectorizer from the scikit-learn library. This approach treats each individual word in the reviews as a feature. The TF-IDF vectorizer calculates the importance of each word in each review relative to its frequency in the entire corpus of reviews. Words that occur frequently in a specific review but are rare in the overall dataset are assigned higher importance, as they are considered more indicative of the sentiment expressed in that particular review.

Character-level TF-IDF: The Character-level TF-IDF representation is also created using the TfidfVectorizer. In this approach, combinations of characters (character n-grams) are treated as features. For example, for a value of n=2, character bigrams are considered, and for n=3, character trigrams are used. This approach allows the model to capture information from smaller units of text, such as specific sequences of characters, which can be valuable for sentiment analysis when sentiment is expressed in phrases or patterns.

TF-IDF Vectorization Process:

The TF-IDF vectorization process involves several steps:

1. Text Preprocessing: The text is converted to lowercase, removing punctuation and special characters, and eliminating common words known as stop words.
2. Term Frequency (TF) Calculation: The TF part of TF-IDF calculates the frequency of each term (word or character n-gram) in each review. The number of times a term appears in a review is divided by the total number of terms in that review, resulting in the term frequency for that term-review pair.
3. Inverse Document Frequency (IDF) Calculation: The IDF part of TF-IDF measures the importance of each term in the entire dataset. It is computed by dividing the total number of reviews by the number of reviews that contain the specific term. The IDF value is then logarithmically scaled to reduce the impact of very common terms.
4. TF-IDF Calculation: Finally, the TF-IDF value for each term-review pair is obtained by multiplying the TF value with the IDF value. This results in a numerical representation that captures the importance of each term in each review relative to the entire dataset.

By performing feature transformation with the TF-IDF vectorizer, the textual data is converted into numerical vectors that can be easily fed into machine learning algorithms for sentiment analysis. This transformation effectively represents the textual information in a format that allows the model to process and analyze the reviews to predict sentiment labels accurately.

Naïve Bayes:

In the proposed methodology, the sentiment analysis task is approached as a supervised learning problem, where the goal is to predict the sentiment rating (e.g., 1, 2, 3, 4, or 5) of product reviews based on their textual content. The objective is for the model to learn patterns and relationships in the data, enabling it to make accurate predictions on new, unseen data.

For sentiment analysis, the chosen classification algorithm is the Naïve Bayes classifier, specifically the MultinomialNB variant. Naive Bayes classifiers are probabilistic algorithms widely used in text classification tasks. They are based on Bayes' theorem and make the "naive" assumption that features (words or word combinations) are conditionally independent, given the class label. Despite this simplifying assumption, Naive Bayes classifiers have demonstrated effectiveness in various natural language processing (NLP) tasks, including sentiment analysis.

Validation:

The dataset is divided into two sets: the training set and the test set, in order to verify the effectiveness of the Naive Bayes classifier. The classifier is trained on a subset of the data using the training set, which enables it to discover the connections between words and the sentiment scores that go along with them. On the other hand, the testing set is used to gauge how effectively the model generalizes to fresh, untested data in order to determine how well it performs. An accuracy curve is plotted to examine how the classifier's accuracy changes with various training set sizes. This entails progressively expanding the training set's size while training the classifier on subsets of the training data. The accuracy curve shows how the model performs better while training with more data. Understanding the trade-off between model performance and the quantity of training data available is made easier by this analysis.

By rigorously validating the model's performance using the testing set and analyzing the accuracy curve, potential issues such as overfitting or underfitting can be identified and addressed. This ensures that the Naive Bayes classifier is reliable and capable of making accurate predictions on new product reviews, providing valuable insights into customer sentiments and preferences. The combination of feature transformation and model validation enhances the robustness and effectiveness of the sentiment analysis system, enabling businesses to make informed decisions and improve customer satisfaction.

Precision-Recall Curve:

The precision-recall curve is a graphical representation that illustrates the trade-off between precision and recall for different classification thresholds in a binary classification problem. Precision is the ratio of true positive predictions to the total positive predictions made by the model, measuring the accuracy of positive predictions. On the other hand, recall is the ratio of true positive predictions to the total actual positive instances in the dataset, indicating the model's ability to correctly identify positive instances. By considering both precision and recall, the precision-recall curve provides a comprehensive view of the model's performance across different thresholds.

In our project, the precision-recall curve is particularly valuable when dealing with imbalanced datasets, where one class may have significantly more instances than the other. In such cases, the model may achieve high accuracy but struggle to detect instances of the minority class. The precision-recall curve helps us evaluate the model's performance more effectively, as it considers the trade-off between precision and recall for both classes. A model with high precision and recall across various thresholds is preferred, as it indicates a balance between correctly predicting positive instances and minimizing false positives.

To handle the multiclass classification task in our sentiment analysis, where there are three sentiment classes (unhappy, ok, and happy), the "one-vs-rest" (OvR) approach is employed for constructing the Precision-Recall (PR) curve. The OvR approach allows us to plot separate PR curves for each sentiment class, providing insights into the classifier's performance for individual classes. This approach enables a more detailed and informative evaluation, allowing us to identify any class-specific issues, such as imbalanced data or variations in precision and recall among sentiment categories.

By utilizing the OvR approach and plotting separate PR curves for each class, a deeper understanding of the classifier's strengths and weaknesses is gained for different sentiment categories. This analysis helps us fine-tune the model and make data-driven decisions to improve its performance. Overall, the precision-recall curve and the OvR approach enhance the evaluation process in multiclass sentiment analysis, enabling us to make more informed choices

and create a more robust sentiment classifier.

ROC Curve:

In our sentiment analysis project, the ROC curve is constructed to assess the performance of the binary classifier, which distinguishes between positive and negative instances. The ROC curve plots the true positive rate (sensitivity) against the false positive rate (1-specificity) at various classification thresholds. This graph helps us understand the classifier's ability to discriminate between the positive class (e.g., 'happy') and the negative class (e.g., 'unhappy' and 'ok').

Since the ROC curve is not directly applicable to the multiclass classification task, label binarization is used to convert the multiclass targets into binary format, allowing us to create separate ROC curves for each class using the "one-vs-all" approach. This enables us to assess the binary classifier's performance for each sentiment class independently.

By plotting multiple ROC curves, the visualization, and comparison of how well the classifier distinguishes between positive and negative instances for each sentiment class is done. The area under the ROC curve (AUC) quantifies the overall performance of the classifier. Higher AUC values indicate better discriminative ability and overall classifier performance.

Using the ROC curve and AUC metrics in the multiclass sentiment analysis setting enhances our understanding of the classifier's effectiveness, helping us identify strengths and weaknesses for each sentiment category.

Algorithm 1 Text Preprocessing and TF-IDF Feature Transformation with Naïve Bayes Classifier

Require: Dataset with 'brand', 'manufacturer', 'reviews.rating', and 'reviews.text'

Ensure: Transformed TF-IDF features and Naive Bayes classifier performance

1: Import necessary libraries
2: Load train and test data from CSV files
3: Preprocess data by dropping irrelevant columns and handling missing values
4: Convert categorical variables to numerical labels
5: Split data into training and test sets
6: Initialize word-level TfidfVectorizer and fit on data to build vocabulary and calculate IDF values
7: Transform train_data into word-level TF-IDF features (train_features_word)
8: Initialize character-level TfidfVectorizer and fit on train_data to build vocabulary and calculate IDF values
9: Set batch size as 4000 for batch processing
10: Process train_data in batches to transform data into character-level TF-IDF features
11: Concatenate transformed batches into single sparse matrix (train_features_char)
12: Split data and target labels into training and testing sets
13: Initialize and train Naive Bayes classifier (MultinomialNB) on training data
14: Predict sentiment labels for the test set
15: Calculate accuracy and generate classification report
16: Print accuracy and classification report

C MATHEMATICAL FORMULAE

TF-IDF Formula:

1. Term Frequency (TF) for term t in document d:

$$TF(t,d) = \frac{\text{Number of occurrences of term } t \text{ in document } d}{\text{Total number of terms in document } d} \tag{1}$$

2. Inverse Document Frequency (IDF) for term t in the entire dataset:

$$IDF(t) = \log\left(\frac{\text{Total number of documents}}{\text{Number of documents containing term } t}\right) \tag{2}$$

3. TF-IDF for term t in document d:

$$TF - IDF(t,d) = TF(t,d) \times IDF(t) \tag{3}$$

Naïve Bayes Formula: The Naive Bayes classifier calculates the probability of a class label C given the input features x. The classifier assumes that the features are conditionally independent given the class label.

$$P(C|x) = \frac{P(x|C) \times P(C)}{P(x)} \tag{4}$$

Where: - $P(C|x)$ is the probability of class C given the features x. - $P(x|C)$ is the probability of the features x given class C. - $P(C)$ is the prior probability of class C. - $P(x)$ is the probability of the features x.

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN} \tag{5}$$

$$\text{F1-Score} = \frac{2 \times (Precision \times Recall)}{Precision + Recall} \tag{6}$$

True Positive (TP): The number of correctly predicted positive instances. False Positive (FP): The number of incorrectly predicted positive instances. True Negative (TN): The number of correctly predicted negative instances. False Negative (FN): The number of incorrectly predicted negative instances.

ROC Curve: The ROC curve is created by plotting the true positive rate (sensitivity) against the false positive rate (1 - specificity) for different classification thresholds.

$$\text{True Positive Rate (Sensitivity)} = \frac{TP}{TP + FN} \tag{7}$$

$$\text{False Positive Rate (1 - Specificity)} = \frac{FP}{FP + TN} \tag{8}$$

Precision-Recall (PR) Curve: The PR curve is created by plotting precision against recall for different classification thresholds.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{9}$$

$$\text{Recall (Sensitivity)} = \frac{TP}{TP + FN} \tag{10}$$

D INSTRUMENTATION DETAILS

pandas: Pandas is a powerful library for data manipulation and analysis. It provides data structures like DataFrame and Series that allow easy handling of structured data. The functions from pandas allow us to load datasets, perform data cleaning, filtering, and transformation, and manage data effectively.

numpy: NumPy is a fundamental package for scientific computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with a wide range of mathematical functions to operate on these arrays. NumPy functions are essential for performing numerical operations on data efficiently.

matplotlib.pyplot: This is a plotting library used for creating visualizations in Python. It is a part of the Matplotlib library and provides a simple interface for creating various types of plots, such as line plots, bar plots, scatter plots, and histograms. Matplotlib is widely used for data visualization tasks.

seaborn: Seaborn is a data visualization library based on Matplotlib. It provides a higher-level interface for creating attractive and informative statistical graphics. Seaborn functions are particularly useful for visualizing data distributions, relationships, and patterns in datasets.

wordcloud: The WordCloud library is used for generating word clouds, which visually represent the most frequent words in a text corpus. Word clouds help to get a quick overview of the most prominent terms in the dataset.

sklearn.metrics.accuracy_score: This function from scikit-learn is used to calculate the accuracy of a classification model's predictions. It compares the predicted labels to the true labels and computes the percentage of correct predictions.

sklearn.metrics.classification_report: The classification report function in scikit-learn generates a comprehensive report of various classification metrics for each class in a multi-class classification problem. It provides metrics like precision, recall, F1-score, and support for each class, along with macro and weighted averages.

sklearn.metrics.precision_recall_curve: This function computes precision-recall pairs for different probability thresholds in binary classification. It is used to plot the precision-recall curve, which visualizes the trade-off between precision and recall at different decision thresholds.

scipy.sparse.hstack: This function from the SciPy library is used to horizontally stack sparse matrices. It is commonly used when combining multiple feature matrices before feeding them into a machine learning model.

sklearn.feature_extraction.text.TfidfVectorizer: The TfidfVectorizer from scikit-learn is used to transform text data into numerical vectors using the TF-IDF (Term Frequency-Inverse Document Frequency) representation. It assigns importance to each word based on its frequency in the review and the inverse document frequency in the entire dataset.

sklearn.model_selection.train_test_split: This function is used to split the dataset into training and testing sets for model validation. It allows us to evaluate the model's performance on unseen data.

scikitplot: Scikit-plot is a wrapper library around scikit-learn and matplotlib that provides simple functions for plotting various visualizations used in machine learning, such as confusion matrices, ROC curves, and precision-recall curves.

string: The string module provides several constants and functions for working with strings in Python. It is used to remove punctuation and special characters from the text during preprocessing.

sklearn.metrics.roccurve: The rocCurve function in scikit-learn calculates the Receiver Operating Characteristic (ROC) curve, which plots the true positive rate (sensitivity) against the false positive rate (1-specificity) at different classification thresholds.

sklearn.preprocessing.label_binarize: This function is used to convert multiclass labels into binary format (one-hot encoding) for use in ROC curve calculations.

sklearn.multiclass.OneVsRestClassifier: The OneVsRestClassifier is a strategy used to extend binary classification algorithms to multi-class problems. It fits multiple binary classifiers, one for each class, and combines their results to make multi-class predictions.

E SYSTEM BLOCK DIAGRAM

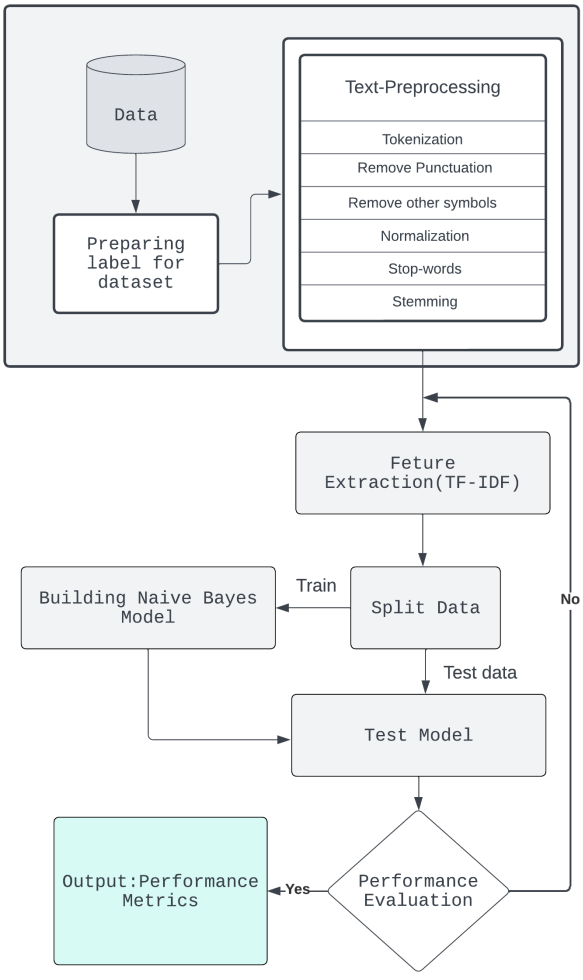

Figure 1: Block Diagram for Naïve Bayes Classifier

IV EXPERIMENTAL RESULTS

Sure, here are the experimental results from our sentiment analysis project:

1. Text Data Visualization (Figure 2 & Figure 5): Exploratory data analysis (EDA) was performed on the textual data to gain insights. Four sub-figures were used to display histograms of text length and word count distributions in the reviews, helping to understand the distribution of these features across all reviews. Additionally, a Word Cloud was created to visualize the most frequently occurring words in the dataset, with the size of each word corresponding to its frequency in the reviews, providing a quick overview of the salient features present in the textual data.

2. Accuracy Curve (Figure 3): An accuracy curve was plotted to study how the model's performance changes with varying training data sizes. The classifier was progressively trained on subsets of the training data, and its accuracy on the testing set was measured. The accuracy curve illustrates the trade-off between model performance and the quantity of training data available, enabling identification of the optimal training data size for achieving better sentiment prediction results.

3. Confusion Matrix (Figure 4): The confusion matrix reveals how well the classifier performs for each sentiment class. Specifically, it displays the number of correctly and incorrectly classified instances for 'Happy', 'Unhappy', and 'Ok' sentiment categories. In this analysis, it is observed that 'Happy' and 'Unhappy' were classified better than 'Ok', suggesting possible class-specific issues that need attention.

4. ROC Curve (Figure 6): The Receiver Operating Characteristic (ROC) curve is used to evaluate the binary classifier's performance. In our case, it is considered 'Happy' and 'Unhappy' as positive classes and 'Ok' as the negative class. The ROC curve shows the true positive rate (sensitivity) against the false positive rate (1-specificity) at different classification thresholds. The AUC values of 0.89, 0.8, and 0.92 for 'Happy', 'Ok', and 'Unhappy', respectively, indicates good discriminative ability of the classifier.

5. PR Curve (Figure 7): The Precision-Recall (PR) curve offers a comprehensive view of the classifier's performance for each sentiment class. The "one-vs-rest" approach is employed to handle the multiclass classification task. The PR curve shows the precision-recall trade-offs for 'Happy', 'Unhappy', and 'Ok'. 'Happy' exhibits the best curve, followed by 'Unhappy', while 'Ok' forms an inverted curve, indicating challenges in its classification.

Our proposed methodology combines Hybrid Naive Bayes with word-level and character-level TF-IDF representations to achieve accurate sentiment analysis. The Naive Bayes classifier, particularly MultinomialNB, is selected for its effectiveness in text classification tasks. The dataset is divided into training and testing sets, to validate the model's performance and optimize its accuracy with the help of various visualizations and analysis techniques. The hybrid approach and thorough evaluation empower businesses to gain valuable insights from customer feedback and make informed decisions to improve customer satisfaction.

V DISCUSSION AND ANALYSIS

In addition to providing visualizations, the sample datasets and a detailed evaluation of the model's performance are presented in tables. The classification report includes essential metrics such as precision, recall, F1-score, and support for each class, along with macro and weighted averages. This thorough evaluation ensures accurate sentiment prediction and empowers data-driven decision-making for businesses.

The experimental results demonstrate the effectiveness of the hybrid Naive Bayes sentiment analysis method, utilizing word- and character-level TF-IDF representations. Visualizations, such as the accuracy curve, Word Cloud, and text data distribution, offer valuable insights into the dataset. The classifier performs well in correctly identifying "Happy" and "Unhappy" sentiments but faces challenges in classifying "Ok" sentiments accurately. Addressing class imbalance and optimizing the classifier could lead to improved outcomes.

The ROC and PR curves provide a comprehensive assessment of the classifier's performance, showcasing the precision-recall trade-offs and discriminative power of each sentiment class. Future enhancements could focus on handling unbalanced data, conducting aspect-based analysis, supporting sentiment analysis in multiple languages, and integrating contextual data. Overall, the proposed methodology presents a robust framework for sentiment analysis, with significant implications for businesses seeking to enhance customer satisfaction and make informed decisions based on customer feedback.

VI CONCLUSION

This study used Hybrid Naive Bayes with word- and character-level TF-IDF representations to do sentiment analysis on product reviews. After gaining an understanding of the dataset through exploratory data analysis and visualization, the TF-IDF is used to convert the text data into numerical vectors. Accuracy, ROC, and PR curves were used to evaluate the Naive Bayes classifier's performance after it had been trained and tested on the dataset. While 'Happy' and 'Unhappy' emotions exhibited promising accuracy and discriminative capacity, correctly categorizing 'Ok' feelings proved challenging, possibly because of a class imbalance. In the future, the model's performance across all sentiment classes might be improved by resolving class imbalance and improving it, which would make it a useful tool for companies looking to understand the sentiment of their customers.

REFERENCES

[1] P. Frasconi, G. Soda, and A. Vullo, "Text categorization for multi-page documents: A hybrid naive bayes hmm approach," in *Proceedings of the 1st ACM/IEEE-CS joint conference on Digital libraries*, 2001, pp. 11–20.

[2] A. M. Kibriya, E. Frank, B. Pfahringer, and G. Holmes, "Multinomial naive bayes for text categorization revisited," in *AI 2004: Advances in Artificial Intelligence:*

17th Australian Joint Conference on Artificial Intelligence, Cairns, Australia, December 4-6, 2004. Proceedings 17. Springer, 2005, pp. 488–499.

[3] S. Qaiser and R. Ali, "Text mining: use of tf-idf to examine the relevance of words to documents," *International Journal of Computer Applications*, vol. 181, no. 1, pp. 25–29, 2018.

[4] S. L. Ramdhani, R. Andreswari, and M. A. Hasibuan, "Sentiment analysis of product reviews using naive bayes algorithm: A case study," in *2018 2nd East Indonesia Conference on Computer and Information Technology (EIConCIT)*. IEEE, 2018, pp. 123–127.

[5] Shreyaswankhede, "sentiment-analysis-on-online-product-reviews." [Online]. Available: https://github.com/shreyaswankhede/Sentiment-Analysis-on-Online-Product-Reviews

[6] H. Zhang and D. Li, "Naïve bayes text classifier," in *2007 IEEE International Conference on Granular Computing (GRC 2007)*. IEEE, 2007, pp. 708–708.

Rijan Ghimire is currently pursuing his undergraduate degree in Electronics, Communication, and Information at IOE, Thapathali Campus. His research interests encompass various areas, including data mining, network communications, and optimization theory and technology.(THA076BEI022)

Sujan Bhattrai is currently pursuing his undergraduate degree in Electronics, Communication, and Information at IOE, Thapathali Campus. His research interests cover various areas, including data mining, deep learning, operating system, robotics, and power electronics.(THA076BEI037)