JCBI
Journal of Computing & Biomedical Informatics
ISSN: 2710 - 1606
SICAIET
2024
Research Article
Collection: Artificial Intelligence and Emerging Technologies

Diagnostic Prediction based on Medical Notes using Machine Learning

Nazir Ahmad[1*], H. M. Shafiq Ur Rehman[1], Mubasher H. Malik[2], Syed Ali Nawaz[1], and M. Abdul Qadoos Bilal[3]

[1]Department of Information Technology, IUB, Bahawalpur, 63100, Pakistan.
[2]Department of Computer Science, ISP, Multan, 59300, Pakistan.
[3]College of Information and Computer, TUT, Taiyuan, 03000, China.
[*]Corresponding Author: Nazir Ahmad. Email: nazeerrana@iub.edu.pk

Academic Editor: Salman Qadri **Published:** February 01, 2024

Abstract: Clinical experts have extracted clinically relevant information from clinical notes through manual review, which has had scaling and financial issues. This is particularly relevant for different diseases since clinical notes prevail over structured data. The availability of this data gives a wonderful opportunity for natural language processing (NLP) to automatically extract clinically relevant information that might delay or prevent the onset of disease, but it also poses several challenges. In this work, we sought to investigate the current state of the art and suggest possible future research pathways that might expedite the general use of natural language processing in disease-related clinical notes. In this study, Kaggle, an open-source platform for machine learning challenges, provides the dataset. The patient's age, gender, diagnoses, and other vitals are all included in the dataset's text format. The dataset collection contains information from many categories. Each stage plays an important role in predicting patient therapy based on clinical notes, from dataset preparation through model training and testing. Two feature engineering methods, term frequency-inverse document frequency and bag of words are used for feature extraction. Six distinct machine learning (ML) methods, Naive Bayes, Light GBM, Random Forest (RF), Logistic Regression, Support Vector Machines (SVM), and Extra Tree Classifiers were employed for analysis. Various sample sizes of the dataset have been used in the proposed study. Based on the findings, logistic regression is the most effective algorithm for predicting medical therapy, with an accuracy of 85.94%.

Keywords: Medical notes; Machine learning; Bag of words; Random Forest; Logistic Regression.

1. Introduction

A medical note is a portion of a patient's private electronic health record (EHR) that details their medical visit. All healthcare team members, including doctors, nurses, PAs, techs, radiologists, and therapists, are responsible for taking notes after each interaction with a patient, whether in person or through telemedicine. Documenting a patient's medical history, current sickness history, diagnosis, prescriptions, allergies, therapy, and general care in a systematic manner requires accurate and thorough medical notes [1]. Each interaction with a patient is documented in some detail in their medical record; these may range from an initial consultation to a second opinion, a follow-up visit, a procedure visit, a therapy visit, or even a diagnostic testing visit. The healthcare provider's office visit note for a consultation, second opinion, or follow-up, for example, has sections with all the details needed to care for the patient, like the patient's Chief Complaint (CC), which is a short medical term or phrase that describes the main problem that made the patient go to the doctor in their initial visit [2].

When evaluating a patient, the attending physician needs to diagnose the disease for proper treatment, and this may be accomplished by asking the patient to identify their primary complaint. The provider may then use this information to guide the kind of further history they collect as part of the assessment and the kind of physical examination they do in light of the stated condition. The History of

the Present Disease (HPI) is an in-depth description of how the patient's illness got worse from the first signs to the day of the office visit. Review of Systems (RoS) is a collection of questions broken down by body system that can be used to find out what's wrong and figure out what's sick. In the physical examination note section, the doctor puts down what he or she sees, hears, or measures. The RoS records, shows what the patient answers in response to those questions. The doctor may use exertion and probing to find out about the size, position, consistency, texture, location, and soreness of a body part or organ. They may also use a stethoscope to check the heart beat and valve function. They may also measure the person's height, weight, and pulse [3].

The keeping of correct clinical records is an important part of both good professional work and providing high-quality healthcare. Whether the notes are kept on paper or electronically, good clinical record keeping should allow for continuity of care and make it easier for doctors to talk to each other. As a result, everyone on the multidisciplinary team (doctors, surgeons, nurses, pharmacists, physiotherapists, occupational therapists, psychologists, pastors, managers, or students) must keep clinical records up to date as needed [4]. Patients should be able to see their own records so that they know what was done and what choices were looked into. In addition to their value in auditing healthcare quality, clinical records may be useful in examining significant occurrences, patient complaints, and compensation claims. Keeping accurate clinical records is crucial.

Clinical notes that accurately record a patient's medical history are crucial. All pertinent clinical information should be included in a medical record for future use. Always keep in mind that if anything does not get recorded, it did not occur. This maintains continuity, which is vital in the event of a contentious medical decision. As many different medical personnel are now engaged in the care of a single patient, clinical notes must be consistent throughout therapy. In order to guarantee that all relevant healthcare personnel have access to the most up-to-date and correct information, clinical notes must be prepared properly and with adequate detail. The patient will benefit from this since unnecessary tests will not need to be repeated and incorrect diagnoses and treatments will be avoided [5].

Good clinical records also improve the healthcare system as a whole by speeding up the decision-making process for an individual patient, allowing more resources to be allocated to those patients who really need them. Last but not least, inaccurate or lacking clinical data may be detrimental to a patient's long-term health. Remember that "the obligation to share information might be as important as the responsibility to protect patient privacy," as stated in the sixth principle of the NHS report on patient information, the Caldicott report [6]. Keeping thorough, up-to-date records of care provided might be used as evidence in the event of a complaint or lawsuit. According to the GMC's Good Medical Practice and the Nursing and Midwifery Council's professional standards, keeping records is an essential component of practice and a key part of providing safe and effective care [7] [5].

Present-day human coders spend a lot of time and energy on inefficient manual processes; automating annotation will reduce this burden significantly. It is hardest to find the right medical code among hundreds of high-dimensional codes when the clinical notes are just a bunch of free text that isn't grouped in any manner. Using Convolutional Neural Networks (CNN) and Long Short-Term Memory (LTSM) networks, a lot of progress has been made in the last three years on the MIMIC-III full-label inpatient clinical notes dataset, which is the gold standard for hard benchmarks. This development begs the age-old issue of how close automated ML systems are to the working performance of human programmers [8].

Technologies like automated medical coding (AMC) and automatic clinical coding (ACC) employ NLP to produce medical diagnostic and procedure codes automatically from medial notes [9]. They play a crucial role in modern healthcare because they provide timely access to comprehensive, accurate patient records that improve diagnosis and allow for better, more coordinated treatment [10].

Laboratory test results, vital signs, medicines, and other therapies delivered to persons at risk for AD dementia, and comorbidities may all be included in EHR data. Neuroimaging scans (such as positron emission tomography (PET) and magnetic resonance imaging (MRI)) and cerebrospinal fluid (CSF) collection for biomarker testing are two expensive and/or invasive procedures that people may get to look for signs of Alzheimer's disease. The electronic health record (EHR) could also include findings from

these tests. Studies have indicated that longitudinal clinical EHR data (data obtained at various periods in time) may be used to track the temporal development of Alzheimer's disease and dementia. The extensive use and accessibility of medical devices over the years have generated enormous amounts of clinical EHR data, which may supplement the traditional resources of dementia specialists. Researchers have been prompted to investigate the potential of artificial intelligence (AI), which is gaining prominence in the area of healthcare innovation due to the unmet demands for dementia expertise and the enormous datasets necessary [11].

Machine learning (ML), a subfield of artificial intelligence, can analyze the correlation between input variables and clinical outcomes, unearth previously unknown patterns in massive datasets, and draw conclusions that guide more informed clinical judgment. However, subject matter experts must still verify the accuracy of computational hypotheses produced by ML models before they can be used in clinical decision-making [12].

Rule-based knowledge engineering was used in the first studies of automated document categorization, featuring the incorporation of a human-created set of guidelines for expert intelligence [16]. NLP and ML algorithms, such as kernel and logistic regression approaches, have lately been used for risk stratification, which in turn has facilitated clinical decision-making. Researchers have used machine learning and natural language processing to automatically sort clinical papers into groups based on ASD [17] asthma [18], heart failure criteria [19], bad drug effects [20], and rheumatoid arthritis activity [21]. Some studies have used technology to automate the categorization of clinical documents to enhance clinical workflows and patient safety [22]. Neural network models using the distributed representation technique [23] are now the gold standard for document categorization jobs. In theory, a deep neural network may acquire to represent complex facts on its own, doing away with the requirement for hand-crafted feature engineering in clinical knowledge representation. [24] employed CNN with dispersed word representation to compete on a sentence-level medical text categorization exam [25]. For broad sentiment analysis, computer scientists have used convolutional neural networks (CNNs) or a recurrent neural networks version called Long Short-Term Memory (LSTM) to learn semantic representations in texts [26]. Character-level applications of CNN have been developed for a variety of text categorization problems [27].

[28] discussed unsupervised learning jobs across many note types and document sources showed promising results when using the clustering technique to discover medical subdomains in a clinical note. This was accomplished by representing the data using language and semantic kinds. Experts used a support vector machine (SVM) with a bag of words to categorize patients admitted to the hospital for treatment of a suspected sickness (UMLS concepts).

However, few researchers address the medical subdomain categorization issue by comparing and assessing the performance of supervised shallow and deep learning algorithms utilizing diverse kinds of data. A supervised machine learning classifier trained to identify medical subdomains within clinical notes has the potential to enhance clinical downstream applications at the subspecialty level if it is used effectively [29]. The medical subdomain classifier has the potential to improve our understanding of common syntactic and semantic structures in specialist notes and, more practically, to direct patients with unresolved medical difficulties toward the appropriate medical specialist for treatment. Using a supervised machine learning-based NLP pipeline, we were able to create classifiers for certain medical subdomains that can place clinical notes into appropriate categories [22].

2. Research Objectives

This research study aims to provide a framework for predicting therapies from clinical notes.

- Other goal is to use data-cleaning strategies to eliminate noise from the dataset.
- Use feature engineering to glean the best characteristics for treatment outcome prediction.
- We will use a variety of machine learning models for the prediction of possible treatment based on the medical notes of patients.

3. Materials and Methods

The proposed methodology describes the detail, from the dataset to the final findings, how to forecast medical therapy by analyzing clinical notes using NLP and ML. These datasets often exist in an unstructured form and have a great deal of noise embedded within the records. Since the data we are working with in this study is textual, there is a high probability that the medical records will include spurious information. Data cleansing is essential before using any ML model. If we want better outcomes, we need cleaner data. Numerous operations, including normalization, lowercase-to-uppercase conversion, tokenization, and the removal of stop words, are required in the preprocessing of data. The next step, after the preprocessing stage, is feature engineering, which involves the extraction of features from text data. To train the ML model and make treatment predictions, these characteristics are crucial. When attempting to extract features from text data, feature engineers often resort to tools like TF-IDF and Bag of Words.

The next step is to run ML models, which may be done when feature extraction has been completed. In this procedure, data is divided into a training set and a test set at 70:30 ratio. This indicates that the model will be trained using 70% of the data and then tested using 30% of the data. The ML models employed in this study are Naive Bayes, Logistic Regression, Extra Tree Classifier, Light Gradient Boosting, Random Forest, and Support Vector Machine. Precision, recall, f1-score, and accuracy score are the four metrics used for assessing the performance of the models. The best-performing model in this study will be determined when training and testing are complete and compared using assessment parameters. When you have a firm grasp of this study framework in its entirety, you'll have a far better grasp of the intricate process behind data analysis and treatment prediction. Figure 1 shows the entire framework with the flow of data from the dataset to training models.

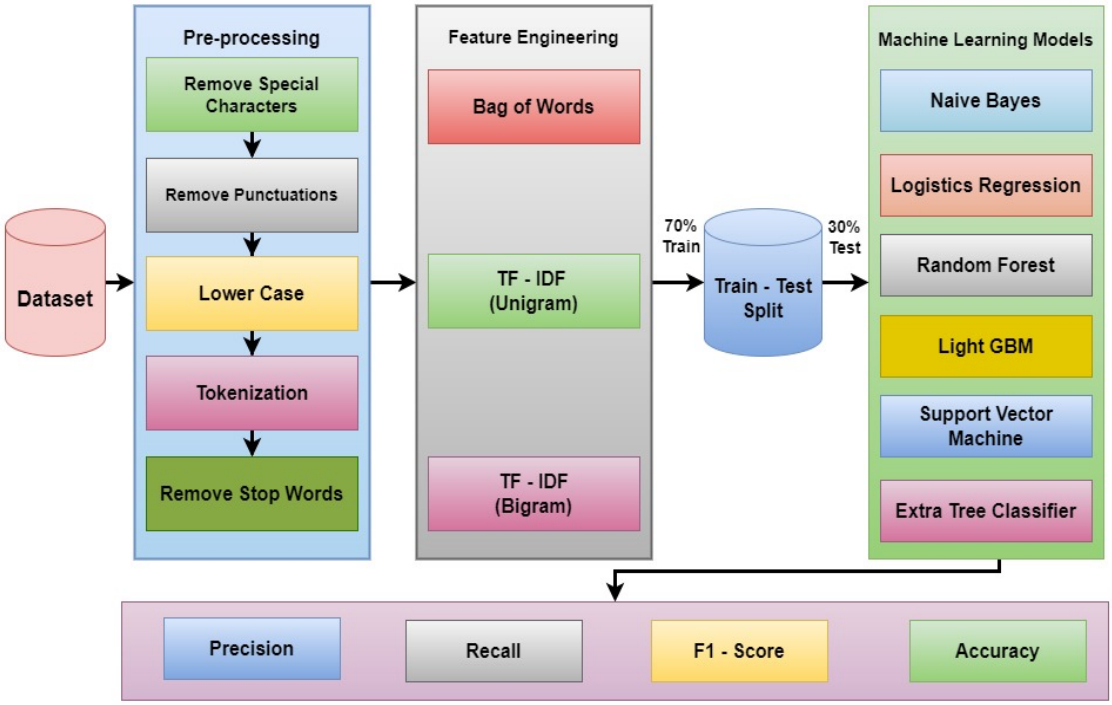

Figure 1. Research Framework

3.1 Data Collection

The dataset was gathered from Kaggle, an open-source machine learning platform. Approximately 2000 individual medical records are used in the dataset. In the first column of the dataset set is the actual text of the medical notes, and in the second is the actual therapy that was administered. Complete information about a patient's disease, potential medical features, symptoms, and other characteristics was considered. Due to the text's inherent lack of structure and plenty of background noise, it is required to clean the dataset before applying feature extraction algorithms to the data. Otherwise, doing so would result in inaccurate predictions.

3.2 Preprocessing

Machine learning-trained algorithms may use their knowledge of existing patterns in features to predict the value of a certain target variable in previously unknown data. The final trained model may be seen as a mathematical function that turns X values (the features) into y predictions accurately (the target). Due to their mathematical nature, ML algorithms can only handle numerical data. In addition, these numerical representations should reflect how the algorithm understands the data since each algorithm runs under a unique set of constraints and assumptions. Now, let's imagine that we have a character that represents the colour of a car and that it can take on the values red, blue, and grey. If we were to give numerical values to each colour, such as red = 1, blue = 2, and grey = 3, a machine learning system unfamiliar with the concept of colour may mistake red for a larger number and assign it a greater priority. Preprocessing entails transforming raw features into a form that a machine learning algorithm can use to learn. It has been shown that data preparation for machine ML is an art that requires a careful evaluation of the raw data in order to choose suitable strategies and preprocessing procedures.

Data preprocessing is essential to data mining and analysis because it prepares unprocessed data for further processing. Data obtained from the real world, whether it's text, images, or videos, is often unorganized and disorderly. In addition to the possibility that it contains errors and inconsistencies, it is usually incomplete and lacks any discernible structure. Processing information in the form of 1s and 0s is the simplest for computers. Thus, calculating structured data, such as whole numbers and percentages, is straightforward. Unfortunately, text and image data must be cleaned and processed before analysis.

3.2.1 Remove Special Characters

Special characters are symbols that can't be represented by the standard ASCII set of letters and numbers. These symbols are often used in remarks, references, monetary figures, etc. The use of these symbols causes algorithmic noise and does not improve text comprehension. The good strategy in text mining is that these unwanted letters, digits, and symbols may be removed using regular expressions (regex).

3.2.2 Remove Punctuations

The next step of omitting special characters may be combined with this one. It's simple to get rid of punctuation marks. Using string.punctuation, just keep anything that isn't on this list.

3.2.3 Lower Case

One could wonder, "How should I handle capitalization when it occurs at the beginning of a phrase or in proper nouns?" It's usual practice to write everything in lowercase letters for readability. As a result, text mining and (NLP) processes may keep moving along smoothly. Since we have the lower() method, it's a breeze.

3.2.4 Remove stop words

The removal of stopwords is beneficial since it reduces the dimensional space and a few stopwords won't drive your analysis if you're using bag of words based techniques like countVectorizer or TF-IDF, which operate on counts and frequency of the words. On the other hand, if you're trying to use the text's semantics, like in a seq2seq model, then you'll get unclear results if you remove stopwords. Remove stop words is a standard preprocessing technique used in many NLP applications. Removing the phrases that are overused across the whole corpus is the core idea.

3.3 Tokenization

Tokenization is a simple process that may be used to successfully turn a large, unstructured data collection into a manageable string. As well as its more well-known functions in cybersecurity and the creation of NFTs, tokenization also plays a crucial part in NLP. Tokenization is a technique used in natural language processing to parse text into smaller, more manageable chunks so that semantic labels may be more accurately applied. The initial stage in NLP is to collect relevant information (a phrase) and parse it into manageable chunks (words). Take the query, "What restaurants are nearby?" as an example of a data string. Tokenization has used the string to break it down into distinct components so that a computer can understand it.

3.4 Feature Engineering

Feature engineering is the process of determining and changing data into features that can be used in supervised learning. Machine learning may need new or enhanced characteristics to be trained on before it can perform well on novel tasks. You probably already know that a feature is any quantifiable characteristic that may be used in a prediction model. Color, tone of voice, and other such characteristics fall within this category. Feature engineering, in its simplest form, is the application of statistical or machine learning techniques to raw data in order to extract the characteristics of interest.

3.5 Features

The "features" that compose a dataset may be used as a means of explanation or communication. This may be done on the basis of size, location, age, time, colour, etc. Features, which are also called traits, variables, fields, and characteristics, are often stored in datasets as columns. Knowing what "features" are can help you prioritize them during data preparation depending on your organization's goals.

3.5.1 Bag of Words

In NLP, BoW is a text modeling technique employed. Feature extraction from text data is a simple explanation of the method. This approach is simple to apply and flexible, making it ideal for feature extraction from documents. To indicate the frequency with which words appear in a document, a "bag of words" might be created. We only count words and pay no attention to spelling, punctuation, or sentence structure. The term "bag of words" is used to describe a manuscript in which the order and structure of the words have been ignored. The model cares solely about the presence or absence of recognized words, not their precise placement. The BoW method enables us to translate texts of different lengths into fixed-length vectors, making it simpler for machine learning algorithms to deal with text despite its inherent disorder and lack of organization. Machine learning models also function on numerical data rather than textual data at a greater resolution. Accurately, we use the BoW technique to convert text into a numerical vector.

3.5.2 TF-IDF

One mathematical way to figure out how important a word is in a document is to use the term frequency-inverse document frequency (TF-IDF) method. We do this by increasing the total number of times a word appears in all documents by the opposite of how often it appears in documents. It's useful for determining how important words are in a document using machine learning techniques for Natural Language Processing, but that's far from where it's most important NLP. If, however, the term "Bug" occurs several times in one document but not in others, this is likely to indicate its importance. The word "Bug," for instance, would likely be associated with the subject "Reliability" if we were attempting to determine which categories certain NPS answers were in.

3.5.3 Train-Test Split

The train-test split process can be used to get an idea of how well ML algorithms will do predictions using data that wasn't used to train the model. Not long from now, we will be able to get information that you can use to compare different machine learning methods to your predictive modeling problem. It's simple and easy to understand the method, but you shouldn't always use it. When the dataset isn't fair, it takes more factors to work with a small dataset and type it in.

3.6 Machine Learning Models

ML model is a mathematical representation of the results of the training procedure. The field of research known as machine learning examines algorithms that can learn from past data and experiences to refine and perfect themselves automatically. When applied to computers, a machine learning model may be thought of as a piece of software that can learn to detect patterns or behaviours by analyzing existing data. The training data is analyzed by the learning algorithm, which then produces a machine learning (ML) model that represents the observed patterns.

In this research, Naive Bayes, Random Forest, Logistic Regression, SVM, Light Gradient Boosting, and Extra Tree Classifier ML models are applied

3.6.1 Naïve Bayes

Naive Bayes classifiers are a kind of "probabilistic classifier" used in statistics. They are based on applying Bayes' theorem under the premise of substantial independence between the features. They

aren't the most complicated Bayesian network models, but they may give great results when used with kernel density estimates.

3.6.2 Logistic Regression

Logistic regression is the method of choice when the dependent variable is of the binary kind (binary). Prediction is the goal of logistic regression analysis, as it is with other types of regression as well. Logistic regression is a statistical technique used to summarise data and provide an explanation for the association between a single binary dependent variable and a set of independent variables with nominal, ordinal, interval, or ratio levels. In spite of the fact that logistic regressions might be difficult to read, the Intellectus Statistics application streamlines research implementation and provides a transparent explanation of the results.

3.6.3 Random Forest

A random forest is a group of individual decision trees. A classification is reached by tallying the votes of each random forest tree that makes a forecast. Simple but powerful, the wisdom of a large number of people is the basis for Random Forest. A large number of significantly uncorrelated models (trees) functioning as a committee will exceed any of the individual component models, according to the data scientist's reasoning for the success of the random forest model.

The low degree of agreement amongst models is crucial. If more than one tree is at fault in the same way, the others will correct for it. Some trees may be wrong, but the forest as a whole will become better since the bulk of the trees are accurate. So, to make use of Random Forest's full potential, we need:

- To create models that beat random guessing, we must guarantee that our features include a true signal.
- Individual tree predictions (and their corresponding errors) should be uncorrelated.

3.6.4 Support Vector Machine

The support vector machine method looks for a hyperplane in N-dimensional space (N minus the number of traits) so that each piece of data can be put into a unique category.

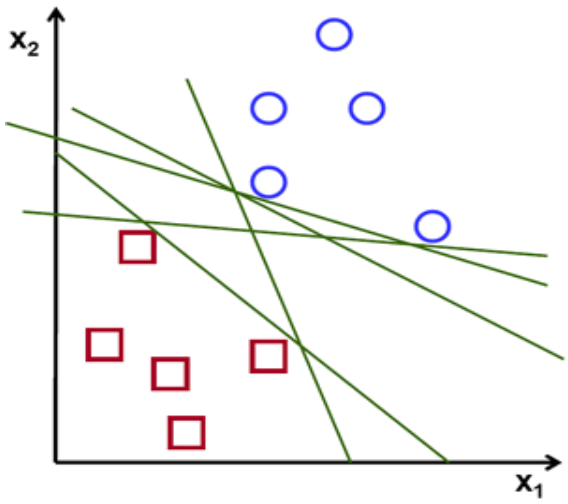

Figure 2. Possible Hyperplane

Any of a plethora of possible hyperplanes might be used to draw a line between the two collections of data. The goal is to find a plane in which the gap between the two group's data points is the largest. Maximizing the margin distance improves future data classification accuracy. Decision boundaries, in the form of hyperplanes, may be used to organize the points in the data. Data points that are on each side of the hyperplane are interpreted differently. The size of the hyperplane is determined by the total number of features. When there are just two input characteristics, the hyperplane is a straight line. The hyperplane is reduced to two dimensions when there are only three input characteristics. It's hard to see more than three features working in tandem.

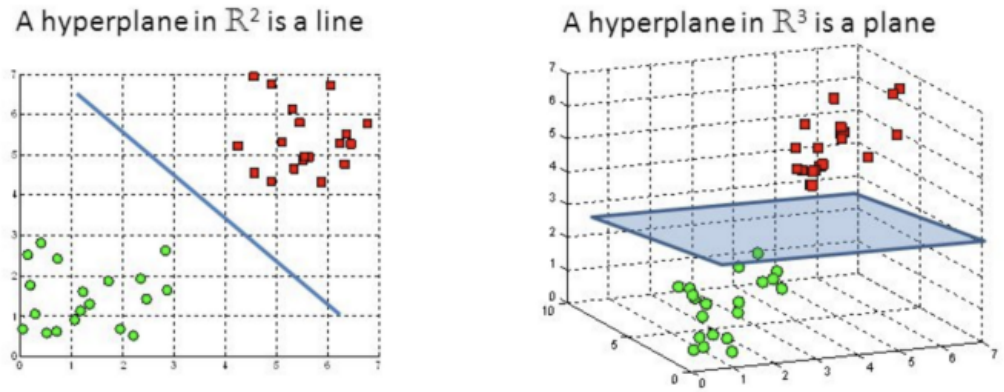

Figure 3. 2D and 3D Hyperplanes

3.6.5 Light Gradient Boosting

An open-source tool called Light Gradient Boosted Machine (or LightGBM for short) can be used to carry out the gradient boosting process quickly and effectively. In the process of boosting cases, LightGBM gives more weight to those with stronger gradients and adds a kind of independent feature selection to the gradient-boosting method. This could make the training process go much faster and improve the accuracy of the guesses. In machine learning events, LightGBM is now the most popular method for jobs that need to use tabular data for regression and classification prediction modeling. Extra

3.6.6 Tree Classifier

ExtraTrees is an ensemble ML strategy that, like Random Forests, trains a large number of decision trees and then uses the aggregated results from all of these trees to make a single prediction. However, Extra Trees and Random Forest are quite similar with just a few minor distinctions. Bagging is used in Random Forest to pick out distinct permutations of the training data to guarantee that the decision trees are sufficiently dissimilar. However, when it comes to training decision trees, Extra Trees use the complete dataset. The values at which a feature is divided to form child nodes are chosen at random, guaranteeing adequate variation across individual decision trees. On the other hand, with a Random Forest, we employ an algorithm to greedily search for and choose the value at which to divide a feature. These two distinctions apart, the similarities between Random Forest and Extra Trees are substantial. In what ways, therefore, do these alterations manifest themselves?

3.7 Evaluation Parameters

There are four evaluation parameters to measure the performance of machine learning models for treatment prediction using clinical notes. Before that, it is important to have an understanding of the confusion matrix.

3.7.1 Confusion Matrix

We may think of a binary classifier as one that labels occurrences as either "positive" or "negative". The classifier concludes that the instance belongs to the target class. A classifier trained to identify pictures of cats, for instance, would label such pictures as "positive" (when correct). Not a member of the class we are seeking to identify; the classification is negative. So, a classifier trained on cat photographs would perform well to label as "negative" any image that also contains canines but no felines. Precision, memory, and the F1-Score are all based on the principles of True Positive, True Negative, False Positive, and False Negative. These are shown in the following table (where 1 represents a positive prediction).

Table 1. Explanation of True and false positive and negative

Prediction	Actual Value	Type	Explanation
1	1	True Positive	Predited Positive and was positive
0	0	True Negative	Predited Negative and was Negative
1	0	False Positive	Predited Positive and was Negative
0	1	False Negative	Predited Negative and was positive

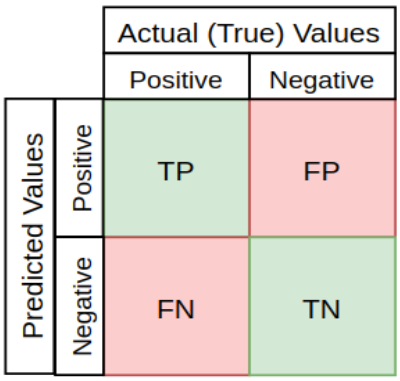

Figure 4. Confusion Matrix

Classifier estimates may be made fairer by considering metrics like Accuracy, Precision, Recall, and F1-score, since the significance of different sorts of mistakes varies between uses.

3.7.2 Precision

Precision is the percentage of times a model produces an accurate forecast (true positives). This is the formula for precision.

$$\frac{\text{True Positives}}{\text{True Positives + False Positives}} = \frac{\text{N. of Correctly Predicted Positive Instances}}{\text{N. of Total Positive Predictions you Made}} = \frac{\text{N. of Correctly Predicted People with Cancer}}{\text{N. of People you Predicted to have Cancer}}$$

3.7.3 Recall

When evaluating a classifier's performance, recall is used to determine how many out of all the positive examples in the data it properly predicted. This trait is also known as "sensitivity" at times. Formula for Recall is given below.

$$\frac{\text{True Positives}}{\text{True Positives + False Negatives}} = \frac{\text{N. of Correctly Predicted Positive Instances}}{\text{N. of Total Positive Instances in the Dataset}} = \frac{\text{N. of Correctly Predicted People with Cancer}}{\text{N. of People with Cancer in the Dataset}}$$

3.7.4 F1-Score

An F1-Score combines the value of precision and recall. The harmonic mean is a common way to explain this relationship. The harmonic mean is just an alternative to the more common arithmetic mean that is said to be more appropriate for ratios (such as precision and recall) than the latter. F1-score in this situation is calculated using the following formula:

$$2 * \frac{\text{Precision * Recall}}{\text{Precision + Recall}}$$

3.7.5 Accuracy

Most of the time, accuracy, which shows how many correct guesses there were compared to the total number of predictions, is used as the main way to judge models.

This is a popular statistic for judging a model's quality since it is easy to understand. Yet, it is frequently instructive to dig a little further.

$$\frac{\text{True Positives + True Negatives}}{\text{True Positives + True Negatives + False Positives + False Negatives}} = \frac{\text{N. of Correct Predictions}}{\text{N. of all Predictions}} = \frac{\text{N. of Correct Predictions}}{\text{Size of Dataset}}$$

4. Results and Discussion

Bag of Words (BoW) and Term Frequency-Inverse Document Frequency (TF-IDF) were utilized as feature engineering strategies with Naive Bayes in the first trial. Two weights, 1 and 2 grams are utilized in conjunction with Naive Bayes and TF-IDF. There are five distinct approaches used throughout the tests. Initially, 300 records are used in the studies. Using the BoW approach in conjunction with Naive Bayes yields a maximum accuracy of 86.75 percent on a dataset of 300 records. Following the use of these 500 records, the accuracy with BoW is 87.41%. The maximum improvement in accuracy using BoW while utilizing 1000 records is 85.17 percent. After processing 1500 records, we achieved an accuracy of 84.87%. Finally, we applied Naïve Bayes to the whole dataset, and as a result, we achieved 85.54% accuracy using BoW. The results show that Naïve Bayes achieved the maximum accuracy when combined with BoW. The table 2 shows the entire performance of Naïve Bayes.

Table 2. Evaluation Matrix of Naïve Bayes

Model	Records	Feature Engineering	Precision	Recall	F1-Score	Accuracy
Naïve Bayes	300	BoW	0.870915	0.86747	0.868056	**0.86747**
		TF-IDF 1-gram	0.843902	0.771084	0.782879	0.771084
		TF-IDF 2-gram	0.817037	0.722892	0.740801	0.722892
	500	BoW	0.878322	0.874126	0.875493	**0.874126**
		TF-IDF 1-gram	0.928956	0.72028	0.785342	0.72028
		TF-IDF 2-gram	0.952488	0.657343	0.769721	0.657343
	1000	BoW	0.845166	0.851724	0.847774	**0.851724**
		TF-IDF 1-gram	1	0.834483	0.909774	0.834483
		TF-IDF 2-gram	1	0.834483	0.909774	0.834483
	1500	BoW	0.857337	0.8487	0.852237	**0.8487**
		TF-IDF 1-gram	0.989777	0.817967	0.890744	0.817967
		TF-IDF 2-gram	1	0.806147	0.89267	0.806147
	2000	BoW	0.865796	0.855446	0.860294	**0.855446**
		TF-IDF 1-gram	0.985766	0.667327	0.788303	0.667327
		TF-IDF 2-gram	1	0.653465	0.790419	0.653465

The graphical representation of the best results on different quantities of datasets using the Naïve Bayes classifier is shown below.

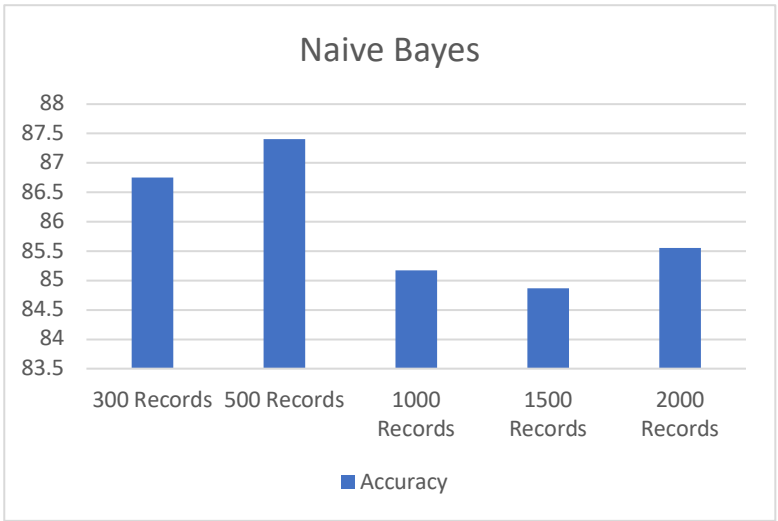

Figure 5. Results by Naive Bayes Classifier

4.1 Logistic Regression

Two feature engineering methods, Bag of Words (BoW) and TF-IDF, were used with Logistic Regression in the first trial. As part of Logistic Regression, we use TF-IDF with a 1- and 2-gram weighting scheme. There are five methods for carrying out the tests. As a first step, we use 300 data points from past trials. Logistic Regression's highest accuracy when combined with the TF-IDF (1-gram) method yields 85.54 percent when applied to a dataset of 300 records. Once 500 records have been utilized, TF-IDF accuracy increases to 85.32 percent (1-gram). The best improvement in accuracy with BoW while utilizing 1000 records is also 84.83%. After processing 1500 records, we were able to improve accuracy to 85.58%. In the end, we applied Logistic Regression on the whole dataset and achieved 86.04% accuracy using BoW. The results showed that while using BoW, Logistic Regression yielded the maximum accuracy.

Table 3. Evaluation Matrix of Logistic Regression

Model	Records	Feature Engineering	Precision	Recall	F1-Score	Accuracy
Logistic Regression	300	BoW	0.820799	0.81928	0.818962	0.819277
		TF-IDF 1-gram	0.857183	0.85542	0.855804	**0.855422**
		TF-IDF 2-gram	0.831325	0.83133	0.831325	0.831325
	500	BoW	0.86772	0.83916	0.846259	0.839161
		TF-IDF 1-gram	0.874173	0.85315	0.858372	**0.853147**
		TF-IDF 2-gram	0.858241	0.81119	0.823174	0.811189
	1000	BoW	0.917498	0.84828	0.875364	**0.848276**
		TF-IDF 1-gram	0.836527	0.82414	0.8298	0.824138
		TF-IDF 2-gram	1	0.83448	0.909774	0.834483
	1500	BoW	0.893639	0.85579	0.873021	**0.855792**
		TF-IDF 1-gram	0.842066	0.81797	0.82942	0.817967
		TF-IDF 2-gram	0.843625	0.81797	0.8303	0.817967
	2000	BoW	0.884931	0.85941	0.870997	**0.859406**
		TF-IDF 1-gram	0.835467	0.82574	0.829235	0.825743
		TF-IDF 2-gram	0.840804	0.82574	0.831104	0.825743

The graphical representation of the best results on different quantities of datasets using the Logistic Regression classifier is shown below.

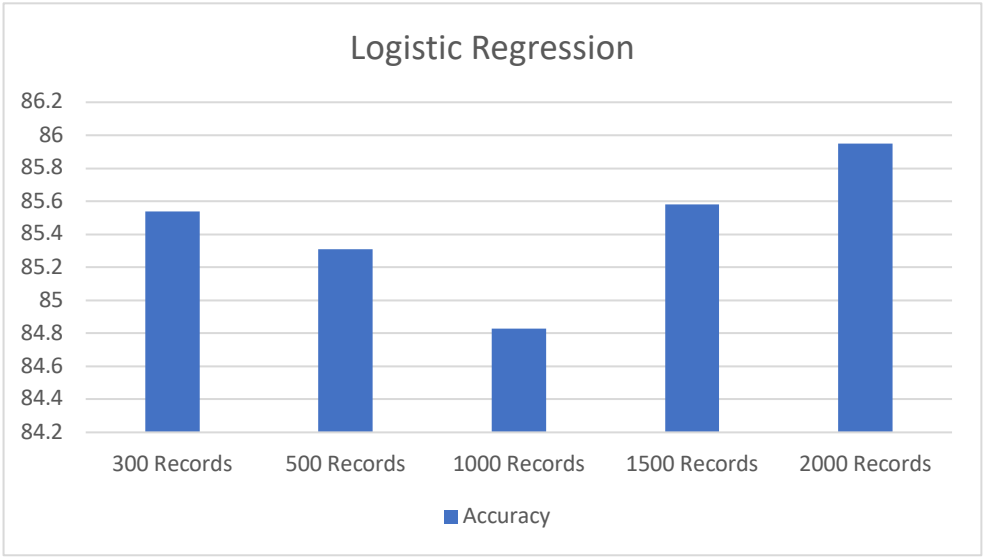

Figure 6. Graph of results by Logistic Regression

4.2 Light Gradient Boosting

Bag of Words (BoW) and Term Frequency Inverse Document Frequency (TF-IDF) were utilized as feature engineering strategies in the initial experiment with Light GBM. TF-IDF may be used with Light GBM at two different weights, 1 and 2 grams. There are five methods for carrying out the tests. As a first step, we use 300 data points from past trials. Using the TF-IDF (2-grams) method with Light GBM, the highest accuracy achieved was 80.72% on 300 records. By the end of the 500th record, the accuracy with TF-IDF was 79.72 percent (2-grams). Similarly, TF-IDF improves accuracy by 82.41% when applied to 1000 records (1-grams). More than 1,500 records have been used, with the best accuracy of 86.53 percent. After training on the whole dataset using Light GBM, we improved accuracy by 83.56% using BoW. This study concludes that Light GBM achieved the maximum accuracy when combined with BoW.

Table 4. Evaluation Matrix of Light GBM

Model	Records	Feature Engineering	Precision	Recall	F1-Score	Accuracy
Light GBM	300	BoW	0.79907	0.79518	0.796086	0.795181
		TF-IDF 1-gram	0.804587	0.79518	0.797003	0.795181
		TF-IDF 2-gram	0.813486	0.80723	0.808482	**0.807229**
	500	BoW	0.842525	0.78322	0.799337	0.783217
		TF-IDF 1-gram	0.842525	0.78322	0.799337	0.783217
		TF-IDF 2-gram	0.845416	0.7972	0.810075	**0.797203**
	1000	BoW	0.887754	0.82414	0.850781	0.824138
		TF-IDF 1-gram	0.895168	0.82414	0.8539	**0.824138**
		TF-IDF 2-gram	0.896362	0.82069	0.852703	0.82069
	1500	BoW	0.910704	0.86525	0.885505	**0.865248**
		TF-IDF 1-gram	0.89169	0.85579	0.871914	0.855792
		TF-IDF 2-gram	0.900354	0.85816	0.877083	0.858156
	2000	BoW	0.88223	0.83564	0.854421	**0.835644**
		TF-IDF 1-gram	0.888914	0.83366	0.856429	0.833663
		TF-IDF 2-gram	0.882199	0.83564	0.854753	0.835644

The graphical representation of the best results on different quantities of datasets using the Light GBM classifier is shown below.

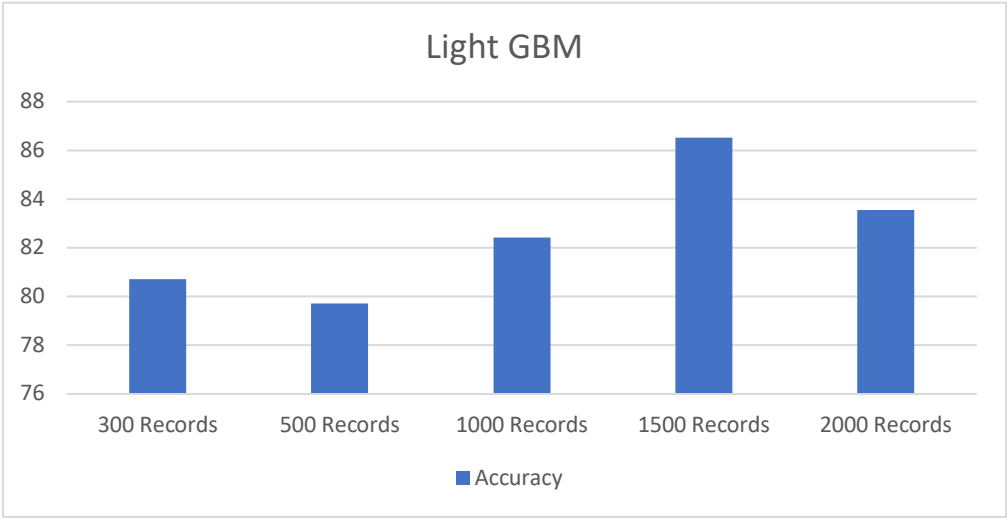

Figure 7. Graph of results by Light GBM

4.3 Random Forest

Bag of Words (BoW) and TF-IDF were utilized as feature engineering approaches with Random Forest in the first trial. Random Forest employs TF-IDF with two different weights, 1 and 2. There are five

methods for carrying out the tests. As a first step, we use 300 data points from past trials. Using the TF-IDF (1-gram) method in conjunction with Random Forest, we can get an accuracy of 80.72 percent on a dataset of 300 records. Following this, 500 records have been utilized, and the accuracy with BoW is 77.62%. Using the same 1000-record dataset, BoW can improve accuracy by a maximum of 20%. After processing 1500 records, we were able to improve accuracy to 80.85%. At last, Random Forest was applied to the whole dataset, and TF-IDF accuracy was improved to 80.99%. (1-gram). It was determined that Random Forest produced the maximum accuracy while using TF-IDF (1-gram).

Table 5. Evaluation Matrix of Random Forest

Model	Records	Feature Engineering	Precision	Recall	F1-Score	Accuracy
Random Forest	300	BoW	0.761192	0.75904	0.759673	0.759036
		TF-IDF 1-gram	0.807229	0.80723	0.807229	**0.807229**
		TF-IDF 2-gram	0.775333	0.74699	0.752481	0.746988
	500	BoW	0.852813	0.77622	0.796891	**0.776224**
		TF-IDF 1-gram	0.865818	0.76224	0.790724	0.762238
		TF-IDF 2-gram	0.870907	0.73427	0.774599	0.734266
	1000	BoW	0.857576	0.8	0.825382	**0.8**
		TF-IDF 1-gram	0.857576	0.8	0.825382	0.8
		TF-IDF 2-gram	0.871567	0.78276	0.823633	0.782759
	1500	BoW	0.843523	0.80615	0.824352	0.806147
		TF-IDF 1-gram	0.845755	0.80851	0.826585	**0.808511**
		TF-IDF 2-gram	0.842505	0.80378	0.822682	0.803783
	2000	BoW	0.835449	0.80594	0.819858	0.805941
		TF-IDF 1-gram	0.839275	0.8099	0.823675	**0.809901**
		TF-IDF 2-gram	0.835909	0.80396	0.818366	0.80396

The graphical representation of the best results on different quantities of datasets using the Random Forest classifier is shown below.

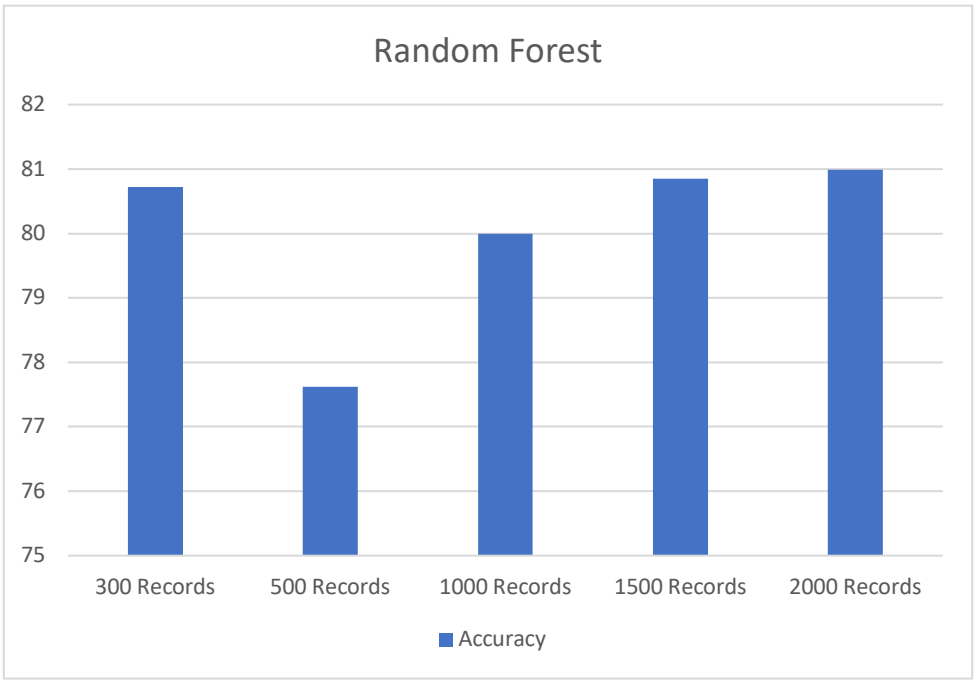

Figure 8. Graph of results by Random Forest

4.4 Support Vector Machine

Two feature engineering methods, Bag of Words (BoW) and TF-IDF, were utilized with Support Vector Machine in the first trial. As part of a Support Vector Machine, TF-IDF is applied with a gram weight of 1 and a gram weight of 2. There are five methods for carrying out the tests. As a first step, we use 300 data points from past trials. When employing the TF-IDF (2-gram) approach with Support Vector Machine, the highest accuracy achieved was 84.33% on a dataset of 300 records. Once 500 records have been utilized, TF-IDF accuracy increases to 85.32 percent (1-gram). The maximum accuracy improvement using TF-IDF while utilizing 1000 records is 81.72 percent (1-gram). Then, TF-IDF was used, and the maximum accuracy achieved was 83.69% with 1500 records (1-gram). In the end, we applied Support Vector Machine to the whole dataset and improved accuracies using TF-IDF by 84.16 percentage points (2-gram). The results show that Support Vector Machine achieved the maximum accuracy when fed TF-IDF (2-gram).

Table 6. Evaluation Matrix of Support Vector Machine

Model	Records	Feature Engineering	Precision	Recall	F1-Score	Accuracy
Support Vector Machine	300	BoW	0.786035	0.78313	0.782754	0.783133
		TF-IDF 1-gram	0.843345	0.84337	0.843236	0.843373
		TF-IDF 2-gram	0.843986	0.84337	0.843557	**0.843373**
	500	BoW	0.844472	0.81818	0.825412	0.818182
		TF-IDF 1-gram	0.874173	0.85315	0.858372	**0.853147**
		TF-IDF 2-gram	0.86772	0.83916	0.846259	0.839161
	1000	BoW	0.837774	0.82759	0.832278	0.827586
		TF-IDF 1-gram	0.855625	0.81724	0.833805	**0.817241**
		TF-IDF 2-gram	0.851946	0.7931	0.819361	0.793103
	1500	BoW	0.820064	0.8156	0.817753	0.815603
		TF-IDF 1-gram	0.864966	0.83688	0.850126	**0.836879**
		TF-IDF 2-gram	0.85454	0.82033	0.836709	0.820331
	2000	BoW	0.804134	0.80396	0.803454	0.80396
		TF-IDF 1-gram	0.847226	0.84158	0.843565	**0.841584**
		TF-IDF 2-gram	0.84612	0.82574	0.833427	0.825743

The graphical representation of the best results on different quantities of datasets using the Support Vector Machine classifier is shown below.

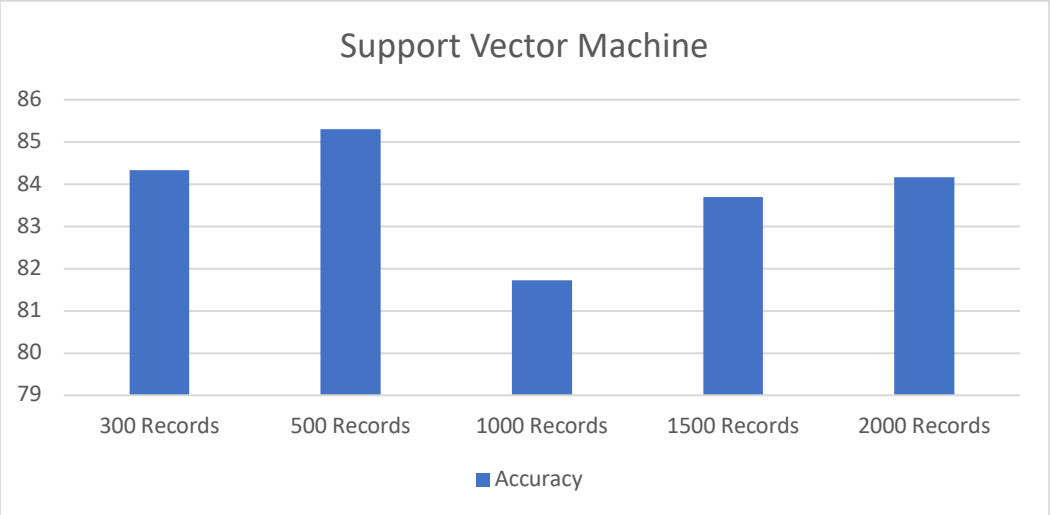

Figure 9. Graph of results by Support Vector Machine

4.5 Extra Tree Classifier

The Extra Tree Classifier was tested in conjunction with the Bag of Words (BoW) and TF-IDF feature engineering methods. Specifically, Extra Tree Classifier employs TF-IDF with a 2-gram weight and a 1-gram weight. There are five methods for carrying out the tests. As a first step, we use 300 data points from past trials. When employing the TF-IDF (1-gram) method with Extra Tree Classifier, the highest accuracy achieved was 78.31% on a dataset of 300 records. Thereafter, when 500 records were utilized, the accuracy increased to 80.41 percent using BoW. In a similar vein, the best improvement in accuracy with BoW while employing 1000 records is 80.69 percent. Then, TF-IDF was applied to the same set of 1500 records, and the best accuracy achieved was 80.85%. (1-gram). Ultimately, we applied Extra Tree Classifier on the whole dataset and improved accuracy by 81.38% using BoW. The results show that when combined with BoW, Extra Tree Classifier produces the maximum accuracy.

Table 7. Evaluation Matrix of Extra Tree Classifier

Model	Records	Feature Engineering	Precision	Recall	F1-Score	Accuracy
Extra Tree Classifier	300	BoW	0.771845	0.77108	0.771353	0.771084
		TF-IDF 1-gram	0.789586	0.78313	0.784542	**0.783133**
		TF-IDF 2-gram	0.775333	0.74699	0.752481	0.746988
	500	BoW	0.856584	0.8042	0.817669	**0.804196**
		TF-IDF 1-gram	0.854128	0.79021	0.80702	0.79021
		TF-IDF 2-gram	0.870907	0.73427	0.774599	0.734266
	1000	BoW	0.863205	0.8069	0.831403	**0.806897**
		TF-IDF 1-gram	0.863969	0.80345	0.829954	0.803448
		TF-IDF 2-gram	0.869942	0.78621	0.824377	0.786207
	1500	BoW	0.840714	0.80615	0.822922	0.806147
		TF-IDF 1-gram	0.843269	0.80851	0.825379	**0.808511**
		TF-IDF 2-gram	0.839443	0.80142	0.81999	0.801418
	2000	BoW	0.838473	0.81386	0.825409	**0.813861**
		TF-IDF 1-gram	0.833596	0.80792	0.819414	0.807921
		TF-IDF 2-gram	0.826228	0.79604	0.808693	0.79604

The graphical representation of the best results on different quantities of datasets using the Extra Tree Classifier is shown below.

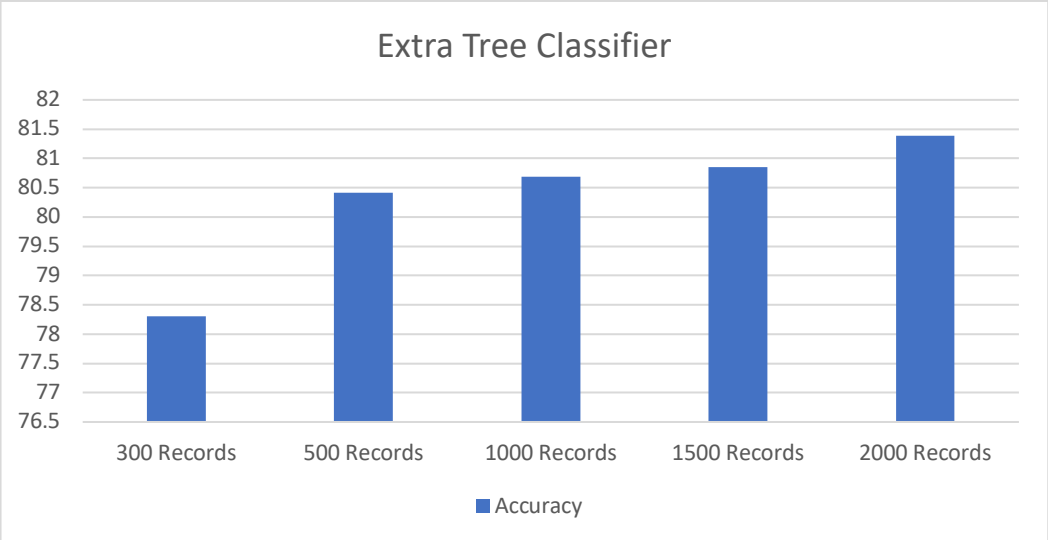

Figure 10. Graph of results by Extra Tree Classifier

5. Conclusion

In the end, we uncovered a number of interesting details about the dataset as well as the effectiveness of feature engineering methods and machine learning models. The dataset utilized in this investigation contains a total of 2000 individual records with various clinical notes. Different types of textual information on patients are included in the various clinical notes, each of which is associated with a certain medical specialty. In the second set of columns, we provide therapy options based on the patient's medical history as documented in clinical notes. Before analyzing the text, it must be cleaned using various Natural Language Processing methods, such as the removal of stop words, punctuation, tokenization, etc. After the text has been cleaned, several feature engineering methods have been applied to it, including TF-IDF(1-gram and 2-grams) and Bag of Words (BoW). Then, the machine learning models' training and testing datasets are divided at 70:30. In order to anticipate therapies based on patient's clinical notes, six distinct machine learning models are used. Among these ML models are the Naive Bayes, Logistic Regression, Random Forest, Light GBM, Support Vector Machine, and Extra Tree Classifiers. Naive Bayes achieved the maximum accuracy, 86.75% with BoW, on a dataset of 300 records. Naive Bayes again outperformed all other methods with an accuracy of 87.41% using just 500 records from the training set. Naive Bayes again performed best when utilizing 1000 records, this time with an accuracy of 85.17 percent using BoW. Once again, after using up to 1500 records, Light GBM comes out on top with BoW. With an accuracy of 86.53 percent, Light GBM easily beat out other models. Using the BoW feature engineering approach, Logistic Regression performed best at the complete dataset level, with an accuracy of 85.94%. For this dataset, it is concluded that machine learning models do well when fed a Bag of Words (BoW).

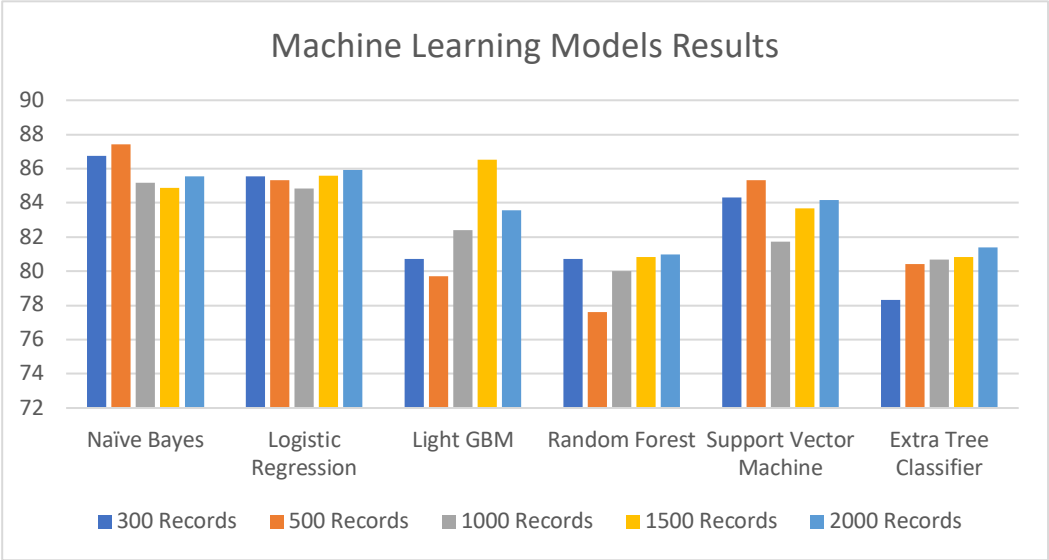

Figure 11. Overall accuracy scores of machine learning models

References

1 C. S. Lee, A. J. Tyring, Y. Wu, S. Xiao, A. S. Rokem, N. P. DeRuyter, Q. Zhang, A. Tufail, R. k. Wang and A. Y. Lee, "Generating retinal flow maps from structural optical coherence tomography with artificial intelligence," Scientific Reprots, vol. 9, pp. 1-11, 2019.

2 B. J. Marafino, J. M. Davies, N. S. Bardach, M. L. Dean and R. A. Dudley, "N-gram support vector machines for scalable procedure and diagnosis classification, with applications to clinical free text data from the intensive care unit," Journal of the American Medical Informatics Association, vol. 21, no. 5, pp. 871-875, 2014.

3 S. C. Derderian, C. Jeanty, M. C. Walters, E. Vichinsky and T. C. MacKenzie, "In utero hematopoietic cell transplantation for hemoglobinopathies," Frontiers in Pharmacology, vol. 5, no. 278, 2015.

4 Y. Li, J. Li, W. Li and H. Du, "A state-of-the-art review on magnetorheological elastomer devices," Smart Materials and Structures, vol. 23, no. 12, pp. 1-13, 2014.

5 S. A. Moqurrab, U. Ayub, A. Anjum, S. Asghar and G. Srivastava, "An accurate deep learning model for clinical entity recognition from clinical notes," IEEE Journal of Biomedical and Health Informatics, vol. 25, no. 10, pp. 3804-3811, 2021.

6 H. Lu, L. Ehwerhemuepha and C. Rakovski, "A comparative study on deep learning models for text classification of unstructured medical notes with various levels of class imbalance," BMC medical research methodology, vol. 22, no. 1, pp. 1-12, 2022.

7 E. Lehman, S. Jain, K. Pichotta, Y. Goldberg and B. C. Wallace, "Does BERT Pretrained on Clinical Notes Reveal Sensitive Data?," arXiv preprint arXiv:2104.07762, pp. 1-9, 2021.

8 B.-H. Kim and V. Ganapathi, "Read, Attend, and Code: Pushing the Limits of Medical Codes Prediction from Clinical Notes by Machines," in Procedding of 6th Machine Learning for Healthcare Conference, 2021.

9 F. Pethani and A. G. Dunn, "Natural language processing for clinical notes in dentistry: A systematic review," Journal of Biomedical Informatics, vol. 138, pp. 1-10, 2023.

10 S. Razdan and S. Sharma, "Internet of Medical Things (IoMT): Overview, Emerging Technologies, and Case Studies," IETE technical review, vol. 39, no. 4, pp. 775-788, 2022.

11 C. Lindvall, C.-Y. Deng, N. D. Agaronnik, S. Samineni, R. Umeton, W. M. Jenkins, K. L. Kehl, j. A. Tulsky and A. C. Enziger, "Deep Learning for Cancer Symptoms Monitoring on the Basis of Electronic Health Record Unstructured Clinical Notes," JCO Clinical Cancer Informatics, vol. 6, pp. 1-10, 2022.

12 I. Spacic and G. Nenadic, "Clinical Text Data in Machine Learning: Systematic Review," JMIR medical informatics, vol. 3, no. 8, pp. 1-7, 2020.

13 K. Y. Ngiam and I. W. Khor, "Big data and machine learning algorithms for health-care delivery," The Lancet Oncology, vol. 20, no. 5, pp. 262-273, 2019.

14 M. H. Stanfill and D. T. Marc, "Health Information Management: Implications of Artificial Intelligence on Healthcare Data and Information Management," Yearbook of medical informatics, vol. 28, no. 1, pp. 56-64, 2019.

15 H. S. R. Rajula, G. Verlato, M. Manchia, N. Antonucci and V. Fanos, "Comparison of Conventional Statistical Methods with Machine Learning in Medicine: Diagnosis, Drug Development, and Treatment," Medicina, vol. 56, no. 9, pp. 1-10, 2020.

16 J. Sata, A. Vitale, G. Lopalco, R. M. R. Pereira, H. F. Giordano and J. Makowska, "Efficacy and safety of tocilizumab in adult-onset Still's disease: Real-life experience from the international AIDA registry," eminars in Arthritis and Rheumatism, vol. 57, pp. 1-6, 2022.

17 J. Yuan, C. Holtz, T. Smith and J. Luo, "Autism spectrum disorder detection from semi-structured and unstructured medical data," EURASIP Journal on Bioinformatics and Systems Biology, pp. 1-9, 2017.

18 S. T. Wu, Y. J. Juhn, S. Sohn and H. Liu, "Patient-level temporal aggregation for text-based asthma status ascertainment," Journal of the American Medical Informatics Association, vol. 21, no. 15, pp. 876-884, 2014.

19 R. J. Byrd, S. R. Steinhubl, J. Sun, S. Ebadollahi and W. F. Stewart, "Automatic identification of heart failure diagnostic criteria, using text analysis of clinical notes from electronic health records," International journal of medical informatics, vol. 83, no. 12, pp. 983-992, 2014.

20 G. Gonzalez-Hernandez, A. Sarker, K. O'Connor and G. Savova, "Capturing the Patient's Perspective: a Review of Advances in Natural Language Processing of Health-Related Text," Yearbook of medical informatics, vol. 26, no. 01, pp. 214-227, 2017.

21 C. Lin, E. W. Karlson, H. Canhao, T. A. Miller, D. Dligach, Y. Shen and G. K. Savova, "Automatic Prediction of Rheumatoid Arthritis Disease Activity from the Electronic Medical Records," PloS one, vol. 8, no. 8, pp. 26-35, 2013.

22 J. G. Adeva, J. P. Atxa, M. U. Carrillo and E. A. Zengotitabengoa, "Automatic text classification to support systematic reviews in medicine," Expert Systems with Applications, vol. 41, no. 4, pp. 1498-1508, 2014.

23 S. Rabhi, J. Jakubowicz and M.-H. Metzger, "Deep Learning versus Conventional Machine Learning for Detection of Healthcare-Associated Infections in French Clinical Narratives," Methods of information in medicine, vol. 58, no. 1, pp. 31-41, 2019.

24 H. P. Martinez, Y. Bengio and G. N. Yannakakis, "Learning deep physiological models of affect," IEEE Computational Intelligence Magazine, vol. 8, no. 2, pp. 20-23, 2013.

25 A. Krizhevsky, I. Sutskever and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Communications of the ACM, vol. 60, no. 6, pp. 84-90, 2017.

26 C.-E. Hsu, K.-C. Huang, T.-C. Lin, K.-M. Tong, M.-H. Lee and Y.-C. Chiu, "Integrated risk scoring model for predicting dynamic hip screw treatment outcome of intertrochanteric fracture," Injury, vol. 47, no. 11, pp. 2501-2506, 2016.

27 C. Haug and J. M. Drazen, "Artificial Intelligence and Machine Learning in Clinical Medicine," New England Journal of Medicine, vol. 388, no. 13, pp. 1201-1208, 2023.

28 S. Kocbek, L. Cavedon, D. Martinez, C. Bain, C. M. Manus, G. Haffari, I. Zukerman and K. Verspoor, "Text mining electronic hospital records to automatically classify admissions against disease: Measuring the impact of linking data sources," Journal of Biomedical Informatics, vol. 64, no. 1, pp. 158-167, 2016.

29 J. Chaki, S. T. Ganesh, S. Cidham and S. A. Theertan, "Machine learning and artificial intelligence based Diabetes Mellitus detection and self-management: A systematic review," Journal of King Saud University-Computer and Information Sciences, vol. 34, no. 6, pp. 3204-3225, 2022.