Disease Classification Based on Clinical Notes

By

Sundas Ashraf

(Registration No.: 00000364512

Supervisor: Dr. Usman Qamar

DEPARTMENT OF COMPUTER & SOFTWARE ENGINEERING

COLLEGE OF ELECTRICAL & MECHANICAL ENGINEERING

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY

ISLAMABAD

JULY, 2023

Annex A

THESIS ACCEPTANCE CERTIFICATE

Certified that final copy of MS/MPhil thesis written by **NS Sundas Ashraf** Registration No. **00000364512**, of College of E&ME has been vetted by undersigned, found complete in all respects as per NUST Statutes/Regulations, is free of plagiarism, errors and mistakes and is accepted as partial fulfillment for award of MS/MPhil degree. It is further certified that necessary amendments as pointed out by GEC members of the scholar have also been incorporated in the thesis.

Signature : _____

Name of Supervisor: **Dr Usman Qamar**

Date: 19-07-2023

Signature of HOD: _____
(Dr Usman Qamar)
Date: 19-07-2023

Signature of Dean: _____
(Brig Dr Nasir Rashid)
Date: 19 JUL 2023

Dedicated to my exceptional parents: ***Muhammad Ashraf & Kauser Parveen*** *and my Husband* ***Haseeb*** *whose tremendous support and cooperation led me to this accomplishment.*

Acknowledgement

All praise and glory to Almighty Allah (the most merciful, the most beneficent) who gave me the courage, patience, knowledge and ability to carry out this work and to persevere and complete it satisfactorily. Undoubtedly, HE eased my way and without HIS blessings I can achieve nothing.

I would like to express my earnest appreciation to my advisor Dr. Usman Qamar for boosting my morale and for his continual support, motivation, dedication and valuable guidance. I am blessed to have such a co-operative and kind mentor for my research.

My acknowledgement would be incomplete without thanking the greatest source of my strength, my family. I am abundantly thankful to my beloved parents who raised me when I was not able of walking and proceeded to support me throughout in every phase of my life.

Finally, I would like to express my gratitude to my beloved husband Haseeb who encouraged and supported me through this entire period.

Abstract

Effective disease classification plays a crucial role in healthcare for accurate diagnosis, treatment, planning, and patient management. With the increasing adoption of electronic health records (EHRs), there is a vast amount of clinical data available that can potentially be leveraged for disease classification. Due to the rapid growth in the volume of clinical data generated, Healthcare providers face a significant challenge in extracting meaningful insights from electronic health records. In this regard Natural Language Processing techniques can assist in identifying and extracting important clinical information from these records and assist healthcare practitioners for accurate diagnosis.

The study utilizes a large dataset of EHRs from a diverse patient population, encompassing a wide range of diseases and medical conditions. Natural language processing (NLP) techniques are employed to extract and preprocess clinical notes, ensuring the removal of un-necessary patient information while retaining the essential clinical details. Feature engineering is applied to transform the unstructured clinical text into a structured representation suitable for machine learning algorithms.

A variety of machine learning models, including Support Vector Machines (SVM), Passive Aggressive Classifier, Naïve Bayes and Logistic Regression are trained and evaluated on the dataset. Performance metrics such as accuracy, precision, recall, and F1 score are used to assess the classification models' effectiveness in accurately predicting the presence or absence of specific diseases based on the clinical notes. The results demonstrate that the proposed disease classification system achieves high accuracy of 98% across multiple diseases using SVM classifier.

The research demonstrates the effectiveness of machine learning models in accurately classifying diseases based on these clinical notes. The system's accuracy and performance highlight its potential for enhancing healthcare delivery and decision-making, contributing to improved patient care and outcomes.

Key Words: *Natural Language Processing, Text Classification, Feature Engineering, Support Vector Machine*

Table of Contents

List of Figures

List of Table

List of Abbreviations

KDD	*Knowledge Discovery in Databases*
EHR	*Electronic Health Record*
NLP	*Natural Language Processing*
ML	*Machine Learning*
BOW	*Bag Of Word*
TF	*Term Frequency*
IDF	*Inverse Document Frequency*
SVM	*Support Vector Machine*
NB	*Naïve Bayes*
NLTK	*Natural Language Tool Kit*
CBOW	*Continuous Bag of Words*
LR	*Logistic Regression*
IDE	*Integrated Development Environment*

CHAPTER 1: INTRODUCTION

In today's world, a large amount of data is available. According to research, data of approximately 328.77 million terabytes is generated every day [1]. Such huge volumes of data are useless if we are unable to process and get deep insight out of it for decision making.

Data is useless until it is converted into a valuable knowledge. Manually it is not possible to manage data and convert it into useful information. This will just increase the cost and slow down the process. As the data increases, manual data analysis will become more difficult. To deal with it, proper filtering of data is required. An example is Knowledge Discovery in Databases (KDD). It is shown in figure 1.1.

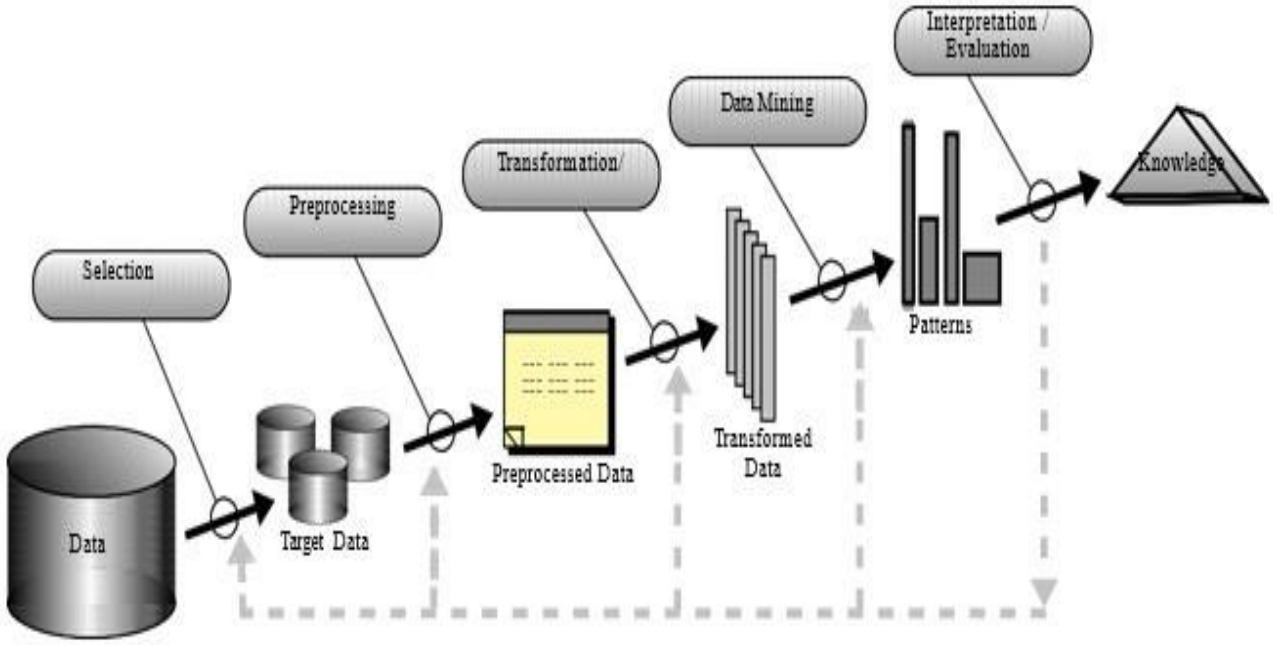

Figure 1-1: Knowledge Discovery in Database

- Data Selection: In this step, data is selected from different or one data repository and converted into a single source.
- Data Cleansing/ Pre-processing: In this step, accuracy and reliability of the selected data is judged. Outliers are detected and removed in cleansing process. A lot of techniques are available for pre-processing / Cleansing of data.

- Data Transformation/Reduction: In this step, data is reduced applying different techniques. Data contains attributes that are of no worth. To eliminate the unnecessary attributes some methods are adopted. More common techniques for this purpose are feature extraction, feature selection, attribute discretization etc. Main purpose to reduce the amount of data is to enhance the performance of knowledge extraction.
- Data Mining: Data mining is applied to discover the patterns/information that are hidden. Data mining is the process used to analyze the data and obtain useful information from it [2].

Interpretation/Evaluation: After the discovery of information patterns, other goals are checked on that according to the domain and scenario

1.1 Evolution in healthcare

In the rapidly evolving landscape of healthcare, the process of generating clinical notes has evolved from traditional paper-based methods to digital format. . With the exponential growth of electronic health records (EHRs) and the wealth of information contained within clinical notes, there arises an opportunity to leverage advanced technologies for disease classification.

The efficient and accurate diagnosis of diseases remains one of the most critical challenges faced by medical professionals. This thesis report delves into the exciting field of Natural Language Processing (NLP) and Machine Learning (ML) to develop a framework for disease classification, aimed at assisting healthcare practitioners in making well-informed and timely decisions.

1.2 Background

Clinical notes, the unstructured narratives recorded by healthcare providers during patient encounters, provide invaluable insights into a patient's medical history, symptoms, and treatment plans. Extracting meaningful information from this vast corpus of unstructured data is a complex and challenging task. Traditional methods of disease diagnosis often rely on manual examination of clinical notes, leading to potential human errors, inefficiencies, and delays in treatment decisions.

1.3 Purpose

The primary objective of this thesis is to design, develop, and evaluate a disease classification system that can efficiently analyze clinical notes to identify and categorize various diseases accurately. By harnessing the power of NLP and ML techniques, this study aims to empower healthcare professionals with a robust decision-support tool, which can significantly improve the accuracy and speed of disease diagnosis, leading to better patient outcomes.

1.4 Significance

The potential impact of this research is substantial and multifaceted. Firstly, it can significantly reduce the burden on healthcare practitioners by automating the laborious and time-consuming task of disease classification from clinical notes. Secondly, it has the potential to minimize diagnostic errors and improve patient safety, ensuring that appropriate treatments are prescribed promptly. Additionally, the proposed framework can aid in early disease detection, facilitating timely interventions and preventive measures.

1.5 Methodology

The methodology adopted for this research involves a comprehensive approach, combining NLP techniques for text preprocessing, feature extraction, and along with a range of state-of-the-art ML algorithms for classification tasks. The process will include pre-processing, model development, and rigorous evaluation to ensure the reliability and generalizability of the proposed solution.

1.6 Relevance to National Needs

Disease classification based on clinical notes using NLP techniques has significant relevance to national needs in healthcare. With the increasing availability of electronic health records, there is a need for more efficient and accurate disease diagnosis tools that can improve patient outcomes and reduce healthcare costs. NLP-based disease classification tools have the potential

to fill this need by automating the process of disease diagnosis, thereby reducing the burden on healthcare providers and improving patient care.

1.7 Organization

The remaining research work is organized in the following manner:

Chapter 2: gives review of the up-to-the-minute algorithms proposed by researchers for classification of disease based using free-text clinical notes.

Chapter 3: consists of the proposed methodology in detail. It includes dataset explanation, feature extraction from raw-text followed by their classification

Chapter 4: all the experimental results are discussed in detail with all desired figures and tables.

Chapter 5: concludes the thesis and reveals future scope of this research

CHAPTER 2: LITERATURE REVIEW

In recent years, the exponential growth of electronic health records (EHRs) and the availability of vast amounts of textual data from clinical notes have presented both opportunities and challenges in the healthcare industry. These clinical notes contain critical information about patient diagnoses, treatments, and outcomes, making them a valuable resource for clinical decision-making and research. However, it remains difficult for healthcare professionals to get meaningful insights efficiently due to un-structured nature of clinical notes. For this purpose various Machine Learning (ML) techniques have emerged as powerful tool for text classification, enabling automatic categorization of clinical notes based on their content. This chapter highlights the some valuable contribution in the field of text classification using clinical notes. This chapter also highlights our research sequence phase which further consist of sub phases as explained in next sub section.

2.1 Research Sequence

In this section, we will explain the steps which we follow to do the literature review for our thesis.

2.1.1 Research Questions

Following are the research questions for our thesis.

- ✓ Which disease have been classified using free-text clinical notes?
- ✓ Which pre-processing techniques are considered for textual data?
- ✓ What are the classical methods that have been adopted by researchers?
- ✓ Which datasets are used for subject mentioned research?

2.1.2 Review Period

Our review period is from 2019 onwards, because we want to stick with recent quality research. For that purpose, we reject old studies.

2.1.3 Objective of Literature Review

Following are the objectives of our literature review:

- ✓ Our main objective is to survey the literature of the research questions described above.
- ✓ To find gaps in the current literature
- ✓ Identify future research areas.
- ✓ To present our study and research in an organized way.

2.1.4 Keywords

Following are the keywords for our literature review:

- ✓ Feature Selection
- ✓ Disease Classification
- ✓ Clinical text data analysis
- ✓ Clinical notes evaluation

2.1.5 Inclusion/Exclusion Criteria

Following are the inclusion and exclusion criteria of our literature review:

- ✓ Paper properly related to keywords are selected
- ✓ Review period described in section 2.1.2 is taken into consideration
- ✓ Authentic and valid researches are included
- ✓ Irrelevant researches are excluded

2.2 Related Work

Several studies have explored the application of machine learning techniques for text classification of clinical notes in the context of disease classification. For instance Akib Mohi Ud Din Khanday et al. [3] classified textual clinical reports into four classes by using classical and ensemble machine learning algorithms. They used feature extraction techniques like Term frequency/inverse document frequency (TF/IDF), Bag of words (BOW) and report length. These features were supplied to traditional and ensemble machine learning classifiers. Logistic regression and Multinomial Naïve Bayes showed better results than other ML algorithms by having 96.2% testing accuracy.

Theresa A. Koleck et al. [4] documented a systematic literature review for Natural Language Processing (NLP) techniques to process and analyze different symptom information present in Electronic Health Record (EHR) free-text narratives. They had selected 27 eligible articles out of 1964 records from PubMed and EMBACE. A wide variety of symptoms are addressed in these studies including shortness of breath, pain, nausea, dizziness, disturbed sleep, constipation, and depressed mood. Their findings showed that various NLP tools, classification methods, and manually curated rule-based processing are being used to extract information from EHR free-text narratives for categorization of general, cardiology, and mental health disease.

Daniel J. Feller et al. [5] explored different NLP models for Automated HIV Risk Assessment using clinical notes. They have used two NLP methods for feature extraction i.e. Automated keyword identification and Automated topic modeling. They have developed three different predictive algorithms based upon Random forest ML classifier i.e. Baseline model with only structured EHR data, baseline plus NLP topics, baseline plus NLP clinical keywords. Baseline plus NLP clinical keywords showed highest precision of 74% for classification of HIV disease.

Seyedmostafa Sheikhalishahi et al. [6] performed a systematic literature review for NLP of clinical notes on Chronic disease. They analyzed 106 articles out of 2652 searched records. Their findings showed that there is a significant increase in the use of machine learning methods as compared to rule-based approaches. However deep learning methods are in continuous phase of evolution and development. Most frequently chronic diseases are classified into four groups i.e. Circulatory system, neoplasms, endocrine, nutritional & metabolic diseases and other

disease. They concluded that SVM and Naïve Bayes are most frequently used NLP methods for text classification purpose.

Theresa A. Koleck et al. [7] proposed a method that combines standardized vocabularies, clinical expertise, and natural language processing (NLP) techniques to generate comprehensive symptom vocabularies and identify accurately symptom information in EHR clinical notes. Five different disease categorizes have been considered in this research i.e. Constipation, depressed mood, disturbed sleep, fatigue, and palpitations. They identified up to 11 times more additional keywords for each class making NLP system more efficient resulting F-measure score from 0.80 to 0.96.

Huiying Liang et al. [8] used deep learning techniques for extraction of clinically relevant information from Electronic Health Record (EHR). They used Logistic regression classifier to build hierarchical diagnostic system. Primarily it classified EHR notes into main five broad organ system i.e. Respiratory, gastrointestinal, neuropsychiatric, genitourinary, and systemic or generalized conditions. Then with in each organ system there is sub organ classification i.e. respiratory system, is further divided into upper and lower respiratory conditions. This model showed average F1 score of 88.5%.

DEVESH UPRETI [9] has used Symptom2Disease dataset. He classified 24 disease classes and SVM and word embedding techniques and showed 93% of accuracy.

2.3 Review Analysis

Although a lot of work has been done using Machine Learning classifier and NLP techniques. Most of past researchers used their own private datasets those are not publically available. Based upon availability of their limited datasets. They had categorized mainly 1 disease or general group of disease.

UCI dataset [17] that we are targeting for this research has not been used for disease classification also disease classes that we are focusing on to classify have not been addressed by past researchers.

Table 2-1: Literature review summary of features extraction and classification techniques

Year	Type	Author	Methodology	No of classes	Accuracy
Springer, 2020	Journal	AkibMohi Ud Din Khanday et al. [3]	TF/IDF, Bag of Words(BOW), reportlength, ensemble classifier (Logisticregression+ Multinomial Naïve Bayes)	4	96.2%
PubMed, 2019	Journal	Theresa A. Koleck et al. [4]	NLP tools, classification methods, and manually curated rule-based processing	3	-
PubMed, 2019	Journal	Daniel J. Feller et al. [5]	(TF-IDF), Latent Dirichlet allocation (LDA), Random forest	1	74%
PubMed, 2019	Journal	Seyedmostafa Sheikhalishahi et al.[6]	SVM and Naïve Bayes	4	-
PubMed, 2021	Journal	Theresa A. Koleck et al. [7]	Nimble minor tool	5	80%to96 %
Nature Medicine ,2019	Journal	Huiying Liang et al. [8]	Logistic regression classifiers	5	88.5%
2023	Kaggle	DEVESH UPRETI[9]	Word Embedding, SVM	24	93%

CHAPTER 3: METHODOLOGY

This thesis presents a method for automatic classification of disease from clinical text data. The proposed methodology consists of two main phases i.e. feature extraction and classification. However, before the feature extraction phase, some pre-processing is applied to the raw data. Then the first phase extracts a number of features for classification purpose using TF-IDF [10] and Word2vec [11] techniques.

The classification phase builds on the extracted features and uses State of the art Machine Learning Algorithms i.e. Support Vector Machine (SVM) [12] , Logistic Regression [13], Passive Aggressive Classifier [14] and Naïve Bayes [15] (NB) at the end to accurately identify diseases. Figure 3-1 shows the constituent steps of the proposed methodology

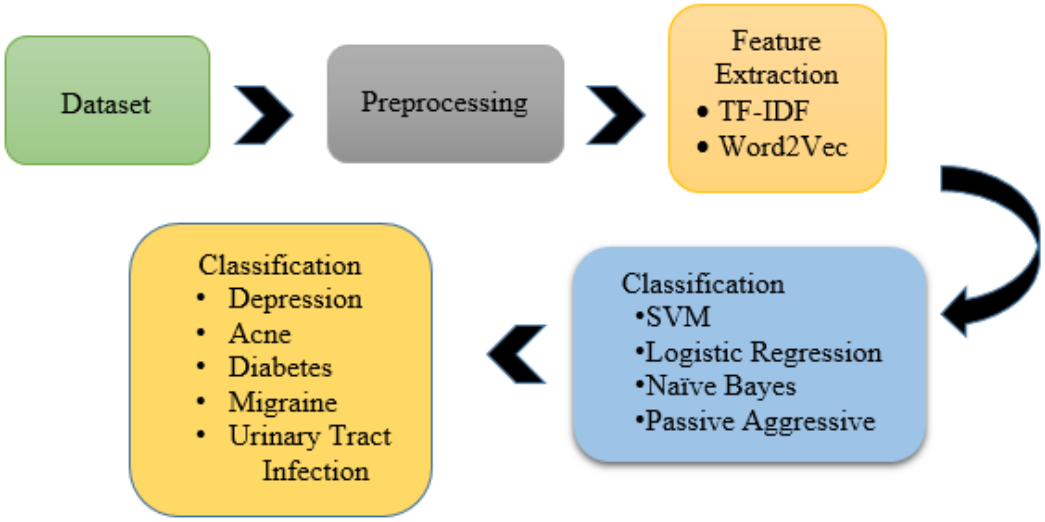

Figure 3-2: system flow diagram of proposed methodology

3.1 Dataset Information

3.1.1 UCI Machine Learning Dataset

Dataset has been taken from UCI machine learning repository [16]. Figure 3-2 shows the class summary and respective number of class samples of this dataset.

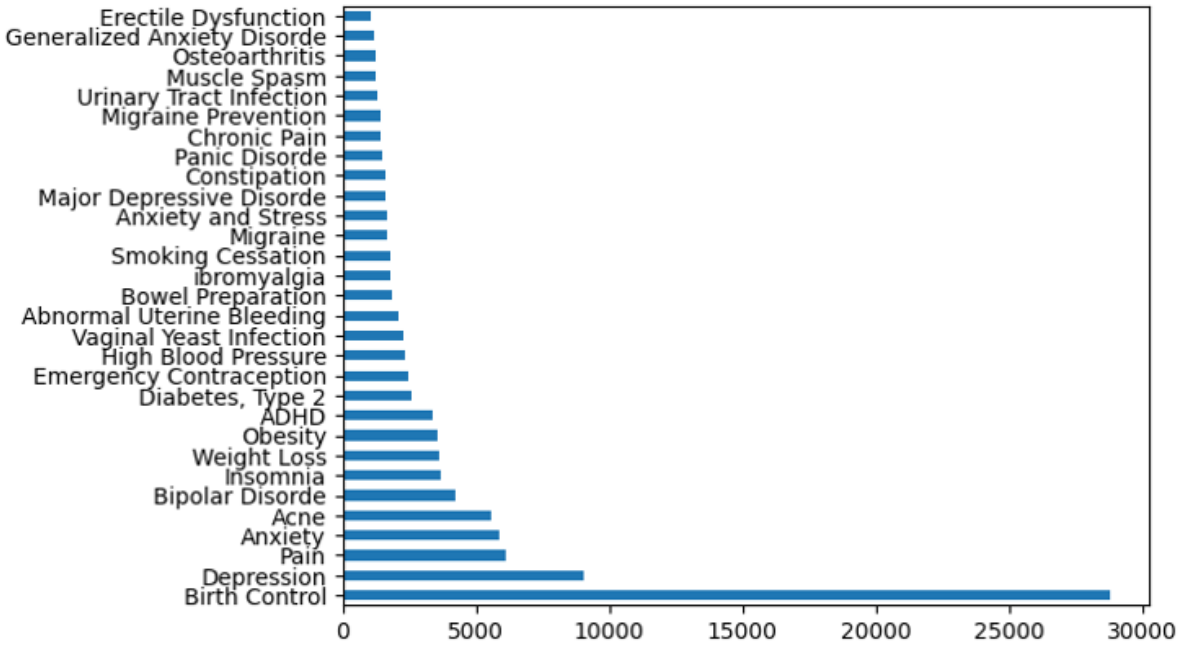

Figure 3-2: UCI Dataset

3.1.2 Symptom2Disease Kaggle Dataset

Another dataset used in this research is Symptom2Disease[17] that is downloaded from Kaggle Repository[18]. Figure 3-3 shows the class summary and respective number of class samples of this dataset

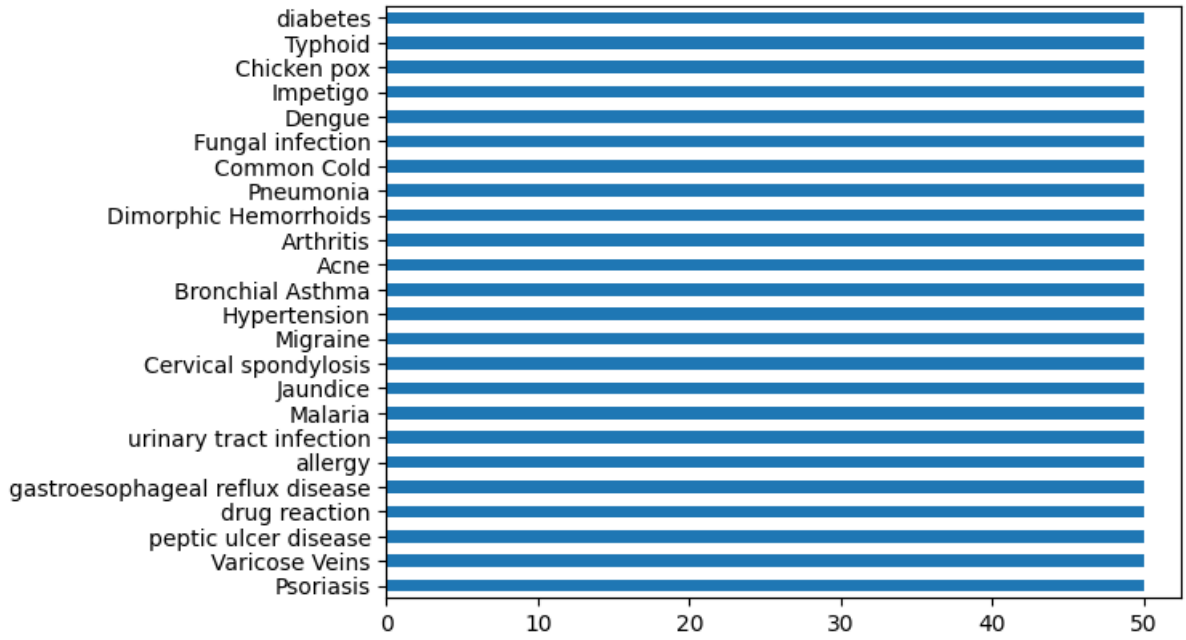

Figure 3-3: Kaggle Dataset

3.1.3 Target Disease

We have set only overlapping disease classes from these two datasets. These are Depression, Acne, Diabetes, Migraine and Urinary Tract Infection. Table 3-1 shows target disease classes and respective number of class samples that has been used for further processing.

Table 3-1: Target Disease Class and respective number of samples

Target Disease Class	No of Samples
Depression	19197
Acne	5588
Diabetes	2554
Migraine	1694
Urinary Tract Infection	1316

3.2 Data Preprocessing

Data preprocessing refers to steps and techniques applied to raw data before it can be used for specific task. Mainly it involves data cleaning, data transformation to ensure its quality, consistency, and suitability for the intended task.

Our Data preprocessing phase is comprised of 3 techniques. Figure 3-4 shows the flow diagram of Data Preprocessing activity.

Figure 3-4: Data Preprocessing Flow

3.2.1 Tokenization

First step in our preprocessing is Tokenization. It is a fundamental task in Natural Language Processing (NLP) that involves breaking down text into smaller units called tokens. These tokens can be words, characters or sub words [19]. Tokenization is the first step in most NLP tasks, as it helps to convert unstructured text data into a structured format that can be processed further. Figure 3-5 shows tokenization example applied on the data.

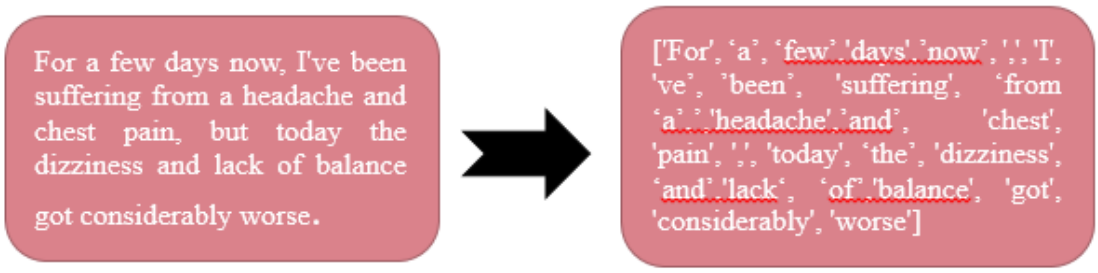

Figure 3-5: Tokenization example

3.2.2 Stop Word & Special Character Removal

After tokenization, second preprocessing strategy applied on tokenization output is Stop word & Special character removal. Stop words are commonly used words that are often considered insignificant in the context of natural language processing and text mining tasks [20]. These words are typically filtered out or excluded from text processing operations because they do not carry significant meaning and occur frequently in a language. Examples of stop words in English include "the," "and," "of," "in," "is," "to," "a," and so on. We have compared each token with pre-defined stop word list given by NLTK which contains 545 stop words.

Special character removal refers to the process of eliminating non-alphanumeric or non-standard characters from a text document. Special characters include punctuation marks, symbols, emojis, diacritical marks, mathematical symbols, and other non-alphabetic or non-numeric characters. Main aim of this pre-processing technique is noise reduction from text data and helps us to focus on main content. Figure 3-6 shows stop word & Special Character removal example applied on the tokens.

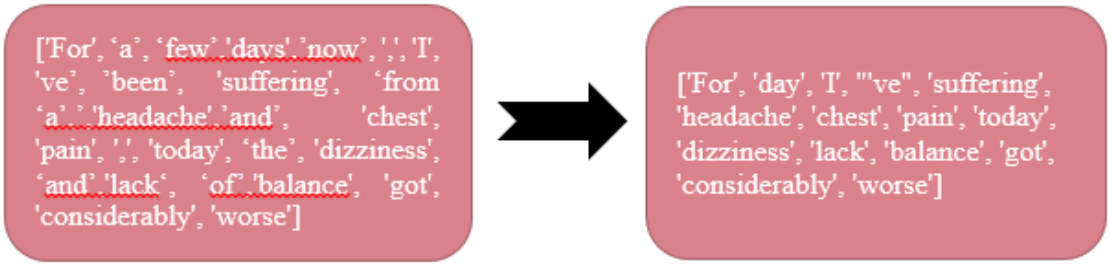

Figure 3-6: Stop Word & Special Character Removal Example

3.2.3 Lemmatization

Lemmatization is a technique used in natural language processing (NLP) preprocessing to reduce words to their base or dictionary form, known as the lemma. The goal of lemmatization is to convert inflected or derived words into a common base form, which helps in reducing word variations and simplifying the analysis of text data [21].

Unlike stemming, which simply removes word affixes to derive the root form, lemmatization takes into consideration the context and meaning of the word. It applies linguistic rules and morphological analysis to produce the lemma. Lemmatization is considered more powerful operations while creating keywords or dictionary because it focuses on main context and morphological analysis of words instead of stemming which only removes or cut the beginning and ending of word. Table 3-2 shows the example of stemming and lemmatization process

Table 3-2: Stemming and Lemmatization Example

Word/Tokens	Stemming	Lemmatization
studies	Studi	study
studying	Studi	studying
cries	Cri	cry
cry	Cri	cry

3.3 Feature Extraction

Feature extraction process is basically to get meaningful insights or discriminative information from pre-processed data. These features or information must be able to differentiate each individual class separately. Our Feature Extraction phase consists of two techniques TF-IDF [10] and Word2Vec [11] then these extracted features will be passed to Machine Learning Algorithm for further classification.

3.3.1 TF-IDF

TF-IDF stands for Term Frequency-Inverse Document Frequency. It is one of the popular feature extraction technique commonly used in natural language processing (NLP) to represent text data numerically. It aims to measure the importance of a term within a document in a collection of documents.

TF-IDF consists of two main components:

Term Frequency (TF)

This component measures the frequency of a term within a document. It is calculated as the ratio of the number of times a term appears in a document to the total number of terms in that document. The intuition behind TF is that the more frequent a term is in a document, the more important it is in representing that document.

Inverse Document Frequency (IDF)

This component measures the rarity or uniqueness of a term across the entire collection of documents. It is calculated as the logarithm of the ratio of the total number of documents to the number of documents containing the term. The intuition behind IDF is that terms that appear in fewer documents are more informative and carry more weight in distinguishing between documents.

The TF-IDF score for a term in a document is calculated by multiplying the TF value and IDF value for that term. The higher the TF-IDF score, the more important the term is in the specific document. The TF-IDF technique assigns higher weights to terms that are frequent within a document but rare across the entire collection, which helps in identifying important and distinctive terms for each document. Once the TF-IDF scores are calculated for all terms in the

documents, they can be used as features to represent the documents numerically. This allows machine learning algorithms to work with text data, as they typically require numerical inputs. Figure 3-7 shows the mathematical representation of TF-IDF algorithm

$$TF(t,d) = \frac{number\ of\ times\ t\ appears\ in\ d}{total\ number\ of\ terms\ in\ d}$$

$$IDF(t) = log\frac{N}{1+df}$$

$$TF - IDF(t,d) = TF(t,d) * IDF(t)$$

Figure 3-7: TF-IDF Mathematical Representation

.

Algorithm Example

Let's consider we have two documents

Document 1	Text processing is necessary
Document 2	Text processing is necessary and important

Assumption: we have assumed all preprocessing is applied on this set of example.

Step 1:

To calculate Term frequency (tf) of each document.

Table 3-3: Term Frequency Calculation

Word	**Doc1**	**Doc2**
	Term Frequency (tf)	
Text	¼	1/6
Processing	¼	1/6
Is	¼	1/6
Necessary	¼	1/6
And	0	1/6
Important	0	1/6

Step 2:

To calculate inverse document frequency (IDF) of each term which refers to log of the total number of document divided by no of documents that contain the text.

Table 3-4: Inverse Document Frequency Calculation

Word	**IDF**
Text	Log(2/2)=0
Processing	Log(2/2)=0
Is	Log(2/2)=0
Necessary	Log(2/2)=0
And	Log(2/1)=0.3
Important	Log(2/1)=0.3

Step 3:

To calculate term frequency and inverse document frequency (TF-IDF) of each term which refers to multiplication of term frequency (TF) and inverse document frequency (IDF)

Table 3-5: TF-IDF Calculation

Word	**TF-IDF**
Text	0
Processing	0
Is	0
Necessary	0
And	0.05
Important	0.05

The reference table 3-5 shows TFIDF of some words are zero and some words are non-zero depending on their frequency in the document and across all documents.

3.3.2 Word2Vec

Word2Vec is a popular feature extraction technique published in 2013 and widely used in natural language processing (NLP) tasks, particularly for word representation [19]. It is a shallow neural network-based model that learns continuous word embeddings from large amounts of text data. Word2Vec represents words as dense vectors in a high-dimensional space, where words with similar meanings are located closer to each other. There are two main architectures used in Word2Vec: Continuous Bag-of-Words (CBOW) and Skip-gram. CBOW tries to predict a word on the basis of its neighbors, while Skip Gram tries to predict the neighbors of a word. In simpler words, CBOW tends to find the probability of a word occurring in a context. So, it generalizes over all the different contexts in which a word can be used

CBOW

The model predicts a target word from the given context or words around the target word by predicting its probability in the context in which it is being used in. In simple words, it can be said that CBOW approach is equivalent to fill in the blank method i.e., take out a word from the sentence and then its asked to guess the word (target). Figure 3-8 displays the CBOW model architecture

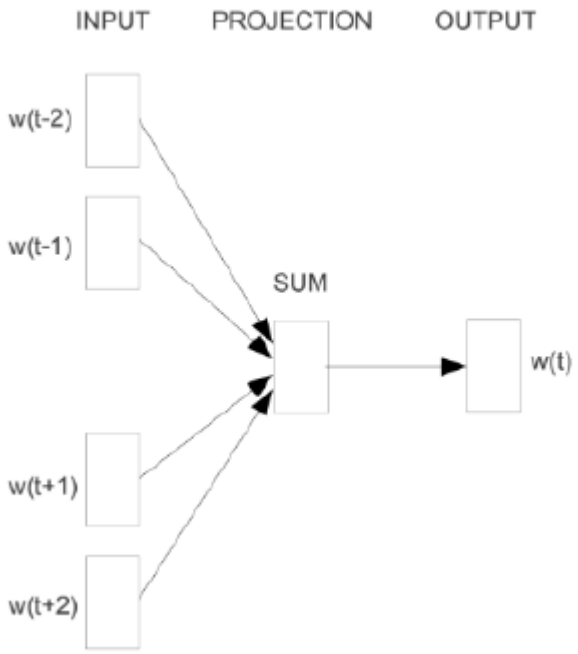

Figure 3-8: CBOW Model Architecture

Skip Gram

The skip-gram method works in the reverse of the CBOW method. The aim of the model is to predict the context window of words using the given word. In simple words, its equivalent to a word is given and then its asked to guess the words before and after to the given term**.** Figure 3-9 displays the Skip-gram model architecture.

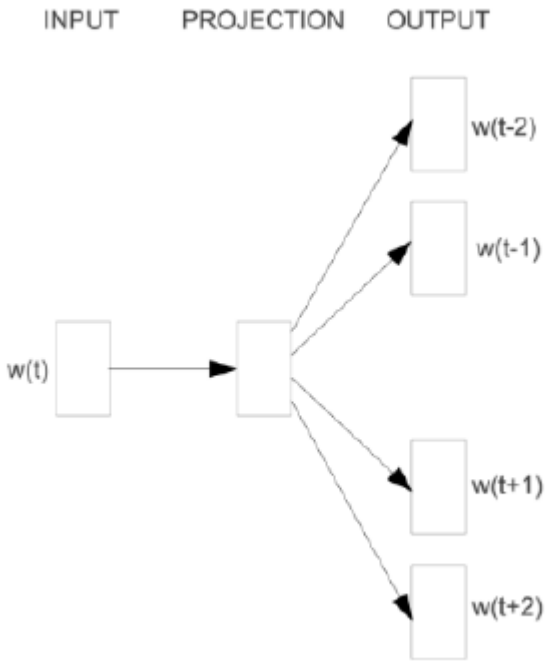

Figure 3-9: Skip-Gram Model Architecture

Both the CBOW and Skip-gram models in Word2Vec utilize artificial neural networks. Initially, each word in the vocabulary is assigned a random n-dimensional vector. During the training process, the neural network algorithm learns the optimal vector representation for each word using either the CBOW or Skip-gram approach.

In our approach, we have chosen Skip-gram model due to its specific advantages over the CBOW method.

- Skip-gram model creates two vector representations of each word where required by capturing two meanings/semantics for each word. For example, there will be two vectors created for the word “Apple”, one for fruit representation and one for technology representation.
- Skip-gram with negative sub-sampling surpasses the CBOW method in performance.

3.3.3 N-gram Model

"n-gram" refers to a contiguous sequence of n items. The items can be words, characters, or even phonemes, depending on the application. The value of n determines the length of the sequence [20]. For example, in a word-based n-gram model with n=3, the n-grams would consist of three consecutive words. Figure 3-8 shows example of N-gram model. So far, we have made three different n-gram patterns for feature extraction. These patterns are listed below:

Unigram pattern: consist of one word for extracting a semantic pattern.

Bigram pattern: consist of two words.

Trigram pattern: consist of three words.

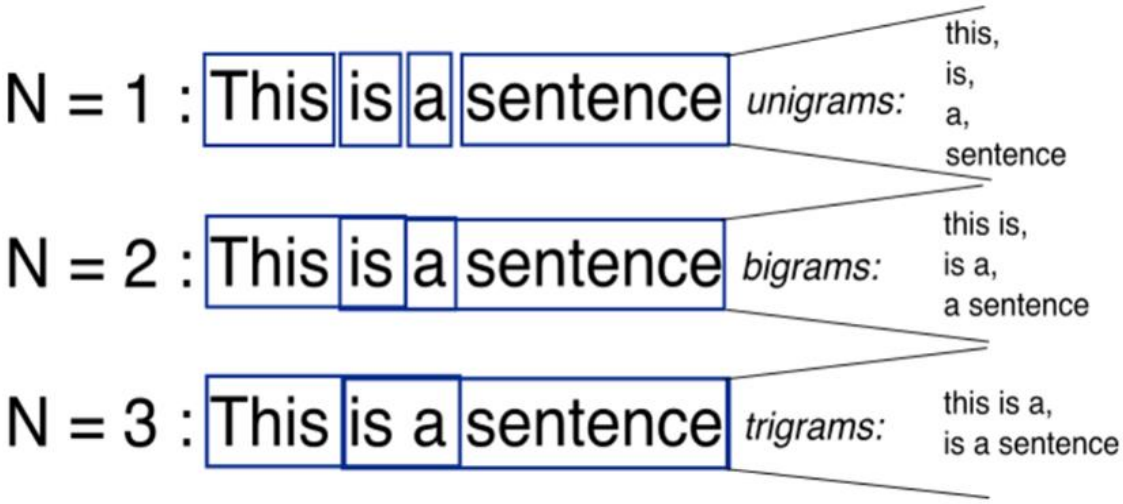

Figure 3-10: NGRAM Model Patterns

3.4 Classification

Classification in machine learning refers to the task of categorizing or labeling data into predefined classes or categories based on their features or attributes. It is a supervised learning approach where a machine learning model learns from a labeled dataset to make predictions or decisions about the class membership of new, unseen instances. It is shown in Fig 3-9(a) and Fig 3-9(b) for binary and multi class.

The process of classification involves two main components: training and prediction. During the training phase, the model is presented with a dataset that consists of input samples (often referred to as features) and their corresponding class labels. The model analyzes the features and learns the underlying patterns or relationships between the input and output. This training

phase typically involves selecting an appropriate algorithm, such as decision trees, support vector machines, or neural networks, and optimizing its parameters to achieve accurate predictions.

Once the model is trained, it can be used for prediction or inference on unseen data. Given a new instance with unknown class labels, the model applies the learned patterns to classify it into one of the predefined classes. The output of the classification process is the predicted class label or a probability distribution over multiple classes, depending on the specific algorithm used.

Classification finds applications in various domains, such as email spam detection, sentiment analysis, fraud detection, image recognition, medical diagnosis, and many others. It is a fundamental problem in machine learning and forms the basis for many other advanced techniques and algorithms.

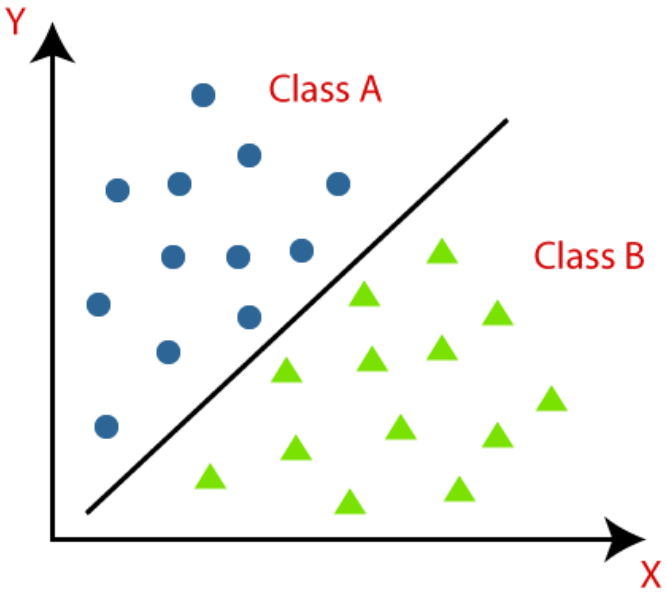

Figure 3-11(a): Binary Classification

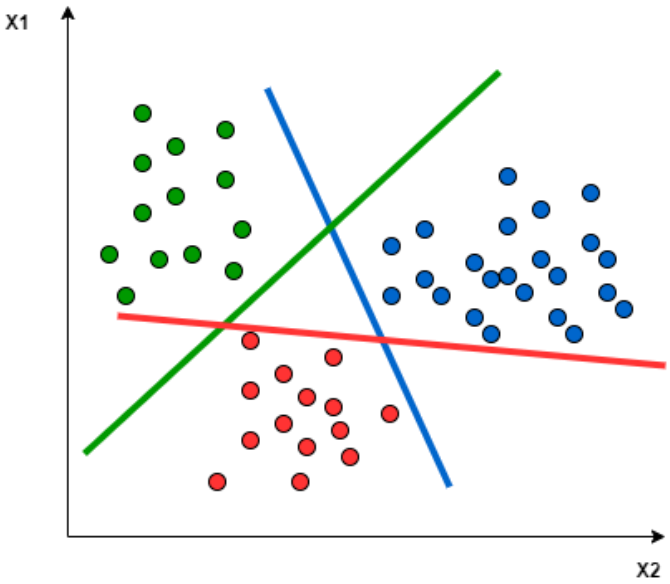

Figure 3-11(b): Multi Classification

For the classification of textual features different Machine learning models are used i.e. Support Vector Machine(SVM), Logistic Regression(LR), Passive Aggressive Classifier and Naïve Bayes.

3.4.1 Support Vector Machine Classifier (SVM)

A support vector machine (SVM) is a supervised **machine learning** algorithm which is used for both regression and classification problems. SVM is a fast and reliable classification technique that performs well with a limited amount of data to analyze. For the classification of data SVM finds the hyperplane that does not only separates the two classes but also maximizes the margin (i-e the distance between the margin and the closest data point of each class). Figure 3-10 shows how the algorithm identifies the best hyper-plane In Figure 3-10-(a), hyper-plane B is selected as the right hyper-plane as it divides the two classes better. In Figure 3-10-(b), all the hyper-planes (A, B and C) are separating the classes accurately. In this case the hyper-plane having the maximum margin from the closest data point will be selected, therefore, hyper-plane C will be preferred. In figure 3-10-(c), apparently, B may be a better classifier but SVM selects the hyper-plane that classifies the classes precisely earlier to maximizing border. Later, the right hyper-plane is A, as it has no classification error. SVM can also ignore the outliers (noisy data points) and maximize the margin as shown in Figure 3-10-(d). Until now, we have only visualized the linear kernel but SVM can also solve a linearly non-separable problem

using complex kernels e.g. radial basis function (RBF) kernel, polynomial of higher degree, Gaussian, Sigmoid, hyperbolic tangent, Laplace RBF etc. Figure 3-10-(e) shows a circular hyperplane for the data that is not linearly separable.

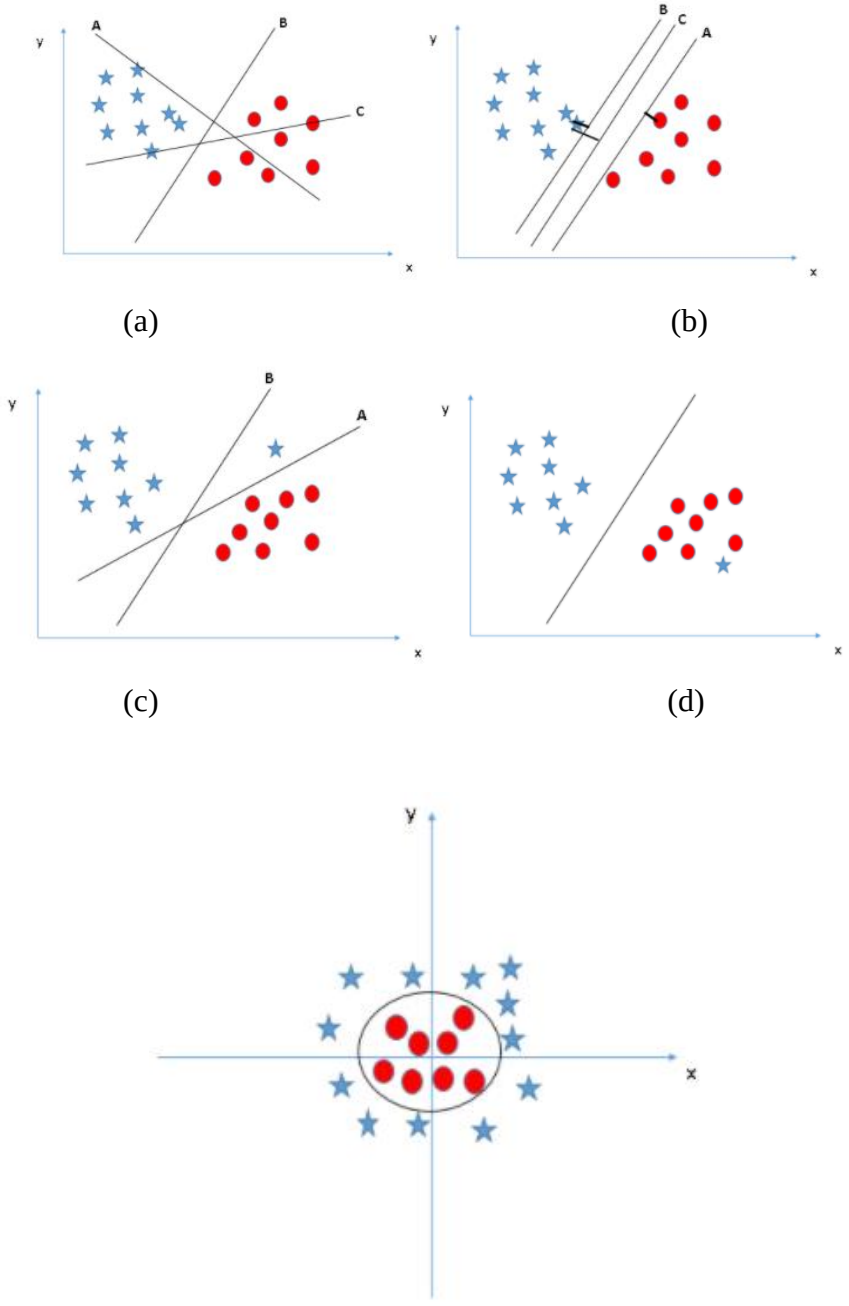

Figure 3-12: SVM with Different Hyper Plane

In order to classify disease based on textual data the SVM performs better classification as compared to other classifiers. The hyper parameters of SVM includes the 'Linear' kernel function.

3.4.2 Logistic Regression

Logistic regression is a popular method for binary classification in natural language processing (NLP) tasks. It can be used to classify text data into two categories, such as positive/negative sentiment, spam/ham email, or toxic/non-toxic comments. In this context, logistic regression models the probability of the input belonging to a certain class.

3.4.2.1 Multinomial Logistic Regression

For 5 class disease classification we have used extension of logistic regression model i.e. multinomial logistic regression also known as soft max regression. Multinomial logistic regression extends binary logistic regression to handle multiple classes. It estimates the probabilities of each class using the soft max function and assigns the class with the highest probability as the predicted class. In this approach, a separate set of coefficients is estimated for each class, and the model maximizes the likelihood of the observed data.

3.4.2.2 Softmax Activation Function

The softmax function is used as the activation function in the output layer of neural network models that predict a multinomial probability distribution [28].

The output of a Softmax is a vector (say v) with probabilities of each possible outcome. The probabilities in vector v sums to one for all possible outcomes or classes [29].

Mathematically, Softmax is defined as

$$S(y)_i = \frac{\exp(y_i)}{\sum_{j=1}^{n} \exp(y_j)}$$

where,

y	is an input vector to a softmax function, S. It consist of n elements for n classes (possible outcomes)
y_i	the i-th element of the input vector. It can take any value between -inf and +inf
$exp(y_i)$	standard exponential function applied on y_i. The result is a small value (close to 0 but never 0) if $y_i < 0$ and a large value if y_i is large. eg • $\exp(55) = 7.69e{+}23$ (A very large value) • $\exp(-55) = 1.30e$-24 (A very small value close to 0) **Note**: $\exp(*)$ is just $\mathbf{e}^*$ where $\mathbf{e} = 2.718$, the Euler's number.
$\sum_{j=1}^{n} \exp(y_j)$	A normalization term. It ensures that the values of output vector $S(y)_i$ sums to 1 for i-th class and each of them and each of them is in the range 0 and 1 which makes up a valid probability distribution.
n	Number of classes (possible outcomes)

3.4.2.3 Loss Function

The loss function typically used by logistic regression is called the "logistic loss" or "binary cross-entropy loss." It is specifically designed for binary classification problems where the target variable has two possible outcomes.

The logistic loss function is defined as:

$$L(y, \hat{y}) = -(y * \log(\hat{y}) + (1 - y) * \log(1 - \hat{y}))$$

Where:

- $L(y, \hat{y})$ represents the loss between the true target value y and the predicted value $\hat{y}$.

- y is the true target value (0 or 1) of the binary classification problem.
- ŷ is the predicted probability of the positive class (usually represented by 1) given the input features.

The goal of logistic regression is to minimize this loss function by adjusting the model's parameters (coefficients) to improve the accuracy of the predicted probabilities ŷ. This optimization process is typically performed using techniques such as gradient descent or other iterative optimization algorithms.

It's worth noting that for multi-class classification problems, logistic regression can be extended using techniques like one-vs-rest or softmax regression. In such cases, different variations of the loss function may be used, such as the multi-class cross-entropy loss or depending on the specific implementation or algorithm being used.

3.4.2.4 Multi Class Cross Entropy

Multi-class cross entropy is a loss function commonly used in machine learning, particularly in multi-class classification problems. It measures the dissimilarity between predicted class probabilities and the true class labels [30].

To understand multi-class cross entropy, let's break down the components:

1. **Multi-class classification**: It refers to a classification task where the goal is to assign an input instance to one of several possible classes. For example, classifying an image into categories such as "cat," "dog," or "horse."
2. **Cross entropy**: Cross entropy is a concept from information theory that quantifies the difference between probability distributions. In the context of classification, it measures the dissimilarity between the predicted class probabilities and the true class probabilities.

The formula for multi-class cross entropy is as follows:

$$\text{Cross entropy} = -\Sigma(y\_true * \log(y\_pred))$$

- The summation (Σ) is taken over all the classes.

- y_true is a one-hot encoded vector representing the true class label. It contains 1 in the position corresponding to the true class and 0s elsewhere.
- y_pred is a vector representing the predicted class probabilities for each class. The values in y_pred are typically obtained from a softmax activation function, ensuring they sum up to 1.

The formula essentially calculates the logarithm of the predicted probabilities of the true class, and the negative sign is applied to flip the direction of the loss value, making it a minimization problem.

By minimizing the multi-class cross entropy loss, the model is encouraged to assign high probabilities to the correct classes and low probabilities to the incorrect classes, thus improving its ability to classify instances accurately.

3.4.3 Passive Aggressive Classifier

Passive Aggressive classifier is another machine learning model that falls under the category of online machine learning model. Passive Aggressive Classifier is an online learning algorithm where a model is trained in a increment manner by feeding it instances sequentially, individually or in small groups called mini-batches. Main principle of this classifier is as follow:

Passive: If the prediction is correct, keep the model and do not make any changes. i.e., the data in the example is not enough to cause any changes in the model.

Aggressive: If the prediction is incorrect, make changes to the model. i.e., some change to the model may correct it

3.4.4 Naïve Bayes Classifier

Naive Bayes classifier is a probabilistic machine learning algorithm commonly used for text classification tasks. It's based on the Bayes' theorem and assumes that the features (words or terms) in the input data are conditionally independent of each other. In the context of disease classification based on text data, the Naive Bayes classifier can be used to predict the most likely disease given a set of symptoms or textual information about a patient.

3.4.4.1 Multinomial Naïve Bayes

For multi class classification variant of Naïve Bayes, Multinomial Naïve Bayes is commonly used for discrete features such as word frequencies in text classification tasks. It assumes that the features follow a multinomial distribution. It can be suitable for problems where the input features are counts or frequencies.

3.4.5 Stratified Distribution

Stratified sampling is a technique used in statistics and machine learning to ensure that the distribution of classes or categories in a dataset is preserved when splitting it into train and test sets [27]. When performing stratified sampling in the context of train-test splitting, it means that the resulting train and test sets will have similar proportions of different classes or categories as the original dataset.

In classification tasks, it is important to have representative samples of each class in both the training and testing sets to ensure that the model learns and evaluates its performance on all classes equally. Without stratification, there is a risk of imbalanced class distribution in either the training or testing set, leading to biased model performance.

3.4.6 K fold Distribution

In machine learning, k-fold cross-validation is a technique used to assess the performance of a model and estimate its generalization ability [31]. It involves dividing the dataset into k subsets or "folds" of approximately equal size. The model is then trained and evaluated k times, each time using a different fold as the validation set and the remaining folds as the training set.
The k-fold cross-validation process can be summarized as follows:

1. The dataset is divided into k subsets or folds.
2. The model is trained k times, each time using k-1 folds as the training set and one fold as the validation set.
3. The performance of the model is evaluated using a chosen evaluation metric (e.g., accuracy, precision, recall) on the validation set.
4. Steps 2 and 3 are repeated k times, with each fold being used as the validation set exactly once.

5. The performance results from each iteration are averaged to obtain a single performance estimate for the model.

The k-fold cross-validation technique helps to overcome issues such as overfitting or under fitting by providing a more robust estimate of a model's performance. It allows for better assessment of how well the model generalizes to unseen data by using multiple validation sets from different portions of the dataset.

The k-fold distribution refers to the distribution of the dataset across the k folds. It ensures that each fold represents a similar distribution of the data, avoiding biases caused by an uneven representation of certain classes or patterns. It is essential to have a representative distribution across the folds to obtain reliable performance estimates for the model.

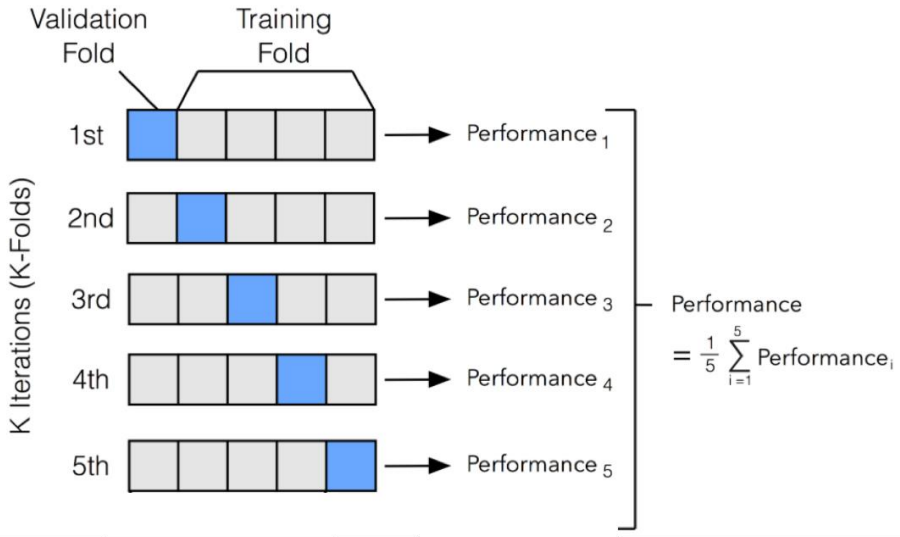

Figure 3-13: K fold cross validation

CHAPTER 4: EXPERIMENTAL RESULTS

In this chapter, the performance of the proposed disease classification model using textual keywords is evaluated using Python, Colab online IDE also section 4.2 highlights our experimental setup. We have performed experimentation based on two different feature extraction techniques, different N-gram Models and different pre-processing techniques. It is observed from the experimentation that SVM performs better for classifying disease from extracted keywords.

4.1 Performance Measures

In order to validate the performance of our proposed methodology we have computed various parameters from confusion matrix including accuracy, precision, recall, f1-score etc. The details of these parameters is discussed in the next subsections as follows:

4.1.1 Confusion Matrix

Confusion Matrix is an easiest way for measuring the performance of a classifier whereas the output can be of two or more classes. It is basically a table that gives the information of "actual class" vs. "predicted/ class"[23]. It contains the following parameters called as "True positive (TP)", "True Negative (TN)", "False positive (FP)", " False Negative (FN)" as shown in Figure below

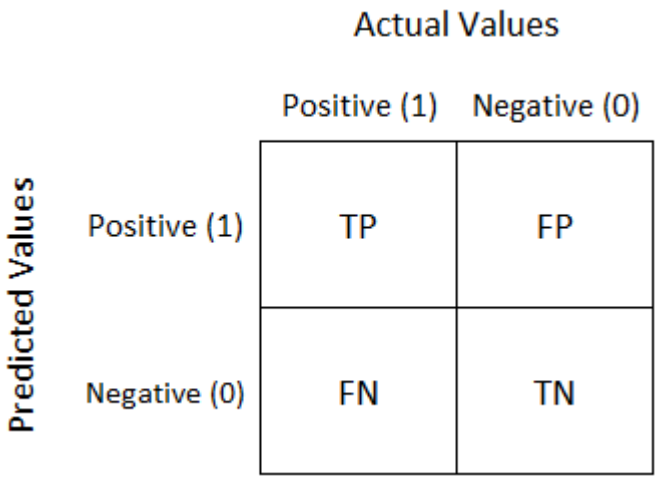

Figure 4-1: Confusion Matrix

TP specifies the number of samples the classifier predicted positive were actually positive

TN specifies the number of samples the classifier predicted negative were actually negative

FP specifies the number of samples the classifier predicted positive were actually negative

FN specifies the number of samples the classifier predicted negative were actually positive

4.1.2 Accuracy

It is ratio of correct predictions and total number of predictions. It tells overall accuracy of Machine Learning Model [25].

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

4.1.3 Recall

It is known as the true positive recognition rate: (i-e, the number of samples the classification model predicted were positive that were actually positive) [26]. Mathematically, it can be calculated as follows

$$Recall = \frac{TP}{TP + FN}$$

It is also termed as sensitivity.

4.1.4 Precision

Precision is defined as the when a positive value is predicted, how often is the prediction correct [26]. Mathematically, it can be calculated as

$$Precision = \frac{TP}{TP + FP}$$

4.1.5 F1 score

F measure is the harmonic mean of precision or recall [26].

$$F1\ score = \frac{2 * Precision * Recall}{Precision + Recall}$$

4.2 Experimental Setup

In this section, we outlined the experimental setup used for conducting the research and implementing the proposed methodology. The following subsections describe the software tools and libraries that have been utilized

4.2.1 Software Tools and Libraries

The implementation of the research was carried out using the following software tools and libraries:

- Colab: Google Colab is an online development environment [32] that provides a Jupyter Notebook interface, allowing for seamless execution of Python code and access to various resources.

- Python: The experiments were implemented using Python, a popular programming language known for its simplicity and extensive range of libraries [33].

- NLTK (Natural Language Toolkit): NLTK is a powerful library for natural language processing (NLP) tasks in Python [34]. It provides a wide range of functionalities, including tokenization, stemming, part-of-speech tagging, and more.

- sklearn (Scikit-learn): Scikit-learn is a machine learning library that offers a broad range of algorithms and tools for data analysis and modeling [35]. It provides efficient implementations of popular machine learning algorithms, such as classification, regression, clustering, and dimensionality reduction.

4.3 Result and Discussion

We have evaluated proposed methodology on two dataset i.e. UCI dataset and Symptom2Disease dataset and 97.7% and 98% accuracies are achieved using SVM Classifiers. SVM outperforms in text classification for five disease problem as compared to other three ML algorithms Multinomial LR, Multinomial NB and Passive Aggressive Classifier.

We have also evaluated different N gram Model. Performance of 3-gram Model is remarkable as compared to 1 and 2-gram. After careful analysis, it is shown that 3-gram model surpasses 1 and 2-gram models in generating more accurate and relevant keyword for disease text classification.

While evaluating feature extraction technique TF-IDF outperforms word2vec for our domain problem. We have also experimented different pre-processing techniques on raw data, upon implementing stemming, lemmatization and a combination of both techniques it becomes evident that employing only lemmatization leads to superior outcomes in text processing and analysis. Details of these experiments are explained in next subsection.

4.3.1 Dataset Distribution

We have used 70 to 30% ratio for training and validation accuracy and used 250 unseen data samples for testing We have trained different classifiers on training data set and acquired validation accuracy and testing accuracy.

Table 4-1: Dataset Distribution

Data Distribution	Samples
Training	21244
Validation	9105
Testing	250

4.3.2 N gram Model Result

We have experimented unigram, bigram and trigram model for further feature extraction it has been observed from acquired result that accuracy for 3-gram model is higher than 1 and 2-gram model. In Table 4-1 we have shown summary of the result. We have compared different evaluation parameters. Based upon this comparison we concluded that Trigram model is highly effective for making keywords for text classification. We have applied 4 different Machine Learning classifiers to evaluate these three model. Unigram model gives 15455 number of keywords. Bigram gives 292204 number of keywords for further classification and Trigram gives 791522 number of keywords. Although Number of keywords generated from trigram model is higher than other than two models but this model also gives higher accuracy.

After thorough analysis, it has been observed that 3-Gram Model implemented with Support Vector Machine (SVM) classifier outperformed other combinations in terms of accuracy, precision, Recall and F1-score.

This finding indicates that the utilization of trigrams, combined with the SVM classifier, yielded superior results when compared to other n-gram models and classifiers.

The accuracy achieved by the 3-gram model with the SVM classifier suggests its potential for effectively capturing and leveraging the contextual information present within the text data. This combination displayed a higher level of accuracy in comparison to the other tested models and classifiers.

Table 4-2: Experimentation results of different N-gram Model

Feature Extraction	Features Dimension	Classifiers	Validation Accuracy	Testing Accuracy	Recall	Precision	F1-Score
Unigram	(21244, 15455) (9105, 15455) (250, 15455)	SVM	97.3%	81.6%	78%	78%	74%
		Logistic Regression	96.5%	80.4%	77%	78%	72%
		Passive Aggressive Classifier	97.7%	82.8%	83%	76%	74%
		Naïve Bayes	87.2%	70.8%	47%	70%	47%
Bigram	(21244, 292204) (9105, 292204) (250, 292204)	SVM	97.5%	97.6%	92%	97%	95%
		Logistic Regression	95.9%	97.2%	90%	99%	94%
		Passive Aggressive Classifier	97.9%	94.0%	93%	92%	92%
		Naïve Bayes	80%	82.0%	82%	80%	76%
Trigram	(21244, 791522) (9105, 791522) (250, 791522)	**SVM**	**97.7%**	**98.0%**	**94%**	**97%**	**95%**
		Logistic Regression	95.6%	96.4%	87%	99%	92%
		Passive Aggressive Classifier	97.9%	94.0%	93%	92%	92%
		Naïve Bayes	80%	82.0%	82%	80%	76%

Unigram Model Performance:

Fig 4-2 shows confusion matrices of testing phase of unigram model. We have used stratified technique for generalized model performance. Average accuracy achieved by each classifier SVM, Logistic Regression, Passive Aggressive Classifier and Naïve Bayes is 81%, 80%, 82% and 70% respectively.

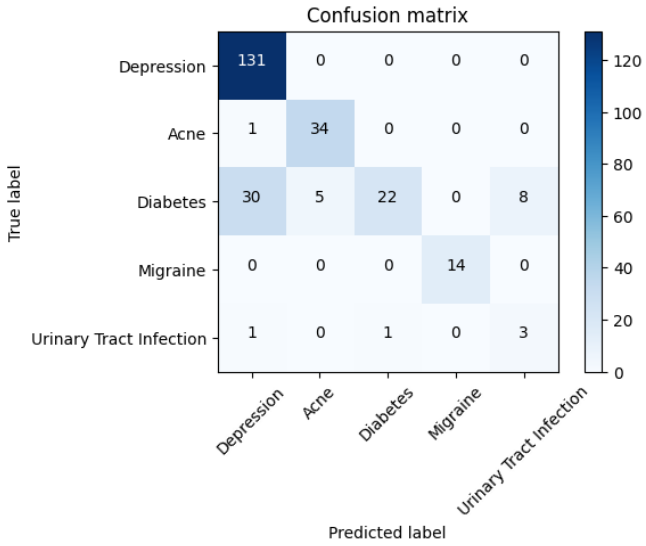

Figure 4-2(a): SVM, Confusion Matrix

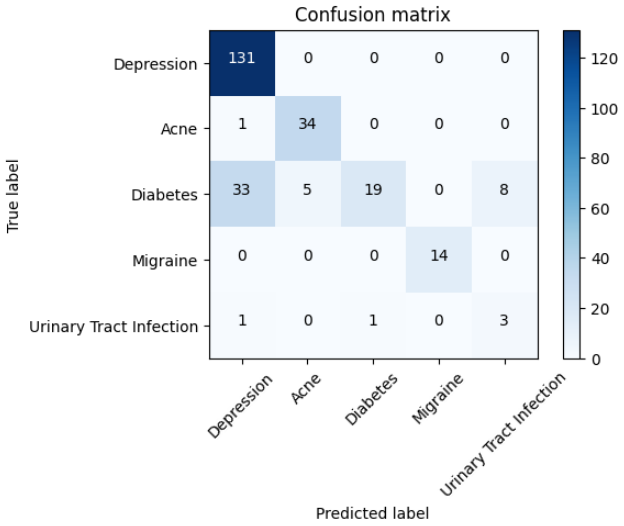

Figure 4-2(b): Logistic Regression, Confusion Matrix

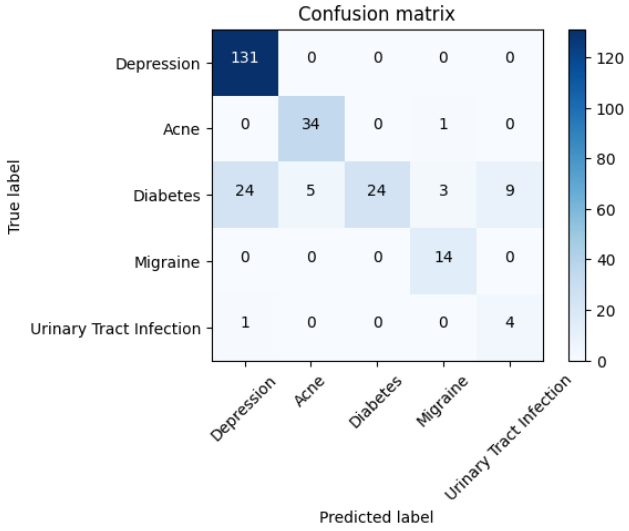

Figure 4-2(c): Passive Aggressive Classifier, Confusion Matrix

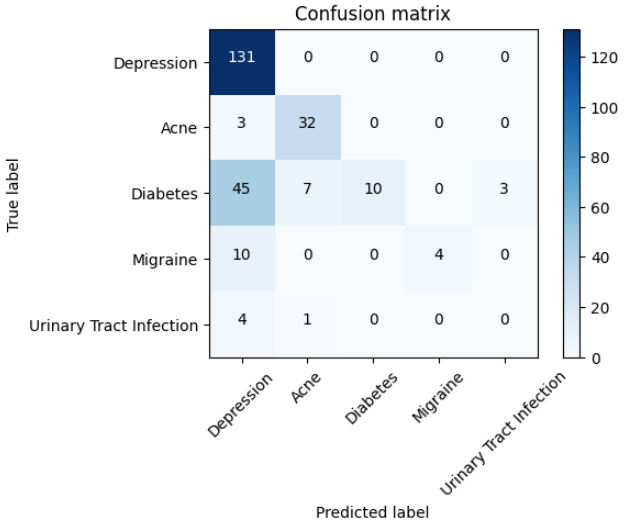

Figure 4-2(d): Naïve Bayes, Confusion Matrix

Figure 4-2: Confusion Matrix Plot for Unigram Model

Bigram Model Performance

Fig 4-3 shows confusion matrices of testing phase of bigram model. We have used stratified technique for generalized model performance. Average accuracy achieved by each classifier SVM, Logistic Regression, Passive Aggressive Classifier and Naïve Bayes is 70%, 97%, 97% and 70% respectively.

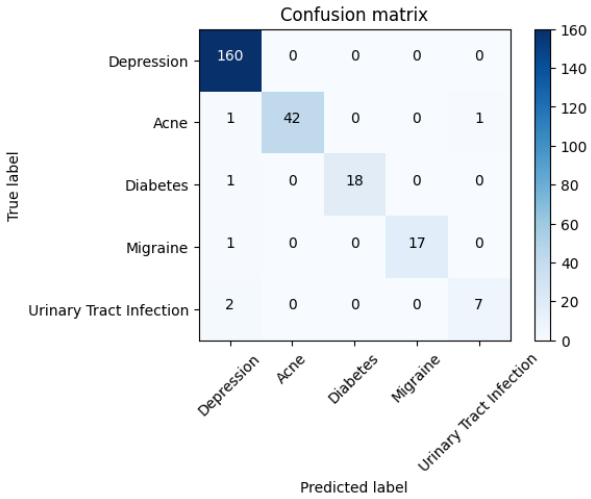

Figure 4-3(a): SVM, Confusion Matrix

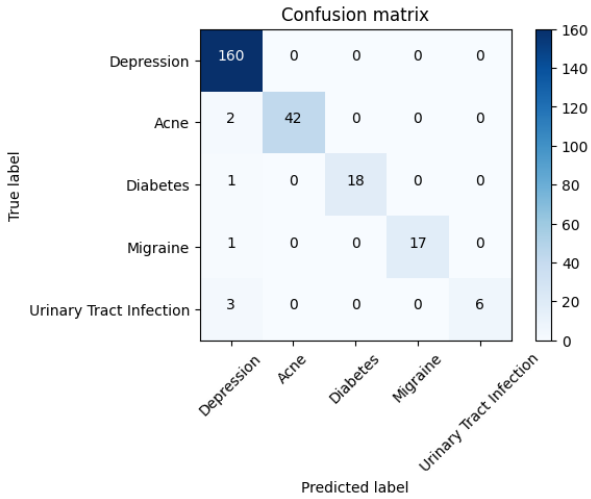

Figure 4-3(b): Logistic Regression, Confusion Matrix

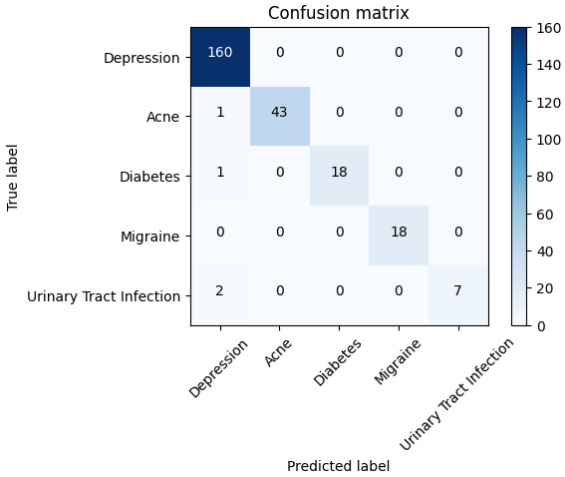

Figure 4-3(c): Passive Aggressive, Confusion Matrix

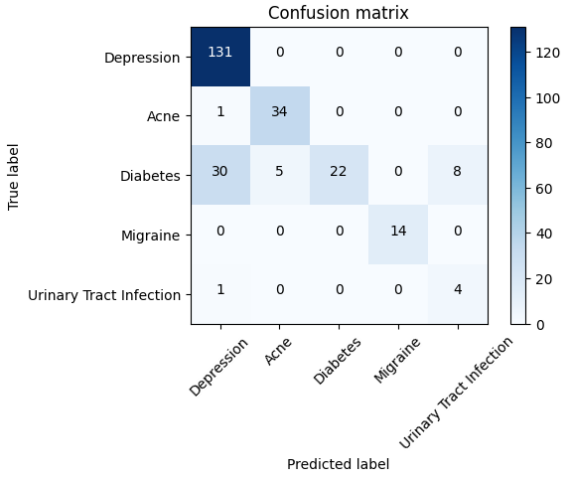

Figure 4-3(d): Naïve Bayes, Confusion Matrix

Figure 4-3: Confusion Matrix Plot for Bigram Model

Trigram Model Performance

Fig 4-4 shows confusion matrices of testing phase of trigram model. Average accuracy achieved by each classifier SVM, Logistic Regression, Passive Aggressive Classifier and Naïve Bayes is 98%, 96%, 94% and 82% respectively.

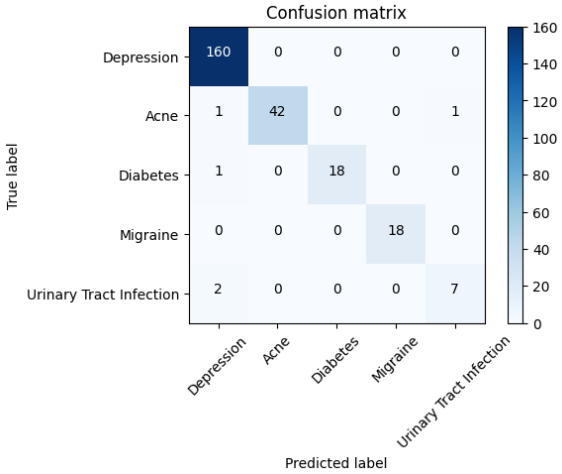

Figure 4-4(a): SVM, Confusion Matrix

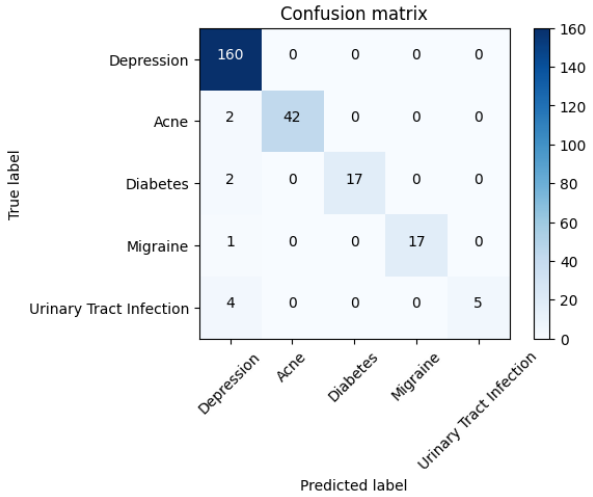

Figure 4-4(b): Logistic Regression, Confusion Matrix

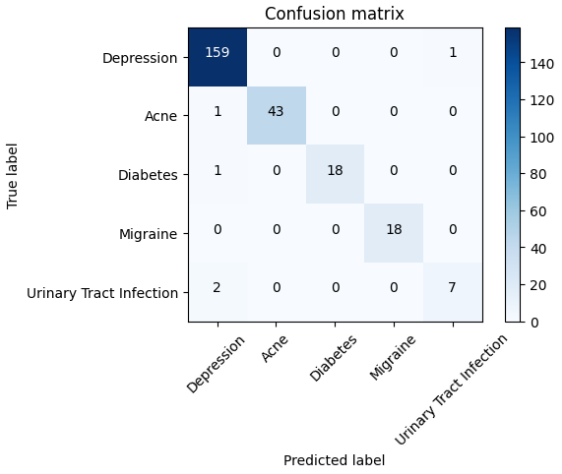

Figure 4-4(c): Passive Aggressive, Confusion Matrix

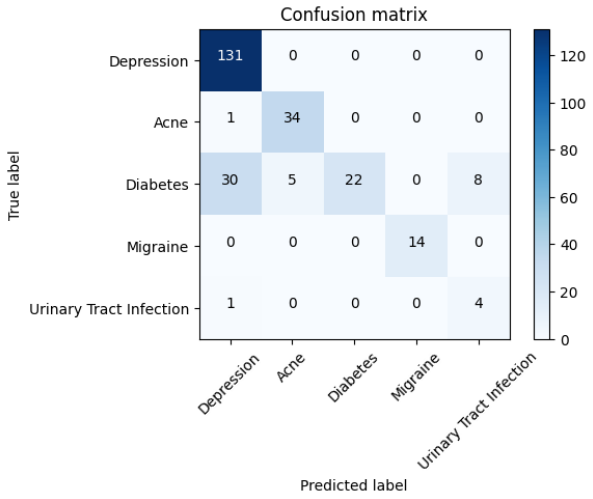

Figure 4-4(d): Naïve Bayes, Confusion Matrix

Figure 4-4: Confusion Matrix Plot for Trigram Model

4.3.3 Pre-processing result

We have applied various preprocessing techniques such as stemming, lemmatization, and a combination of both for text classification. After evaluating the results, I found that the accuracy achieved through lemmatization alone is significantly higher than the other techniques.

During the preprocessing stage, stemming involves reducing words to their base or root form. This technique can help in reducing the dimensionality of the text data by eliminating suffixes and prefixes. Lemmatization, on the other hand, aims to transform words to their base form based on their context and dictionary definitions. It provides more meaningful results compared to stemming as it takes into account the part of speech and context of the word.

In my experimentation, I observed that the accuracy of the text classification task was highest when I used only lemmatization. This indicates that preserving the original meaning and context of the words was crucial for accurately classifying the text. Stemming, while helpful in some cases, might have led to a loss of information that affected the classification accuracy. Table 4-3 gives summary of result. We can see 98.0% accuracy achieved through lemmatization and TF-IDF techniques.

Table 4-3: Experimentation results of different Pre-processing techniques

Feature Extraction	Preprocessing	Validation Accuracy	Testing Accuracy	Recall	Precision	F1-Score
TF/IDF	Stemming	97.7%	82.0%	82%	80%	76%
	Lemmatization	**97.7%**	**98.0%**	**94%**	**97%**	**95%**
	Stemming+Lemmatization	97.7%	91.2%	87%	84%	85%
word2vec	Stemming	95.1%	93.6%	91%	85%	87%
	Lemmatization	**94.8%**	**96.2%**	**85%**	**91%**	**87%**
	Stemming+Lemmatization	95.0%	93.2%	88%	90%	88%

4.3.4 Feature Extraction result

We conducted experiments using clinical text data to address a five-class disease classification problem. In this study, we applied two popular natural language processing techniques, TF-IDF and Word2Vec, and compared their performance. Table 4-3 gives summary of performance. After careful evaluation, we observed that TF-IDF yielded superior results compared to Word2Vec.

TF-IDF, short for Term Frequency-Inverse Document Frequency, is a numerical statistic that reflects the importance of a term in a given document corpus. It calculates the product of the term frequency (TF) and the inverse document frequency (IDF) to assign weights to individual terms. TF-IDF captures the relevance of a term to a particular document by considering its frequency in that document and its rarity across the entire corpus.

On the other hand, Word2Vec is an algorithm that represents words in a vector space, enabling semantic similarity calculations between words. It learns continuous word embeddings by considering the context in which words appear within a large corpus. Word2Vec captures semantic relationships between words and can be particularly useful for various natural language processing tasks.

However, in the context of the five-class disease classification problem using clinical text data, TF-IDF outperformed Word2Vec. The reasons for this superiority could be attributed to the following factors:

1. Domain-specific relevance: Clinical text data contains specialized medical terminology, which is well-captured by TF-IDF. By considering the term frequencies and inverse document frequencies, TF-IDF assigns higher weights to the terms that are most relevant to the diseases under consideration. This domain-specific relevance contributes to its superior performance.
2. Feature representation: TF-IDF represents each document in the dataset as a vector of weighted terms. This representation emphasizes the importance of individual terms in distinguishing between different classes of diseases. In contrast, Word2Vec focuses on capturing the semantic relationships between words, which may not be as effective for disease classification tasks.
3. Data sparsity: Clinical text data often suffers from data sparsity, with limited examples available for each disease class. TF-IDF is less affected by this issue because it

considers the rarity of terms across the entire corpus. By assigning higher weights to rare terms specific to certain disease classes, TF-IDF can better discriminate between the classes, even with limited examples.

Based on these observations, it can be concluded that TF-IDF is more suitable for the five-class disease classification problem using clinical text data. Its ability to capture domain-specific relevance, focus on term importance, and handle data sparsity contribute to its superior performance compared to Word2Vec in this context.

CHAPTER 5: CONCLUSION & FUTURE WORK

5.1 Conclusion

This report aimed to investigate the classification of 5 different diseases based on clinical notes using Support Vector Machines (SVM) and TF-IDF (Term Frequency-Inverse Document Frequency) technique. Throughout the study, we have gained valuable insights into the potential of machine learning algorithms and natural language processing in disease classification.

Our results indicate that SVM, combined with TF-IDF, proved to be a powerful tool in accurately classifying the diseases based on the information extracted from clinical notes. By employing feature extraction techniques like TF-IDF, we were able to transform the textual data into numerical representations, allowing SVM to effectively learn patterns and make accurate predictions.

The use of SVM in this classification task provided several benefits. SVM is known for its ability to handle high-dimensional data and its robustness to outliers. It also demonstrated good generalization performance, which is essential for reliable disease classification.

TF-IDF played a crucial role in extracting relevant features from the clinical notes. By assigning weights to each term based on its frequency and importance in the dataset, TF-IDF effectively captured the distinguishing characteristics of each disease. This enabled the SVM model to make informed decisions and achieve high classification accuracy.

The successful classification of the 5 diseases based on clinical notes using SVM and TF-IDF has significant implications for healthcare and medical research. Accurate disease classification can assist healthcare professionals in making informed decisions, providing timely treatments, and improving patient outcomes. Furthermore, this approach can contribute to the field of medical research by aiding in the identification of patterns and associations between clinical notes and disease diagnoses.

5.2 Contribution

In recent years, the analysis of clinical text data has gained significant attention due to its potential in improving healthcare outcomes. One crucial task in this field is disease

classification, where the goal is to automatically categorize clinical text into specific disease categories. This contribution focuses on presenting five disease classifications that have shown promising results in handling clinical text data. The first classification involves Depression. The second classification tackles Acne issue. The third classification centers on Diabetes problem. The fourth classification addresses Migraine problem. Lastly, the fifth classification encompasses infectious diseases like Urinary tract infection. By exploring these disease classifications for clinical text data, we contributed to the development of accurate disease classification systems that can assist healthcare professionals in diagnosing and treating patients more efficiently and effectively.

5.3 Future Work

Despite the promising results, there are still areas for further improvement. Fine-tuning the SVM hyper parameters, exploring alternative feature extraction methods, implementing deep learning method, Convolutional Neural Network for classification and incorporating more extensive datasets are potential avenues for future research. Additionally, the inclusion of other machine learning algorithms for comparison purposes would provide a more comprehensive understanding of disease classification techniques.

REFERENCES

[1] https://explodingtopics.com/blog/data-generated-per-day

[2] Zucco, Chiara. "Data Mining in Bioinformatics." Encyclopedia of Bioinformatics and Computational Biology: ABC of Bioinformatics (2018): 328.

[3] Khanday, Akib Mohi Ud Din, et al. "Machine learning based approaches for detecting COVID-19 using clinical text data." *International Journal of Information Technology* 12 (2020): 731-739.

[4] Koleck, Theresa A., et al. "Natural language processing of symptoms documented in free-text narratives of electronic health records: a systematic review." *Journal of the American Medical Informatics Association* 26.4 (2019): 364-379.

[5] Feller, Daniel J., et al. "Using clinical notes and natural language processing for automated HIV risk assessment." *Journal of acquired immune deficiency syndromes (1999)* 77.2 (2018): 160.

[6] Sheikhalishahi, Seyedmostafa, et al. "Natural language processing of clinical notes on chronic diseases: systematic review." *JMIR medical informatics* 7.2 (2019): e12239.

[7] Koleck, Theresa A., et al. "Identifying symptom information in clinical notes using natural language processing." *Nursing research* 70.3 (2021): 173.

[8] Liang, Huiying, et al. "Evaluation and accurate diagnoses of pediatric diseases using artificial intelligence." *Nature medicine* 25.3 (2019): 433-438.

[9] Kaggle competition https://www.kaggle.com/datasets/jessicali9530/kuc-hackathon-winter-2018?select=drugsComTrain_raw.csv

[10] Liu, Qing, et al. "Text features extraction based on TF-IDF associating semantic." *2018 IEEE 4th international conference on computer and communications (ICCC)*. IEEE, 2018.

[11] Alshari, Eissa M., et al. "Senti2vec: an effective feature extraction technique for sentiment analysis based on word2vec." *Malaysian Journal of Computer Science* 33.3 (2020): 240-251.

[12] Gould, Joshua. "Support Vector Machine (SVM) Documentation." (2004).

[13] Bagley, Steven C., Halbert White, and Beatrice A. Golomb. "Logistic regression in the medical literature:: Standards for use and reporting, with particular attention to one medical domain." *Journal of clinical epidemiology* 54.10 (2001): 979-985.

[14]Passive Aggressive classifier documentation: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.PassiveAggressiveClassifier.html

[15] Naïve Bayes Classifier documentation:
https://en.wikipedia.org/wiki/Naive_Bayes_classifier

[16] UCI Dataset: https://www.kaggle.com/datasets/jessicali9530/kuc-hackathon-winter-2018?select=drugsComTrain_raw.csv

[17] Kaggle Dataset:https://www.kaggle.com/datasets/niyarrbarman/symptom2disease

[18]Dataset Repositort:https://www.kaggle.com/datasets

[19]NLP tokenization documentationhttps://towardsdatascience.com/tokenization-for-natural-language-processing-a179a891bad4

[20] NLP Stopwords Removal:https://towardsdatascience.com/text-pre-processing-stop-words-removal-using-different-libraries-f20bac19929a

[21] NLP Pre-processing Documentation:https://www.guru99.com/stemming-lemmatization-python-nltk.html

[22]NLP Word2vec Documentation:
https://en.wikipedia.org/wiki/Word2vec#:~:text=Word2vec%20was%20created%2C%20patented%2C%20and,at%20Google%20over%20two%20papers.

[23]NLP N-gram Model: https://www.kdnuggets.com/2022/06/ngram-language-modeling-natural-language-processing.html

[24] Machine Learning confusion Matrix: https://www.geeksforgeeks.org/confusion-matrix-machine-learning/

[25]Machine learning accuracy parameters: https://developers.google.com/machine-learning/crash-course/classification/accuracy

[26]Machine Learning Evaluation parameters: https://machinelearningmastery.com/precision-recall-and-f-measure-for-imbalanced-classification/

[27]Machine Learning train/test split: https://scikitlearn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

[28] Softmax function: https://machinelearningmastery.com/softmax-activation-function-with-python/

[29] Softmax Documentation:https://towardsdatascience.com/softmax-activation-function-how-it-actually-works-d292d335bd78

[30] Multi class cross entropy:https://towardsdatascience.com/cross-entropy-for-classification-d98e7f974451

[31] K fold distribution: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html

[32] Online Google Colab: https://colab.research.google.com/?utm_source=scs-index

[33] Python Language: https://www.python.org/downloads/

[34]Python NLTK Library: https://www.nltk.org/

[35] Python Scikit Library: https://scikit-learn.org/stable/