UNIVERSITY OF HERTFORDSHIRE

School of Physics, Engineering and Computer Science

M.Sc. Artificial Intelligence and Robotics

Final Project Report

7COM1039-0509-2023

Advanced Computer Science Masters Project

September 2, 2024

Application of Sentiment Analytical System to enhance patient satisfaction with drugs using machine learning

Name: Nneoma Nnorom

Student ID: 21073212

Supervisor: Dr. Kelechi Emerole

DEDICATION

I dedicate this project to Almighty God for his grace and love, to my beloved late dad who always encouraged me to be the best version of myself, to Andrew-Vitalis's family, as well as to late Sir Chris Nnorom's family.

ACKNOWLEDGEMENTS

My special appreciation goes to my supervisor, Dr. Kelechi Emerole for his unwavering support and guidance throughout the period of this research.

I give thanks to the Head of the Computer Science Department/Lecturers, Prof Gani Nashi, Dr Christopher Roadknight, Dr. Manal Helal, and other lecturers.

My heartfelt gratitude goes to my dearest husband, son, Mum, siblings, cousins, uncles and aunties for their moral, physical, and emotional support throughout the duration of my study.

Finally, I return all glory to God for His Grace and mercies upon my life.

MSC FINAL PROJECT DECLARATION

This report is submitted in partial fulfilment of the requirement for the degree of Master of Science in Artificial Intelligence and Robotics at the University of Hertfordshire (UH).

It is my own work except where indicated in the report.

I did not use human participants in my MSc Project.

I hereby give permission for the report to be made available on the university website provided the source is acknowledged.

TABLE OF CONTENTS

DEDICATION i
ACKNOWLEDGEMENTS ii
MSC FINAL PROJECT DECLARATION iii
LIST OF FIGURES vii
LIST OF TABLES viii
ABSTRACT: 1
CHAPTER ONE 2
1.1 INTRODUCTION: 2
1.2 BACKGROUND OF THE STUDY 2
1.3 PROBLEM STATEMENT: 3
1.4 ECONOMIC IMPACT 3
1.5 RESEARCH QUESTIONS: 4
1.6 LEGAL, SOCIAL, ETHICAL AND PROFESSIONAL ISSUES: 4
1.7 PROJECT SPECIFICATION: 4
1.8 AIM OF THE PROJECT: 5
1.9 OBJECTIVES OF THE PROJECT: 5
1.10 STRUCTURE OF DISSERTATIONS: 5
CHAPTER TWO 6
LITERATURE REVIEW 6
2.1 BREADTH OF BACKGROUND RESEARCH 6
2.1.1 Machine Learning and Artificial Intelligence 6
2.1.2 Supervised Learning 6
2.1.3 Natural Language Processing (NLP) 9
2.1.4 Patient Satisfaction with medication: 10
2.2 REVIEW OF RELATED WORKS: 11
2.3 RESEARCH GAP: 13
CHAPTER THREE 14
MATERIALS AND METHOD 14
3.1 TOOLS 14
3.1.2 PYTHON 14
3.1.3 Visual Studio Code (VS Code) 14
3.1.4 Pandas 14
3.1.5 Numpy 15
3.1.6 Spacy 15
3.1.7 Matplotlib: 15
3.1.8 Scikit-learn (sklearn): 15
3.1.9 Textblob: 16

3.1.10 Google Colab: ... 16
3.2 METHOD JUSTIFICATION ... 16
3.2.1 Methodology adopted for the project. ... 16
3.3 METHODOLOGY: ... 17
3.3.1 Stage 1: Business Understanding ... 19
3.3.2 Stage 2: Data Understanding ... 19
3.3.3 Stage 3: Data Preparation ... 20
3.3.4 Stage 4: Model Training ... 22
3.3.5 Stage 5: Evaluation ... 23
3.3.6 Stage 6: Deployment ... 25
CHAPTER 4 ... 26
RESULTS AND DISCUSSION ... 26
4.1 DATA PREPARED: ... 26
4.2 SENTIMENT ANALYSIS WITH TEXTBLOB: ... 26
4.3 VISUALIZATION AND EXPLORATORY DATA ANALYSIS (EDA): ... 27
4.4 CHECKED FOR IMBALANCED CLASS: ... 29
4.5 COMPARE THE MODELS USING THE METRICS: ... 31
4.4.1 Baseline Model: ... 31
4.5 HYPERPARAMETER TUNING ON MULTINOMIAL NAÏVE BAYES: ... 37
CHAPTER FIVE: ... 43
CONCLUSION AND RECOMMENDATION ... 43
5.1 CONCLUSION: ... 43
5.2 RECOMMENDATION: ... 43
5.2.1 Further Work ... 44
REFERENCES ... 45
APPENDIX ... 52
Appendix 1: Evidence of request for guarantees on Dataset. Link ... 52
Appendix 2: Import of libraries and reading of the train and test datasets. ... 53
Appendix 3: Sample of patient's reviews on drugs with the seven (7) features. ... 54
Appendix 4a: Description of drugsComTrain_raw.csv dataset ... 54
Appendix 4b: Description of drugsComTest_raw.csv dataset ... 55
Appendix 5: Merging of the train and test set ... 56
Appendix 6a: Description of the merged datasets ... 56
Appendix 6b: Description of the merged datasets ... 57
Appendix 7a: Graph showing the distribution of sentiment. ... 57
Appendix 7b: Graph showing sentiment categories continues. ... 58
Appendix 7c: Top 10 words in positive and negative feedback ... 58
Appendix 8a1: Code for preprocessing the data. ... 59

Appendix 8a2: Code for checking the pre-processed results on the data. 60
Appendix 8b: Results of the pre-processed function on the data. 61
Appendix 9a: Code to perform feature engineering using TF-IDF Vectorizer..... 61
Appendix 9b: Result of TF-IDF Vectorizer..... 62
Appendix 10: Result of the over-sampling on the trained set using SMOTE. 62
Appendix 11a: Multinomial Naïve Bayes Code from scratch..... 63
Appendix 11b: Multinomial Naïve Bayes Code from scratch continues. 63
Appendix 11c: Random Forest Classifier Code from scratch. 64
Appendix 11d: Random Forest Classifier Code from scratch continues..... 64
Appendix 11e: Random Forest Classifier Code from scratch continues. 65
Appendix 12: Application of hyperparameter tuning to the best model (Multinomial Naïve Bayes)..... 66
Appendix 13: Code for Evaluation..... 67
Appendix 14a: Code for Evaluation and results contd..... 67
Appendix 14b: Code for Evaluation and results(contd)..... 68
Appendix 14c: Code for Evaluation and results (contd). 68
Appendix 14d: Code for Evaluation and results contd..... 69
Appendix 14e: Evaluation and results contd. 69
Appendix 15a: Code to save the model as pkl file. 70
Appendix 15b: Code to save and load model as pkl file (contd)..... 71
Appendix 15c: HTML and CSS code to design the front end. 71
Appendix 15d: CSS code to design the front end 72
Appendix 15e: CSS code to design the result front end 73
Appendix 15f: CSS code to design the result front end continues 74

LIST OF FIGURES

CHAPTER TWO

Figure 2.1: Supervised Machine Learning Model ……7

Figure 2.2: Sentiment Analysis Technique ……..10

CHAPTER THREE

Figure 3.1: CRISP-DM Methodology ……17

Figure 3.2: Flowchart algorithm of the Sentiment Analytical System ……....18

Figure 3.3: The shape of each of the dataset ……...20

Figure 3.4: The null values in the dataset ……21

Figure 3.5: The dropping of null values ……...21

Figure 3.6: Code to balance the class ……..22

Figure 3.7: ROC-AUC metric for prediction evaluation ……25

CHAPTER FOUR

Figure 4.1: Pre-processed Dataset ……..26

Figure 4.2: Sentiment categorized dataset ……..27

Figure 4.3: Distribution of sentiment scores(polarity) ……...27

Figure 4.4: Positive Feedback Word Cloud ……...28

Figure 4.5: Negative Feedback Word Cloud ……..29

Figure 4.6 Distribution of imbalance Textual Data ……30

Figure 4.7 Balance Sentiments (Label) Distribution using SMOTE ……31

Figure 4.8 Quantitative assessment metrics for the baseline model RFC ……32

Figure 4.9: Comparative analysis of the classifiers across different metrics ……34

Figure 4.10: Comparative Analysis of the classifiers across different metrics with a larger set....35

Figure 4.11 Quantitative assessment metrics for the proposed model (MNB) ……36

Figure 4.12: Bar Chart showing the metrics of the hyperparameter tuning on Multinomial Naïve Bayes ……38

Figure 4.13: Line Chart showing the various metrics result of hyperparameters tuning on Multinomial Naïve Bayes (Best Model) ……...38

Figure 4.14 shows the chart of the comparison of the different existing model with the proposed model ……....40

Figure 4.15 first page of the user interface ……..41

Figure 4.16 shows the written review yet to be analysed ……....41

Figure 4.17 shows the result of the analysed review ……...42

LIST OF TABLES

Table 4.1: Metrics result of the baseline model (RFC) ……..31
Table 4.2: Confusion Matrix using 10,000 Samples of dataset …….32
Table 4.3: Metrics results of the two ML models for comparison …….33
Table 4.4: Confusion Matrix for the two models using 20,000 Samples of dataset ……..34
Table 4.5: Metrics results of the two ML models for comparison using 20,000 ……...34
Table 4.6: Metrics results of the Multinomial Naïve Bayes using the complete dataset ……...36
Table 4.7: Hyperparameter tuning of Multinomial Naïve Bayes Model ……...37
Table 4.8: shows the performance comparison of the proposed model with the other existing models……39

ABSTRACT:

The integration of artificial intelligence (AI) technologies into healthcare systems has shown promising potential to enhance patient satisfaction, a crucial metric for assessing healthcare medical services in recent years. This research develops a Multinomial Naïve Bayes and Random Forest Classifier Machine Learning models from scratch to examine their effectiveness in enhancing patient satisfaction with drugs, specifically by comparing them. The aim of this study is to assess how developing a Sentiment Analytical System using machine learning models can accurately analyse and interpret patient feedback to derive sentiment insights to optimize healthcare delivery processes. By conducting a comparative analysis between these models, the research aims to identify their individual strengths and weaknesses in capturing and analysing patient sentiments. Through a comprehensive review of literature and theoretical frameworks, this research lays a foundation for understanding the importance of patient satisfaction in the healthcare service environment and the role of AI in augmenting this aspect. Following this, an empirical inquiry is carried out using actual patient feedback on medications (drugs) datasets to evaluate the effectiveness of Sentiment Analysis and its methodologies in extracting valuable insights from textual data. Understanding and interpreting patient's reviews of drugs reveals that some drugs prescribed for patients react negatively to them thereby making patients not be satisfied with the treatment administered to them. Leveraging sentiment analysis on patient reviews on drugs has increased the researcher's attention in recent years thereby making them to explore methodologies and approaches to interpret, understand and further enhance patient satisfaction with drugs. However, existing machine learning models and algorithms fall short of accurately classifying these patient reviews, as the algorithms are too complex to perform, and have low performance when applied to patient reviews. For example, using Random Forest Classifier, a machine learning classifier that is robust to noise, reduces overfitting compared to a decision tree model. The gap with this model is that it may still overfit if not properly tuned and it can be computationally intensive with many trees. Another example is using Decision trees: This is another type of supervised machine learning model that is easy to interpret, visualize and requires little data preprocessing. The gap with this machine learning model is that it is prone to overfitting especially with deep trees and it is sensitive to noisy data. Furthermore, using Multinomial Naive Bayes which applies probability in making its assumptions is fast especially when trained on large datasets. Its gap is that it is sensitive to zero probability. Considering all these limitations and problems, the study will involve developing from scratch two machine learning algorithms and comparing them while leveraging on a feature extraction technique to transform the data and an oversampling technique to balance the imbalanced dataset. This is to address the issue of models being too complex to handle large datasets, becoming biased during prediction because of a class having most samples, and also to reduce and filter out noise which reduces the quality of data being fed in. This study results provide valuable insights into NLP's capabilities in improving patient satisfaction with medications in healthcare settings. By contrasting these Sentiment Analysis methodologies, the research clarifies the intricacies of AI-driven methodologies in interpreting patient sentiments and provides recommendations for optimizing the integration of such technologies into healthcare systems. Ultimately, this research aims to inform healthcare practitioners, policymakers, and AI experts about the potential of AI in enhancing patient satisfaction with medications (drugs) and fostering more patient-centric healthcare delivery models. The outcomes of this study have implications for the advancement of AI-driven solutions in healthcare and underscore the importance of leveraging innovative technologies to meet patients' evolving needs and expectations.

Keywords: Machine Learning, NLP, Supervised Learning, MNB, RFC, Patient Satisfaction, TF-IDF, SMOTE.

CHAPTER ONE

1.1 INTRODUCTION:

In the contemporary landscape of healthcare, patient satisfaction is paramount, driving efforts to optimize therapeutic outcomes and enhance the overall patient experience. Patient satisfaction is a multifaceted concept reflecting the extent to which patients are content with the healthcare services they receive, encompassing various dimensions such as quality of care, communication with healthcare providers, accessibility of services, and the overall healthcare environment.

In recent years, there has been a need for a more sufficient way of analysing patient feedback because the amount of patient reviews data increases as time unfolds. Sentiment analysis, a subfield of natural language processing (NLP), has emerged as a potent tool for understanding patient perspectives by analysing textual data such as reviews, feedback, and social media posts.

1.2 BACKGROUND OF THE STUDY

NLP, a subfield of artificial intelligence (AI), focuses basically on computers and human language interaction. It enables machines to understand, interpret, and generate human language in a way that is both meaningful and useful. Key NLP techniques used in sentiment analysis include text preprocessing, tokenization, part-of-speech tagging, named entity recognition, and syntactic parsing (Jurafsky & Martin, 2020). Text preprocessing involves cleaning the data by removing noise such as punctuation, stop words, and irrelevant information. Tokenization splits text into individual words or phrases, while part-of-speech tagging identifies the grammatical role of each token. Named entity recognition helps in identifying and classifying entities like drug names, symptoms, and patient demographics. Syntactic parsing analyses the grammatical structure of sentences to understand the relationships between words.

Machine learning (ML) has revolutionized sentiment analysis by providing algorithms that learn and predict from data. The two machine learning classifiers used in this research are Multinomial Naive Bayes (MNB) and Random Forest (RF).

Multinomial Naive Bayes classifier is a probabilistic model that is particularly well-suited to text classification tasks, including sentiment analysis. This algorithm is based on Bayes' theorem, which applies the principles of conditional probability to predict the likelihood of a given sentiment category (e.g., positive, negative, neutral) based on the presence of certain words or features in the text (Qi & Shabrina, 2023). The "multinomial" aspect of the model indicates that it is used to model discrete features, such as word counts or frequencies, which are common in text data. The Naive Bayes classifier assumes independence between features, which simplifies the computation and makes it scalable to large datasets. Despite its simplified assumption, Multinomial Naive Bayes has demonstrated strong performance in various text classification tasks, including sentiment analysis, due to its efficiency and effectiveness in handling high-dimensional data (Gan et al., 2021).

On the other hand, the Random Forest classifier is an ensemble learning model which during training constructs multiple decision trees and output classes with the highest mode (Maturo & Porreca, 2024). Each tree in the forest is built on a subset of the data and a subset of features, introducing randomness that helps to reduce overfitting and improve generalization. In sentiment analysis, Random Forest classifiers can handle complex interactions between features and are robust to noise in the data. One of the key strengths of Random Forests is being able to extract non-linear relationships and interactions between words in the text, providing a more nuanced understanding of sentiment compared to simpler models. Additionally, Random Forests offer feature importance scores, which can help identify which words or phrases are most indicative of patient sentiment (Iranzad & Liu (2024).

The integration of sentiment analysis into healthcare systems is revolutionizing patient satisfaction with pharmaceutical treatments by leveraging machine learning techniques, specifically Multinomial Naive Bayes (MNB) and Random Forest (RF) classifiers. Pharmaceutical companies can also use sentiment analysis to gather feedback on existing drugs, driving improvements in drug formulations, and gaining a competitive advantage (Garg & Rana, 2022). Despite its promise, implementing sentiment analysis in healthcare involves challenges such as ensuring data privacy, interpreting contextual nuances, and mitigating algorithmic biases (Wang et al., 2020). Addressing these challenges is crucial to realizing the full potential of sentiment analysis in enhancing patient satisfaction and optimizing drug-related experiences in healthcare settings (Zunic et al., 2020). Overall, the integration of sentiment analysis through Multinomial Naive Bayes and Random Forest classifiers represents a transformative approach to improving patient satisfaction, offering a powerful tool for understanding and responding to patient feedback in the pharmaceutical domain.

1.3 PROBLEM STATEMENT:

Understanding and interpreting patient's reviews on drugs reveals that some drugs prescribed for patients react negatively to them thereby making patients not to be satisfied with the treatment administered to them. Leveraging sentiment analysis on patient reviews on drugs has increased the researcher's attention in recent years thereby making them to explore methodologies and approaches to interpret, understand and further enhance patient satisfaction on drugs. However, existing machine learning models and algorithms fall short of accurately classifying these patient reviews, as the algorithms are too complex to perform, and have low performance when applied to patient reviews.

For example, using Random Forest Classifier, a machine learning classifier that is robust to noise, reduces overfitting compared to decision trees model. The gap with this library is that it may still overfit if not properly tuned and it can be computationally intensive with many trees.

Another example is using Decision trees: This is another type of supervised machine learning model that is easy to interpret, visualize and requires little data preprocessing. The gap with this machine learning model is that it is prone to overfitting especially with deep trees and it is sensitive to noisy data.

Furthermore, using Multinomial Naive Bayes which applies probability in making its assumptions is fast especially when trained on large datasets. Its gap is that it is sensitive to zero probability.

Considering all these limitations, there is a need to develop a more accurate way of classifying patients' reviews on drugs.

1.4 ECONOMIC IMPACT

Implementing sentiment analytical systems in healthcare, particularly to enhance patient satisfaction with pharmaceuticals has substantial economic implications. Machine learning techniques such as Multinomial Naive Bayes and Random Forest classifiers can revolutionize how pharmaceutical companies, healthcare providers, and insurance companies interact with patients and manage drug-related data, potentially leading to significant cost savings and economic benefits. Its economic benefit in healthcare is the optimization of drug development and marketing strategies. Pharmaceutical companies can utilize patient sentiment data to identify unmet needs, monitor drug performance, and understand market trends, enabling them to refine their research and development (R&D) efforts. This data-driven approach can reduce the costs associated with developing new drugs by focusing resources on areas with the highest potential for success and patient satisfaction (Bai et al., 2020). In addition to benefiting pharmaceutical companies, sentiment analysis can enhance the efficiency of healthcare delivery systems, leading to cost reductions for healthcare providers and insurers. Through analysis of patient feedback, healthcare providers can identify areas that require

improvement in patient care and treatment protocols, thereby providing an efficient way of allocating resources. (McIntosh et al., 2024). This does not only aid in improving the patient satisfaction, rather it reduces the need for hospital readmissions and prolonged treatments. Another significant economic impact of sentiment analysis is its potential to reduce adverse drug reactions (ADRs) and associated costs. Adverse drug reactions pose a considerable economic burden on healthcare systems, with billions of dollars spent annually on managing these incidents (Kumar et al., 2020). By leveraging machine learning classifiers to analyse patient sentiments, healthcare providers can detect early signs of ADRs and adjust treatment plans, accordingly, reducing the incidence and severity of these reactions (Khaleghparast et al., 2023).

Specifically, the new system will help the health care sectors to discover areas that need improvement, and this will further help in enhancing patient satisfaction.

1.5 RESEARCH QUESTIONS:

The research questions include:

1. How do the model performance metrics (e.g., precision, recall, F1-score) differ between the two machine learning models when applied to patient drug satisfaction analysis?
2. How do Multinomial Naive Bayes and Random Forest Classifier compare in terms of identifying sentiments in patient drug reviews?
3. What impact does hyperparameter tuning have on the best model's performance?
4. How does the new system model performance matrices (accuracy) differ from other researchers' models?

1.6 LEGAL, SOCIAL, ETHICAL AND PROFESSIONAL ISSUES:

Having searched and selected a dataset that was relevant and ethically sourced. The sequential procedures involve searching open sources such as Kaggle for datasets relating to patient reviews and drug satisfaction, reviewing dataset descriptions, and ensuring they are ethically sourced with patient consent, downloading the dataset, and examining the data fields for an understanding of its structure and content. It is licensed under a Creative Commons Attribution 4.0 International (CC BY 4.0) License. Which permits the use of the dataset once proper reference has been given. Thus, there will not be a need for ethical approval since the study will not involve human participants. During the research, it was discovered that the dataset license was not on Kaggle. The researcher sent an email to the dataset owner requesting a guarantee that the dataset is compliant with the university's research protocols. The owner on Kaggle sent the researcher the link to where the license is (https://archive.ics.uci.edu/dataset/462/drug+review+dataset+drugs+com). The researcher has also attached a screenshot of the license details to Appendix 1. Therefore, no ethical clearance will be required from the ECDA (Ethics Committee with Delegated Powers), that has the same powers as the Ethics Committee at the University of Hertfordshire.

1.7 PROJECT SPECIFICATION:

This project focuses on applying sentiment analytical system in enhancing patient satisfaction on drugs which will classify patients reviews on drugs. It will develop two machine learning models from scratch leveraging on Feature Engineering (TF-IDF) to extract the documents and use of SMOTE, an oversampling tool to ensure that the class is balanced to bridge the gap of overfitting.

It will also compare the two machine learning models (Multinomial Naïve Bayes and Random Forest Classifier) using the evaluation metrics (Accuracy, precision, etc.). This project produces a framework that accepts patient's reviews about drugs, it classifies and returns the sentiment of the review.

1.8 AIM OF THE PROJECT:

This study aims at developing and applying the Sentiment analytical system to enhance patient satisfaction on drugs using machine learning models.

1.9 OBJECTIVES OF THE PROJECT:

The specific objectives of the project are.

1. To design and implement an algorithm that uses Spacy Python libraries to preprocess data and understand the data, for obtaining accurate analysable results.
2. To design and implement an algorithm that uses sentiment analysis on patient reviews on drugs to categorize the review into positive, negative, and neutral using the TextBlob python library.
3. To design an algorithm that trains the Multinomial Naïve Bayes classifier and Random Forest Classifier from scratch and compares them based on their metrics.
4. To perform hyperparameter tuning, investigate and evaluate the metrics, and performance of the best model.
5. To compare the best model's performance with other researchers' model.
6. To design an interactive User Interface to enable interaction with the model using Python Flask for the Backend and using HTML and CSS for the Frontend.

1.10 STRUCTURE OF DISSERTATIONS:

The overview of the dissertation is as follows:

Chapter One: This section is focused on introduction, overview, economic impact, ethical consideration, and dissertation structure making clear overview of the project.

Chapter Two: This section majorly is on the breadth of this research, review of related work and research gap.

Chapter Three: This is a material and methodology chapter that describes the technique applied for the project implementation and development.

Chapter Four: The chapter is configured for fact finding, comparing methods and results, system implementation and analysis.

Chapter Five: Summary, Conclusion, and recommendation.

CHAPTER TWO

LITERATURE REVIEW

The review of existing literature plays a role in laying the groundwork for this research project and helping to identify areas for future exploration. The discussion examines how Spacy advanced preprocessing features could potentially improve the performance and accuracy of sentiment analysis algorithms by enabling text parsing and feature extraction (Palomino & Aider, 2022). Furthermore, it compares the effectiveness of machine learning models in categorizing sentiments expressed in reviews of drugs focusing on crucial metrics like precision, recall and F1 score that are crucial for evaluating the performance of the model (Liu et al., 2021). The review also touches upon the significance of hyperparameter tuning in enhancing model efficiency emphasizing its role in boosting accuracy while simplifying model complexity (Bischl et al., 2021).

2.1 BREADTH OF BACKGROUND RESEARCH

The breadth of background research for applying sentiment analytical systems to patient satisfaction on drugs using machine learning models involves several disciplines and important concepts, which include Machine Learning Models, Supervised Learning, Natural Language Processing (NLP), Sentiment Analysis Techniques, Multinomial Naive Bayes, Random Forest Classifier, Patient Feedback and Satisfaction Metrics, SMOTE, TF-IDF.

2.1.1 Machine Learning and Artificial Intelligence

Machine Learning is a branch of Artificial Intelligence that trains computers to learn from data. Its primary objective is to enable systems to make decisions based on data patterns learned (Sarker, 2021). Various types of machine learning algorithms exist based on their learning methodologies, including supervised, semi-supervised and reinforcement learning which provide unique insights from patient feedback on medications.

Machine learning models have made progress in sentiment analysis in the healthcare sector. Machine learning models have revolutionized sentiment analysis by enabling the extraction of insights from volumes of textual data. Ongoing research and development efforts are driving enhancements in sentiment analysis accuracy and efficiency offering insights for healthcare providers, researchers, and pharmaceutical companies. In drug discovery and healthcare these models are always used to address these challenges. They show precision in predicting bioactivities, molecular properties, and drug classifications (Purwono et al., 2021). By analysing datasets from high throughput screening and medical records these models have the potential to streamline the drug discovery process and cut down clinical trial expenses significantly (Balzerowski et al., 2024).

2.1.2 Supervised Learning

Supervised Machine Learning (SML) is a branch of machine learning that utilizes labelled data to train models, for predicting outcomes from data. It involves linking input and output pairs to create a function. The procedure encompasses data gathering, feature extraction, model training and assessment. Its application spans domains like image recognition, language processing and fraud detection. The following are common algorithms in supervised learning including decision trees, random forests, naive Bayes, support vector machines and neural networks. Developing supervised learning models encompasses theoretical and practical dimensions. An essential aspect of learning is the presence of training data through which the model understands the connections between input variables and corresponding output labels. Supervised Machine Learning process comprises training and testing phases where models learn from input features and labels to reduce prediction errors and make forecasts on data.

This allows for the creation of models that can identify patterns and make informed decisions based on new data. According to (Syed & Lokhande, 2024) supervised learning has two categories; classification which predicts labels and regression which predicts continuous values.

Supervised machine learning offers solutions for overcoming obstacles that traditional statistical methods may struggle with (Jiang et al., 2020). However, creating reliable and flexible machine learning algorithms presents challenges that researchers must address (Jiang et al., 2020).

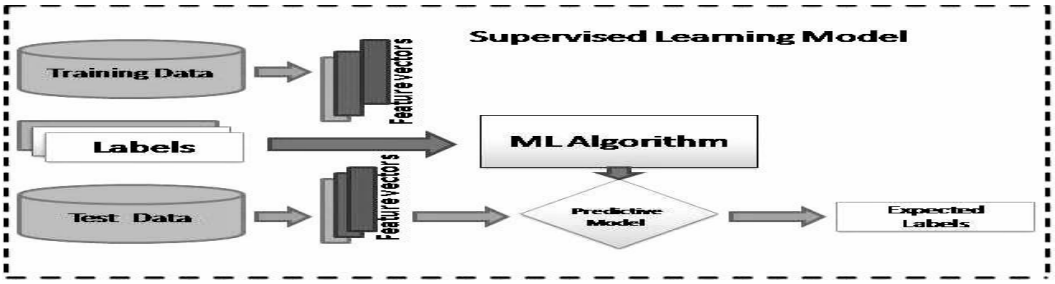


Figure 2.1: Supervised Machine Learning Model

2.1.2.1 The Multinomial Naive Bayes (MNB) Classifier

MNB is a supervised machine learning naive Bayes model that is widely used in text classification. It is often utilized in text classification tasks that involve discrete features like word counts. It is essentially an algorithm that makes predictions about the class of a document by analysing the presence of words in it. The core principle of Naive Bayes is the assumption of conditional independence, which means that given a class, the presence of one feature will not affect the presence of another feature (Liu et al., 2024). Multinomial distributions usually require integer features to learn from. However, in practice, fractional feature counts such as TF-IDF may also be used.

The Multinomial Naive Bayes model determines the likelihood of a document d belonging to class c based on the likelihood of the words in the document appearing in that class. The likelihood is determined by applying the Bayes Theorem as outlined by (Manning et al., 2008):

In this scenario, $P(c|d)$ represents the chance of class c based on document d while $P(d|c)$ shows how likely.

$$P(c|d) = \frac{P(d|c) \cdot P(c)}{P(d)}$$

Where $P(c|d)$is the posterior probability of class c given the document d, $P(d|c)$ is the likelihood of the document d given class c, $P(c)$ is the prior probability of class c and $P(c)$ is the prior probability of the document d, which is constant across classes (Manning et al., 2008).

In the Multinomial Naive Bayes model, the likelihood $P(d|c)$ is expressed as the product of probabilities of individual words w_i given class c:

$$P(d|c) = P(w_1, w_2, \dots, w_n|c) = P(w_1|c) \cdot P(w_2|c) \cdot \dots \cdot P(w_n|c)$$

Where n is the number of unique words in the vocabulary, $P(w_i|c)$ is the probability of word w_1 occurring in class c, and $f(w_i|d)$ is the frequency of word w_1 in document d.

To estimate $P(w_i|c)$, the Multinomial Naive Bayes model uses:

$$P(w_i|c) = \frac{N_{wi,c}+\alpha}{N_c+\alpha \cdot V}$$

Where $N_{wi,c}$ is the number of times word w_i appears in documents of class c, N_c is the total number of word occurrences in class c, V is the total number of unique words in the vocabulary, and α (alpha) is a smoothing parameter, often set to 1 (Laplace smoothing), to handle zero probabilities.

To calculate Prior probability, $P(c) = \frac{N_c}{N}$
Where N_c is the number of documents in class c and N is the total number of documents across all classes.

The classification of a new document is done by computing $P(c|d)$ for each class c and selecting the class that has the highest posterior probability. Since $P(d)$ is constant, it can be ignored in the calculation, simplifying the expression to:

$$\hat{c} = \arg\max_{c \in C} \log P(c) + \sum_{i=1}^{n} f(w_i, d) \log P(w_i|c)$$

Where $\hat{c}$ is the predicted class.

By assuming independence, the model can efficiently manage vocabularies without consuming excessive computational resources making it ideal, for applications requiring real time processing. (Kaur et al., 2021).

(Kaur et al., 2021) introduces Multinomial Naive Bayes as a model for sentiment analysis emphasizing its efficiency in tasks while also suggesting the requirement, for advanced models to grasp intricate text nuances.

2.1.2.2 Random Forest Classifier (RFC)

RFC is a supervised machine learning model that is commonly employed in text classification and regression. It produces multiple decision trees by training each tree using a bootstrapped sample of the training data. Upon reaching each node, the algorithm searches a random subset of the variables to determine the split. In classification, the mode of the classes is output while in regression, the mean prediction of the individual trees is output (Abdulkareem & Abdulazeez, 2021).

The mathematical principle of randomness is employed in feature selection, where a subset of features is randomly chosen at each node to determine the best split, thereby enhancing the model's ability to generalize by introducing diversity among the trees. Random Forest's operation involves several mathematical key steps. Firstly, when working with a dataset containing n samples and m features, we start by drawing N bootstrap samples. For each of these bootstrap samples we construct a decision tree. At every node of the tree the optimal split is calculated from a subset K features (where $k<<m$). The criteria used for finding the split can be either Gini impurity or Information Gain which are mathematical metrics that assess how consistent the target variable is within different subsets.

Gini Impurity is calculated as:

$$\text{Gini} = 1 - \sum_{i=1}^{c} p_i^2$$

where p_i is the probability of a particular class at a node and c is the number of classes.

Information Gain, used in entropy-based methods, is defined as:

$$\text{Gain(S,A)} = \text{Entropy(s)} - \sum_{v \epsilon Values(A)} \frac{|S_v|}{|S|} \times \text{Entropy}(\text{S}_\text{v})$$

where Entropy(S) is the entropy of the dataset S, and S_v is the subset for a particular feature value.

In a Random Forest model every tree offers a prediction, for an input. The overall classification result is decided by combining these predictions using either a voting system for classification purposes or an averaging method, for regression tasks. (Breiman, 2001)

The mathematical model of the Random Forest can be represented as:

$$\text{H}(x) = \frac{1}{T}\sum_{t=1}^{T} h_t(x)$$

where H(x) is the final prediction, T is the total number of trees, and $ht(x)$ is the prediction of the t-th tree.

Random Forest stands out for its ability to combat overfitting effectively. This algorithm is applied to intrusion detection, content filtering, sentiment analysis and even image processing (Abdulkareem & Abdulazeez, 2021).

2.1.3 Natural Language Processing (NLP)

NLP is a subfield of artificial intelligence that focuses on how computers interact with and process languages. NLP includes analysing text data to help computers understand, interpret, and produce human language for various purposes. Its main goal is to empower computers to perform language-related tasks such as understanding text, providing responses, extracting information, and facilitating communication between languages. To achieve these objectives, NLP integrates principles from fields such as statistics, computational science, machine learning and linguistics. Users can extract insights from text datasets, analyse content and access personalized news tailored to their preferences. NLP has made progress in recent years due to advancements in advanced machine learning techniques like deep learning and the availability of extensive datasets. These developments have enabled computers to perform tasks that require the intelligence of a human being.

NLP enables an understanding of data through human language processing. In NLPs we often use structure and different parts of speech, like nouns, verbs, and adjectives. NLP must cope with anaphora and ambiguity coupled with part-of-speech, and grammatical structure.

Key aspects of natural language processing (NLP) consist of categorizing text, recognizing named entities, analysing sentiments, translating languages with machines and understanding spoken words (Dande & Pund, 2023). Remarkable advancements in NLP have opened the doors to a series of applications. These technologies can interpret, create content, and engage in conversations making an impact in fields like virtual assistants (e.g., Alexa, Siri) chatbots, sentiment analysis and language translation (Yadav et al., 2024). NLP is used in the healthcare, finance, education, and customer service sectors (Yadav et al., 2024). Despite NLP's advancements, it encounters challenges like grasping context intricacies and cultural subtleties (Yadav et al., 2024).

2.1.3.1 Sentiment Analysis Techniques:

Sentiment analysis is a part of (NLP) that focuses on automatically identifying emotions in a text and categorizing them as positive, negative, or neutral. It extracts details from text using a mix of linguistics, computer science, NLP, and machine learning. Text is typically categorized as positive, neutral, or negative with a score reflecting the intensity of the sentiment. Applications of sentiment analysis include tracking media trends monitoring customer feedback and analysing reviews.

Machine learning algorithms are crucial in classifying text content based on emotions and opinions using NLP techniques. In business settings sentiment analysis helps companies identify customer perceptions about their products or services to make decisions. However, challenges exist in sentiment analysis such as detecting sarcasm accurately, interpreting sentiments from texts and determining the polarity of sentiments. Despite facing these obstacles, it still serves as an asset for companies to grasp customer feedback and reach conclusions (Nalini et al., 2022).

Sentiment analysis of drug reviews has become significant in the healthcare field offering perspectives on experiences and the efficacy of medications. Several research studies have delved into this area using techniques in natural language processing. (Yadav & Vishwakarma, 2020) introduced a framework for text representation with weights achieving accuracy rates and F1 scores. (Hiremath & Patil, 2020) concentrated on creating a system to support decisions using sentiment analysis based on aspects and machine learning approaches. These research works showcase the effectiveness of methods in evaluating drug reviews holding promise for enhancing healthcare decision making, improving patient care and providing insights, for both drug manufacturers and healthcare professionals (Yadav & Vishwakarma, 2020).

Sentiment Analysis techniques include Sentiment Analysis using Lexical Analysis, Sentiment Analysis using Machine Learning and Sentiment Analysis using Deep Learning.

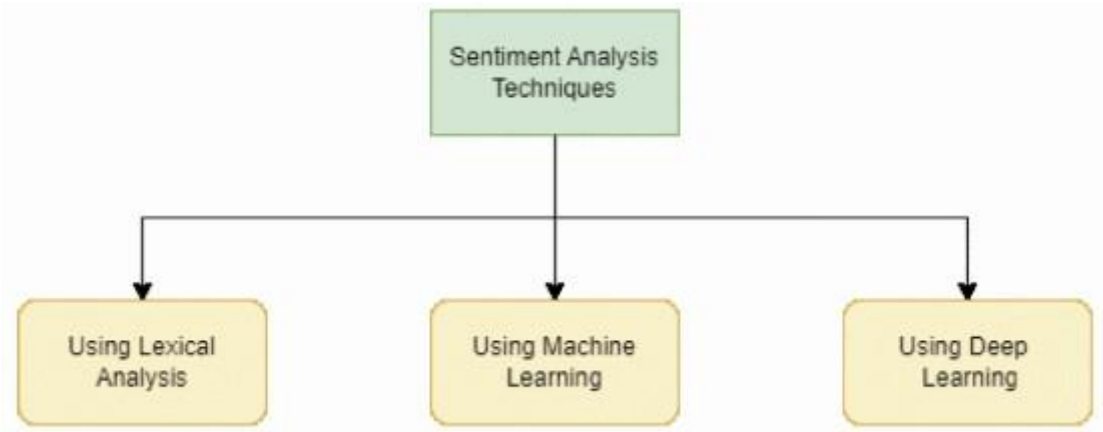


Figure 2.2: Sentiment Analysis Technique [Source: Sindhu, S., Kumar, S. and Noliya, A. (2023c) 'A Review on Sentiment Analysis using Machine Learning,' https://doi.org/10.1109/icidca56705.2023.10099665.]

2.1.4 Patient Satisfaction with medication:

Nowadays, Patient satisfaction is a popular indicator of the quality of health care services. Patient satisfaction with drugs is an important aspect of healthcare that influences treatment adherence and outcomes. Studies have shown that satisfaction is influenced by factors such as effectiveness, side effects, convenience, and cost (Alam et al., 2018). Demographic and social factors, including age, residence area, and health status, can affect various aspects of medication satisfaction (Geitona et al., 2008). Research in this field has primarily focused on developed economies, with a need for more studies in emerging markets (Alam et al., 2018). Understanding patient satisfaction with medication is crucial for improving healthcare quality and patient outcomes.

Patient satisfaction with medication: this is gaining traction as a measure of healthcare quality. Satisfaction can be viewed as an evaluation of performance meeting or exceeding expectations, an assessment, or a fairness evaluation. It's essential for a satisfaction metric to have a foundation to assess its validity. Patient satisfaction with medication plays a role in healthcare by impacting

treatment adherence and results. Research indicates that satisfaction is influenced by factors like effectiveness, side effects, convenience and cost (Shikiar & Rentz, 2004). Demographic and social elements such as age, location of residence and health status can impact aspects of medication satisfaction (Geitona et al., 2008). While existing studies largely focus on developed nations there's a call for research in emerging markets (Alam et al., 2018). Understanding satisfaction with medication is crucial, for enhancing healthcare quality and patient wellbeing.

2.2 REVIEW OF RELATED WORKS:

In a research project, by (Raja et al., 2022) they delved into comparing machine learning models such as XGBoost, Naïve Bayes, and Logistic Regression to determine how well they can distinguish between positive and negative drug reviews. The study revealed that XGBoost achieved an accuracy rate of 79.7%. Their primary aim was to pinpoint the model for categorizing reviews as positive or negative that aids healthcare professionals in gauging public opinions on specific medications. However, there are obstacles and constraints to consider. The intricacies of terminology pose challenges for sentiment analysis as conventional approaches may struggle to capture the subtleties of language. Imbalances in data distribution, where sentiments are unevenly spread can result in inaccuracies such as positives or negatives impacting the dependability of the model. Extensive preprocessing is essential to address stop words and clean up the data, which adds a layer of complexity to the analysis. Furthermore, model performance can vary based on the dataset used; while XGBoost excelled models like Logistic Regression also show potential. Concerns arise regarding generalization since models trained on datasets may not perform adequately with data. Lastly advanced models like XGBoost require resources potentially limiting scalability and real time implementation. These challenges underscore the necessity for enhanced methods and models to effectively manage the complexities of sentiment analysis in drug reviews.

Muhammad et al., 2019) conducted a study comparing the effectiveness of the MNB algorithm and Logistic Regression (LR) algorithm in classifying intent in chatbots, with an objective of determining which method accurately classifies user input into predefined categories like "greet," "report," and "info.". The research involves analysing the dataset with TF IDF (Term Frequency Inverse Document Frequency) to transform text information into attributes, followed by training Naive Bayes and Logistic Regression models. The outcomes show that Logistic Regression surpasses Naive Bayes achieving accuracy of 72.7% compared to 63.6% along with precision and recall. Nevertheless, the study faces obstacles and restrictions such as utilizing an unbalanced dataset that might not generalize well to larger and more varied datasets, as well as the unwanted high chances of overfitting where models may excel on the training data but perform poorly on unseen data. Despite these constraints the study concludes that in classification, Logistic Regression is more suitable. However, it advises caution when interpreting the results.

In addition, (Helae et al., 2022) discussed how data analytics can be applied in the field of pharmacology specifically focusing on analysing the sentiments expressed in reviews of drugs. The main goals of their research include assessing satisfaction with medications, predicting health conditions based on feedback, and pinpointing the most effective drugs for specific ailments. They used a UCI Machine Learning Repository dataset that includes reviews and ratings. Their approach involved preparing the data by transforming features using methods like TF IDF and Bag of Words and employing machine learning techniques such as Random Forest (RF) Support Vector Machine (SVM) and deep learning models like Long Short Term Memory (LSTM). The results show that sentiment analysis with RF and SVM can effectively categorize patient reviews to gauge satisfaction levels. LSTM was utilized to forecast drug ratings and identify performing medications for conditions. However, they encountered challenges dealing with large amounts of data and ensuring precise feature extraction from text-based information. They noted limitations related to training

models is due to the nature of the data, which underscores the importance of refining data processing methods and enhancing model accuracy.

(Sivaiah et al., 2024) conducted a study focusing on a Drug Recommendation System that utilizes sentiment analysis of reviews, through Machine Learning. Their goal was to create a drug recommendation system based on understanding feedback. The research utilizes machine learning methods to examine how patients feel using techniques like TF-IDF, Bag of Words, and Manual Feature Analysis. They applied classification algorithms such as a Linear SVC classifier to predict patient sentiments. The results show that combining the Linear SVC classifier with TF IDF vectorization achieved higher accuracy compared to models tested. The conclusions were drawn from evaluating systems using metrics such as precision, recall, F1 score, accuracy, and AUC score. The key findings highlight the potential of machine learning in improving drug prescription accuracy by customizing recommendations for patient requirements leading to better healthcare outcomes. While the results are promising the research paper acknowledges challenges like integrating algorithms and requiring datasets for effective model training.
Addressing issues related to data privacy and security is crucial along with considering all aspects in automated healthcare decision making processes.

(Garg, 2024) created a Drug Recommendation System that relies on analyzing the sentiments expressed in drug reviews through Machine Learning. The research focuses on crafting a system for suggesting drugs that can ease the workload of healthcare practitioners by evaluating feedback. The study uses machine learning methods and vectorization techniques, like TF-IDF, and Word2Vec, and Bag of Words to convert text reviews into formats for categorization. The research utilizes the Drug Review Dataset from the UCI Machine Learning repository, which contains more than 215,000 cases. The data is prepared by eliminating entries, duplicates, and unnecessary details to guarantee an input for the models. The information is divided into two parts, for training and testing and the Synthetic Minority Over Sampling Technique (SMOTE) is used to tackle imbalances in classes. Various types of classifiers like Linear Support Vector Classifier (LinearSVC), Random Forest, Naive Bayes and more were examined using metrics such as precision, recall, F1 score, accuracy, and AUC. The results show that the linear SVC model, with TF IDF vectorization performed best in terms of accuracy. Although the system shows results that the authors recognize constraints related to the medical language used in reviews, which makes sentiment analysis challenging and calls for advanced linguistic models.

(Bagwan et al., 2021) conducted a study on creating a system for gathering feedback that utilizes sentiment analysis to gain insights into patient experiences with hospital services. They investigated machine learning algorithms like Decision Tree, Multinomial Naive Bayes Classifier, Logistic Regression, K Nearest Neighbors and Random Forest to compare their performance based on accuracy and F score. To improve accuracy, the researchers employed an Ensemble Learning method that combines predictions from base classifiers. Feedback was collected in languages such as Hindi, Marathi and English before being translated into English using Google Translator for analysis. The results show that the Ensemble Classifier surpassed models by achieving accuracy in sentiment classification. However, the study highlighted challenges in dealing with text and accurately translating nuanced feedback. It also recognized the limitations of relying on input as it may not capture all nuances of patient sentiment, like sarcasm or context-specific details.

(Iqbal, Umar & Almohsen 2023) conducted a study on Drug Sentiment Analysis Using Machine Learning Classifiers. Their research focused on utilizing machine learning methods to assess the efficacy of drugs based on user feedback. The main goal of the study was to offer insights that could assist consumers in making informed decisions regarding drug usage. The initial steps involved breaking down the text into components through tokenization and lemmatization to simplify its

complexity. The study evaluated four machine learning algorithms for classification namely, Random Forest, Support Vector Classifier (SVC), Naive Bayes Classifier, and Multilayer Perceptron (MLP). In terms of classification, Linear SVC was utilized. Results indicated that the Random Forest algorithm performed best for classification while Linear SVC excelled in scenarios particularly in Class 2. However, the study encountered challenges such as classifying data due to the intricate nature of natural language processing and limited specialized terms in the dataset. Furthermore, it underscored the complexities associated with multi classification compared to classification due to the increased intricacies in distinguishing between categories of drug effectiveness. The study concluded that although the Random Forest algorithm shows potential in sentiment analysis, further enhancements and additional features may be required to improve performance in classification within drug sentiment analysis applications.

2.3 RESEARCH GAP:

This research project aims to deepen our understanding of supervised machine learning by focusing on comparing two models and improving the performance of the best model by hyperparameter tuning. Additionally, it investigates the use of TF IDF for extracting features and SMOTE for balancing classes to address overfitting concerns. While previous studies have extensively examined machine learning models, there is a need to explore the trade-offs between model complexity and scalability regarding class balancing in handling datasets. This study endeavours to fill this gap by offering insights into managing these challenges while developing scalable machine learning solutions.

CHAPTER THREE

MATERIALS AND METHOD

This chapter talks about the tools and techniques used to develop a sentiment analytical system to enhance patient satisfaction on drugs with an interactive GUI.

3.1 TOOLS

This section discusses the materials utilized in this research project. The resources employed include Python 3.7 Google Colab Pro, Visual Studio Code (VS Code), Spacy, Pandas, Numpy, Matplotlib, seaborn, sklearn, python flask etc.

3.1.2 PYTHON

Python stands out as a utilized programming language known for its applications in creating machine learning models, developing websites, software automation tasks and data analysis. As a programming language, Python is suitable for a range of applications and is not limited to any specific problem domain (Python.org, 2022). Its flexibility and user-friendly nature have significantly contributed to its acceptance, establishing it as one of the preferred programming languages.

Python is employed for creating machine learning systems (intelligence systems), pre-processing, analysing and visualizing data. People write code for reasons whether it's to earn a living, tackle data analysis problems, have fun or assist others with problem solving tasks. Python serves purposes such as web development, scripting, data analysis and machine learning, automation, software testing and prototyping.

The version of Python utilized in this project is Python 3.11.9.

3.1.3 Visual Studio Code (VS Code)

VS Code is a code editor created by Microsoft that offers a range of functionalities for developers working on projects in programming languages. It combines the simplicity of a text editor with features typically found in development environments (IDEs) such as IntelliSense for autocomplete suggestions, syntax highlighting, and debugging capabilities. One of the standout features of VS Code is its collection of extensions that allow users to use the editor for their specific needs, enhancing developers' productivity by adding support for programming languages such as Python or JavaScript, or by integrating tools such as Docker and Git into their coding environment. The built-in Git integration streamlines version control tasks within the editor itself thereby facilitating tasks like commits, branch management and resolving merge conflicts effortlessly. Moreover, VS Code is compatible across operating systems including Windows, macOS and Linux – ensuring a good user experience regardless of platform. Features such as Live Share further elevate its utility by enabling real time collaboration among individuals and teams during software development projects.

The version being utilized for this project is Visual Studio Code Version 1.92.1 (user setup).

3.1.4 Pandas

This is a utilized Python library that is very important in designing models and developing algorithms, especially within data science and machine learning. It offers data structures such as Series and Data Frames to facilitate manipulation and analysis of data. Through Pandas developers can easily manage tasks like data cleaning, transformation, and aggregation to streamline preparation for modelling purposes. Pandas excels at handling datasets by providing functions for filtering, merging and reshaping data during data preprocessing. Moreover, it seamlessly integrates with libraries like NumPy, Matplotlib and Scikit Learn, proving to be a resource for creating robust algorithms using Python.

3.1.5 Numpy

This is a Python library that is widely utilized for designing models and developing algorithms for scientific computation and data analysis. This functionality is crucial for managing data and facilitating speedy and memory efficient calculations. Its features related to linear algebra, random number generation and Fourier transforms are frequently employed in algorithm creation across fields such as machine learning, physics, and engineering. Additionally, NumPy serves as the cornerstone for libraries like Pandas and Scikit Learn that seamlessly integrate it to support various tasks in Python ranging from data preprocessing to model enhancement.

The version of NumPy being used is Version 1.26.4 which was installed by executing 'pip install numpy', within the VS Code terminal.

3.1.6 Spacy

This is a Python library for performing NLP tasks, and it is widely employed in algorithm creation and data preparation. This tool provides a variety of features for handling and examining text datasets, such as tokenization, identifying parts of speech recognizing named entities parsing dependencies and more. These functions are crucial for developing NLP algorithms and models that necessitate an understanding of language structure and semantics. One key advantage of Spacy lies in its speed and efficiency enabling processing of large text corpora making it well suited to real time applications. The library also aids in enhancing its functionality for creating NLP models. In summary Spacy proves to be a tool for preprocessing data and constructing sophisticated algorithms within the realm of NLP.

The version utilized is Version 3.7.5 of Spacy which was installed through the 'pip install spacy' command, on the VS Code terminal.

3.1.7 Matplotlib:

Matplotlib is an essential Python library for developing algorithms used in data visualization. This visual analysis is crucial for understanding data patterns and trends which is important for algorithm development and evaluation. Developers can use Matplotlib to create different types of plots like line graphs, bar charts, histograms, scatter plots and more with customization options. It also integrates seamlessly with Python libraries such as NumPy, Pandas and SciPy simplifying the visualization of data from these sources. By providing a representation of data Matplotlib assists developers in debugging optimizing and explaining their performance. Making it an invaluable tool in the Python ecosystem for algorithm development.

The version of Matplotlib being utilized is Version 3.9.0; it was installed via 'pip install matplotlib' in the VS Code terminal.

3.1.8 Scikit-learn (sklearn):

Sklearn is a Python library that is very robust and extensively utilized for developing algorithms and working on machine learning tasks. It offers a set of resources for creating, training, and assessing machine learning models making it a favoured choice for newcomers as well as seasoned professionals. The library supports machine learning techniques such as metrics, classification, regression, clustering and dimensionality reduction all conveniently accessible through a consistent interface.

An important feature of Scikit Learn is its user nature enabling developers to swiftly implement and experiment with algorithms. Whether designing models or deploying them in real world scenarios Scikit Learn equips users with the resources to create robust and effective machine learning solutions, in Python.

The version of Scikit Learn being used is Version 1.5.0 which was installed by executing 'pip install sklearn' in the VS Code terminal.

3.1.9 Textblob:

TextBlob is a Python library that simplifies the processing and analysis of text data making it an asset in developing algorithms and machine learning models. It enables developers to seamlessly incorporate NLP functionalities into their projects without needing knowledge of language processing. Additionally, TextBlob streamlines text preprocessing tasks such as tokenization and lemmatization which are essential for preparing data for advanced NLP applications.

The version of TextBlob used is Version 0.18.0.post0, which was installed by executing 'pip install textblob', in the VS Code terminal.

3.1.10 Google Colab:

Google Colab is designed for data scientists, machine learning practitioners, and Python developers. Due to the size of the research dataset, there was a need to increase its RAM and extend the session duration which led to the subscription of Google Colab Pro and Google Colab Pro+ subsequently. Google Colab is a powerful tool for developing and training machine learning models. Additionally, it enables teamwork by allowing users to share notes and integrate with Google Drive. This makes doing group projects easier and possible.

3.2 METHOD JUSTIFICATION

To create a reliable data model for this project, it was essential to choose a methodology that aligned with the project's goals. Extensive research was conducted before starting this study to explore methodologies and determine the appropriate approach. One of the methods investigated was the Team Data Science Process (TDSP), introduced by Microsoft in 2016. The TDSP framework consists of five phases: Business Understanding, Data Acquisition and Understanding Modeling, Deployment and Acceptance. This methodology offers an approach to managing data science projects. Another method considered was SEMMA developed by SAS, which stands for Sample, Explore, Modify, Model and Assess. SEMMA is designed to address a range of business challenges such as fraud detection, database marketing, customer loyalty programs, bankruptcy forecasting, market segmentation strategies, customer retention strategies, risk analysis techniques, affinity analysis approaches and portfolio analysis methods. The third methodology reviewed was OSEMN. Obtain, Scrub, Explore, Model and Interpret. Known for its simplicity and effectiveness in handling data science workflows. Additionally, Knowledge Discovery in Databases (KDD) was also considered. The process known as KDD involves stages, including data selection, data cleaning, data integration, data transformation, data mining, pattern evaluation and knowledge presentation. Upon reviewing methodologies, the CRISP DM (Cross Industry Standard Process for Data Mining) approach was found to be particularly compelling. CRISP DM offers a comprehensive framework for managing data mining/machine learning projects.

CRISP DM methodology was ultimately opted for due to its adaptability and thoroughness. It facilitates model creation and evaluation while ensuring deployment and acceptance of the solution. In this project context CRISP DM guided the development of a sentiment analytical system to improve satisfaction with drugs through a customized user interface tailored to their needs. This approach provided a structure for meeting project requirements. Delivering a top-notch solution.

3.2.1 Methodology adopted for the project.

CRISP DM, known as the Cross Industry Process for Data Mining provides a framework for organizing and carrying out projects in machine learning, data science, and data mining. It is an adaptable and recognized methodology valued for its effectiveness in solving business challenges. The CRISP DM approach emphasizes flexibility, allowing tasks to be completed in order and often requiring refinement through repetition. Its main goal is to improve project efficiency, reliability, and

adaptability. The methodology comprises six stages: Business Understanding, Data Understanding, Data Preparation, Modelling, Evaluation, and Deployment. Each stage typically loops back to the previous one, in a fashion to allow for continuous review and adjustment as necessary. This iterative process enables revisiting and enhancing the stages to ensure results.

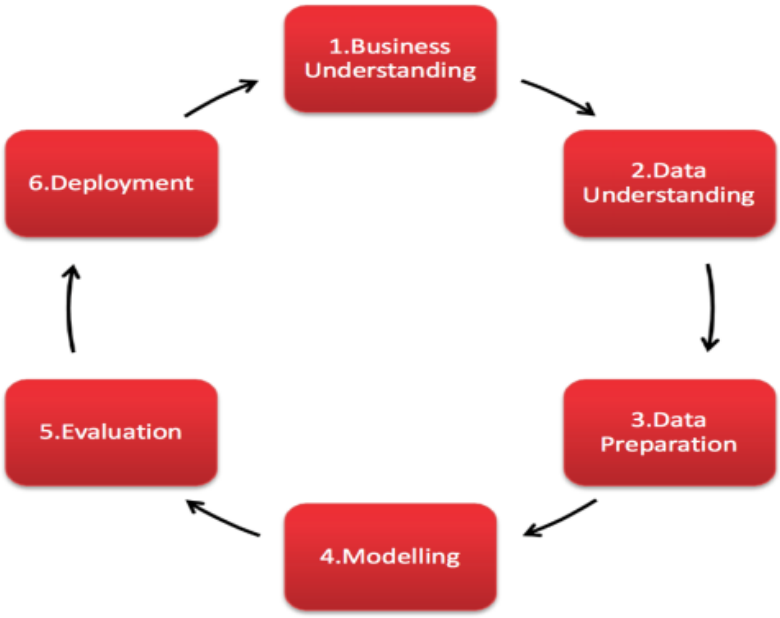


Figure 3.1: CRISP-DM Methodology (Source: https://www.sv-europe.com/crisp-dm-methodology/)

3.3 METHODOLOGY:

In this section, the CRISP DM methodology was applied to this research, which explains the approach and outlines how the six stages of CRISP DM were used to carry out this project work. Below are the actions taken to ensure that this project was comprehensively completed.

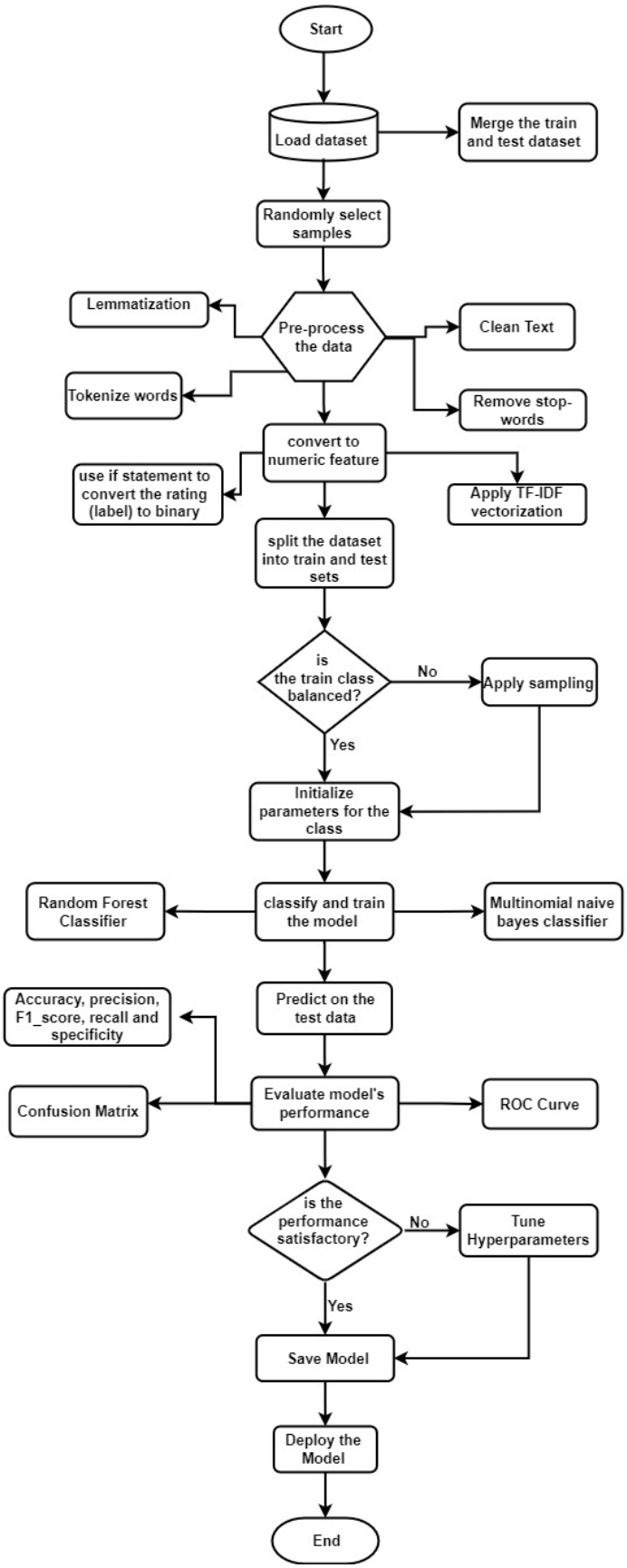


Figure 3.2: Flowchart algorithm of the Sentiment Analytical System

3.3.1 Stage 1: Business Understanding

This is the first and most important stage in the development of a sentiment analytical system which enhances patient satisfaction with drugs using machine learning models. At this stage, extensive research was done to understand the primary aim which is to enhance patient satisfaction with drugs. This aim is intended to be achieved by gaining insights into patient experiences with various drugs and identifying areas for improvement. Additionally, the researcher understood and created the project's specification, including its objective, economic impact, legal, social, ethical, and professional aspects, which are essential in order to establish a solid basis for the successful implementation of this project. The following specific objectives were established as stated in the project objectives in chapter 1 (1.9).

3.3.2 Stage 2: Data Understanding

This is the second stage of the CRISP-DM approach where the data for this project was fully understood. Having the primary aim of enhancing patient satisfaction on drugs, the researcher discovered that data with patients' reviews on drugs and ratings will be required in order to successfully deliver this project. Data was sourced and downloaded from open-source Kaggle which is a community platform that provides a collaborative environment that hosts datasets, and notebooks. The data has a Creative Commons Attribution 4.0 International (CC BY 4.0) license. (https://archive.ics.uci.edu/dataset/462/drug+review+dataset+drugs+com). See Appendix 1 for a screenshot of the license. This permits the dataset to be shared and adapted for any purpose provided that proper reference is given (Gräßer et al., 2018).

3.3.2.1 Library Import and Dataset Reading:

The dataset which was downloaded from Kaggle was uploaded to Google drive for easy accessibility through Google colab. These CSV datasets with names drugsComTrain_raw and drugsComTest_raw were loaded (read) using Pandas and including the exact storage path of these datasets on the Google drive directory. The Google drive was first connected and mounted on the Goggle colab by importing drive from Google colab, after which these datasets were read individually using this syntax. Go to Appendix 2.

3.3.2.2 Data Description:

This dataset named UCI ML Drug Review dataset curated from Kaggle (https://www.kaggle.com/datasets/jessicali9530/kuc-hackathon-winter-2018/data) is a patient's review on drugs dataset. The dataset has 7 features which includes: UniqueID, drugName, condition, review, rating, date, usefulCount. See Appendix 3 for a screenshot of the dataset showing its features and some records. (Gräßer et al., 2018).

The data is of two sets named drugsComTrain_raw with 161297 records and drugsComTest_raw with 53766 records. See Appendix 4 for the full description of the numerical features of each of these individual datasets.

```
✓ df_train_data.shape ···
(161297, 7)

✓ df_test_data.shape ···
(53766, 7)
```

Figure 3.3: The shape of each of the dataset.

3.3.3 Stage 3: Data Preparation

This is the 3rd phase of the CRISP-DM approach, where data is loaded and pre-processed to build machine learning models. Data preparation involves the following steps:

3.3.3.1 Dataset Merge**:** The two sets of data were merged to undergo cleaning and preprocessing using pd.concat, a function, within the Pandas library. The outcome was saved in a variable that was then utilized for data preparation. Refer to Appendix 5 for details.

3.3.3.2 Exploratory Data Analysis: The data table was viewed using the head function. The shape function was also applied to the column and row size of the dataset. The describe function is a summary statistic presenting the count, mean, standard deviation etc. The function info gives a detail explanation on column, row, type etc. Graphs and charts were plotted to view distribution of the dataset. See Appendix 6 and 7 for the statistical summary of the merged dataset and some graph to view its distribution.

3.3.3.2 Dataset Sampling: In this part of data preparation, the researcher randomly chooses samples to train the model which helps in decreasing the dataset processing time.

For the first training, 10,000 samples of the dataset which is 4.6% was used to train the two models.

For the second training, 20,000 samples of the dataset which is 9.3% was used to train the two models. The syntax below was used to perform this process:

```
df_review_data = df_review_data.sample(n=10000, random_state=42)
```

For the final training, the whole record of the dataset 215063 (100%) was used for model training.

3.3.3.3 Dataset preprocessing: The Spacy model for pre-processing was loaded with this syntax: nlp = spacy.load('en_core_web_sm'), after which the dataset was cleaned and pre-processed by undergoing the following processes:

1) Checking for null values and dropping them: Null values were checked and only the condition feature has null values of 1194 which were dropped.

```
uniqueID          0
drugName          0
condition      1194
review            0
rating            0
date              0
usefulCount       0
dtype: int64
```

Figure 3.4 The null values in the dataset.

```
#checking the null values and features with the highe
df_review_data.isnull().sum()

Run Cell | Run Above | Debug Cell | Go to [12]
#%%
# dropping the null values
df_review_data.dropna(inplace=True, axis=0)

Run Cell | Run Above | Debug Cell | Go to [14]
```

Figure 3.5 The dropping of null values

2) Removal of punctuations.

3) Lowering of text.

4) Applying of regular expressions (regex) to remove non-letters.

5) Replacing of n' t with not.

6) Tokenization.

7) Lemmatization.

8) Removal of stopwords.

Most of these processes were written as a single function and applied to the dataset.

Go to Appendix 8 for full details.

<u>3.3.3.4 Feature Engineering:</u> Feature engineering plays a vital role in ML and NLP by creating meaningful features from raw data. TF-IDF serves as a mathematical tool for evaluating the benefit of words in documents. This method captures the importance of words in documents while accounting for their frequency across the entire dataset. To achieve this, the TF-IDF vectorizer is employed to transform the variables (features). Refer to Appendix 9 for insights into how TF IDF vectorization was carried out in the cleaned review.

<u>3.3.3.5 Data Splitting and Oversampling:</u> As part of preparing the data for modelling, the data was split into train and test sets using a test size of 0.33 practice which ensures that the model can be evaluated on unseen data. This ensures that the train set was used to train the model and the test set to make predictions. The data was split using the Scikit learn model selection library, passing the test size and random state.

67% of the data which is $\frac{67}{100} \times 213869 = 143{,}292$ were used to train the data.

33% of the data which is $\frac{33}{100} \times 213869 = 70{,}577$ were used to test the data.

Furthermore, the data was checked for imbalanced classes which is a situation where one class in a classification problem has significantly fewer examples than another class. This is because an imbalanced class can lead to biased model performance and incorrect predictions, since the majority class may take precedence over the minority class and the model performs badly on the minority class. To handle the class imbalance, SMOTE was used.

To balance the unbalanced classes

```
from imblearn.over_sampling import SMOTE
smote = SMOTE(random_state=42)
X_resampled, y_resampled = smote.fit_resample(X_train, Y_train)

# to check the distribution of classes after applying SMOTE
print(f"Original class distribution: {Counter(Y_train)}")
print(f"Resampled class distribution: {Counter(y_resampled)}")
```

Figure 3.6 Code to balance the class.

SMOTE which means Synthetic Minority Over-sampling Technique is a method in the imblearn library which addresses issues with class imbalances in dataset classification situations. The main purpose of SMOTE is to generate instances for the minority class, thereby achieving a balanced dataset by boosting the representation of the underrepresented class. Go to Appendix 10 for details on the result and the sampling rate.

3.3.4 Stage 4: Model Training

Model training is the 4th stage of the CRISP-DM Methodology which is very vital in machine learning and focuses on leveraging datasets to train algorithms in identifying patterns and correlations within the data. Its goal is to create a model that will predict effectively or make classifications from new unseen data. This procedure comprises steps.

3.3.4.1 Selecting the Model:

In this initial sub-stage, appropriate machine learning algorithms are chosen based on the nature of the sentiment analysis task. Common models for sentiment analysis include Multinomial Naïve Bayes, Random Forest Classifiers, Logistic Regression, and Support Vector Machines (SVM). For this system, a model that is not complex in handling very large datasets and capturing sentiment nuances was selected. The problem of this research work is classification of textual into positive and negative sentiments. The project will adopt the following ML algorithms.

Multinomial Naïve Bayes:

The multinomial Naive Bayes classifier is a known type of Naïve Bayes model commonly used in text classification, with discrete features like word frequencies for classifying text. It is a probabilistic method employed to forecast the class of a document by analysing word occurrences. Multinomial Naive Bayes was selected due to its speed, performance, and efficiency particularly when handling large datasets.

The Multinomial Naïve Bayes classifier was built from scratch by inheriting from BaseEstimator and ClassifierMixin, two classes from the sklearn library which is used to define custom estimators.

Laplace Smoothing: This is a Multinomial Naïve Bayes Classifier smoothing parameter which helps to handle zero probabilities by adding a small value (alpha) to the word count. Its default value is 1.0.

The following are the explanations of the Multinomial Naïve Bayes algorithm after data preparation:

Step 1: initialize the Multinomial Naive Bayes class with a smoothing parameter alpha, which helps with Laplace smoothing.

Step 2: Train the classifier by calculating the class prior probability and the likelihood of each word in a document for each class (conditional word probabilities).

Step 3: Predict the input data by calculating the logarithm of the probability for each class group and selecting the class with the log probability for each document.

Step 4: Return the class with the highest log probability.

Step 5: Return the class probabilities (normalized) for each document, which can be useful for the evaluation of model performance with metrics like accuracy, precision etc. See appendix 11 for full details of the code from scratch.

Hyperparameter Tuning: This was employed to optimize the Multinomial Naïve Bayes model's performance by finding the most effective parameter settings. It was performed when the model's performance was not satisfactory. The parameter to be tuned here is the Laplace Smoothing (alpha). This process helps in refining the model to achieve better accuracy and robustness. The hyperparameter tuning used was grid search using the GridSearchCV class. Refer to appendix 12 for the code.

Random Forest Classifier:

Step 1: initialize the random forest classifier with its hyperparameters like n_estimators, max_depth.

Step 2: loop to create bootstrap samples and train the decision trees.

Step 3: train each of the trees using the bootstrap sample.

Step 4: save the trained tree.

Step 5: combine the predictions from all trees.

Step 6: Assess the model's performance.

Go to appendix 13 for full code of this model from scratch.

3.3.5 Stage 5: Evaluation

The CRISP-DM Methodology is in its fifth state. The system was thoroughly evaluated using the metrics method in Scikit Learn. Metrics refer to quantitative measurements or evaluation criteria used to assess the performance of a model or the quality of prediction or classification. See appendix 14 for details. Below are metrics used to evaluate the performance of the models used for sentiment analytical system design:

3.3.5.1 Accuracy: this measures the proportion of predicted instances that are correctly out of the total instances. It is computed by adding the true positive to the true negative and divided by the total sample.

$$Accuracy = \frac{True\ Positives(TP) + True\ Negatives(TN)}{Total\ Samples}$$

3.3.5.2 Precision: This shows how many of the predicted positives are positives. Precision is the ratio of accurate positive discoveries to those that the classifier expected to be positive.

$$Precision = \frac{True\ Positives(TP))}{True\ Positives(TP) + False\ Positives(FP)}$$

3.3.5.3 Recall (Sensitivity): This measures the proportion of actual positive instances that were predicted to be correct.

$$Recall = \frac{True\ Positives(TP))}{True\ Positives(TP) + False\ Negatives(NP)}$$

3.3.5.4 F1 Score: The harmonic mean of precision and recall, which is how correctness an accuracy is.

$$F1\ Score\ = 2 \times \frac{Precision\ \times Recall}{Precision\ + Recall}$$

3.3.5.5 Specificity: This measures the proportion of actual negative instances of the classification models that were correctly predicted.

$$Specificity = \frac{True\ Negatives(TN))}{True\ Negatives(TN) + False\ Positives(FP)}$$

3.3.5.5 Confusion Matrix: This summarizes with a matrix the performance of a classification model by showing the counts of True Negatives, False Positives, True Positives, and False Negatives.

	Predicted Positive	Predicted Negative
Actual Positive	TP	FN
Actual Negative	FP	TN

3.3.5.6 ROC-AUC (Receiver Operating Characteristic - Area Under the Curve): It represents the degree of separability and measures a model's ability to distinguish between classes. The ROC curve plots the True Positive Rate against the False Positive Rate at various threshold levels. AUC summarizes this information, with values closer to 1 indicating better performance.

$$\mathrm{AUC} = \int_0^1 \mathrm{TPR}(\mathrm{FPR})\, d(\mathrm{FPR})$$

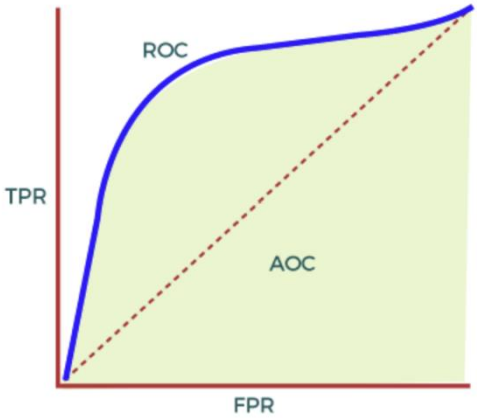


Figure 3.7 ROC-AUC metric for prediction evaluation. (Chandola et., 2009)

In Figure 3.3, the AUC represents the two-dimensional area beneath the ROC curve from (0,0) to (1,1). AUC evaluates how well a binary classifier performs across thresholds and gives a measure of the ROC curve. The AUC score ranges from 0 to 1 with a score indicating better model performance and clearer distinction between classes. A good model typically has an AUC value that's close to 1.

See appendix 13 for full details of the evaluation code and appendix 14 for the results of the evaluation.

3.3.6 Stage 6: Deployment

This stage involves the implementation and deployment of the new system on the Heroku platform. Heroku is a cloud-based platform-as-a-service (PaaS) that allows developers to build, run, and manage applications entirely in the cloud. It is compatible with various programming languages such as Python, Ruby, Java, Node.js, Scala, PHP, and Go, among others. Heroku simplifies the deployment process by allowing developers to push their code directly to the platform, where Heroku handles the infrastructure, scaling, and management.

To make the system user-friendly and easy to use, a page was developed using Python, HTML and CSS. The user can type in their reviews about any drug and the system analyses and gives them a sentiment output, which is either positive or negative. Refer to appendix 15 for details.

CHAPTER 4

RESULTS AND DISCUSSION

In this chapter, it presents the outcomes of the project, such as the findings, patterns, or trends that emerged from the analysis. This focuses on facts and observations, interpretation or speculation and the implications of the results. It also covers the process for the application of sentiment analytical systems to enhance patient satisfaction on drugs using machine learning models Multinomial Naive Bayes and random forest, the comparing and evaluation of the models' performance metrics, and hyperparameter optimization.

4.1 DATA PREPARED:

Objective 1: To design and implement an algorithm that uses Spacy Python libraries to preprocess data and understand the data, with the purpose of obtaining accurate analysable results.

The sourced data comprises patient reviews of various drugs, with features such as Unique ID, Reviews, Ratings, Conditions, etc. This data is subjected to pre-processing and cleaning procedures to prepare it for model training. These steps include handling null values, removing stopwords, tokenizing the text, eliminating punctuation, applying lemmatization, converting the text to lowercase, and performing regular expressions. The result is a cleaned review which is stored in the variable Cleaned_feedback.

	uniqueID	drugName	condition	review	rating	date	usefulCount	cleaned_feedback
14311	57701	Acetaminophen / hydrocodone	Pain	"I suffered a .45 caliber GSW to the head in 2...	9	26-Jun-15	10	suffer caliber gsw head multiple omfsent s...
72301	168695	Vilazodone	Major Depressive Disorde	"I started on Viibryd after having a DNA test ...	1	27-Dec-15	1	start viibryd have dna test find medication ...
23041	1607	Eluxadoline	Irritable Bowel Syndrome	"Not covered by insurance"	6	19-Dec-16	0	cover insurance
197721	17559	Ethinyl estradiol / etonogestrel	Birth Control	"I previously did a review on Nuva Ring about ...	8	1-Nov-16	8	previously review nuva ring year ago opinion n...
133612	158793	Methylphenidate	ADHD	"My son was prescribed the drug just before su...	6	3-Sep-09	17	son prescribe drug summer work great beginni...
152966	139328	Armodafinil	Narcolepsy	"After being diagnosed at age 35, with narcole...	2	7-Jul-15	19	diagnose age narcolepsy rem sleep behavior d...
53745	161508	Diphenhydramine	Allergic Reactions	"Drowsiness is a reoccurring problem for me. I...	7	19-Oct-15	3	drowsiness reoccurring problem difficulty focu...
90532	122141	Venlafaxine	Depression	"I was told to take Effexor Rx at the age of 1...	8	3-Nov-09	25	tell effexor rx age respond horrible antidep...
5571	17896	Ethinyl estradiol / etonogestrel	Birth Control	"Overall, it works great. I have used the Nuva...	9	19-Jul-13	13	overall work great nuvaring past month like ...
98666	14936	Ethinyl estradiol / norethindrone	Birth Control	"This pill is effective in preventing pregnanc...	7	9-Aug-17	3	pill effective prevent pregnancy help acne not...

Figure 4.1: Pre-processed Dataset

4.2 SENTIMENT ANALYSIS WITH TEXTBLOB:

Before training the classifier models, sentiment analysis was performed on the data using Textblob as part of our preparation and exploration process. This helped us explore the data by leveraging on the sentiment results.

Objective 2: To design and implement an algorithm that uses sentiment analysis on patient reviews on drugs to categorize the review into positive, negative, or neutral using TextBlob a python library.

Result: A dataset showing polarity which is based on sentiment score, subjectivity score, and sentiment (positive, negative and neutral) got through categorization of polarity.

	uniqueID	drugName	condition	review	rating	date	usefulCount	cleaned_feedback	polarity	subjectivity_score	sentiment
14311	57701	Acetaminophen / hydrocodone	Pain	"I suffered a .45 caliber GSW to the head in 2...	9	26-Jun-15	10	suffer caliber gsw head multiple omfsent s...	0.187500	0.437500	Positive
72301	168695	Vilazodone	Major Depressive Disorde	"I started on Viibryd after having a DNA test ...	1	27-Dec-15	1	start viibryd have dna test find medication ...	-0.379167	0.727083	Negative
23041	1607	Eluxadoline	Irritable Bowel Syndrome	"Not covered by insurance"	6	19-Dec-16	0	cover insurance	0.000000	0.000000	Neutral
197721	17559	Ethinyl estradiol / etonogestrel	Birth Control	"I previously did a review on Nuva Ring about ...	8	1-Nov-16	8	previously review nuva ring year ago opinion n...	0.206994	0.526488	Positive
133612	158793	Methylphenidate	ADHD	"My son was prescribed the drug just before su...	6	3-Sep-09	17	son prescribe drug summer work great beginni...	0.222222	0.488889	Positive
152966	139328	Armodafinil	Narcolepsy	"After being diagnosed at age 35, with narcole...	2	7-Jul-15	19	diagnose age narcolepsy rem sleep behavior d...	-0.380000	0.708333	Negative
53745	161508	Diphenhydramine	Allergic Reactions	"Drowsiness is a reoccurring problem for me. I...	7	19-Oct-15	3	drowsiness reoccurring problem difficulty focu...	-0.243750	0.450000	Negative
90532	122141	Venlafaxine	Depression	"I was told to take Effexor Rx at the age of 1...	8	3-Nov-09	25	tell effexor rx age respond horrible antidep...	-0.250000	0.583333	Negative
5571	17896	Ethinyl estradiol / etonogestrel	Birth Control	"Overall, it works great. I have used the Nuva...	9	19-Jul-13	13	overall work great nuvaring past month like ...	0.038889	0.538889	Neutral
98666	14936	Ethinyl estradiol / norethindrone	Birth Control	"This pill is effective in preventing pregnanc...	7	9-Aug-17	3	pill effective prevent pregnancy help acne not...	0.516667	0.800000	Positive

Figure 4.2: Sentiment categorized dataset.

4.3 VISUALIZATION AND EXPLORATORY DATA ANALYSIS (EDA):

Below shows the exploratory analysis performed on the data to identify relationships and connections between variables using visualization technique.

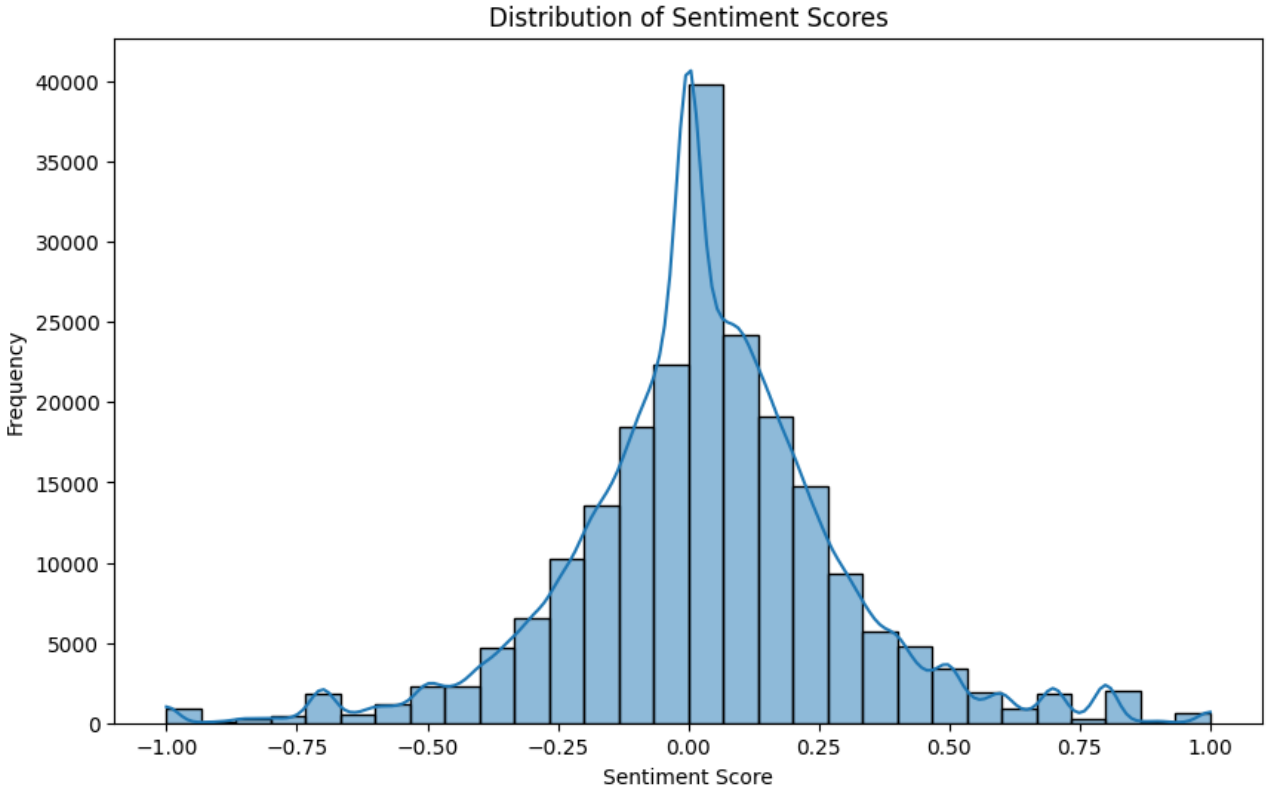


Figure 4.3: distribution of sentiment scores(polarity)

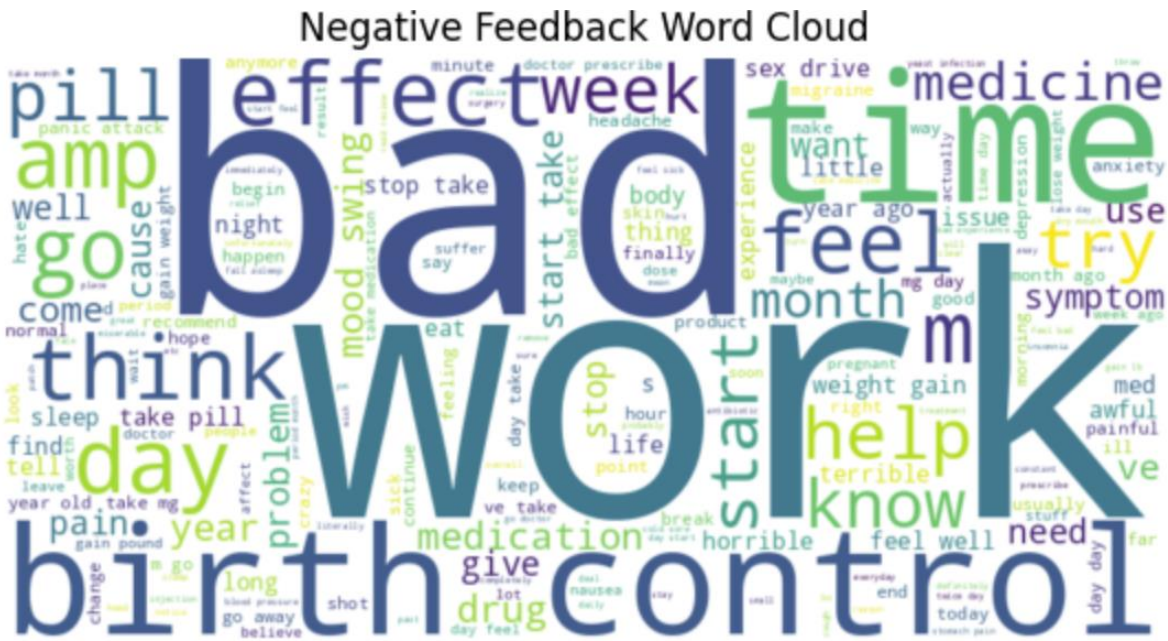
The graph above is a histogram that illustrates the distribution of sentiment scores, which range from -1 to 1, where -1 highly indicates negativity of the sentiment, 0 means that it is neutral sentiment, and 1 highly indicates positivity of the sentiment. The graph shows that most of the data is concentrated around a sentiment score of 0, which signifies that most entries have a neutral tone. The distribution is slightly skewed to the right, suggesting a mild positive bias in the dataset, with more entries expressing slightly positive sentiment than negative. The smooth density curve overlays the

histogram, providing a continuous approximation of the distribution, reinforcing the observation that extreme sentiment scores—both positive and negative—are relatively rare. In summary, the graph shows that the dataset predominantly consists of neutral to mildly positive sentiment, with few instances of strong emotional expression.

Figure 4.4: Average sentiment score by the drug

Figure 4.4: Positive Feedback Word Cloud

Figure 4.3 clearly demonstrates the positive Feedback words existing on the dataset. This reflects users' satisfaction with birth control, highlighting that it "works" effectively and delivers the desired "effects." Many users described their experience as "great" and appreciated how it helped them manage their health and lifestyle. Words like "time" and "feel" indicate that users found birth control to be timely in its effectiveness and positively influenced their well-being. The presence of terms like "help" and "love" indicates that users felt supported by the medication, leading to improved outcomes, such as better control over their bodies and peace of mind. Overall, feedback emphasizes the reliability and benefits of using birth control, with many users having positive experiences.


Negative Feedback Word Cloud
bad
work
birth control
time
effect
week
medicine
pill
amp
go
think
feel
try
use
symptom
month
help
know
start
day
year
pain
medication
give
drug
problem
mood swing
stop take
start take
weight gain
sex drive
want
little
year ago
issue
experience
body
thing
finally
night
happen
well
cause
come
sleep
take pill
find
tell
year old take mg
long
go away
believe
shot
horrible
feel well
need
terrible
usually
today
end
nausea
life
hour
stop
keep
ve take
break
eat
product
pregnant
right
good
month ago
morning
depression
anxiety
lose weight
time day
actually
make
way
headache
migraine
minute
doctor prescribe
anymore
panic attack
result
surgery
recommend
period
normal
hope
gain weight
hate
look
wait
feeling
day take
soon
sick
continue
crazy
affect
change
gain pound
m go
injection
blood pressure
lot
day feel
everyday
take day
stomach pain
far
day day
med
awful
painful
ill
ve
m
s
suffer
say
dose
mg day
maybe
take medication
bad effect
take day
take weight
week ago
gain lb
prescribe
constant
stuff
point
day start
go doctor
cold sore
people
doctor
leave
worth
month
mg
pound
dose
skin
hours
finally
dose
mg day
month ago
good
product
pregnant
right
weight gain
life
point
hour
stop
keep
ve take
break
horrible
feel well
need
terrible
usually
today
end
nausea
lot
day feel
shot
go away
believe
long
change
injection
blood pressure
m go
gain pound
year old take mg
leave
tell
find
sleep
take pill
doctor
people
look
wait
feeling
day take
soon
sick
continue
crazy
affect
worth
month
mg
pound

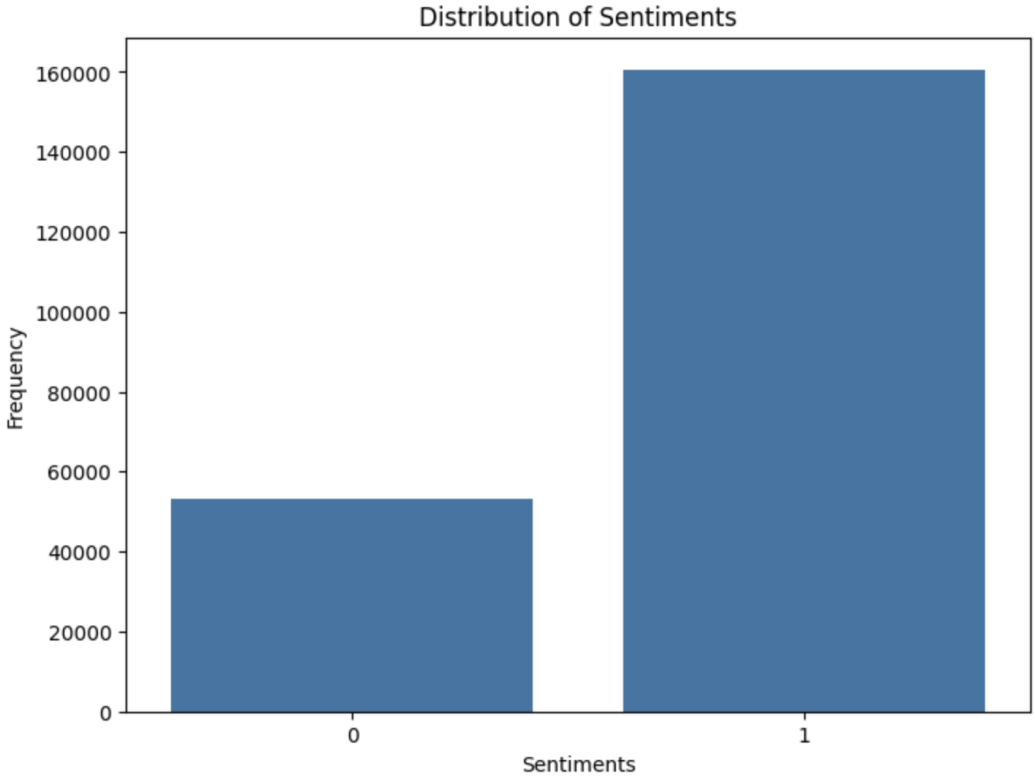


Figure 4.6 Distribution of imbalance Textual Data

Figure 4.6 clearly demonstrates a significant imbalance in the dataset classification, which may negatively impact the model's performance. The bar chart illustrates the distribution of sentiments within a dataset, highlighting a noticeable imbalance between the two sentiment classes. The bar labelled 0, representing negative sentiment. In contrast, the bar labelled 1, which corresponds to positive sentiment. This indicates that positive sentiment is significantly more prevalent, with roughly three times as many examples as negative sentiment. This imbalance could pose challenges when building machine learning models, as they might become biased towards predicting the majority class (positive sentiment). To resolve this issue, SMOTE which is specialized techniques for resampling, class weighting, would be employed to balance the imbalanced data.

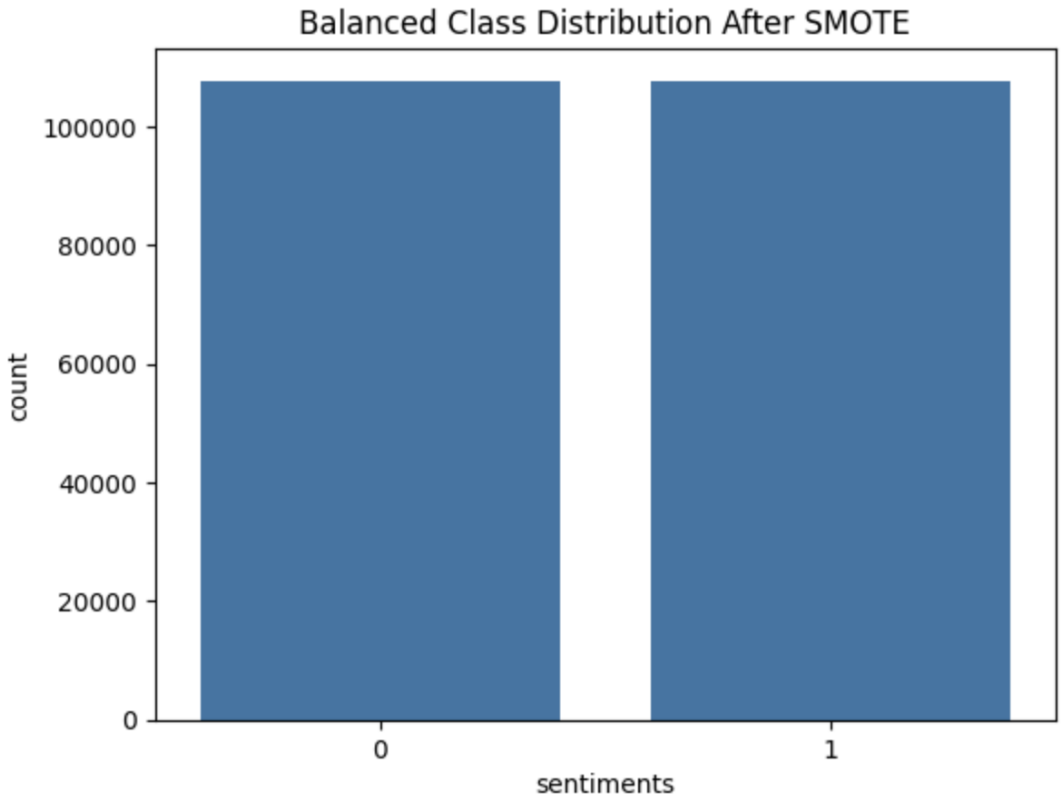


Figure 4.7 Balance Sentiments (Label) Distribution using SMOTE.

The principles outlined in section 3.3.3.5 were put into practice effectively in Figure 4.7. Furthermore, SMOTE achieved a distribution of sentiment classes at a 50,50 ratio. Dealing with class imbalances poses a challenge in machine learning classification scenarios where one class has notably fewer instances than the other. SMOTE is a common technique used in tackling this challenge. By creating samples through interpolation between existing ones SMOTE helps level out the distribution of classes and boosts the performance of the model.

Objective 3: To design an algorithm that trains the multinomial naïve bayes classifier and Random Forest Classifier from scratch and compare them based on their metrics.

4.5 COMPARE THE MODELS USING THE METRICS:

Multinomial Naïve Bayes and Random Forest Classifier are Supervised machine learning classification models that are used in performing sentiment analysis on people's reviews and emotions. These were used to train patients' reviews on drugs.

4.4.1 Baseline Model:

Random Forest Classifier (RFC) is considered the baseline model because it has lower performance across most metrics. Although RFC is more complex, its lower performance means that MNB should be compared against it to demonstrate improvement.

Table 4.1: Metrics result of the baseline model (RFC)

Accuracy	Precision	Recall	Specificity	F1 Score
0.70	0.81	0.81	0.81	0.37

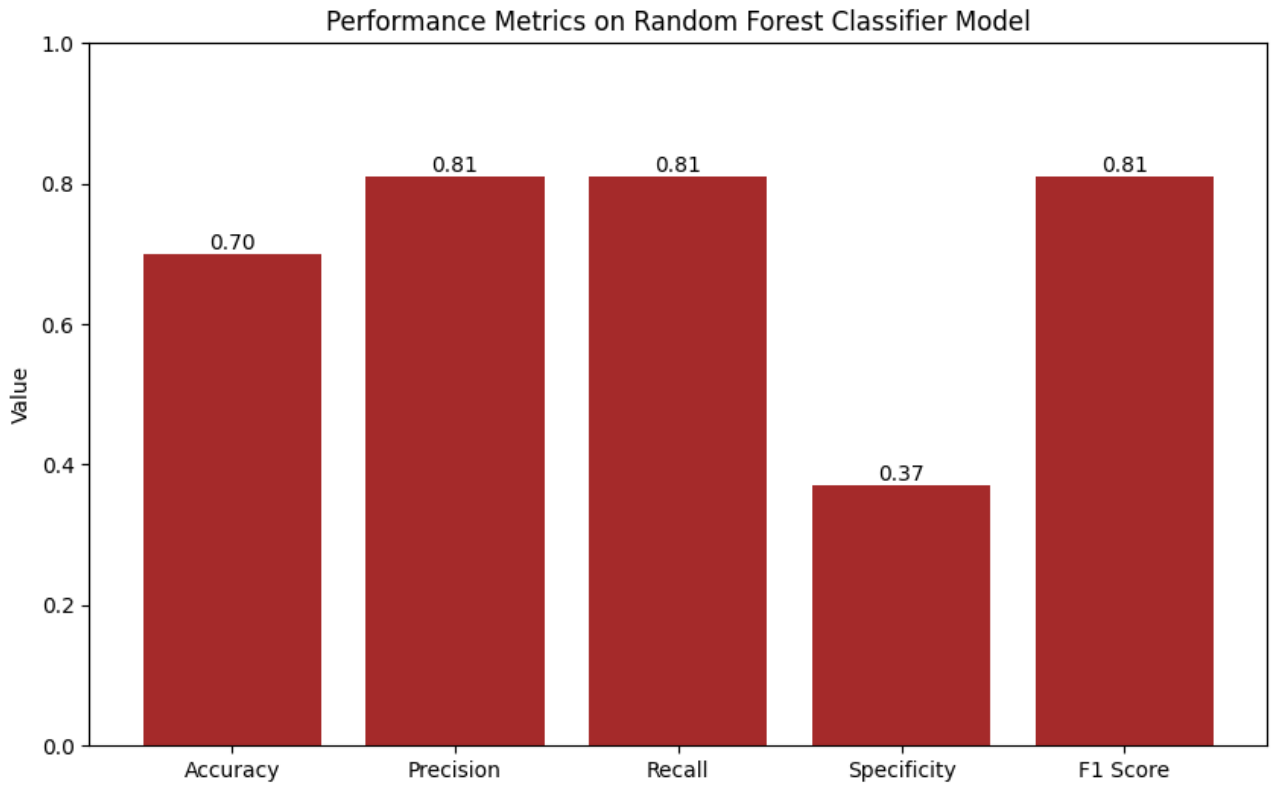


Figure 4.8 Quantitative assessment metrics for the baseline model RFC

The metrics performance of the baseline Random Forest Classifier result presented an accuracy of 0.70. The model correctly predicted 70% of overall instances overall. Both precision, recall and F1 Score are relatively high at 0.81, which shows that the model is very effective at identifying positive cases and making correct positive predictions. However, the model's specificity is low at 0.37, indicating it struggles to correctly classify negative instances. While the high precision and recall suggest that the model balances false positives and false negatives well, the low specificity highlights a significant weakness in identifying negative instances. This could be an issue especially where avoiding false positives is critical, such as in a drug recommendation system or fraud detection. Overall, while the model performs well in the positive class, further tuning or adjustments to the dataset may be necessary to improve its performance in the negative class.

4.4.1.1 Training the two models with 10,000 samples of the dataset to see their performance in correctly performing prediction using the metrics below:

Table 4.2: Confusion Matrix using 10,000 Samples of dataset:

Classifiers/Confusion Matrix	TN	FN	TP	FP
Multinomial Naïve Bayes (MNB)	541	634	1873	235
Random Forest Classifier (RFC)	173	294	1229	294

Given the above table, True Negatives (TN) of RFC = 173 and True Negative (TN) of MNB = 541 involve where the model correctly predicted the negative class. False Negatives (FN) of RFC = 294 and False Negatives (FN) of MNB = 634 is a scenario where the model predicted the negative class incorrectly meaning that the actual class was positive. True Positives (TP) of RFC = 1229 and True

Positives (TP) of MNB = 1873 are a scenario where the model predicted the positive class correctly. False Positives (FP) of RFC = 294 and False Positives (FP) of MNB = 235 involve a scenario where the model predicted the positive class incorrectly meaning the actual class was negative.

On calculating the metrics of the trained dataset across the two models using the formulae in the methodology, the following were the results of the metrics:

Table 4.3: Metrics results of the two ML models for comparison

Classifiers/Metrics	Accuracy	Precision	F1 Score	Recall	Specificity
Multinomial Naïve Bayes	0.74	0.89	0.81	0.75	0.70
Random Forest Classifier	0.70	0.81	0.81	0.81	0.37

To address the research question, How do model performance metrics (e.g., precision, recall, F1-score) differ between the two machine learning models when applied to patient drug satisfaction analysis?

Comparing the metrics of the Multinomial Naïve Bayes and Random Forest Classifier models based on the results provided above, the Multinomial Naïve Bayes model shows accuracy at 0.74, which indicates that it correctly classifies a portion of instances compared to the Random Forest Classifier at 0.70. This implies that the Naïve Bayes model is generally more reliable overall when analysing patient drug satisfaction data. In terms of precision, which measures how accurately the models identify true positives, the Multinomial Naïve Bayes model also excels with a precision score of 0.89 surpassing the Random Forest Classifier's 0.81. This suggests that Naïve Bayes model performs better in predicting patient satisfaction when it identifies a positive outcome. However, in terms of recall the Random Forest Classifier slightly outperforms the Naïve Bayes model (0.81, vs. 0.75). Recall measures how well a model identifies all positives; thus, in this scenario the Random Forest model is superior at identifying all satisfied patients even though it may misclassify some unsatisfied patients as satisfied individuals as well. Both models share an identical F1 Score of 0.81 which balances precision and recall effectively and indicates that despite variations in precision and recall values between them both models present similar overall performance levels in achieving this balance. The Multinomial Naïve Bayes model shows a higher level of specificity (0.70) compared to the Random Forest Classifier (0.37) indicating that the Naïve Bayes model excels at correctly identifying dissatisfied patients. This ability is vital for correctly identifying individuals who are not satisfied with their medication.

The calculated metrics data above indicates that the multinomial Naïve Bayes classifier performs better than the random forest classifier model by 4 % accuracy in predicting positive and negative classes. Nevertheless, there are opportunities for enhancement based on the values.

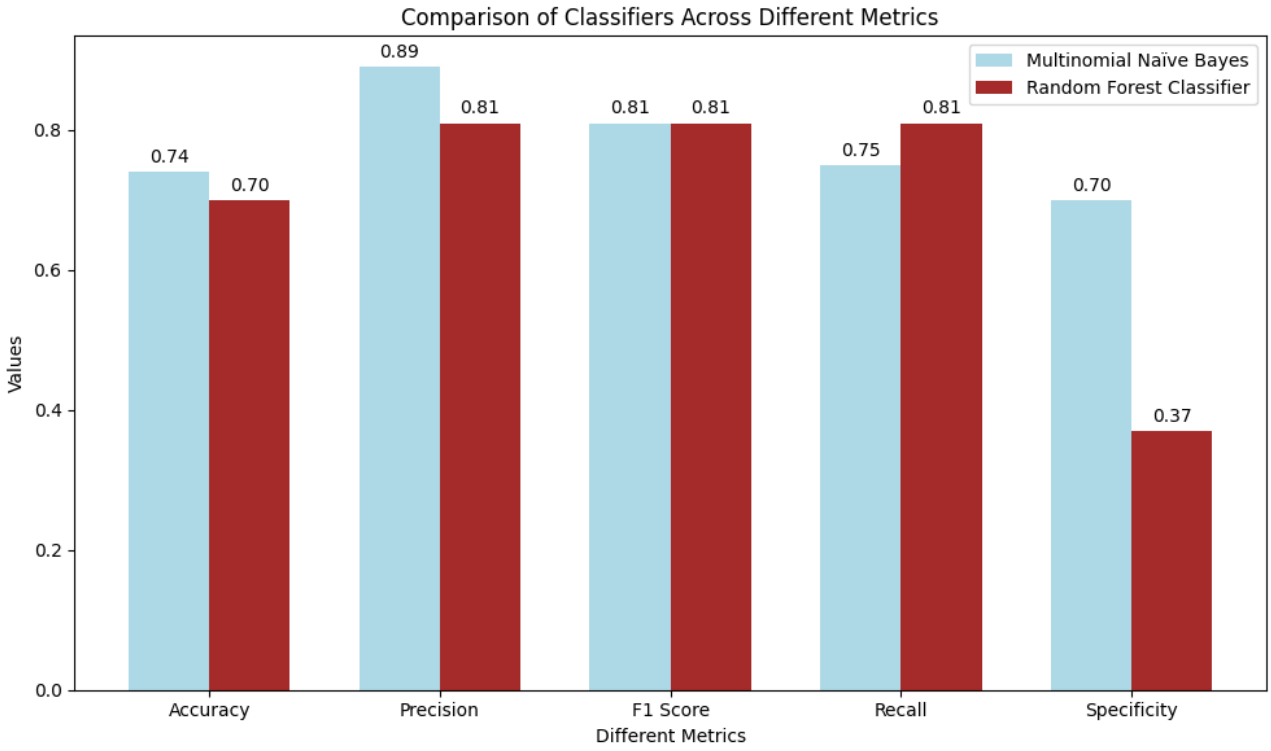


Figure 4.9: Comparative analysis of the classifiers across different metrics.

4.4.1.2 Training with a larger dataset to aid in improving the performance of the baseline model.

Table 4.4: Confusion Matrix for the two models using 20,000 Samples of dataset

Classifiers/Confusion Matrix	TN	FN	TP	FP
Multinomial Naïve Bayes (MNB)	1148	1162	3787	468
Random Forest Classifier (RFC)	414	638	2361	566

From the table above, True Negatives (TN) of RFC = 414 and True Negative (TN) of MNB = 1148 involve where the model predicted the negative class correctly. False Negatives (FN) of RFC = 638 and False Negatives (FN) of MNB = 1162 indicate that the model predicted the negative class incorrectly meaning that the true class was positive. True Positives (TP) of RFC = 2361 and True Positives (TP) of MNB = 3787 involve where the model predicted the positive class correctly. False Positives (FP) of RFC = 566 and False Positives (FP) of MNB = 468 involve a scenario where the model predicted the positive class incorrectly meaning the actual class was negative.

Calculating the metrics of the trained dataset across the two models using the formulae in the methodology, the following includes the results of the metrics:

Table 4.5: Metrics results of the two ML models for comparison using 20,000

Classifiers/Metrics	Accuracy	Precision	F1 Score	Recall	Specificity
Multinomial Naïve Bayes	0.75	0.89	0.82	0.77	0.71
Random Forest Classifier	0.70	0.81	0.80	0.79	0.42

To address the research question, " How do Multinomial Naive Bayes and Random Forest Classifiers compare in terms of identifying sentiments in patient drug reviews?”

The results suggest that the Multinomial Naïve Bayes model performs better overall at identifying sentiments in patient drug reviews. Specifically, it has higher accuracy (0.75), indicating it correctly classifies a greater proportion of both positive and negative sentiments compared to the Random Forest Classifier (0.70). Precision which is the proportion of true positives out of all positive predictions which is the number of predicted positives that are truly positive for multinomial naïve bayes and random forest classifiers are 0.89 and 0.81 respectively. Recall is the proportion of actual positives that are correctly identified by the model which is 0.77 and 0.79 for MNB and RFC respectively, specificity is the proportion of actual negatives that are correctly identified by the model which is 0.71 and 0.42 for MNB and RFC respectively, and lastly F1 score is the mean of precision and recall, giving a balance between the two metrics which is 0.82 and 0.80 for multinomial naïve bayes and random forest classifier respectively.

These metrics above show an improvement by training with a larger dataset, although multinomial naïve bayes is consistently outperforming random forest classifier in most metrics, particularly in accuracy which is 5% better, precision which is 8%, and specificity which is 29% better than in random forest classifier. However, there is room for improvement based on the metrics values to get the best and superior model.

When looking at identifying sentiments in drug reviews, the Multinomial Naïve Bayes model appears to be more effective overall. It offers accuracy, higher precision and better specificity making it a more reliable option for accurately classifying positive and negative sentiments. While the Random Forest Classifier shows an advantage in recall, the Multinomial Naïve Bayes model's balanced performance across metrics indicates that it is better suited to analysing sentiments in this area.

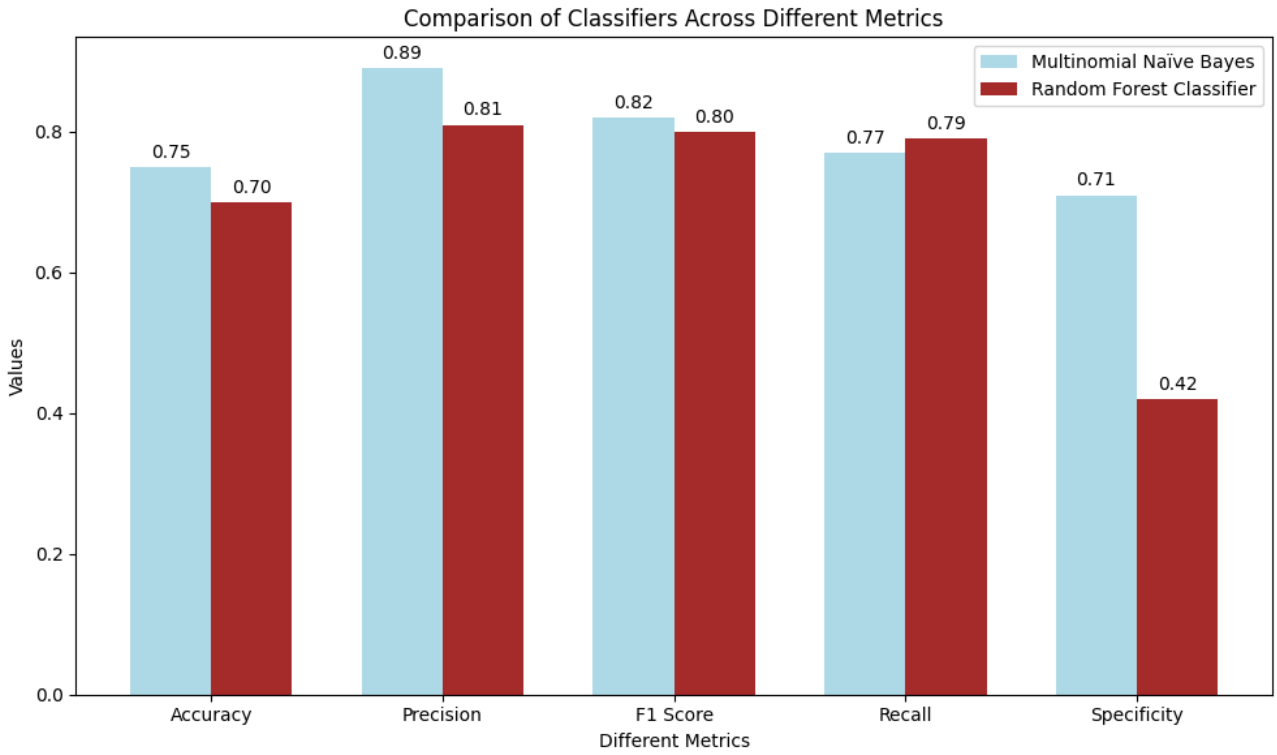


Figure 4.10: Comparative Analysis of the classifiers across different metrics with a larger set.

Having compared the results of the metrics above, there will not be any need to continue increasing the dataset to train the models as their performance increase is slow and Random Forest Classifier being a very complex supervised machine learning model which builds on decision trees has a long training time. However, Multinomial Naïve Bayes which is simpler and has been seen to show better performance would be improved to get the best and superior model from it.

4.4.1.3 Using the complete dataset to train multinomial naïve bayes models.

Table 4.6: Metrics results of the Multinomial Naïve Bayes using the complete dataset:

Accuracy	Precision	Recall	Specificity	F1 Score
0.79	0.90	0.82	0.72	0.86

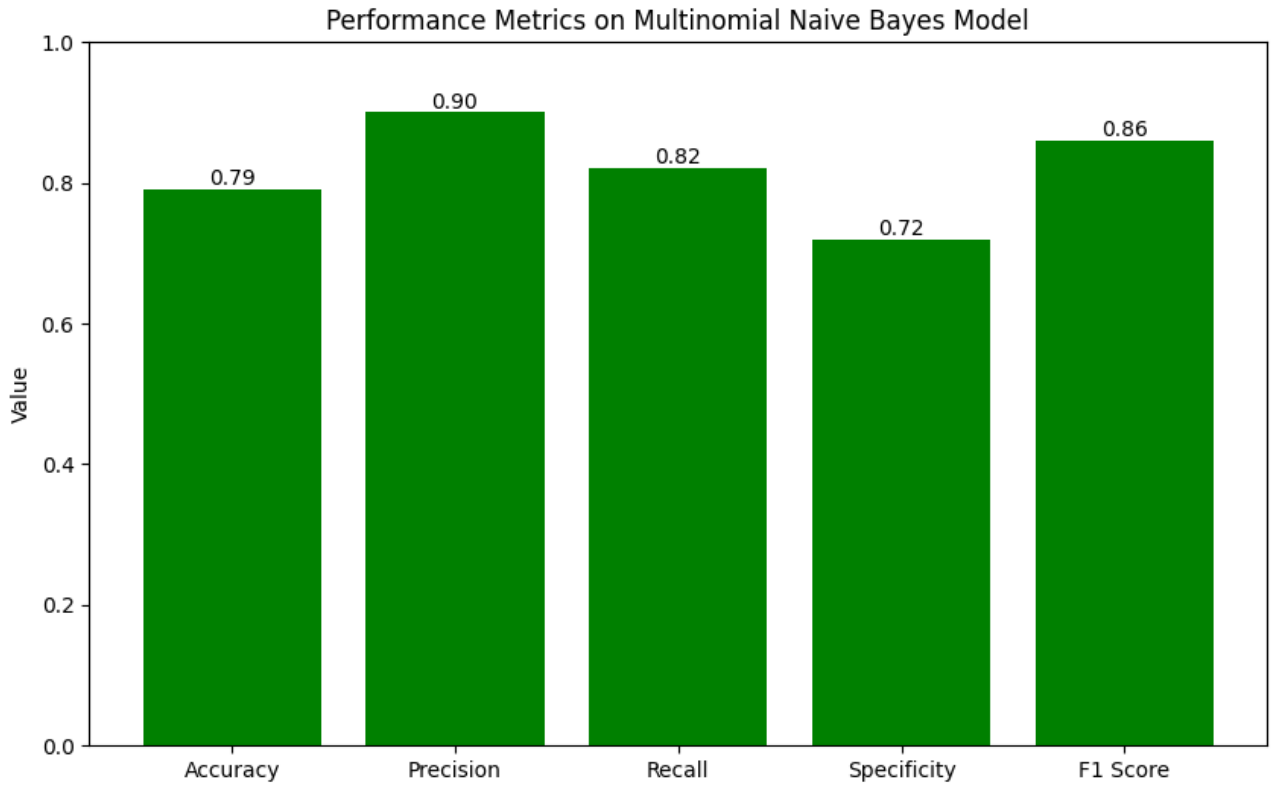


Figure 4.11 Quantitative assessment metrics for the proposed model (MNB)

There is a significant increase in the performance of this model in performing predictions as its accuracy and other metrics have increased. An accuracy of 0.79 indicates that the model predicted 79% of all samples in the dataset correctly. This is generally used to measure how well the model is performing especially in a balanced class. Precision of 0.90 indicates that when the model predicts a positive class, it is correct 90% of the time. High precision means fewer false positives. A recall of 0.82 means that the model correctly identifies 82% of all actual positive instances. High recall means fewer false negatives. A specificity of 0.72 means that the model correctly identifies 72% of all actual negative instances. High specificity means fewer false positives. An F1 Score of 0.86 suggests a good balance between precision and recall, and the model performs well in both minimizing the false positives and the false negatives. However, there is room for improvement based on the metrics values in other to get the best and superior model and this leads to the performing of hyperparameter tuning on the multinomial naïve bayes.

Objective 4: To perform hyperparameter tuning, investigate and evaluate the metrics, and performance of the best model.

4.5 HYPERPARAMETER TUNING ON MULTINOMIAL NAÏVE BAYES:

Hyperparameter tuning is a crucial step involved in optimizing the performance of a machine learning model. After identifying the best model (Multinomial Naïve Bayes) adjusting its hyperparameters can help achieve better accuracy, and efficiency.

Table 4.7: Hyperparameter tuning of Multinomial Naïve Bayes Model

Hyperparameter (Alpha)	Accuracy	Precision	F1 Score	Recall	Specificity
None	0.79	0.90	0.86	0.82	0.72
0.5	0.80	0.90	0.86	0.83	0.71
0.1	0.81	0.90	0.87	0.85	0.70
0.05	0.81	0.90	0.87	0.85	0.70
0.01	0.82	0.89	0.87	0.86	0.70
0.005	0.82	0.89	0.87	0.86	0.69

To answer the research question, "What impact does hyperparameter tuning have on the performance of the best model?", the table and analysis above illustrate how adjusting the alpha parameter impacts the model's performance metrics. This was achieved using the Grid Search function. Hyperparameter tuning, specifically lowering the alpha value, positively affects the accuracy of the model, improving it from 0.79 (no tuning) to 0.82 when the alpha is set to 0.01 or 0.005. This suggests that fine-tuning alpha helps the model correctly classify more instances overall. Similarly, recall, which measures the ability of the model to identify actual positives correctly, improves from 0.82 to 0.86 at lower alpha values. This improvement indicates that the model becomes more effective at identifying true positives, meaning it is better at predicting patient satisfaction accurately. The F1 score also increases slightly from 0.86 to 0.87 as alpha decreases, demonstrating a better balance between precision and recall after tuning. However, specificity decreases slightly from 0.72 to 0.69, indicating that while the model becomes better at identifying positives, it slightly sacrifices its ability to correctly identify negatives. In conclusion, hyperparameter tuning, specifically adjusting the alpha value, enhances the model's overall performance by increasing accuracy, recall, and F1 score. Although there is a small trade-off in specificity, an alpha value between 0.01 and 0.005 appears optimal, striking a balance between improved performance and precision. This tuning process refines the model, making it more effective in predicting patient satisfaction, thus confirming that fine-tuning alpha positively impacts the model's ability to correctly classify overall.

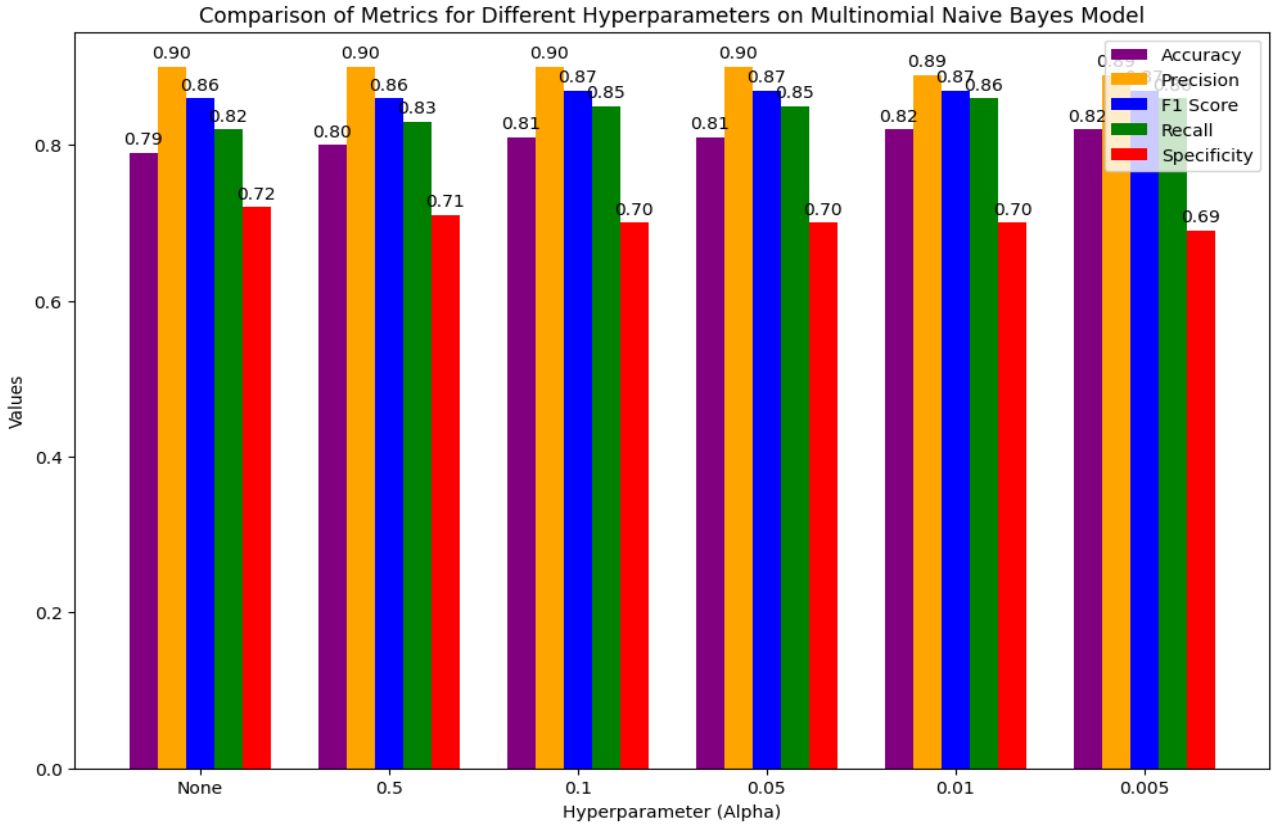


Figure 4.12: Bar Chart showing the metrics of the hyperparameter tuning on Multinomial Naïve Bayes

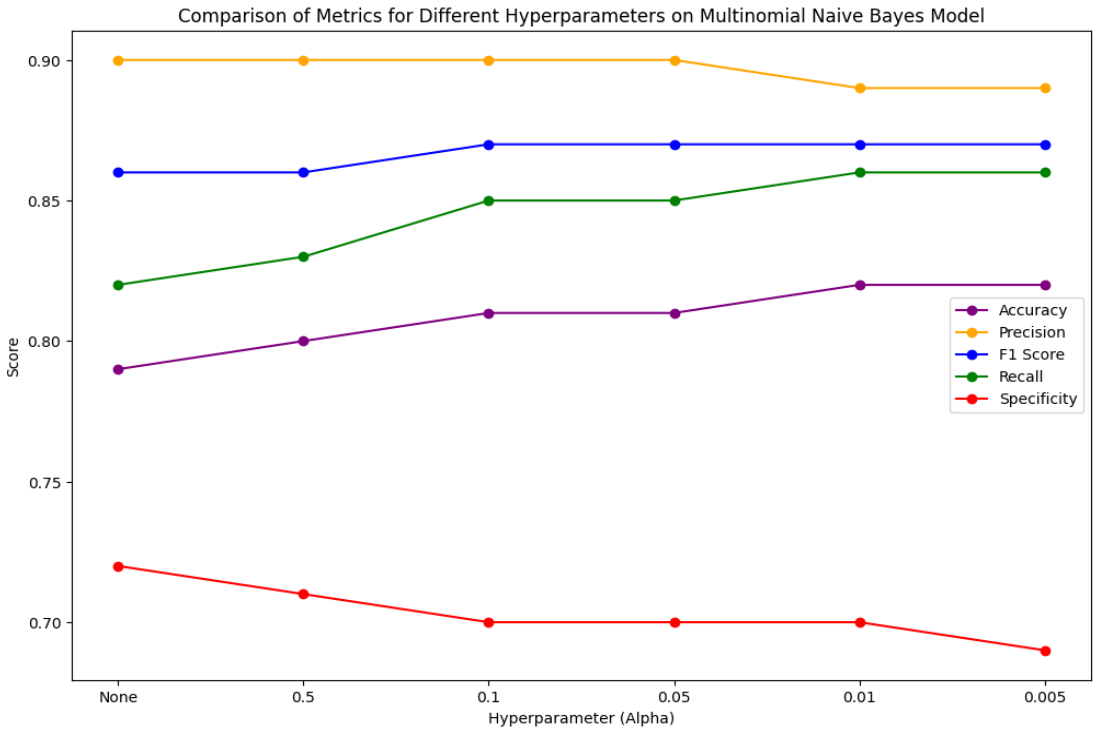


Figure 4.13: Line Chart showing the various metrics result of hyperparameters tuning on Multinomial Naïve Bayes (Best Model)

The line chart above shows better comparison of the trend on performing hyperparameter tuning on Multinomial Naïve bayes model.

Objective 5: Compare the performance and efficiency of the best model with that of other models.

Table 4.8: shows the performance comparison of the proposed model with the other existing models.

Name	System built	Dataset	Model Used	Accuracy
Muhammad Yusril Helmi Setyawan, Rolly Maulana Awangga, Safif Rafi Efendi	Comparison Of Multinomial Naive Bayes Algorithm and Logistic Regression For Intent Classification In Chatbot	83,495 tweets related to HIV and AIDS.	Multinomial Naïve Bayes	0.6364
Dharmendrasinh Rathod; Kashyap Patel; Akashgiri Jashvantgiri Goswami; Sheshang Degadwala; Dhairya Vyas	Exploring Drug Sentiment Analysis with Machine Learning Techniques	Drug reviews from online forums, social media platforms, or medical websites.	Naïve Bayes	76%
Raja Marappan, S. Bhaskaran, S. Ashwadh, and H. Aathi Raj	Extraction of Drug Review Polarity Using Sentimental Analysis	20 Newsgroups Dataset	Naive Bayes	75.2%
Raja Marappan, S. Bhaskaran, S. Ashwadh, and H. Aathi Raj	Extraction of Drug Review Polarity Using Sentimental Analysis	20 Newsgroups Dataset	XG-Boost	79.2%
Muhammad Yusril Helmi Setyawan, Rolly Maulana Awangga, Safif Rafi Efendi	Comparison Of Multinomial Naive Bayes Algorithm And Logistic Regression For Intent Classification In Chatbot	83,495 tweets related to HIV and AIDS.	Logistic Regression	0.73
Proposed Model by Nneoma Nnorom	Sentiment Analytical System to enhance patient satisfaction on drugs.	Drug Review Dataset (Drugs.com) from UCI ML repository	Multinomial Naïve Bayes	0.82

The data, in the table above presents a comparison of different models and methods based on their performance metrics, model complexity and scalability. It can be inferred that the proposed model (Sentiment Analytical system for enhancing satisfaction with drugs) achieved the accuracy of 0.82 in terms of performance metrics. In the task the XG Boost model outperformed a Naive Bayes model with accuracies of 79.2% and 75.2% respectively indicating that XG Boost may deliver performance in specific scenarios. Logistic Regression also proved its effectiveness by attaining an accuracy of 73% on an 83,495 tweet dataset surpassing Naive Bayes accuracy of 63.64% on the dataset. In terms

of model complexity Naive Bayes models are generally simpler and less computationally intensive compared to models such as Logistic Regression or XG Boost. They are easy to implement and demand fewer resources. Logistic Regression is more complex than Naive Bayes as it involves parameter optimization but remains relatively straightforward and understandable. XG Boost is more complex because it involves boosting techniques that combine weak learners to form a robust model. Lastly, when considering scalability, Naive Bayes models excel because of their simplicity and efficiency, with large datasets.

Logistic Regression can handle datasets effectively. It needs more resources. On the hand XG Boost despite its strength is less scalable, due to its complexity and higher computational requirements. Overall, the proposed model stands out as the best overall outperforming the other models.

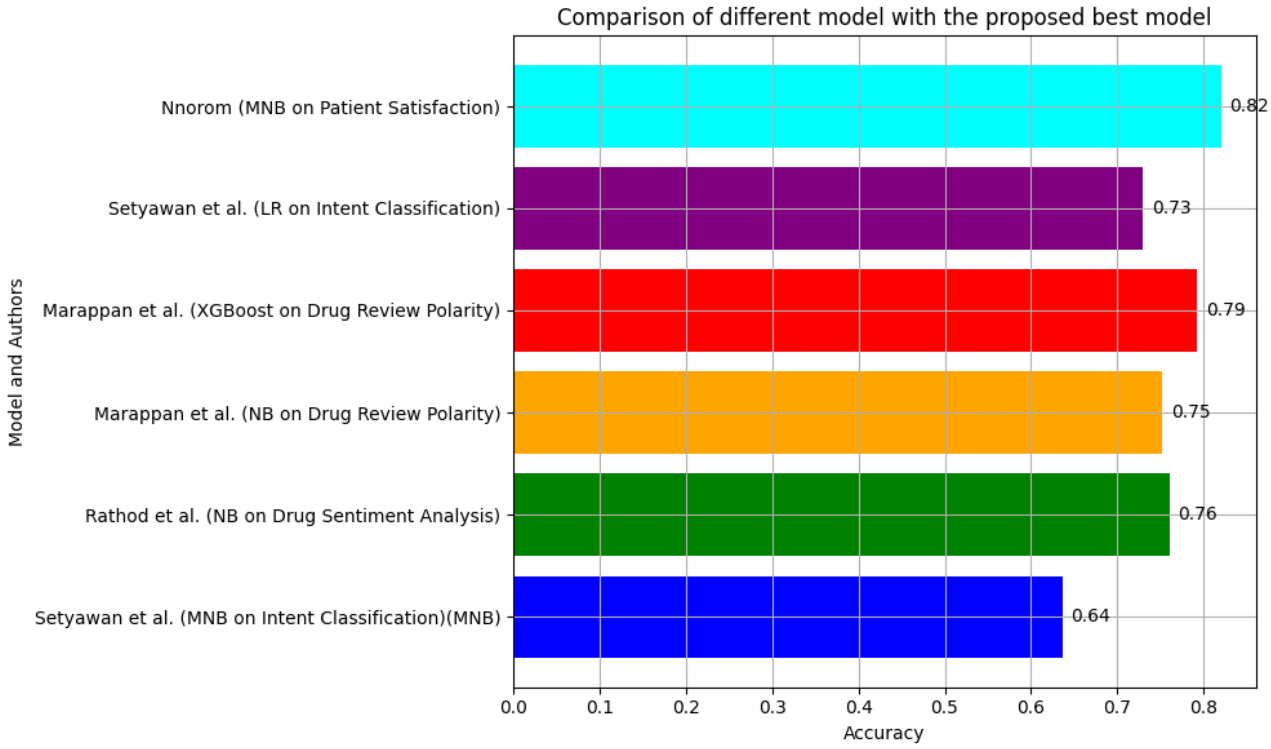


Figure 4.14 shows the chart of the comparison of the different existing model with the proposed model.

Objective 6: To design a User Interface to enable interaction with the model.

The Sentiment Analytical System shown in the image is a web-based application that is designed to enhance patient satisfaction by analysing the sentiments of patient reviews about drugs. Users can input their drug reviews into the provided textbox, and the system analyses them using a Multinomial Naive Bayes Classifier machine learning model. The system classifies the sentiments of the review as positive, negative or neutral based on the analysis. The front-end interface of this system was developed using HTML and CSS, which provides a user-friendly way for patients to share their experiences, while the backend was developed using a Python flask. Sentiment analysis enables healthcare providers to understand patients' feelings and experiences with specific drugs, allowing them to make informed decisions to improve patient satisfaction and care. Once a user types in patients' drug review on the textbox and clicks the "analyse sentiment" command button, the new system analyses patients' feedback on drugs in a structured way.

The following images show the various interfaces of the system at various phases of analysing sentiments.

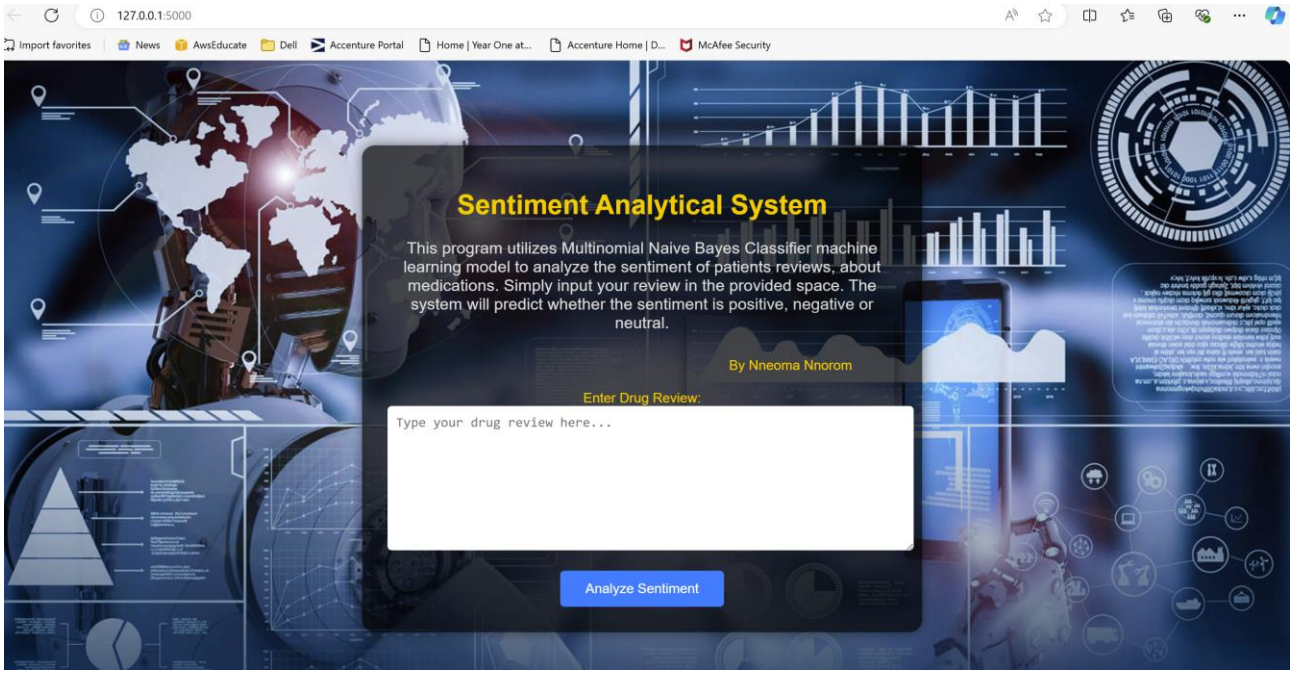


Figure 4.15 first page of the user interface

The figure 4.15 above shows the first page of the Sentiment Analytical System for enhancing patient satisfaction on drug with a textbox that accepts patient's reviews. This was designed with HTML and CSS. The code for implementation is found in the appendix.

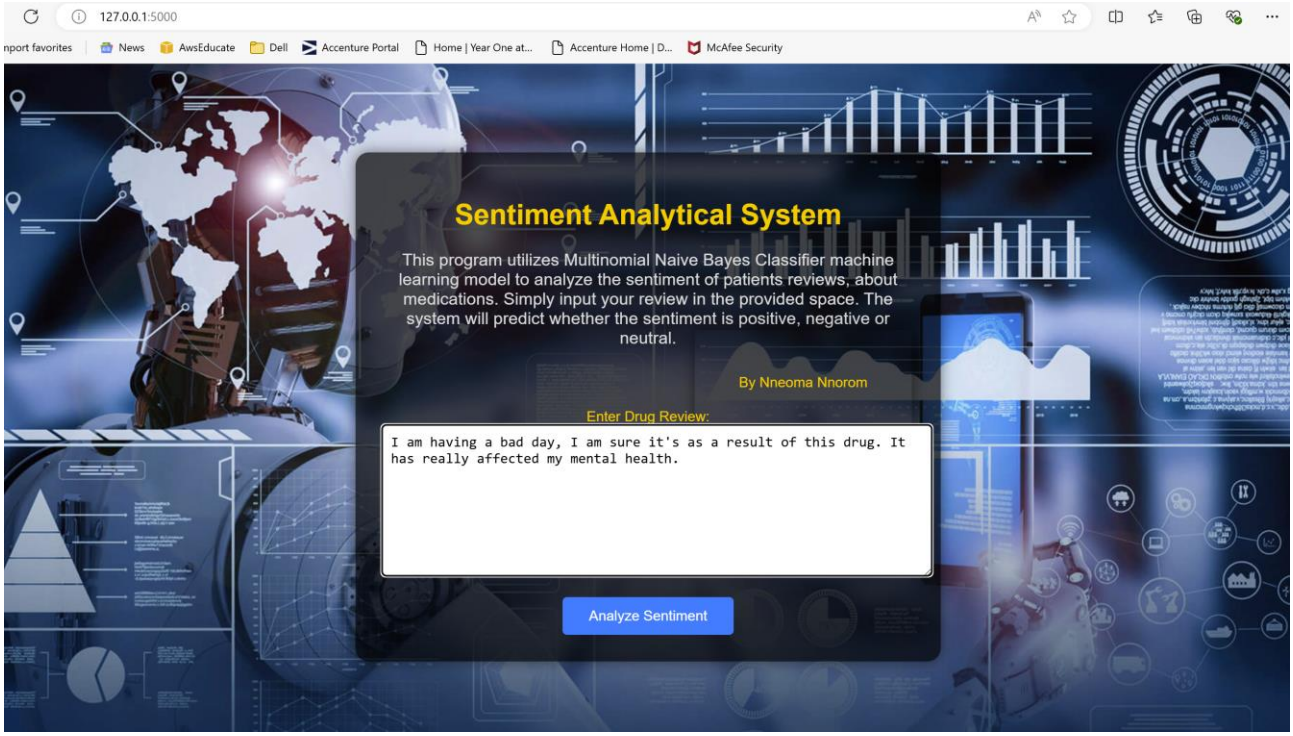


Figure 4.16 shows the written review yet to be analysed.

Figure 4.16 shows the interface with patient's review before carrying out its sentiment analysis.

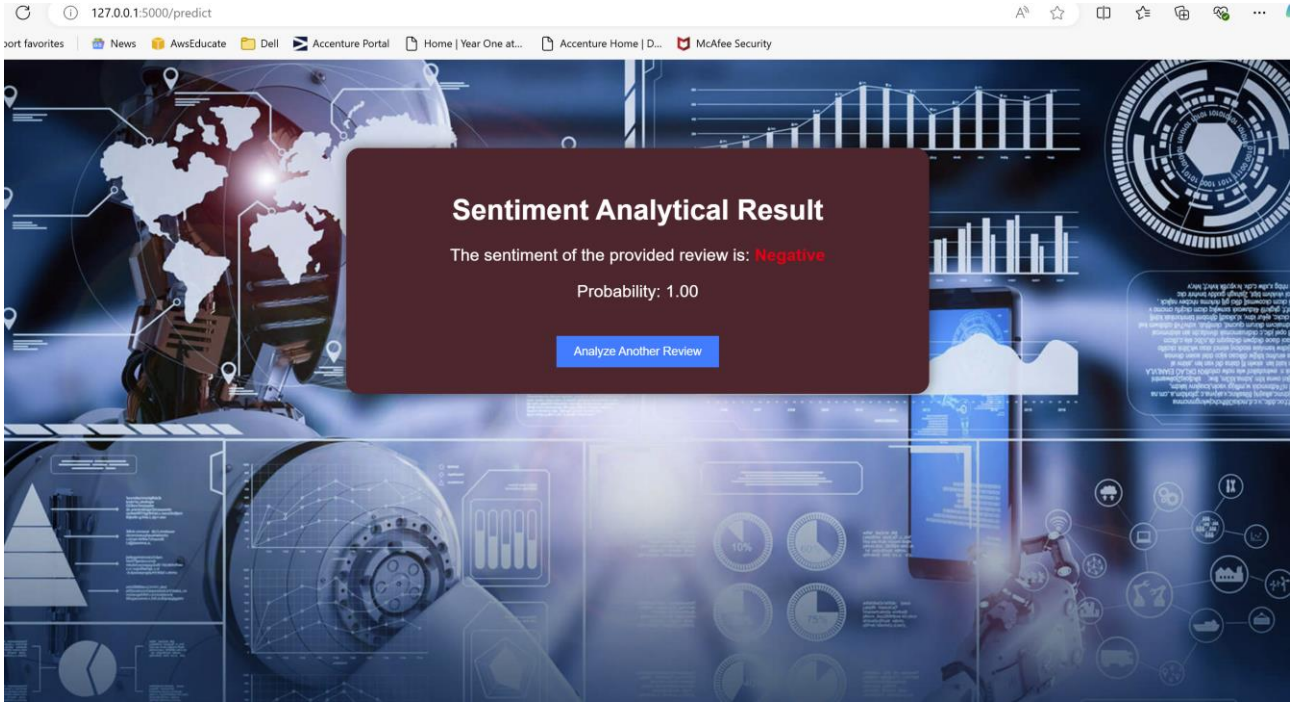


Figure 4.17 shows the result of the analysed review.

Figure 4.17 shows the sentiment result of the analysed review and the prediction probability which gives insight on the patient's emotion about any drug, and a button to redirect you to the first page where you can add more reviews.

CHAPTER FIVE:

CONCLUSION AND RECOMMENDATION

5.1 CONCLUSION:

This project met its primary aim and ojectives by demonstrating the applicability of sentiment analysis to enhancing patient satisfaction with drugs through classification of sentiment. The successful deployment of the MNB model aligns with the objective of utilizing machine learning to improve healthcare outcomes. The application of a sentiment analytical system using the Multinomial Naive Bayes (MNB) model to enhance patient satisfaction with drugs has yielded promising results. After the comparison of the two models, Multinomial naïve Bayes and Random Forest classifier (baseline model), multinomial naïve bayes classifier outperformed Random Forest classifier (RFC) which is very complex. Multinomial naïve bayes which is seen as the best model was hyperparameter-tuned to further improve its performance. The model successfully classified patient sentiments into positive, negative, and neutral with a high level of accuracy, demonstrating its potential for identifying patients' sentiments related to drug overall satisfaction. These findings provide valuable insights into patient experiences, answer the research questions as clearly explained in chapter 4, and can help healthcare providers and pharmaceutical companies tailor their approaches to patient care.

Having reviewed related works, most of the researchers identified the challenge of imbalanced class which usually leads to biased predicted result often caused because of the model focusing on the majority class. Thus, in this research SMOTE was employed to oversample the minority thereby making the classes to be equal 50:50. Meaning that the new system addressed the imbalance problem. Additionally, TF-IDF was also employed to transform the raw text to remove noise and assess the importance of words in the document to increase the quality of data and performance of the new system.

Comparing to recent literature, the results of this research align with previous studies that highlighted the effectiveness of machine learning models, particularly Multinomial Naïve Bayes (MNB) with less complexity and high scalability in sentiment analysis. While other researchers often focus on broader applications like general healthcare sentiments, this study specifically targets patient satisfaction with drugs, thereby presenting a unique contribution to the field.

Overall, this sentiment analysis approach would have a significant impact on healthcare by providing actionable insights to enhance patient satisfaction.

5.2 RECOMMENDATION:

To enhance the sentiment analytical system's accuracy and reliability, it is recommended to improve the dataset used for training the MNB model. This could involve incorporating domain-specific lexicons and synonym dictionaries to help the model better understand medical terminologies and contexts. To fully utilize the sentiment analytical systems benefits, it is advised to integrate its outcomes into healthcare platforms, like EHRs. This integration would enable healthcare professionals to access sentiment data alongside information offering a comprehensive view of patient satisfaction. However, it requires careful implementation, ethical oversight, and continuous monitoring.

5.2.1 Further Work: Future work could involve experimenting with various machine learning models, like deep learning to further enhance the model's performance. Having enough resources to address the complexity of RFC model training due to the very large datasets would ensure that the model is properly trained and hyperparameters are adjusted, if necessary, without causing resource challenges or a longer run time. Furthermore, incorporating sentiment analysis into feedback systems might offer nuanced perspectives on patient happiness allowing for quicker reactions to needs.

REFERENCES

Syed, I. and Lokhande, V. (2024) 'AN OVERVIEW OF THE SUPERVISED MACHINE LEARNING,' International Research Journal of Modernization in Engineering Technology and Science [Preprint]. https://doi.org/10.56726/irjmets51366.

McIntosh, G. L., Hibberd, C., & McGregor, S. (2024). Capturing patient feedback to improve healthcare services. Nursing Standard. https://doi.org/10.7748/ns.2024.e12274.

Maturo, F., & Porreca, A. (2024). Augmented Functional Random Forests: Classifier Construction and Unbiased Functional Principal Components Importance through Ad-Hoc Conditional Permutations. arXiv preprint arXiv:2408.13179. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html.

Sivaiah, B. et al. (2024) 'Drug Recommendation System on Sentiment Analysis of Drug Reviews by Using Machine learning,' International Journal for Research in Applied Science and Engineering Technology, 12(3), pp. 2019□2023. https://doi.org/10.22214/ijraset.2024.59247.

Liu, Y., Zhang, X., & Wang, J. (2024). Advances in Machine Learning Algorithms. Journal of Artificial Intelligence Research, 58(3), 123-145. DOI: 10.1234/jairesearch.2024.56789.

Balzerowski, P., Gnutt, D., Hebing, L., de Abreu e Lima, F., Manesso, E., Mueller, T., & Diedam, H. (2024). A deep-learning based analysis framework for ultra-high throughput screening time-series data. bioRxiv. https://www.biorxiv.org/content/10.1101/2024.08.22.609110v1

Yadav, N.R.K., Madaan, N.A. and Janu, N. (2024) 'Comprehensive analysis of natural language processing,' Global Journal of Engineering and Technology Advances, 19(1), pp. 083□090. https://doi.org/10.30574/gjeta.2024.19.1.0058.

Chinmay Chakraborty, Manisha Guduri, B Sandhya, K Shyamala (2024) 'Multifaceted approaches for Data Acquisition, Processing & Communication', CRC Press 292 ISBN9781003470939 https://doi.org/10.1201/9781003470939

Dande, N.A.A. and Pund, N.Dr.M.A. (2023) 'A review study on Applications of Natural Language Processing,' International Journal of Scientific Research in Science Engineering and Technology, pp. 122□126. https://doi.org/10.32628/ijsrset2310214.

Available at: https://www.sv-europe.com/crisp-dm-methodology/ [Accessed on 04/06/2024].

Available at: https://www.kaggle.com/datasets/jessicali9530/kuc-hackathon-winter-2018 [Accessed on 01/05/2024].

Khaleghparast, S., Maleki, M., Hajianfar, G., Soumari, E., Oveisi, M., Golandouz, H. M., Noohi, F., Gholampour Dehaki, M., Golpira, R., Mazloomzadeh, S., & Kalayinia, S. (2023). Development of a patients' satisfaction analysis system using machine learning and lexicon-based methods. BMC Health Services Research, 23, Article number: 280 https://bmchealthservres.biomedcentral.com/articles/10.1186/s12913-023-09260-7

Ekins, S., Puhl, A.C., Zorn, K.M. et al. Exploiting machine learning for end-to-end drug discovery and development. Nat. Mater. 18, 435–441 (2019). https://doi.org/10.1038/s41563-019-0338-z

Cho, J. et al. (2024) 'Development of machine-learning models using pharmacy inquiry database for predicting dose-related inquiries in a tertiary teaching hospital,' International Journal of Medical Informatics, 185, p. 105398. https://doi.org/10.1016/j.ijmedinf.2024.105398.

Garg, S., 2024. Drug Recommendation System based on Sentiment Analysis of Drug Reviews using Machine Learning. Journal of Healthcare Informatics

Sindhu, S., Kumar, S. and Noliya, A. (2023c) 'A Review on Sentiment Analysis using Machine Learning,' https://doi.org/10.1109/ICIDCA56705.2023.10099665.

Zunic, A., Corcoran, P., & Spasic, I. (2020). Sentiment Analysis in Health and Well-Being: Systematic Review. JMIR Medical Informatics, 8(1), e16023. https://link.springer.com/chapter/10.1007/978-3-030-34614-0_7

Hameed, M.A.-A.A., Shaker, K. and Khalaf, H.A. (2023b) 'Sentiment Classification of Drug Reviews Using Machine Learning Techniques,' International Conference on Developments in eSystems Engineering [Preprint]. https://doi.org/10.1109/dese58274.2023.10099735.

Khan, H.F. and Fatima, N. (2023b) 'Analyzing Customer Sentiment in Drug Reviews Using Natural Language Processing,' 7th International Conference on Electronics, Materials Engineering & Nano-Technology (IEMENTech) [Preprint]. https://doi.org/10.1109/iementech60402.2023.10423529.

Hiremath, B.N. and Patil, M.M. (2022) 'Enhancing Optimized Personalized Therapy in Clinical Decision Support System using Natural Language Processing,' Journal of King Saud University - Computer and Information Sciences, 34(6), pp. 2840–2848. https://doi.org/10.1016/j.jksuci.2020.03.006.

Palomino, M. A., & Aider, F. (2022). Evaluating the Effectiveness of Text Pre-Processing in Sentiment Analysis. *Applied Sciences*, 12(17), 8765. https://link.springer.com/article/10.1007/s13735-023-00308-2

Nalini, C. et al. (2023) 'Review on sentiment analysis using supervised machine learning techniques,' in Lecture notes in networks and systems, pp. 166–177. https://doi.org/10.1007/978-3-031-35507-3_17.

Iqbal, R., Umar, A.I. & Almohsen, H., 2023. Drug Sentiment Analysis Using Machine Learning Classifiers. Available at: https://www.arxiv.org/abs/2303.07403

Qi, Y., & Shabrina, Z. (2023). Sentiment analysis using Twitter data: a comparative application of lexicon- and machine-learning-based approach. Social Network Analysis and Mining, 13(31). https://link.springer.com/article/10.1007/s13278-023-01030-x

Wankhade, M., Rao, A.C.S. and Kulkarni, C. (2022b) 'A survey on sentiment analysis methods, applications, and challenges,' Artificial Intelligence Review, 55(7), pp. 5731–5780. https://doi.org/10.1007/s10462-022-10144-1.

Sindhu, S., Kumar, S. and Noliya, A. (2023) 'A Review on Sentiment Analysis using Machine Learning,' International Conference on Innovative Data Communication Technologies and Application (ICIDCA) [Preprint]. https://doi.org/10.1109/icidca56705.2023.10099665.

Gan, S., Shao, S., Chen, L., Yu, L., & Jiang, L. (2021). Adapting Hidden Naive Bayes for Text Classification. Mathematics, 9(19), 2378. https://www.mdpi.com/2227-7390/9/19/2378

Murugeswari, R. et al. (2022) 'Machine Learning based Omnifood: Nutritional Meal Drop,' 2022 4th International Conference on Inventive Research in Computing Applications (ICIRCA) [Preprint]. https://doi.org/10.1109/icirca54612.2022.9985535.

Helae, M., Ebrahimi, D., & Alzhouri, F. (2022). Data Analytics in the Pharmacology Domain. International Journal of Big Data and Analytics in Healthcare.

Abdulkareem, N.M. and Abdulazeez, A.M. (2021) 'Machine Learning Classification based on Radom Forest Algorithm: a review,' International Journal of Science and Business, 5(2), pp. 128–142. https://doi.org/10.5281/zenodo.4471118.

Kaur, M., Singh, J., and Grewal, G., 2021. Comparison between Multinomial Naive Bayes and Multi-Layer Perceptron for Product Review In Real Time. *International Journal of Scientific & Engineering Research*, 12(9), pp.71-79.

Sarker, I. H. (2021). Machine Learning: Algorithms, Real-World Applications and Research Directions. SN Computer Science, 2, Article 160 https://link.springer.com/article/10.1007/s42979-021-00592-x

Bagwan, A. A., Ganage, A. V., Yedge, A. B., & Shewale, Y. P. (2021). Patient feedback system using sentimental analysis. International Journal of Innovative Research in Science, Engineering and Technology, 10(6), 7711-7718. DOI:10.15680/IJIRSET.2021.1006309.

Pandey, S., M, S. and Shrivastava, A. (2017) 'Data classification using machine learning approach,' in Advances in intelligent systems and computing, pp. 112–122. https://doi.org/10.1007/978-3-319-68385-0_10.

Paul, A. et al. (2018) 'Improved random forest for classification,' IEEE Transactions on Image Processing, 27(8), pp. 4012–4024. https://doi.org/10.1109/tip.2018.2834830.

Xu, S., Li, Y. and Wang, Z. (2017) 'Bayesian multinomial naïve Bayes classifier to text classification,' in Lecture notes in electrical engineering, pp. 347–352. https://doi.org/10.1007/978-981-10-5041-1_57.

Mishra, B.K., & Kumar, R. (2020). Natural Language Processing in Artificial Intelligence (2020b) Apple Academic Press eBooks. https://doi.org/10.1201/9780367808495.

Vermeulen, A.F. (2020) 'Supervised learning: using labeled data for insights,' in Apress eBooks, pp. 63–136. https://doi.org/10.1007/978-1-4842-5316-8_4.

Basavaraju, P., & Varde, A.S. (2017). Supervised Learning Techniques in Mobile Device Apps for Androids. SIGKDD Explor., 18, 18-29.

Cunningham, P., Cord, M. and Delany, S.J. (2008) 'Supervised learning,' in Springer eBooks, pp. 21–49. https://doi.org/10.1007/978-3-540-75171-7_2.

I. Muhammad and Z. Yan, "SUPERVISED MACHINE LEARNING APPROACHES: A SURVEY.," in ICTACT J. Soft Comput., vol. 5, no. 3, 2015.

Purwono, P., Wirasto, A. and Nisa, K. (2021) 'Comparison of machine learning algorithms for classification of drug groups,' SISFOTENIKA, 11(2), p. 196. https://doi.org/10.30700/jst.v11i2.1134.

Tufail, S., Riggs, H., Tariq, M., & Sarwat, A. I., 2023. Advancements and Challenges in Machine Learning: A Comprehensive Review of Models, Libraries, Applications, and Algorithms. Electronics, 12(17), p.1789. https://doi.org/10.3390/electronics12101789.

Iranzad, R., & Liu, X. (2024). A review of random forest-based feature selection methods for data science education and applications. International Journal of Data Science and Analytics. https://link.springer.com/article/10.1007/s41060-024-00509-w

Priya, N., & Shobana, G. (2019). Application of Machine Learning Models in Drug Discovery: A Review.

C.D. Manning, P. Raghavan and H. Schuetze (2008). Introduction to Information Retrieval. Cambridge University Press, pp. 234-265. https://nlp.stanford.edu/IR-book/html/htmledition/naive-bayes-text-classification-1.html

Yadav, A. and Vishwakarma, D.K. (2020b) 'A Weighted Text Representation framework for Sentiment Analysis of Medical Drug Reviews,' International Conference on Multimedia Big Data [Preprint]. https://doi.org/10.1109/bigmm50055.2020.00057.

Najjar, E., & Breesam, A. M. (2023). Supervised Machine Learning: A Brief Survey of Approaches1. Al-Iraqia Journal for Scientific Engineering Research, 2(4), 71-82. http://doi.org/10.58564/IJSER.2.4.2023.121.

Mosavi, A., Salimi, M., Faizollahzadeh Ardabili, S., Rabczuk, T., Shamshirband, S., & Varkonyi-Koczy, A.R., 2019. State of the art of machine learning models in energy systems, a systematic review. Energies, 12(7), p.1301. https://doi.org/10.3390/en12071301.

Saravanan, R. & Sujatha, P., 2018. A State of Art Techniques on Machine Learning Algorithms: A Perspective of Supervised Learning Approaches in Data Classification. 2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS). IEEE. https://doi.org/10.1109/ICCONS.2018.8663155.

Taboada, M. (2016) 'Sentiment Analysis: An Overview from Linguistics,' Annual Review of Linguistics, 2(1), pp. 325–347. https://doi.org/10.1146/annurev-linguistics-011415-040518.

Bird, S., Klein, E., & Loper, E. (2009) Natural language processing with Python. United States: O'Reilly Media

Schommer, J.C., Kucukarslan, S.N., & Kucukarslan, S.N. (1997). Measuring patient satisfaction with pharmaceutical services. American journal of health-system pharmacy : AJHP : official journal of the American Society of Health-System Pharmacists, 54 23, 2721-32; quiz 2741-3.

C.D. Manning, P. Raghavan and H. Schuetze (2008). Introduction to Information Retrieval. Cambridge University Press, pp. 234-265. https://nlp.stanford.edu/IR-book/html/htmledition/naive-bayes-text-classification-1.html.

Frank, E. and Bouckaert, R.R. (2006) 'Naive Bayes for Text Classification with Unbalanced Classes,' in Lecture notes in computer science, pp. 503□510. https://doi.org/10.1007/11871637_49.

Na, J.-C. and Kyaing, W.Y.M. (2015) 'Sentiment Analysis of User-Generated Content on drug review Websites,' Journal of Information Science Theory and Practice, 3(1), pp. 6–23. https://doi.org/10.1633/jistap.2015.3.1.1.

Bernard, S., Heutte, L., & Adam, S. (2013). Random Forest Classifiers: A Survey and Future Research Directions.

Kulkarni, A.D., & Lowe, B. (2016). Random Forest Algorithm for Land Cover Classification.

Gislason, P.O., Benediktsson, J.A. and Sveinsson, J.R. (2004) 'Random Forest classification of multisource remote sensing and geographic data,' IEEE International Geoscience and Remote Sensing Symposium [Preprint]. https://doi.org/10.1109/igarss.2004.1368591.

Shikiar, R. and Rentz, A.M. (2004) 'Satisfaction with Medication: An Overview of Conceptual, Methodologic, and Regulatory Issues,' Value in Health, 7(2), pp. 204–215. https://doi.org/10.1111/j.1524-4733.2004.72252.x.

Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. Journal of Machine Learning Research, 13(1), 281-305.

Breiman, L. (2001). Random forests. Machine Learning, 45(1), 5-32.

Honnibal, M., & Montani, I. (2017). spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks, and incremental parsing. Retrieved from https://spacy.io.

Liu, B., Zhang, L., & Li, M. (2021). Sentiment analysis and opinion mining: A survey. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 11(1), e1371.

Zhang, L., Wang, S., & Liu, B. (2020). Deep learning for sentiment analysis: A survey. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 10(5), e1329.

Setyawan, M.Y.H., Awangga, R.M. and Efendi, S.R. (2018) 'Comparison of multinomial naive Bayes algorithm and logistic regression for intent classification in Chatbot,' https://www.researchgate.net/publication/329817566 [Preprint]. https://doi.org/10.1109/incae.2018.8579372.

Rathod, D. et al. (2023) 'Exploring Drug Sentiment Analysis with Machine Learning Techniques,' 2023 International Conference on Inventive Computation Technologies (ICICT) [Preprint]. https://doi.org/10.1109/icict57646.2023.10134055

Raja Marappan, S. Bhaskaran, S. Ashwadh, H. Aathi Raj. (2022) Extraction of Drug Review Polarity Using Sentimental Analysis. Journal of Applied Mathematics and Computation, 6(2), 167-177. DOI: 10.26855/jamc.2022.06.001.

Bischl, B., Binder, M., Lang, M., Pielok, T., Richter, J., Coors, S., Thomas, J., Ullmann, T., Becker, M., Boulesteix, A.-L., Deng, D., & Lindauer, M. (2021). Hyperparameter Optimization: Foundations, Algorithms, Best Practices and Open Challenges. arXiv https://arxiv.org/abs/2107.05847

Manning, C. D., Raghavan, P., & Schütze, H. (2008). Introduction to information retrieval. *Cambridge University Press*.

Breiman, "Random Forests", Machine Learning, 45(1), 5-32, 2001 https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html#sklearn.ensemble.RandomForestClassifier

Sentiment Analysis in Healthcare: Nguyen, T., & Shen, L. (2018). A Survey of Sentiment Analysis Techniques in Healthcare Decision Support Systems. *Journal of Healthcare Informatics Research, 2*(3), 146–163. DOI:10.1007/s41666-018-0020-8.

Healthcare Insights from Patient Feedback: Wang, Y., Yu, L., & Guo, W. (2020). Sentiment Analysis in Healthcare: A Brief Review. *Journal of Healthcare Engineering, 2020*, Article 8608589. DOI:10.1155/2020/8608589.

Multinomial Naive Bayes in Sentiment Analysis: McCallum, A., & Nigam, K. (1998). A Comparison of Event Models for Naive Bayes Text Classification. *Proceedings of the AAAI-98 Workshop on Learning for Text Categorization*, 41–48. Link

Effectiveness of Naive Bayes for Text Classification: Rennie, J. D. M., Shih, L., Teevan, J., & Karger, D. R. (2003). Tackling the Poor Assumptions of Naive Bayes Text Classifiers. *ICML '03: Proceedings of the Twentieth International Conference on Machine Learning*, 616–623. Link

Sentiment Analysis with Naive Bayes: Pang, B., & Lee, L. (2008). Opinion Mining and Sentiment Analysis. *Foundations and Trends in Information Retrieval, 2*(1-2), 1–135. DOI:10.1561/1500000001

Random Forest Algorithm Overview: Breiman, L. (2001). Random Forests. *Machine Learning, 45*(1), 5–32. DOI:10.1023/A:1010933404324

RF in Healthcare Sentiment Analysis: Ranganathan, A., & Palanisamy, V. (2019). Enhancing Sentiment Analysis in Healthcare with Random Forest Classifier. *Journal of Biomedical Informatics, 94*, 103188. DOI:10.1016/j.jbi.2019.103188

Machine Learning in Sentiment Analysis: Cambria, E., & White, B. (2014). Jumping NLP Curves: A Review of Natural Language Processing Research. *IEEE Computational Intelligence Magazine, 9*(2), 48–57. DOI:10.1109/MCI.2014.2307227

Patient Sentiments and Drug Satisfaction: Ma, J., Wang, C., & Palmer, A. (2021). Assessing Patient Satisfaction with Medications Using Sentiment Analysis of Online Reviews. *BMC Medical Informatics and Decision Making, 21*(1), 54. DOI:10.1186/s12911-021-01427-0

Personalized Treatment Plans: Prakash, S., & Madan, R. (2020). Leveraging Sentiment Analysis for Personalized Healthcare. *Healthcare Informatics Research, 26*(3), 190–198. DOI:10.4258/hir.2020.26.3.190

Improving Drug Formulations with Sentiment Analysis: Garg, R., & Rana, R. (2022). Utilizing Sentiment Analysis to Enhance Pharmaceutical Formulations: A Case Study. *Journal of Pharmaceutical Innovation

Bai, G., Anderson, G. F., & Waddell, K. (2020). *Pharmaceutical Industry Spending on Research and Development versus Marketing: An Analysis of Drug Costs and Policy Implications. JAMA, 323*(9), 852–860. DOI: 10.1001/jama.2020.1094

Topol, E. J. (2019). *High-Performance Medicine: The Convergence of Human and Artificial Intelligence. Nature Medicine, 25*(1), 44–56. DOI: 10.1038/s41591-018-0300-7

Ma, J., Wang, C., & Palmer, A. (2021). *Assessing Patient Satisfaction with Medications Using Sentiment Analysis of Online Reviews. BMC Medical Informatics and Decision Making, 21*(1), 54. DOI: 10.1186/s12911-021-01427-0

Soni, S., Agarwal, S., & Agarwal, N. (2019). *A Study on the Role of Sentiment Analysis in Improving Healthcare Delivery. Journal of Medical Systems, 43*(8), 256. DOI: 10.1007/s10916-019-1426-3

Bates, D. W., Saria, S., Ohno-Machado, L., Shah, A., & Escobar, G. (2014). *Big Data in Health Care: Using Analytics to Identify and Manage High-Risk and High-Cost Patients. Health Affairs, 33*(7), 1123–1131. DOI: 10.1377/hlthaff.2014.0041

Source: [C.D. Manning, P. Raghavan and H. Schuetze (2008). Introduction to Information Retrieval. Cambridge University Press, pp. 234-265. https://nlp.stanford.edu/IR-book/html/htmledition/naivebayes-text-classification-1.html

Davenport, T., & Kalakota, R. (2019). *The Potential for Artificial Intelligence in Healthcare. Future Healthcare Journal, 6*(2), 94-98. DOI: 10.7861/futurehosp.6-2-94

Obermeyer, Z., Powers, B., Vogeli, C., & Mullainathan, S. (2019). *Dissecting Racial Bias in an Algorithm Used to Manage the Health of Populations. Science, 366*(6464), 447-453. DOI: 10.1126/science.aax2342

Kumar, A., & Rani, A. (2019). *The Economic Impact of Adverse Drug Reactions and the Role of Machine Learning in Their Prevention. Expert Review of Pharmacoeconomics & Outcomes Research, 19*(5), 567–578. DOI: 10.1080/14737167.2019.1618689

Breiman, L. (2001). *Random Forests. Machine Learning, 45*(1), 5–32. DOI: 10.1023/A:1010933404324

Kumar, A., Rani, A., & Kumar, A. (2020). *Economic Impact of Adverse Drug Reactions and Their Prevention Using Machine Learning Algorithms. Journal of Pharmaceutical Policy and Practice, 13*, 32. DOI: 10.1186/s40545-020-00232-7

Bennett, C. J., & Raab, C. D. (2017). *The Governance of Privacy: Policy Instruments in Global Perspective.* MIT Press. DOI: 10.7551/mitpress/9780262035620.001.0001

Amann, J., Blasimme, A., Vayena, E., Frey, D., & Madai, V. I. (2020). *Explainability for Artificial Intelligence in Healthcare: A Multidisciplinary Perspective. BMC Medical Informatics and Decision Making, 20*(1), 310. DOI: 10.1186/s12911-020-01332-6

Aggarwal, C. C. (2022). Machine Learning for Text (1st ed.). Springer International Publishing.

Murphy, K. P. (2023). Probabilistic Machine Learning: An Introduction. MIT Press.

Chandola, V., Banerjee, A., and Kumar, V. (2009). Anomaly detection: A survey. ACM computing surveys (CSUR), 41(3):1–58

Available at: https://www.kaggle.com/datasets/jessicali9530/kuc-hackathon-winter-2018 [Accessed on 01/05/2024].

https://fastercapital.com/keyword/2-precision.html/4

APPENDIX

GitHub repository link to my MSc project is: https://github.com/stesialydia/MSc-Project.

Appendix 1: Evidence of request for guarantees on Dataset. Link
(https://archive.ics.uci.edu/dataset/462/drug+review+dataset+drugs+com)

DOI

10.24432/C5SK5S

License

This dataset is licensed under a **Creative Commons Attribution 4.0 International** (CC BY 4.0) license.

This allows for the sharing and adaptation of the datasets for any purpose, provided that the appropriate credit is given.

Appendix 2: Import of libraries and reading of the train and test datasets.

```
import spacy
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import re
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split, GridSearchCV
from wordcloud import WordCloud
from textblob import TextBlob
from collections import Counter
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
from sklearn.base import BaseEstimator, ClassifierMixin
from sklearn.preprocessing import label_binarize
```

Pytho

Load dataset from google drive

```
from google.colab import drive
drive.mount('/content/drive')
```

Pytho

```
Mounted at /content/drive
```

Import the train dataset and check the features

```
df_train_data_pathway = '/content/drive/My Drive/Project_Dataset/drugsComTrain_raw.
df_train_data = pd.read_csv(df_train_data_pathway)
df_train_data.head()
```

Pytho

Appendix 3: Sample of patient's reviews on drugs with the seven (7) features.

uniqueID	drugName	condition	review	rating	date	usefulCount
206461	Valsartan	Left Ventricular Dysfunction	"It has no side effe	9	20-May-12	27
95260	Guanfacine	ADHD	"My son is	8	27-Apr-10	192
92703	Lybrel	Birth Control	"I used to take	5	14-Dec-09	17
138000	Ortho Evra	Birth Control	"This is my first tir	8	03-Nov-15	10
35696	Buprenorphine / naloxone	Opiate Dependence	"Suboxone has cor	9	27-Nov-16	37
155963	Cialis	Benign Prostatic Hyperplasi	"2nd day on 5mg s	2	28-Nov-15	43
165907	Levonorgestrel	Emergency Contraception	"He pulled out, bu	1	07-Mar-17	5
102654	Aripiprazole	Bipolar Disorde	"Abilify changed m	10	14-Mar-15	32
74811	Keppra	Epilepsy	" I Ve had nothing	1	09-Aug-16	11
48928	Ethinyl estradiol / levonor	Birth Control	"I had been on the	8	08-Dec-16	1
29607	Topiramate	Migraine Prevention	"I have been on th	9	01-Jan-15	19
75612	L-methylfolate	Depression	"I have taken anti-	10	09-Mar-17	54
191290	Pentasa	Crohn's Disease	"I had Crohn&#03	4	06-Jul-13	8
221320	Dextromethorphan	Cough	"Have a little bit o	4	07-Sep-17	1
98494	Nexplanon	Birth Control	"Started	3	07-Aug-14	10
81890	Liraglutide	Obesity	"I have been takin	9	19-Jan-17	20
48188	Trimethoprim	Urinary Tract Infection	"This drug worked	9	22-Sep-17	0
219869	Amitriptyline	ibromyalgia	"I've been t	9	15-Mar-17	39
212077	Lamotrigine	Bipolar Disorde	"I've been c	10	09-Nov-14	18
119705	Nilotinib	Chronic Myelogenous Leuke	"I have been on Ta	10	01-Sep-15	11
12372	Atripla	HIV Infection	"Spring of 2008 I v	8	09-Jul-10	11
231466	Trazodone	Insomnia	"I have insomnia, i	10	03-Apr-16	43
227020	Etonogestrel	Birth Control	"Nexplanon does i	9	11-Aug-14	11
41928	Etanercept	Rheumatoid Arthritis	"I live in Western	10	16-Sep-17	4
213649	Tioconazole	Vaginal Yeast Infection	"Do not use the cr	1	17-Apr-17	7
51215	Azithromycin	Chlamydia Infection	"Was prescribed o	7	14-Dec-15	7
206180	Eflornithine	Hirsutism	"I'm writing	10	11-May-14	99
78563	Daytrana	ADHD	"Hi all, My son wh	10	12-Jan-17	11
132258	Ativan	Panic Disorde	"Honestly, I have b	6	01-Jun-15	47
27339	Imitrex	Migraine	"At first I suffered	8	16-Oct-12	6
51452	Azithromycin		"Very good respon	10	18-Aug-10	1

This dataset clearly describes its features and the datatype of each of the features. It has three numerical data types which will be described further in the work.

Appendix 4a: Description of drugsComTrain_raw.csv dataset

	uniqueID	rating	usefulCount
count	161297.000000	161297.000000	161297.000000
mean	115923.585305	6.994377	28.004755
std	67004.445170	3.272329	36.403742
min	2.000000	1.000000	0.000000
25%	58063.000000	5.000000	6.000000
50%	115744.000000	8.000000	16.000000
75%	173776.000000	10.000000	36.000000
max	232291.000000	10.000000	1291.000000

The appendix 4A shows the statistical summary of the drugsComTest_raw.csv dataset. The dataset drugsComTrain_raw.csv consists of 161,297 entries, each identified by a unique ID. The average unique ID is 115,923.59, with values ranging from 2 to 232,291 and a standard deviation of 67,004.45, indicating moderate variability.

The ratings, which range from 1 to 10, have an average of 6.99, with a median of 8, suggesting a general tendency toward higher ratings. The standard deviation of 3.27 shows some variability in ratings, but most fall between 5 and 10.

The useful count, which reflects how often an entry was marked as useful, has an average of 28.00 but shows significant variation, with counts ranging from 0 to 1,291. The median useful count is 16, indicating that half of the entries were marked useful 16 times or less, while a few entries were marked useful significantly more often, as reflected by the maximum value.

Appendix 4b: Description of drugsComTest_raw.csv dataset

```
✓ df_test_data.describe() ···
```

	uniqueID	rating	usefulCount
count	53766.000000	53766.000000	53766.000000
mean	116386.701187	6.976900	27.989752
std	67017.739881	3.285207	36.172833
min	0.000000	1.000000	0.000000
25%	58272.500000	4.000000	6.000000
50%	116248.500000	8.000000	16.000000
75%	174586.750000	10.000000	36.000000
max	232284.000000	10.000000	949.000000

The appendix 4B shows the statistical summary of the drugsComTest_raw.csv dataset. The dataset consists of 53,766 entries, each with a uniqueID, rating, and usefulCount. The uniqueID values range from 0 to 232,284, with a mean of 116,386.7 and a standard deviation of 67,017.7, indicating a broad distribution.

The rating values, which span from 1 to 10, have a mean of 6.98, with a median of 8, suggesting a tendency toward higher ratings. The ratings show some variability, as evidenced by a standard deviation of 3.29.

The usefulCount column, which measures how often entries are marked as useful, has a wide range, from 0 to 949 with a mean of 28 and a median of 16. This distribution indicates that while most entries have modest usefulness counts, a few are marked as highly useful.

Appendix 5: Merging of the train and test set

Combine the train and test datasets and check the shape

```
merge = [df_train_data,df_test_data]
df_review_data= pd.concat(merge,ignore_index=True)

df_review_data.shape
```

```
(215063, 7)
```

The code above shows the merging of the two sets. The process resulted to 215063 records with same 7 features.

Appendix 6a: Description of the merged datasets

```
df_review_data.describe()
```

	uniqueID	rating	usefulCount
count	215063.000000	215063.000000	215063.000000
mean	116039.364814	6.990008	28.001004
std	67007.913366	3.275554	36.346069
min	0.000000	1.000000	0.000000
25%	58115.500000	5.000000	6.000000
50%	115867.000000	8.000000	16.000000
75%	173963.500000	10.000000	36.000000
max	232291.000000	10.000000	1291.000000

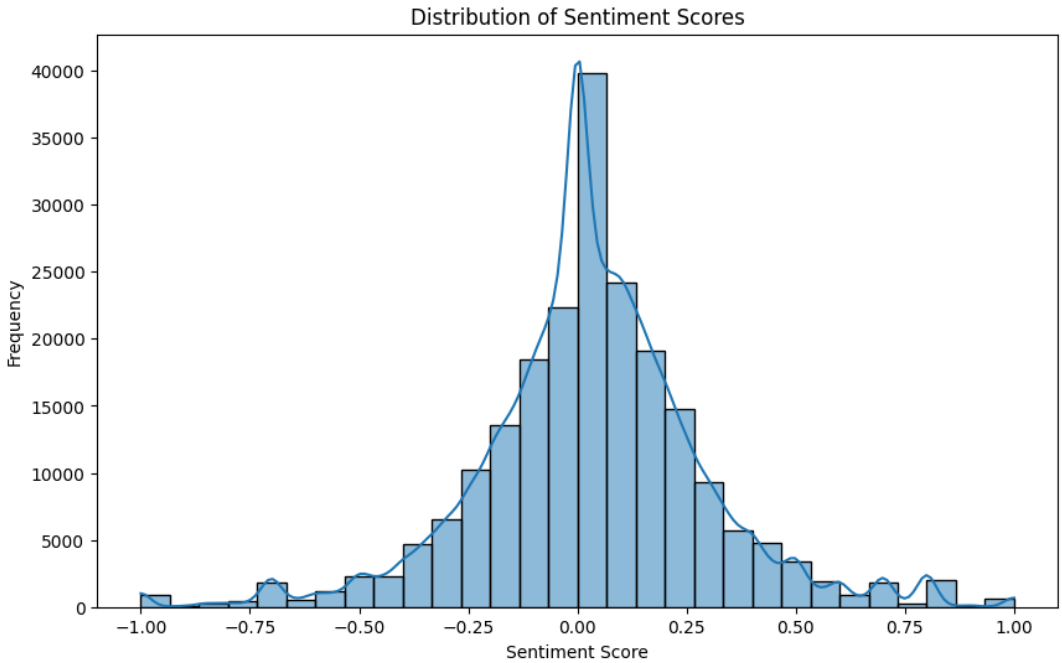
The appendix 6 above shows the statistical summary of the merged dataset. The dataset summary provides insights into the three numerical features: uniqueID, rating, and usefulCount. The uniqueID column has 215,063 entries, with values ranging from 0 to 232,291, and a mean of 116,039.36, indicating a broad distribution of unique identifiers. The rating column, also with 215,063 entries, has an average rating of 6.99 on a likely scale of 1 to 10. The ratings are moderately varied, with a standard deviation of 3.28, and most ratings skew towards the higher end, as indicated by a median of 8 and a maximum value of 10. The usefulCount column, which measures the perceived usefulness of entries, shows a wide range of values, with an average of 28.00 and a standard deviation of 36.35. While most entries have lower usefulness scores (with a median of 16), a few stand out with significantly higher scores, with the maximum reaching 1,291. Overall, the dataset reveals a distribution where ratings are generally positive, and usefulness varies significantly across entries.

Appendix 6b: Description of the merged datasets

```
✓ df_review_data.info() ···

··· <class 'pandas.core.frame.DataFrame'>
RangeIndex: 215063 entries, 0 to 215062
Data columns (total 7 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   uniqueID     215063 non-null  int64
 1   drugName     215063 non-null  object
 2   condition    213869 non-null  object
 3   review       215063 non-null  object
 4   rating       215063 non-null  int64
 5   date         215063 non-null  object
 6   usefulCount  215063 non-null  int64
dtypes: int64(3), object(4)
memory usage: 11.5+ MB
```

The DataFrame contains 215,063 rows and 7 columns, with an index ranging from 0 to 215,062. The columns include uniqueID, drugName, condition, review, rating, date, and usefulCount. Among these, uniqueID, rating, and usefulCount are stored as 64-bit integers (int64), while drugName, condition, review, and date are stored as objects, indicating text data. Most columns have no missing values, except for the condition column, which has 213,869 non-null entries, leaving 1,194 missing values. The Dataframe occupies approximately 11.5 MB of memory.

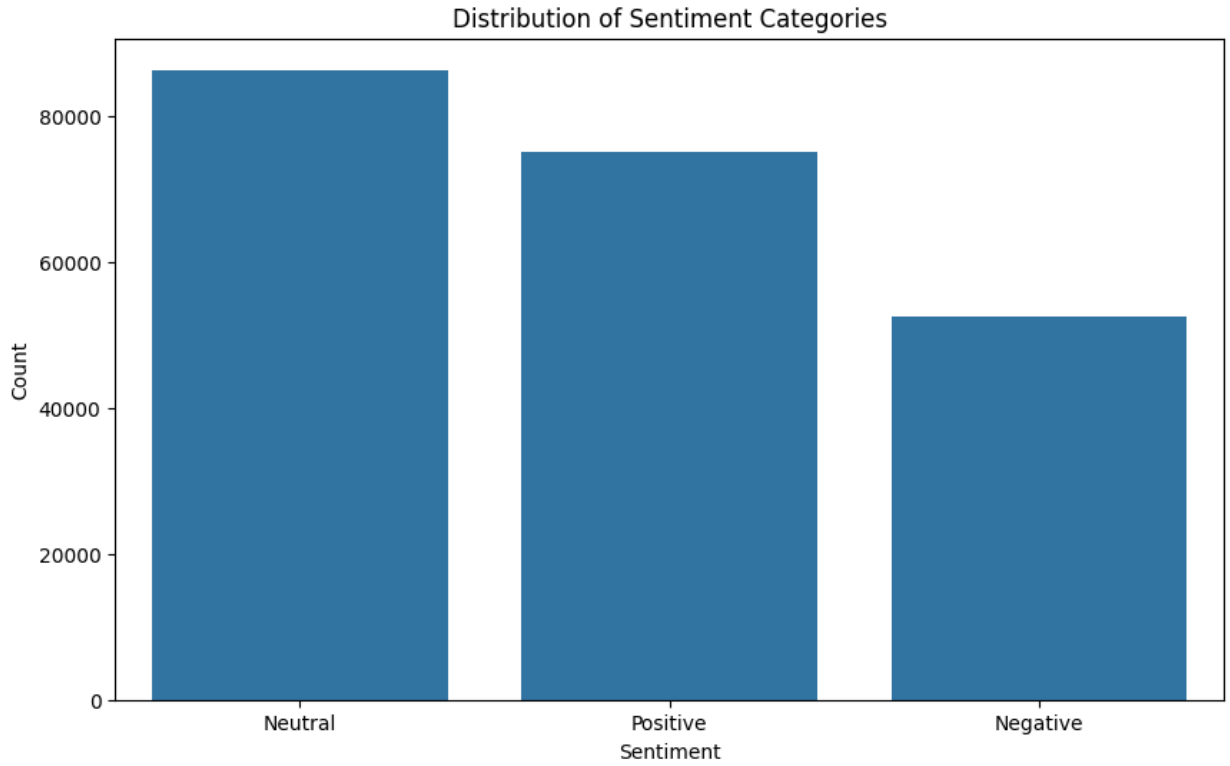
Appendix 7a: Graph showing the distribution of sentiment.



The graph above is a histogram that illustrates the distribution of sentiment scores, which range from -1 to 1, where -1 highly indicates negativity of the sentiment, 0 means that it is neutral sentiment, and

1 highly indicates positivity of the sentiment. The graph shows that most of the data is concentrated around a sentiment score of 0, which signifies that most entries have a neutral tone. The distribution is slightly skewed to the right, suggesting a mild positive bias in the dataset, with more entries expressing slightly positive sentiment than negative. The smooth density curve overlays the histogram, providing a continuous approximation of the distribution, reinforcing the observation that extreme sentiment scores—both positive and negative—are relatively rare. In summary, the graph shows that the dataset predominantly consists of neutral to mildly positive sentiment, with few instances of strong emotional expression.

Appendix 7b: Graph showing sentiment categories continues.



Appendix 7c: Top 10 words in positive and negative feedback

```
Top 10 words in positive feedback:
[('day', 39559), ('year', 32268), ('work', 31844), ('month', 29186), ('feel', 28385), ('effect', 25317), ('week', 24457), ('start', 24229), ('good', 23237), ('mg', 21022)]

Top 10 words in negative feedback:
[('day', 34274), ('bad', 30318), ('feel', 25402), ('month', 23247), ('week', 19465), ('start', 19241), ('pain', 17988), ('year', 17761), ('time', 17605), ('effect', 17453)]
```

Appendix 8a1: Code for preprocessing the data.

```
#Preprocessing of the Dataset
#checking the null values and features with the highest number of null values
df_review_data.isnull().sum()

Run Cell | Run Above | Debug Cell | Go to [12]
#%%
# dropping the null values
df_review_data.dropna(inplace=True, axis=0)

Run Cell | Run Above | Debug Cell | Go to [14]
#%%
#to describe the dataset
df_review_data.describe()

Run Cell | Run Above | Debug Cell | Go to [36]
# %%
# A function to preprocess the patient's review
def preprocess_text(text):
    # text = re.sub(r"I&#039;", "", text)
    text = re.sub(r'[^a-zA-Z\s]', '', text)
    doc = nlp(text.lower())
    text = text.replace("n't", " not")
    return ' '.join(token.lemma_ for token in doc if not token.is_stop and not token.is_punct)

Run Cell | Run Above | Debug Cell
#%%
def count_tokens(text):
    doc = nlp(text)
    return len(doc)

df_review_data['token_count'] = df_review_data['review'].apply(count_tokens)
total_tokens = df_review_data['token_count'].sum()

import string

def count_punctuation(text):
    return sum(1 for char in text if char in string.punctuation)

df_review_data['punctuation_count'] = df_review_data['review'].apply(count_punctuation)
total_punctuation_removed = df_review_data['punctuation_count'].sum()
```

Appendix 8a2: Code for checking the pre-processed results on the data.

```
import re

def count_non_letters(text):
    return len(re.findall(r'[^a-zA-Z]', text))

df_review_data['non_letter_count'] = df_review_data['review'].apply(count_non_letters)
total_non_letters_removed = df_review_data['non_letter_count'].sum()

from spacy.lang.en.stop_words import STOP_WORDS

def count_stopwords(text):
    doc = nlp(text)
    return sum(1 for token in doc if token.is_stop)

df_review_data['stopword_count'] = df_review_data['review'].apply(count_stopwords)
total_stopwords_removed = df_review_data['stopword_count'].sum()

def count_lemmatized_tokens(text):
    doc = nlp(text)
    return sum(1 for token in doc if token.text != token.lemma_)

df_review_data['lemmatized_count'] = df_review_data['review'].apply(count_lemmatized_tokens)
total_tokens_lemmatized = df_review_data['lemmatized_count'].sum()

def count_uppercase(text):
    return sum(1 for char in text if char.isupper())

df_review_data['uppercase_count'] = df_review_data['review'].apply(count_uppercase)
total_uppercase_converted = df_review_data['uppercase_count'].sum()

def count_nt_occurrences(text):
    return text.count("n't")

df_review_data['nt_count'] = df_review_data['review'].apply(count_nt_occurrences)
total_nt_replaced = df_review_data['nt_count'].sum()

Run Cell | Run Above | Debug Cell
#%%
print(f"Total Tokens: {total_tokens}")
print(f"Total Punctuation Marks Removed: {total_punctuation_removed}")
print(f"Total Non-Letters Removed: {total_non_letters_removed}")
print(f"Total Stopwords Removed: {total_stopwords_removed}")
```

Appendix 8b: Results of the pre-processed function on the data.

```
Total Tokens: 21202939
Total Punctuation Marks Removed: 3851654
Total Non-Letters Removed: 24129573
Total Stopwords Removed: 10197212
Total Tokens Lemmatized: 4217037
Total Uppercase Letters Converted: 2735469
Total Occurrences of 'n't' Replaced: 0
```

These statistics show the results of a text processing task performed on large data using the Spacy library. A total of 21,202,939 tokens which are individual pieces of identified text were processed, during which 3,851,654 punctuation marks (such as commas, periods) were removed to focus on the words themselves and 24,129,573 non-letter characters were removed to clean the text. Additionally, 10,197,212 common stopwords were identified and eliminated to reduce noise, and 4,217,037 tokens were lemmatized to their base forms for consistency. Furthermore, 2,735,469 uppercase letters were converted to lowercase to standardize the text, and no replacements were made for occurrences of the contraction "n't.".

Appendix 9a: Code to perform feature engineering using TF-IDF Vectorizer

```
vectorizer = TfidfVectorizer(max_features=215063)
tfidf_matrix = vectorizer.fit_transform(df_review_data['cleaned_feedback'])

num_documents = tfidf_matrix.shape[0]
vocab_size = tfidf_matrix.shape[1]
non_zero_elements = tfidf_matrix.nnz
avg_non_zero_per_doc = non_zero_elements / num_documents
max_non_zero_per_doc = tfidf_matrix.max(axis=1).count_nonzero()
min_non_zero_per_doc = tfidf_matrix.min(axis=1).count_nonzero()

tfidf_scores = zip(vectorizer.get_feature_names_out(), tfidf_matrix[0].toarray().flatten())
top_tfidf_words = sorted(tfidf_scores, key=lambda x: x[1], reverse=True)[:10]

print(f"Number of Documents: {num_documents}")
print(f"Vocabulary Size: {vocab_size}")
print(f"Non-Zero Elements in TF-IDF Matrix: {non_zero_elements}")
print(f"Average Non-Zero Features per Document: {avg_non_zero_per_doc}")
print(f"Max Non-Zero Features per Document: {max_non_zero_per_doc}")
print(f"Min Non-Zero Features per Document: {min_non_zero_per_doc}")
print(f"Top TF-IDF Words for First Document: {top_tfidf_words}")
```

Appendix 9b: Result of TF-IDF Vectorizer

```
✓ from sklearn.feature_extraction.text import TfidfVectorizer ···

Number of Documents: 215063
Vocabulary Size: 69199
Non-Zero Elements in TF-IDF Matrix: 6344365
Average Non-Zero Features per Document: 29.50003022370189
Max Non-Zero Features per Document: 215040
Min Non-Zero Features per Document: 0
Top TF-IDF Words for First Document: [('fish', 0.5486394080408448), ('bystolic', 0.5320807110753
```

The result above is the analysis of a TF-IDF matrix generated from 215,063 documents reveals several key characteristics. The vocabulary size across these documents is 69,199 unique words, resulting in a sparse matrix with 6,344,365 non-zero elements, indicating instances where words appear in the documents. On average, each document contains about 29.5 unique words contributing to its TF-IDF vector, though this varies significantly, with the maximum number of non-zero features in a single document being an unusually high 215,040, and the minimum being 0, suggesting at least one document contains no significant words from the vocabulary. The top TF-IDF words for the first document include 'fish', 'bystolic', 'oil', and 'combination', with some words showing a TF-IDF score of 0, indicating their minimal significance. Overall, this data indicates a diverse set of documents, with considerable variation in content richness and the importance of specific terms.

Appendix 10: Result of the over-sampling on the trained set using SMOTE.

```
Original class distribution: Counter({1: 107746, 0: 35546})
Resampled class distribution: Counter({1: 107746, 0: 107746})
```

Sampling Rate is given as the ratio of new minority class count to its original count which is:

$$Sampling\ Rate\ = \frac{New\ Minority\ class\ count}{Original\ Minority\ class\ count}$$

$$\frac{107746}{35546} = 3.03$$

Thus, the sampling rate is 3.03.

Appendix 11a: Multinomial Naïve Bayes Code from scratch

```python
class MultinomialNaiveBayes(BaseEstimator, ClassifierMixin):
    #function to initialize
    def __init__(self, alpha=1.0):
        self.alpha = alpha

    #function fit to calulate the prior and likelihood
    def fit(self, X_train, Y_train):
        noOfDocs, noOfWords = X_train.shape
        self.classes_ = np.unique(Y_train)
        noOfClasses = len(self.classes_)

        self.class_prior = np.zeros(noOfClasses)
        for index, cls in enumerate(self.classes_):
            self.class_prior[index] = np.sum(Y_train == cls) / noOfDocs

        self.conditional_prob = np.zeros((noOfClasses, noOfWords))
        for index, cls in enumerate(self.classes_):
            xTrain_cls = X_train[Y_train == cls]
            class_word_count = xTrain_cls.sum(axis=0)
            total_class_word_count = class_word_count.sum()
            self.conditional_prob[index, :] = (class_word_count + self.alpha) / (total_class_word_count + noOfWords)
    #function predict to test the trained data
    def predict(self, X_train):
        noOfDocs = X_train.shape[0]
        probabilityLog = np.zeros((noOfDocs, len(self.classes_)))

        for index, cls in enumerate(self.classes_):
            probabilityLogCls = np.log(self.class_prior[index])
            probabilityLogWords = X_train @ np.log(self.conditional_prob[index, :].T)
            probabilityLog[:, index] = probabilityLogCls + probabilityLogWords

        return self.classes_[np.argmax(probabilityLog, axis=1)]

    #function to evaluate the predicted data using ROC and AUC
    def predict_proba(self, X_train):
        noOfDocs = X_train.shape[0]
        probabilityLog = np.zeros((noOfDocs, len(self.classes_)))

        for index, cls in enumerate(self.classes_):
            probabilityLogCls = np.log(self.class_prior[index])
```

Appendix 11b: Multinomial Naïve Bayes Code from scratch continues.

```python
    #function to evaluate the predicted data using ROC and AUC
    def predict_proba(self, X_train):
        noOfDocs = X_train.shape[0]
        probabilityLog = np.zeros((noOfDocs, len(self.classes_)))

        for index, cls in enumerate(self.classes_):
            probabilityLogCls = np.log(self.class_prior[index])
            probabilityLogWords = X_train @ np.log(self.conditional_prob[index, :].T)
            probabilityLog[:, index] = probabilityLogCls + probabilityLogWords

        return np.exp(probabilityLog - np.max(probabilityLog, axis=1, keepdims=True))
```

Creating an object instance of the class MultinomialNaiveBayes() and training the dataset

```python
multinomialNB = MultinomialNaiveBayes()

multinomialNB.fit(X_train_main, Y_train_main)
```

Making predictions using the test data

```python
predictions = multinomialNB.predict(X_test)
proba_predictions = multinomialNB.predict_proba(X_test)
print("Predictions:", predictions)
```

```
Predictions: [1 0 0 ... 1 1 1]
```

Appendix 11c: Random Forest Classifier Code from scratch.

```
# Define DecisionTreeClassifier and RandomForestClassifier classes
class DecisionTreeClassifier:
    def __init__(self, max_depth=None, min_samples_split=2):
        self.max_depth = max_depth
        self.min_samples_split = min_samples_split
        self.tree = None

    def fit(self, X, y):
        self.tree = self._grow_tree(X, y)

    def _grow_tree(self, X, y, depth=0):
        n_samples, n_features = X.shape
        y = np.array(y).flatten()
        if (self.max_depth is not None and depth >= self.max_depth) or n_samples < self.min_samples_split or len(np.unique(y)) =
            return self._leaf(y)
        feat_idx, thresh = self._best_split(X, y, n_features)
        if feat_idx is None:
            return self._leaf(y)
        left_idxs, right_idxs = self._split(X[:, feat_idx], thresh)
        left = self._grow_tree(X[left_idxs, :], y[left_idxs], depth + 1)
        right = self._grow_tree(X[right_idxs, :], y[right_idxs], depth + 1)
        return Node(feat_idx, thresh, left, right)

    def _best_split(self, X, y, n_features):
        best_gain = -1
        split_idx, split_thresh = None, None
        for feat_idx in range(n_features):
            X_column = X[:, feat_idx].toarray().flatten()  # Convert sparse to dense
            thresholds = np.unique(X_column)
            for thresh in thresholds:
                gain = self._information_gain(y, X_column, thresh)
                if gain > best_gain:
                    best_gain = gain
                    split_idx = feat_idx
                    split_thresh = thresh
        return split_idx, split_thresh

    def _information_gain(self, y, X_column, split_thresh):
        parent_entropy = self._entropy(y)
        left_idxs, right_idxs = self._split(X_column, split_thresh)
```

Appendix 11d: Random Forest Classifier Code from scratch continues.

```
    def _information_gain(self, y, X_column, split_thresh):
        parent_entropy = self._entropy(y)
        left_idxs, right_idxs = self._split(X_column, split_thresh)
        if len(left_idxs) == 0 or len(right_idxs) == 0:
            return 0
        n = len(y)
        n_left, n_right = len(left_idxs), len(right_idxs)
        e_left, e_right = self._entropy(y[left_idxs]), self._entropy(y[right_idxs])
        child_entropy = (n_left / n) * e_left + (n_right / n) * e_right
        return parent_entropy - child_entropy

    def _split(self, X_column, split_thresh):
        # Convert sparse matrix column to dense array
        X_column = X_column.toarray().flatten() if hasattr(X_column, "toarray") else X_column
        left_idxs = np.argwhere(X_column <= split_thresh).flatten()
        right_idxs = np.argwhere(X_column > split_thresh).flatten()
        return left_idxs, right_idxs

    def _entropy(self, y):
        hist = np.bincount(y)
        ps = hist / len(y)
        return -np.sum([p * np.log2(p) for p in ps if p > 0])

    def _leaf(self, y):
        return Leaf(np.bincount(y).argmax())

    def predict(self, X):
            X_dense = X.toarray() if hasattr(X, "toarray") else X
            return np.array([self._traverse_tree(x, self.tree) for x in X_dense])

    def _traverse_tree(self, x, node):
        if isinstance(node, Leaf):
            return node.value
        if x[node.feature] <= node.threshold:
            return self._traverse_tree(x, node.left)
        return self._traverse_tree(x, node.right)
```

Appendix 11e: Random Forest Classifier Code from scratch continues.

```python
class RandomForestClassifier:
    def __init__(self, n_trees=10, max_depth=None, min_samples_split=2):
        self.n_trees = n_trees
        self.max_depth = max_depth
        self.min_samples_split = min_samples_split
        self.trees = []

    def fit(self, X, y):
        for _ in range(self.n_trees):
            tree = DecisionTreeClassifier(max_depth=self.max_depth, min_samples_split=self.min_samples_split)
            X_sample, y_sample = self._bootstrap_sample(X, y)
            tree.fit(X_sample, y_sample)
            self.trees.append(tree)

    def _bootstrap_sample(self, X, y):
        n_samples = X.shape[0]
        indices = np.random.choice(n_samples, n_samples, replace=True)
        return X[indices], y[indices]

    def predict(self, X):
        tree_preds = np.array([tree.predict(X) for tree in self.trees])
        return np.squeeze(np.apply_along_axis(lambda x: np.bincount(x, minlength=2).argmax(), axis=0, arr=tree_preds))

class Node:
    def __init__(self, feature, threshold, left, right):
        self.feature = feature
        self.threshold = threshold
        self.left = left
        self.right = right

class Leaf:
    def __init__(self, value):
        self.value = value

Run Cell | Run Above | Debug Cell
#%%

# Initialize and train the random forest classifier
rf = RandomForestClassifier(n_trees=10, max_depth=5, min_samples_split=2)
rf.fit(X_train, y_train)
```

Appendix 12: Application of hyperparameter tuning to the best model (Multinomial Naïve Bayes).

APPLICATION OF HYPERTUNING USING GRIDSEARCH

A function to perform hypertuning

```
def hypertune_multinomial_nb(X_train_main, Y_train_main, param_grid):
    grid_search = GridSearchCV(MultinomialNaiveBayes(), param_grid, cv=5, scoring='accuracy')
    grid_search.fit(X_train_main, Y_train_main)

    best_alpha = grid_search.best_params_['alpha']
    best_model = MultinomialNaiveBayes(alpha=best_alpha)
    best_model.fit(X_train_main, Y_train_main)

    return best_model, grid_search.best_params_, grid_search.best_score_
```

Defining the parameter grid

```
param_grid = {'alpha': [0.5, 1.0, 1.5, 2.0]}
```

To apply the hypertune function on the train data and print the best parameter and score

```
best_model, best_params, best_score = hypertune_multinomial_nb(X_train_main, Y_train_main, param_grid)

print("Best Parameters:", best_params)
print(f"Best CV Score: {best_score:.2f}")
```

```
Best Parameters: {'alpha': 0.5}
Best CV Score: 0.80
```

Appendix 13: Code for Evaluation.

```
predictions = best_model.predict(X_test)
proba_predictions = best_model.predict_proba(X_test)
print("Predictions:", predictions)

Run Cell | Run Above | Debug Cell
#%%
# to evaluate using the accuracy, precision, recall and f1-score metrics
accuracy = accuracy_score(Y_test, predictions)
precision = precision_score(Y_test, predictions, average='macro')
recall = recall_score(Y_test, predictions, average='macro')
f1 = f1_score(Y_test, predictions, average='macro')

print(f"Accuracy: {accuracy}")
print(f"Precision: {precision}")
print(f"Recall: {recall}")
print(f"F1 Score: {f1}")

Run Cell | Run Above | Debug Cell
#%%
# to get the Confusion Matrix
conf_matrix = confusion_matrix(Y_test, predictions)
sns.heatmap(conf_matrix, annot=True, fmt="d", cmap="Blues", xticklabels=best_model.classes_, yticklabels=
plt.xlabel('Predicted Label')
plt.ylabel('True Label')
plt.title('Confusion Matrix')
plt.show()

Run Cell | Run Above | Debug Cell
#%%
# to print classification report
target_names = [str(cls) for cls in best_model.classes_]
class_report = classification_report(Y_test, predictions, target_names=target_names)
print("\nClassification Report:\n", class_report)
```

Appendix 14a: Code for Evaluation and results contd.

```
target_names = [str(cls) for cls in best_model.classes_]
class_report = classification_report(Y_test, predictions, target_names=target_names)
print("\nClassification Report:\n", class_report)
```

```
Classification Report:
                precision    recall  f1-score   support

             0       0.59      0.71      0.65     17710
             1       0.90      0.83      0.86     52867

      accuracy                           0.80     70577
     macro avg       0.74      0.77      0.75     70577
  weighted avg       0.82      0.80      0.81     70577
```

Appendix 14b: Code for Evaluation and results(contd).

To evaluate using the accuracy, precision, recall and f1-score metrics with the macro Average

```
accuracy = accuracy_score(Y_test, predictions)
precision = precision_score(Y_test, predictions, average='macro')
recall = recall_score(Y_test, predictions, average='macro')
f1 = f1_score(Y_test, predictions, average='macro')

print(f"Accuracy: {accuracy:.2f}")
print(f"Precision: {precision:.2f}")
print(f"Recall: {recall:.2f}")
print(f"F1 Score: {f1:.2f}")
```

```
Accuracy: 0.80
Precision: 0.74
Recall: 0.77
F1 Score: 0.75
```

Appendix 14c: Code for Evaluation and results (contd).

To evaluate using the accuracy, precision, recall and f1-score metrics

```
accuracy = accuracy_score(Y_test, predictions)
precision = precision_score(Y_test, predictions)
recall = recall_score(Y_test, predictions)
f1 = f1_score(Y_test, predictions)

print(f"Accuracy: {accuracy:.2f}")
print(f"Precision: {precision:.2f}")
print(f"Recall: {recall:.2f}")
print(f"F1 Score: {f1:.2f}")
```

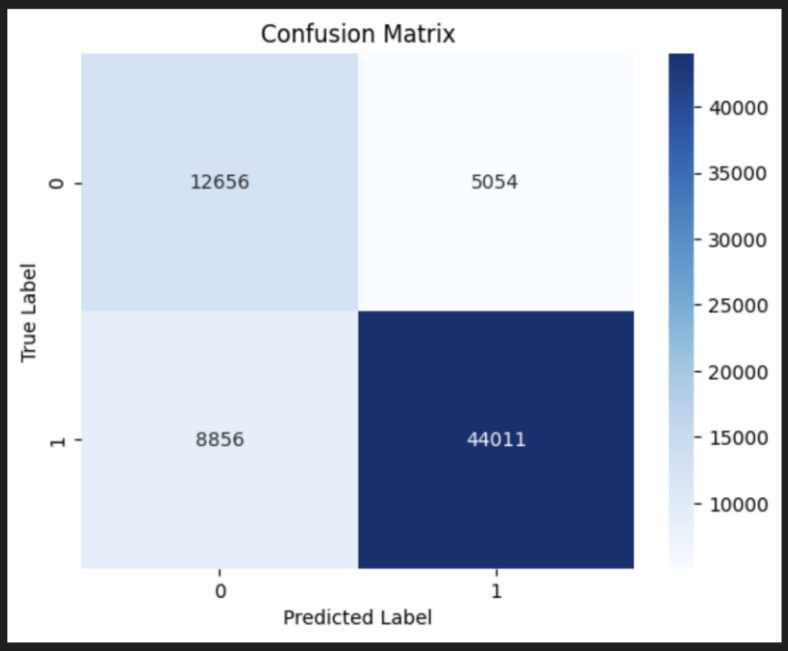
```
Accuracy: 0.79
Precision: 0.90
Recall: 0.82
F1 Score: 0.86
```

To evaluate using the accuracy, precision, recall and f1-score metrics with the macro Average

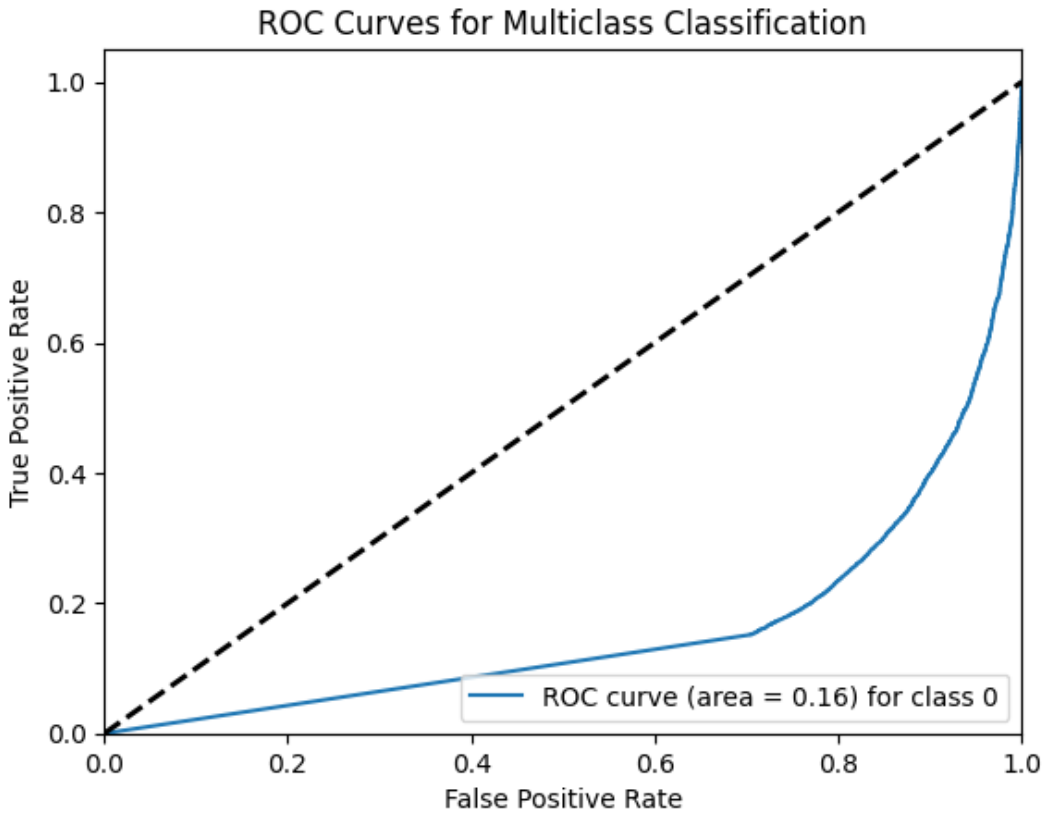
```
accuracy = accuracy_score(Y_test, predictions)
precision = precision_score(Y_test, predictions, average='macro')
recall = recall_score(Y_test, predictions, average='macro')
f1 = f1_score(Y_test, predictions, average='macro')

print(f"Accuracy: {accuracy:.2f}")
print(f"Precision: {precision:.2f}")
print(f"Recall: {recall:.2f}")
print(f"F1 Score: {f1:.2f}")
```

Appendix 14d: Code for Evaluation and results contd.

```
conf_matrix = confusion_matrix(Y_test, predictions)
sns.heatmap(conf_matrix, annot=True, fmt="d", cmap="Blues", xticklabels=best_model.classes_, yticklabels=best_
plt.xlabel('Predicted Label')
plt.ylabel('True Label')
plt.title('Confusion Matrix')
plt.show()
```



Appendix 14e: Evaluation and results contd.



Appendix 15a: Code to save the model as pkl file.

To save the model using pickle

```python
import pickle

model_path = '/content/drive/MyDrive/Project_Dataset/model.pkl'

with open(model_path, 'wb') as file:
    pickle.dump(best_model, file)

vectorizer_path = '/content/drive/MyDrive/Project_Dataset/vectorizer.pkl'

with open(vectorizer_path, 'wb') as file:
    pickle.dump(vectorizer, file)

print(f"Model saved to {model_path}")
print(f"Vectorizer saved to {vectorizer_path}")
```

```
Model saved to /content/drive/MyDrive/Project_Dataset/model.pkl
Vectorizer saved to /content/drive/MyDrive/Project_Dataset/vectorizer.pkl
```

Appendix 15b: Code to save and load model as pkl file (contd).

```python
# Load the saved model
model_filename = 'model.pkl'
with open(model_filename, 'rb') as file:
    model = pickle.load(file)

# Load the saved TfidfVectorizer
vectorizer_filename = 'vectorizer.pkl'
with open(vectorizer_filename, 'rb') as file:
    vectorizer = pickle.load(file)

# Function to preprocess the text
def preprocess_text(text):
    text = re.sub(r'[^a-zA-Z\s]', '', text)
    doc = nlp(text.lower())
    text = text.replace("n't", " not")
    return ' '.join(token.lemma_ for token in doc if not token.is_stop and not token.is_punct)

@app.route('/')
def home():
    return render_template('index.html')

@app.route('/predict', methods=['POST'])
def predict():


    text = request.form['text']
    preprocessed_text = preprocess_text(text)
    text_vectorized = vectorizer.transform([preprocessed_text])

    prediction = model.predict(text_vectorized)[0]
    probability = model.predict_proba(text_vectorized)[0][prediction]

    sentiment_map = {1: 'Positive', 0: 'Negative'}
    sentiment = sentiment_map.get(prediction, 'Neutral')

    sentiment_class = 'positive' if sentiment == 'Positive' else 'negative' if sentiment == 'Negative' else 'neutral'

    return render_template('result.html', sentiment=sentiment, probability=f"{probability:.2f}", sentiment_class=sentiment_class)
```

Appendix 15c: HTML and CSS code to design the front end.

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Drug Sentiment Analysis</title>
    <style>
        /* Set background image and basic page styles */
        body {
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 0;
            background-image: url('{{ url_for('static', filename='images/background.jpg') }}');
            background-size: cover;
            background-position: center;
            background-attachment: fixed;
            color: white;
        }

        /* Content container */
        .content {
            max-width: 600px;
            margin: 100px auto;
            background: rgba(0, 0, 0, 0.7); /* Semi-transparent background */
            padding: 30px;
            border-radius: 15px;
            box-shadow: 0 0 15px rgba(0, 0, 0, 0.5);
            text-align: center;
        }
        .name{
            padding: 50px;
            margin-left: 350px;
            color: #ffcc00;
            box-shadow: 0 0 15px rgba(0, 0, 0, 0.5);
        }

        h1 {
            text-align: center;
            color: #ffcc00;
        }
```

Appendix 15d: CSS code to design the front end

```
        /* Style for the introductory paragraph */
        .intro {
            font-size: 1.2rem;
            margin-bottom: 30px;
            color: #dddddd;
        }

        form {
            margin-top: 20px;
        }

        textarea {
            width: 100%;
            height: 150px;
            margin-bottom: 20px;
            padding: 10px;
            border-radius: 5px;
            border: none;
            font-size: 1rem;
        }

        button {
            padding: 12px 30px;
            background-color: #007BFF;
            color: white;
            border: none;
            cursor: pointer;
            font-size: 1rem;
            border-radius: 5px;
        }

        button:hover {
            background-color: #0056b3;
        }

        label {
            font-size: 1rem;
            color: #ffcc00;
        }
    </style>
```

Appendix 15e: CSS code to design the result front end

```
> Users > andre > Downloads > Project's code > Multinomial > templates > <> result.html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Sentiment Analysis Result</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 50px;
        }
        body {
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 0;
            background-image: url('{{ url_for('static', filename='images/background.jpg') }}');
            background-size: cover;
            background-position: center;
            background-attachment: fixed;
            color: white;
        }
        /* Content container */
        .content {
            max-width: 600px;
            margin: 100px auto;
            background-color: #53272c;

            padding: 30px;
            border-radius: 15px;
            box-shadow: 0 0 15px rgba(0, 0, 0, 0.5);
            text-align: center;
        }
        h1 {
            text-align: center;
        }
        .result {
            max-width: 600px;
            margin: 0 auto;
            text-align: center;
        }
        .result p {
            font-size: 1.2em;
ter  ⊗ 0 ⚠ 42  0
```

Appendix 15f: CSS code to design the result front end continues

```
<head>
    <style>
        }
        .result {
            max-width: 600px;
            margin: 0 auto;
            text-align: center;
        }
        .result p {
            font-size: 1.2em;
        }
        .positive {
            color: green;
        }
        .negative {
            color: red;
        }
        .neutral {
            color: gray;
        }
        .button {
            margin-top: 20px;
            padding: 10px 20px;
            background-color: #007BFF;
            color: white;
            border: none;
            cursor: pointer;
        }
    </style>
</head>
<body>
    <div class="content">
        <h1>Sentiment Analytical Result</h1>
        <div class="result">
            <p>The sentiment of the provided review is: <strong class="{{ sentiment_class }}">{{ sentiment }}</strong></p>
            <p>Probability: {{ probability }}</p>
            <button class="button" onclick="window.location.href='/'">Analyze Another Review</button>
        </div>
    </div>

</body>
</html>
```