Instituto Tecnológico y de Estudios Superiores de Occidente

Reconocimiento de validez oficial de estudios de nivel superior según acuerdo secretarial 15018, publicado en el Diario Oficial de la Federación del 29 de noviembre de 1976.

Departamento de Matemáticas y Física

Maestría en Ciencia de Datos

Predicción de diagnóstico a partir de datos médicos utilizando algoritmos de PLN

TRABAJO RECEPCIONAL que para obtener el **GRADO** de
Maestro en Ciencia de Datos

Presenta:
ASHWIN GURUDEVA BHAT

Director:
Mtro. JUAN FRANCISCO MUÑOZ ELGUEZABAL

Tlaquepaque, Jalisco, May 27, 2024

Predicción de diagnóstico a partir de datos médicos utilizando algoritmos de PLN

ASHWIN GURUDEVA BHAT

Abstract

In the contemporary medical landscape, there is a pressing need for rapid and accurate solutions to health emergencies, as well as access to expert physician insights. Traditional approaches involve clinical consultations where doctors assess patient histories and recommend specialist interventions. However, with the advent of Natural Language Processing (NLP) – a subset of machine learning – there is potential to revolutionize this process. NLP, when applied to medical findings, offers promising avenues for predicting patient diagnoses and identifying early indicators of chronic diseases.

Given the vast repositories of publicly accessible medical data, there is an opportunity to harness advanced models such as Spark NLP, Spacy, and Deep Learning to distill meaningful insights from these findings. Such models can not only aid in patient diagnosis but also provide a broader perspective on population health trends, paving the way for proactive disease prevention strategies.

This document delves into the utilization of diverse NLP algorithms for diagnosing conditions based on medical findings, underscoring the transformative potential of machine learning in clinical analysis.

Predicción de diagnóstico a partir de datos médicos utilizando algoritmos de PLN

ASHWIN GURUDEVA BHAT

Resumen

En el panorama médico contemporáneo, existe una necesidad apremiante de soluciones rápidas y precisas para la salud. emergencias, así como acceso a información de médicos expertos. Los enfoques tradicionales implican consultas clínicas. donde los médicos evalúan los historiales de los pacientes y recomiendan intervenciones especializadas. Sin embargo, con la llegada de El procesamiento del lenguaje natural (PLN), un subconjunto del aprendizaje automático, tiene potencial para revolucionarlo. proceso. La PLN, cuando se aplica a los hallazgos médicos, ofrece vías prometedoras para predecir los diagnósticos de los pacientes y Identificar indicadores tempranos de enfermedades crónicas.

Dados los vastos depósitos de datos médicos de acceso público, existe una oportunidad aprovechar modelos avanzados como Spark PLN, Spacy y Deep Learning para destilar ideas significativas a partir de estos hallazgos. Estos modelos no sólo pueden ayudar a los pacientes diagnóstico, sino que también proporciona una perspectiva más amplia sobre las tendencias de salud de la población, allanando el camino el camino para estrategias proactivas de prevención de enfermedades.

Este documento profundiza en la utilización de diversos algoritmos de PLN para el diagnóstico condiciones basadas en hallazgos médicos, lo que subraya el potencial transformador de Aprendizaje automático en el análisis clínico.

Contents

List of Figures

List of Tables

Dedicated to...

Through these lines, I wish to express my most sincere gratitude to all the people who have contributed significantly to my development, both professionally and personally.

First of all, I would like to express my deep gratitude to my wife, whose unwavering support has been fundamental in my educational career. Without her dedication and sacrifice, it would not have been possible to carry out my university studies. I recognize the tireless effort my wife made in taking care of kids and encouraging me to puruse my education. Despite the adversities and challenges we have faced over the years, I will be eternally grateful to you for your determination.

Likewise, I wish to express my gratitude to the Instituto Tecnológico y de Estudios Superiores de Occidente, for giving me the opportunity to pursue a postgraduate degree. I cannot overlook thanking all the teachers who shared their knowledge and wisdom, and guided my steps in the development of my research project.

Last but not least, I want to express my gratitude to my friends. Although the geographical distance that some of us face, we have always been there for each other. We have shared experiences both in the academic and personal fields, and we have providing mutual support in the best and worst moments of life.

In summary, I want to express my gratitude to all the people who have been an integral part of my growth and development, both professionally and personally. Your contributions have been invaluable, and I am deeply grateful for each one of you.

1 Introduction

With the rapid advancements in both medicine and artificial intelligence (AI), there is a compelling opportunity to harness the strengths of these fields for enhanced healthcare outcomes. By integrating AI, particularly Natural Language Processing (NLP), into medical practices, healthcare professionals can achieve more precise patient assessments and make informed decisions swiftly.

NLP algorithms can analyze medical findings to identify specific health issues, streamline diagnostic reports, and even suggest subsequent medical actions. This not only expedites the patient care process but also aids in locating the most suitable physicians and healthcare facilities in proximity.

Furthermore, the incorporation of AI can elevate the efficiency and accuracy within the medical profession, allowing practitioners to allocate more time to critical cases. Analyzing health data trends over time can also offer insights into disease patterns, facilitating proactive measures for future health challenges.

The recent COVID-19 pandemic underscored the challenges faced by doctors in differentiating between symptoms of common colds and those of the novel coronavirus. Leveraging technologies like NLP can be instrumental in distinguishing such nuances, preparing us better for unforeseen health crises in the future.

1.1 Background

The field of medical data analysis has seen a significant shift towards the adoption of artificial intelligence (AI) technologies, notably Natural Language Processing (NLP).This research delves into the potential of NLP algorithms to predict diagnoses from medical data, drawing inspiration from a plethora of articles that have explored disease identification using AI.

A salient challenge in this domain is the inherent unstructured nature of medical data.Traditional methods often struggle to extract meaningful insights from such data, leading to inefficiencies in patient care. NLP, with its capability to process and analyze unstructured text, presents a promising solution. By harnessing the power of NLP, we aim to enhance the efficiency of the medical field, allowing healthcare professionals to focus on delivering superior patient care.

However, while the potential benefits of using NLP in medical data analysis are vast, it's paramount to address concerns related to data privacy and security. Before subjecting any medical data to NLP algorithms, it's essential to anonymize sensitive information,such as a patient's name or address, to uphold the highest standards of confidentiality and ethical research.

1.2 Justification

The integration of Natural Language Processing (NLP) algorithms into medical diagnostics offers transformative potential. Here are the primary justifications for this research:

- Enhanced Relationship Mapping: NLP algorithms can adeptly analyze vast amounts of unstructured medical data, drawing connections between medical findings and potential diagnoses. This capability can lead to more informed and timely medical decisions.
- Early Disease Detection: In an era where new diseases can emerge and spread rapidly, the ability to detect and diagnose them at the earliest stages are crucial. NLP can sift through patient data, identifying subtle patterns or anomalies that might be overlooked, thereby facilitating early intervention and treatment.
- Mitigating Cognitive Bias: Every medical professional, regardless of their expertise, can be influenced by cognitive biases that might skew their judgment.NLP algorithms, being data-driven and consistent, can help normalize these biases. By providing an objective analysis of patient data, NLP can assist healthcare professionals in making more accurate and unbiased diagnoses.

1.3 Methodology and techniques

We will be creating TF-IDF(Term Frequency, Inverse Document Frequency) based on tokens generated in each experiments. The tokens we will be generating using libraries like spacy, scispacy and HuggingFace based LLM(Large Language Models) transformers. We will also use SMOTE analysis if the data is unbalanced. Then we will use those vectors in classifier model to generate the classification Report. Below are the brief description of the techniques we will be using in further chapters

1.3.1 TF-IDF

TF-IDF stands for "Term Frequency, Inverse Document Frequency." It's a way to score the importance of words (or "terms") in a document based on how frequently they appear across multiple documents.[1]

[1] R. K. Gupta, "An introduction to tf-idf," *medium.com*, 2019

1.3.2 SMOTE

Imbalanced datasets pose a common challenge for machine learning practitioners in binary classification problems. This scenario frequently arises in practical business applications like fraud detection, spam filtering, rare disease discovery, and hardware fault detection. To address this issue, one popular technique is Synthetic Minority Oversampling Technique (SMOTE). SMOTE is specifically designed to tackle imbalanced datasets by generating synthetic samples for the minority class.[2]

[2] S. satpathy, "overcoming-class-imbalance-using-smote-techniques," *Analytics Vidya*, 2023

1.3.3 Medical Analysis using NER.

NER(Named Entity Recognition) is an important building block for NLP used for recognising and classifying named entities in text into predefined categories such as names of persons,

organizations, locations, and more[3] In the context of medical data analysis, NER becomes particularly crucial due to the intricate nature of medical terminologies and the need to extract specific entities like symptoms, diseases, medications, and patient conditions from unstructured clinical narratives.[4]

Several Python libraries, such as spacy and its medical extension medspacy, have been developed to harness the power of NER algorithms tailored for the medical domain [5] These tools not only recognize medical entities but also offer functionalities like section detection, sentence splitting, and assertion negation, which are essential for processing clinical notes

1.3.4 *Pretrained-BIOBERT Model*

BioBERT (Bidirectional Encoder Representations from Transformers for Biomedical Text Mining) is introduced, which is a domain-specific language representation model pre-trained on large-scale biomedical corpora.[6]

1.3.5 *Naive Bayes*

The Naive Bayes classifier is a supervised machine learning algorithm, which is commonly applied in use cases involving recommendation systems, text classification, and sentiment analysis. Because it performs well with data sets with high dimensionality, it is a favored classifier for text classification in particular.[7]

1.3.6 *Decision Tree*

A decision tree is a type of machine learning algorithm that falls under the supervised learning category. Basically, it is a flowchart that shows a clear and visual pathway to a decision or prediction. In fact, the decisions are conditional control statements that "ask questions" to the data based on its attributes and creates new branches. The flowchart starts from a single decision node that branches into two or more nodes, either another decision node or a prediction node. As a consequence, its final structure resembles a tree, that is where the name came from. In addition, this algorithm can be applied to classification tasks or regression tasks.[8]

[3] D. Nadeau and S. Sekine, *A survey of named entity recognition and classification*. Lingvisticae Investigationes, 2007

[4] Özlem Uzuner, B. R. South, S. Shen, and S. L. DuVall2, *challenge on concepts, assertions, and relations in clinical text*. Journal of the American Medical Informatics Association,, 2011

[5] G. K. Savova, Masanz, J. J., P. V. Ogren, J. Zheng, K.-S. Sohn, S., K. C., and C. G. Chute, *Mayo clinical Text Analysis and Knowledge Extraction System (cTAKES): architecture, component evaluation and applications*. Journal of the American Medical Informatics Association,, 2010

[6] S.-H. Tsang, "Biobert: a pre-trained biomedical language representation model for biomedical text mining," *medium.com*, 2023

[7] E. Kavlakoglu, "Classifying data using the multinomial naive bayes algorithm," *ibm.com*, 2024

[8] G. H. dos Santos, "A brief introduction to decision trees," *Analytics Vidya*, 2021

2 Context

2.1 Efficiency in Diagnosis

Hospitals and healthcare facilities often grapple with patient overcrowding. In such scenarios, swift and accurate diagnosis becomes imperative. As healthcare providers continue to focus on the patient of chronic disease, it is essential for stakeholders to accurately measure, report and improve the experience and outcomes of their patients.

A solution to mitigate the above limitations is the use of Artificial Intelligence (AI). AI has a relevant application in the health area and plays an increasingly important role in the area of biomedicine.

2.2 Variability in Diagnosis

Medical professionals, despite their expertise, can sometimes arrive at varying diagnoses for similar medical conditions. This variability often stems from cognitive biases influenced by factors such as their training, experiences, and personal beliefs. Patients often present with undifferentiated symptoms that change over time, making it difficult for clinicians to identify a satisfactory explanation of the patient's presenting problem.

Diagnostic uncertainty has been associated with diagnostic variation (physicians giving different diagnoses to the same patient), over-testing, unnecessary surgeries, more hospitalizations and referrals, and increased health care expenditure. Inappropriate management of diagnostic uncertainty could contribute to diagnostic errors or excess health care utilization. Additionally, rising costs related to diagnostic testing have led to recommendations for cost-containment, requiring physicians to carefully consider the resources they use and diagnostic decisions they make in the midst of uncertainty.Thus, inappropriate management of diagnostic uncertainty can impact both system and patient outcomes.

2.3 Extracting medical information from clinical text with nlp

The integration of Natural Language Processing (NLP) algorithms presents a promising avenue to address these challenges. By harnessing the analytical prowess of NLP, we aim to streamline the diagnostic process, ensuring both speed and accuracy, while also normalizing potential cognitive biases inherent in human judgment.

Pretrained models have revolutionized the field of Natural Language Processing (NLP) and machine learning at large. These models are initially trained on a vast corpora of unstructured

data, enabling them to capture intricate patterns, relationships, and linguistic structures present in the language [1]. The training often involves unsupervised learning tasks, such as predicting the next word or sentence given a sequence of words. Prominent sources for such training data include extensive datasets like Wikipedia, books, and in specialized cases, clinical medical data.

Once pretrained, these models can be fine-tuned on smaller, domain-specific datasets.This fine-tuning process allows the model to adapt to the nuances and intricacies of the 12 target domain, thereby enhancing its performance on specific tasks [2]. The advantage of this two-step approach—pretraining on a large dataset and fine-tuning on a smaller one—is that it significantly reduces the time, computational resources, and data required to achieve state-of-the-art results compared to training a model from scratch.

[1] Devlin, J "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.",2018

[2] Sun, C. "How to Fine-Tune BERT for Text Classification", 2019

3 Problem

The realm of medical diagnostics is riddled with challenges that necessitate innovative solutions. The primary issues this research seeks to address are

- In the context of efficiency in diagnosis the challenge lies in developing a model that not only pinpoints the diagnosis rapidly but also ensures that patients receive timely attention, with prioritization based on the severity and urgency of their condition.
- In the context of variability in diagnosis the challenge is to mitigate these biases to achieve a more consistent and accurate diagnostic outcome.
- In the context of Use of LLM Models in detection of diagnosis the challenge is to train the model is time consuming and requires lot of GPU processing units which are not available free of cost.

4 Objectives

4.0.1 General Objective

To comprehensively analyze medical data with the aim of uncovering patterns and relationships between patient findings and their corresponding diagnoses, leveraging the advanced capabilities of Natural Language Processing (NLP) algorithms.

4.0.2 Specific Objective

- Utilizing Named Entity Recognition (NER) Models: Investigate the relationship between medical findings and diagnoses by employing NER models, with a particular emphasis on tools like Spacy.
- Leveraging Large Language Models: Explore the potential of advanced language models, such as BERT and BIOBERT, in discerning patterns and relationships between medical findings and their corresponding diagnoses.
- Predicting Medical Urgency: Analyze medical findings to determine the type and severity of medical conditions, with the goal of predicting the urgency of care required for patients.

5 Literature Overview

5.1 Efficiency of natural language processing as a tool for analysing quality of life in patients with chronic diseases

In this article[1] ,We see that NLP can be used to extract information, convert unstructured text into a structured format, perform syntactic processing, capture meaning and identify relationships between concepts[2] NLP can be used in various scientific fields and for myriad purposes such as linguistic analysis, information retrieval, text translation, conversational bots, text classification, and human emotion analysis, among others[3] A solution to mitigate the above limitations is the use of Artificial Intelligence (AI). AI has a relevant application in the health area and plays an increasingly important role in the area of biomedicine [4] In this context, machine learning and, specifically, natural language processing (NLP) is going to be the critical methodology for processing unstructured free text.

[1] E. Lázaro, J.-C. Yepez, P. Marín-Maicas, P. López-Masés, T. Gimeno, S. de Paúl, and V. Moscardó, "Computers in human behavior reports," *ScienceDirect*, 2023

[2] A. Klein, H. Cai, D. Weissenbacher, L. Levine, and G. Gonzalez-Hernandez, "A natural language processing pipeline to advance the use of twitter data for digital epidemiology of adverse pregnancy outcomes," *Journal of Biomedical Informatics*, 2020

5.2 Defining and Measuring Diagnostic Uncertainty in Medicine: A Systematic Review

In this article[5], we see that Variability in diagnosis is described as the degree of variation between the diagnostic interpretations when a set of cases are examined by two or more independent clinicians. Clinical Bert and BioBert models have proved that when they are trained over large corpus of data can used to predict the problems in patient.

Using NLP models in a medical domain can be challenging. Pathologists can use different words to describe the same observation or use rare words, making it difficult to train an AI model to represent the images correctly. Key-word and topic selection are readily available for pathology reports. Moreover, a lack of a clean, large-scale, and universal data set for this domain is another challenge in using NLP methods for digital pathology.

[3] L. Guamán, L. Armando, A. Flores, and D. Omar, "Trabajo de titulación previo a la obtención del título de ingeniero en electrónica y telecomunicaciones," *Google Scholar*, 2017

5.3 Extracting medical information from clinical text with nlp

In this article[6] we see that there are three commonly used techniques in the NLP industry for extracting medical information from clinical text. They are described as below

[4] J. Sancho, C. Fanjul, M. D. la Iglesia Vayá, J. Montell, and M. Escartí, "Aplicación de la inteligencia artificial con procesamiento del lenguaje natural para textos de investigación cualitativa en la relación médico-paciente con enfermedad mental mediante el uso de tecnologías móviles," *Revista de Comunicación y Salud*, 2020

5.3.1 Rule-based Techniques

Rule-based systems typically work by defining specific patterns that match the target information, such as named entities or specific keywords, and then extracting that information based on those patterns. Rule-based systems are fast, reliable, and straightforward, but they are

[5] V. Bhise, S. S. R. F.

limited by the quality and number of rules defined, and they can be difficult to maintain and update.

5.3.2 *Statistical Techniques Using Machine Learning Models*

These techniques use statistical algorithms to learn patterns in the data and make predictions based on those patterns. Machine learning models can be trained on large amounts of annotated data, making them more flexible and scalable than rule-based systems. Several types of machine learning models are used in NLP, including decision trees, random forests, support vector machines, and neural networks.

5.3.3 *Transfer Learning*

These techniques are a hybrid approach combining the strengths of rule-based and machine-learning models. Transfer learning uses a pre-trained machine learning model, such as a language model trained on a large corpus of text, as a starting point for fine-tuning a specific task or domain

6 Data and Methods

6.1 Hypothesis

Below are the hypothesis statements we will be testing with our methodology.

- The tokens generated from medical scispacy NER models which only detects words with label pertaining to Disease or Organ failures will produce better classification of medical findings using TF-IDF than the spacy NER models which uses all the words present in the document.
- Similarly tokens generated from LLM (Large Language Models) based on BIOBERT produce better understanding of the medical findings and help classify Diagnosis Category more efficiently using TF-IDF(Term Frequency-Inverse Document Frequency) than the spacy NER models which uses all the words present in the document.

6.2 Dataset: California Medical Review.

The dataset we will use is from California Department of Managed Health Care(DMHC). The link for the data set is given as California Medical Review Dataset. This data consists of medical findings for each patient along with their diagnostics. There is only one file of 26.4 MB which forms the dataset and it is in csv (comma seperated values) format. There are records for 19245 different patients over a period from 2001 till 2016. The dataset consists of 11 columns and they are given as below

Column	Description	Data type
Reference ID	Reference ID of the Patient	Integer
Report Year	Year when the patient was reported	Integer
Diagnosis Category	Diagnosis Category of the patient	String
Treatment Category	Treatment for diagnosis of the patient	String
Treatment Sub Category	Sub Category for the diagnosis found	String
Type	Type of the Medical findings	String
Age Range	Age of Medical Patient	Integer
Patient Gender	Gender of the Medical patient	Binary
Findings	Medical Text of the findings of the patient	String
Determination	Determination of treatment of diagnosis	String

Table 6.1: Dataset Columns Representation.

6.3 Data Exploration

We have 30 different diagnosis category values of which some of them are Not applicable but they are minimal and should not affect the model. We can see that the majority of cases for diagnosis are orthopedic/musculoskeletal issues.

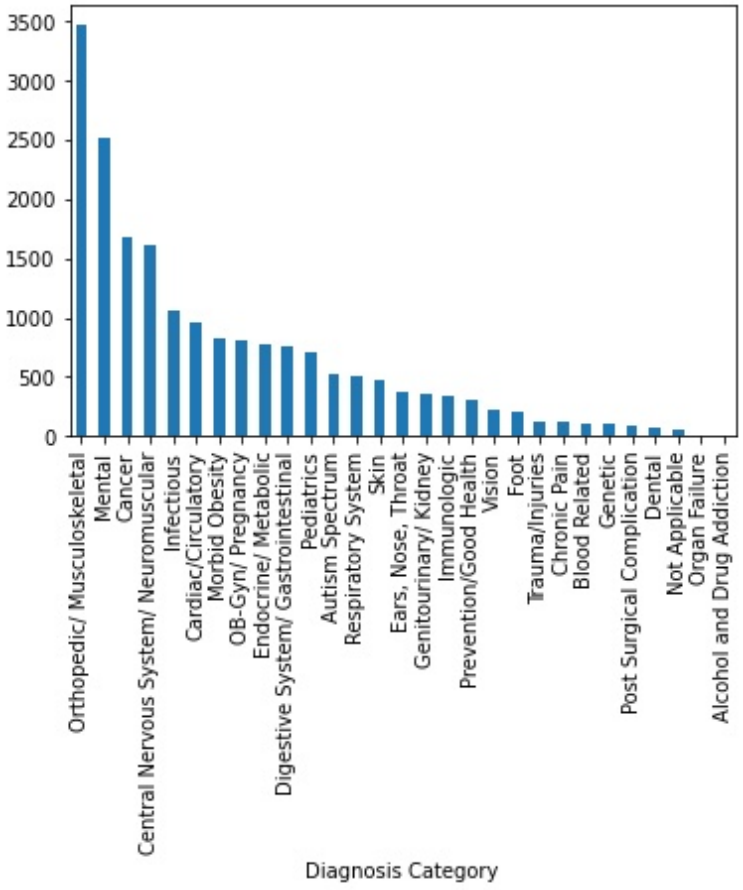

Figure 6.1: Diagnosis Category value counts

Index	Medical Findings
0	Nature of Statutory Criteria/Case Summary: An ....
1	Nature of Statutory Criteria/Case Summary: An....
2	Nature of Statutory Criteria/Case Summary: Th...
3	Nature of Statutory Criteria/Case Summary: An....
.	...
.	...
.	...
19244	A male requested participation in a Phase I st...

Table 6.2: Medical findings record

We see many records as in Table 6.2 starts with a particular string 'Nature of Statutory Criteria/Case Summary' which we can remove them in preprocessing stage. Also while checking the diagnosis between genders we don't see any diagnosis/disease unique among them that means they share the same types of diseases. Most of the cases recorded are from 2015 and least cases are from 2001. Also we see that there are three types of medical records i.e 'Medical Necessity','Experimental/Investigation' and 'Urgent Care'. We also found that there were nan values but it was significantly low compared to total number of records.

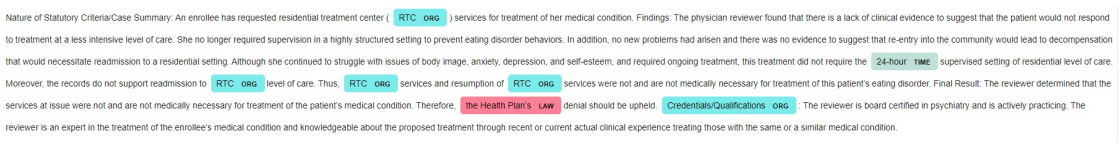

Figure 6.2: Spacy Named Entity Recognition tokenizer

We will also remove stop words and punctuations using the spacy module. Also applying lemmatization where we move the words to its lemma would also help decrease the number of tokens. The medical data is also unstructured and has a lot of whitespaces so we would have to remove them using python re module.

We will also use medical scispacy pre-trained model en_ner_bc5cdr_md to find the labels for medical findings for Diseases or Chemicals. In this case we are only interested in labeling of Diseases as in below Figures 6.3 6.4

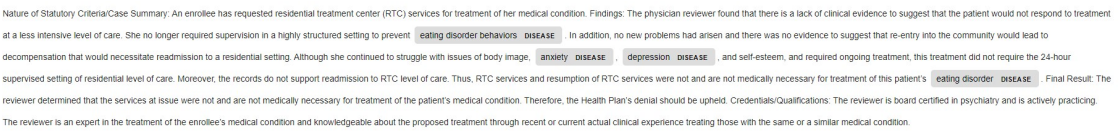

Figure 6.3: Scispacy LLM bc5cdr Named Entity Recognition tokenizer

Similarly we can also apply other pre-trained medical scispacy modules to label organs and cancers as they will not always be classified as diseases. Scispacy module of bionlp13cg provides us that option to find label Organs and Cancer in the text. Some of the examples of labeling based on medical scispacy modules below.

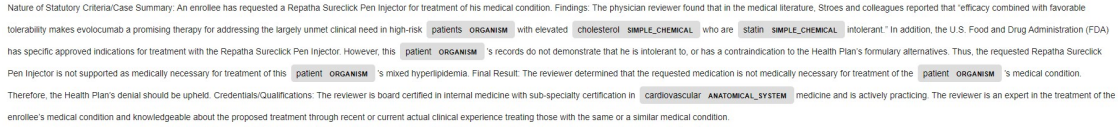

Figure 6.4: Scispacy LLM bionlp13cg Named Entity Recognition tokenizer

Similarly we can use Hugging Face BERT based token classifiers as in Figure 6.5 which have been trained with a large amount of data. One such BERT based tokenizer classifier is "sschet/biobert_diseases_ner". It has been trained over corpus datasets such as ncbi_disease, tner/bc5cdr and bc2gm_corpus.

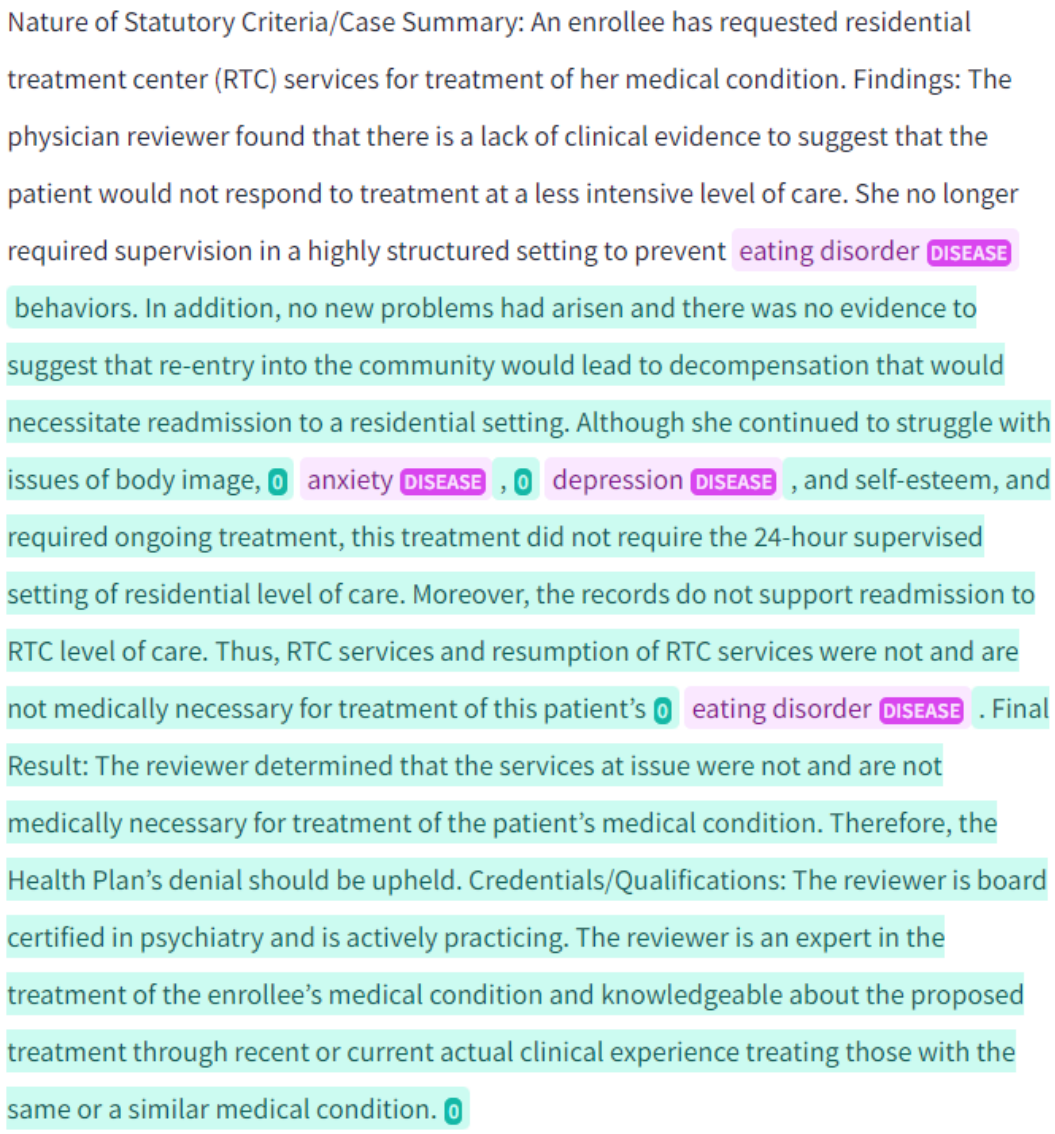

Figure 6.5: Hugging Face sschet/biobert_disease_ner token classifier

6.4 *Tokenizer Methods.*

We will be using the python tokenizer method to preprocess the data to useful information only after which we can tokenize and convert to word vectors for creating a model to categorize diagnosis. We will be using spacy, medical scispacy modules and Hugging Face Biobert tokenizers.

6.5 *Conversion of Text to Numbers.*

In the realm of text representation and information retrieval, Term Frequency-Inverse Document Frequency (TF-IDF) stands out as a prominent technique to quantify the importance of words

within a document relative to a collection of documents or a corpus. The underlying principle of TF-IDF is to weigh words not just by their frequency in a single document (Term Frequency or TF) but also by how unique or rare they are across the entire corpus (Inverse Document Frequency or IDF).

- Term Frequency (TF): Represents the frequency of a word in a specific document. It is calculated as the number of times a word appears in a document divided by the total number of words in that document.
- Inverse Document Frequency (IDF): Measures the significance of a word across a corpus. It is computed as the logarithm of the total number of documents divided by the number of documents containing the word.

The combined TF-IDF score for a word is given by the product of its TF and IDF scores: TF*IDF. This results in a matrix representation where each column corresponds to a word in the vocabulary, and each row captures the TF-IDF value of that word for a particular document. The essence of this representation is to diminish the weight of words that frequently appear in many documents (e.g., common stopwords) while emphasizing words that are more unique and potentially more informative in the context of the entire corpus [1]

[1] G. Salton and C. Buckley, *Term-weighting approaches in automatic text retrieval.* Information processing and management, 1988

Experiments	Dataset shape
Experiment 1	(19245,)
Experiment 2	(19245,)
Experiment 3	(19245,)

Table 6.3: Medical findings before TF-IDF

The text data needs to be converted to vectors and we will use TF-IDF vectorization which will calculate its term frequency and inverse document frequency. We will pass different tokenizer methods to get different vectors in each experiment.

TF-IDF vectorization involves calculating the TF-IDF score for every word in your corpus relative to that document and then putting that information into a vector. we would see the shape of column changed based on number of words we take into consideration in each experiments. We can compare from Tables 6.3 and 6.4 shapes of vectors generated before and after TF-IDF for each experiments.

Experiments	TF-IDF vectors shape
Experiment 1	(19245, 21963)
Experiment 2	(19245, 8571)
Experiment 3	(19245, 3488)

Table 6.4: Medical findings after TF-IDF

We will now apply label encoding to the Diagnosis column, so that both input and output are converted into numeric values which can then be applied to a model for prediction. In this case since we have 30 different diagnosis categories as shown in Table 6.5, they will be labeled from 0 to 29.

Diagnosis Category	Label Encoded Value
Alcohol and Drug Addiction	0
Autism Spectrum	1
Blood Related	2
Cancer	3
Cardiac/Circulatory	4
Central Nervous System/ Neuromuscular	5
Chronic Pain	6
Dental	7
Digestive System/ Gastrointestinal	8
Ears, Nose, Throat	9
Endocrine/ Metabolic	10
Foot	11
Genetic	12
Genitourinary/ Kidney	13
Immunologic	14
Infectious	15
Mental	16
Morbid Obesity	17
Not Applicable	18
OB-Gyn/ Pregnancy	19
Organ Failure	20
Orthopedic/ Musculoskeletal	21
Pediatrics	22
Post Surgical Complication	23
Prevention/Good Health	24
Respiratory System	25
Skin	26
Trauma/Injuries	27
Vision	28
nan	29

Table 6.5: Label encoded values

6.6 *SMOTE analysis.*

As we are getting imbalances in categorical values shown in Table 6.6, we would have to apply oversampling of minority classes using SMOTE analysis. With SMOTE sampling of data, we would get better predictions using models like Naive Bayes or other classification models.

The Synthetic Minority Oversampling Technique (SMOTE), offers a solution to this challenge. Instead of merely oversampling the minority class or under sampling the majority class, SMOTE takes a more nuanced approach of generating synthetic samples in the feature space to balance out the class distribution.

Diagnosis Category	Count
Orthopedic/ Musculoskeletal	3469
Mental	2512
Cancer	1681
Central Nervous System/ Neuromuscular	1620
Infectious	1059
Cardiac/Circulatory	965
Morbid Obesity	824
OB-Gyn/ Pregnancy	801
Endocrine/ Metabolic	779
Digestive System/ Gastrointestinal	758
Pediatrics	709
Autism Spectrum	524
Respiratory System	513
Skin	481
Ears, Nose, Throat	376
Genitourinary/ Kidney	350
Immunologic	343
Prevention/Good Health	300
Vision	224
Foot	214
Trauma/Injuries	127
Chronic Pain	122
Blood Related	101
Genetic	99
Post Surgical Complication	85
Dental	76
Not Applicable	63
Organ Failure	8
Alcohol and Drug Addiction	3

Table 6.6: Diagnosis Category type counts before SMOTE

This technique not only balances the class distribution but also makes the decision boundary in the classifier more general, leading to improved classification performance [2]. A significant advantage of using SMOTE is the enhancement of the classifier's sensitivity to the minority class, ensuring that the model can effectively detect and classify instances from the underrepresented category.

We see in Table 6.6 before SMOTE analysis there are many imbalances for various types of diagnosis category and after applying the SMOTE analysis as in Table 6.8 we can see the number for all types of diagnosis category values are equalized by adding duplicated values to minor classes. This also increased the size of the variables.

[2] A. Fernández, S. García, F. Herrera, and N. V. Chawla, *SMOTE for learning from imbalanced data: progress and challenges, marking the 15-year 31 anniversary.* Journal of artificial intelligence research,, 2018

Diagnosis Category	Count
Orthopedic/ Musculoskeletal	3469
Mental	3469
Cancer	3469
Central Nervous System/ Neuromuscular	3469
Infectious	3469
Cardiac/Circulatory	3469
Morbid Obesity	3469
OB-Gyn/ Pregnancy	3469
Endocrine/ Metabolic	3469
Digestive System/ Gastrointestinal	3469
Pediatrics	3469
Autism Spectrum	3469
Respiratory System	3469
Skin	3469
Ears, Nose, Throat	3469
Genitourinary/ Kidney	3469
Immunologic	3469
Prevention/Good Health	3469
Vision	3469
Foot	3469
Trauma/Injuries	3469
Chronic Pain	3469
Blood Related	3469
Genetic	3469
Post Surgical Complication	3469
Dental	3469
Not Applicable	3469
Organ Failure	3469
Alcohol and Drug Addiction	3469

Table 6.7: Diagnosis Category type counts after SMOTE

Experiments	TF-IDF vectors shape
Experiment 1	(104070, 21963)
Experiment 2	(104070, 8571)
Experiment 3	(104070, 3488)

Table 6.8: TF-IDF vectors shape after SMOTE analysis

6.7 *Naive Bayes.*

First we need to split the data to train and test datasets. We will use the train dataset to train our Naive Bayes model and then we will use the test dataset to predict and check the metrics.

The Naive Bayes classifier is a fundamental algorithm in the realm of machine learning,particularly in the domain of text classification [3]. It is a probabilistic model grounded on Bayes' theorem, with the "naive" assumption that features (or categories) are conditionally independent given the class label. This means that the presence or absence of a particular feature is unrelated to the presence or absence of any other feature, given the class variable.

[3] A. McCallum and K. Nigam, *A comparison of event models for naive Bayes text classification.* AAAI-98 workshop on learning for text categorization, 1998

One of the prerequisites for the optimal performance of the Naive Bayes classifier is a balanced dataset. Imbalanced data can lead to skewed probabilities, which might affect the classifier's decision-making capability [4].

[4] N. Japkowicz and S. Stephen, *The class imbalance problem: A systematic study.* Intelligent data analysis, 2002

According to Bayes' theorem, the probability of a record belonging to a particular class is given by:

$$P(C_i|d) = \frac{P(d|C_i)P(C_i)}{P(d)} \tag{6.1}$$

where:

- $P(C_i|d)$ is the posterior probability of class given predictor (or feature) d
- $P(d|C_i)$ is the likelihood which is probability of predictor d given predictor(or features) C_i
- $P(C_i)$ is the prior probability of class C_i
- $P(d)$ is the prior probability of predictor d

Despite its simplicity and the naive assumption of feature independence, the Naive Bayes classifier has proven effective in various applications, especially in scenarios where dimensionality is high, such as text classification [5].

[5] I. Rish, *An empirical study of the naive Bayes classifier.* IJCAI 2001 workshop on empirical methods in artificial intelligence, 2001

6.8 *Description of the Experiments.*

We will be performing 3 different experiments based on our two hypothesis where we will generate different types of tokens before passing it to the TF-IDF vectorizer after which we will generate vectors. We will be creating 3 different types of token methods for vectorizer.

6.8.1 *Experiment 1*

In the first experiment the method will use the NER(Named Entity Recognition) based spacy python module to remove all punctuations and stopwords and also to decrease the words to its lemma. This will take into consideration all the words present in the document irrespective of whether it is associated with medical terms or not.

The process of lemmatization reduces words to their base or dictionary form, and it can further enhance the accuracy of NER tools. By converting words to their lemmas, NER tools can achieve a more contextual understanding of medical findings, ensuring that variations of a term are recognized as the same entity.

6.8.2 Experiment 2

Transfer learning has emerged as a powerful technique in machine learning, allowing models trained on one task to be repurposed for another related task. In the context of medical Natural Language Processing (NLP), transfer learning becomes particularly invaluable. Given the specialized nature of medical language and the scarcity of annotated medical data, leveraging pretrained models on medical corpora can significantly boost performance on specific medical tasks [6].

A notable resource in this domain is the scispacy library, which offers models specifically tailored for biomedical texts. By utilizing medical corpus data from scispacy, one can filter out irrelevant information, retaining only pertinent medical content. This refined data can then serve as a foundation for encoding purposes in subsequent models [7].

For instance, the open-source model en_ner_bc5cdr_md is trained on the BC5CDR corpus, a dataset rich in annotations related to diseases and chemicals. This model is adept at extracting mentions of diseases and chemicals from text [8]. Given the current focus on disease diagnosis, extracting such labels can significantly streamline the information, distilling vast textual data into actionable insights.

Another noteworthy model is en_ner_bionlp13cg_md, which is trained on the BIONLP13CG corpus. This model specializes in labeling text related to cancer and specific organ issues, which might not be traditionally categorized as diseases but are of paramount importance in medical diagnostics [9].

In the second experiment the method will use medical scispacy pre-trained models(BIONLP/BIONLP13CG) as mentioned above to generate tokens based on diseases or organ failures or cancer. First priority will be given to detection of diseases and then if we don't get any tokens, it will move on to detect organ failures or Cancer. If in both cases we don't detect any tokens then we will use the english based spacy ner tokens. In all cases we will remove the stop words. So we will convert only medically significant words in the document to TF-IDF vectors.

[6] J. R. Zech, M. A. Badgeley, M. Liu, A. B. Costa, Titano, J. J., and E. K. Oermann, *Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: A cross-sectional study.* PLOS Medicine, 2018

[7] M. Neumann, D. King, I. Beltagy, and W. Ammar, *ScispaCy: Fast and Robust Models for Biomedical Natural Language Processing.* arXiv preprint arXiv, 2019

[8] J. Li, Y. Sun, R. J. Johnson, D. Sciaky, C. H. Wei, R. Leaman, and Z. Lu, *BioCreative V CDR task corpus: a resource for chemical disease relation extraction.* Database, 2016

[9] S. Pyysalo, T. Ohta, R. Rak, D. Sullivan, C. Mao, C. Wang, , and S. Ananiadou, *Overview of the ID, EPI and REL tasks of BioNLP Shared Task 2011.* BMC bioinformatics, 2012

6.8.3 Experiment 3

In the third experiment we will be using Hugging Face BIOBERT based token classifier sschet/biobert_diseases_ner. In this case we will generate tokens using pipeline provided by pre-trained model.

BERT, which stands for Bidirectional Encoder Representations from Transformers, is based on transformers, a deep learning model in which every output element is connected to every input element, and the weightings between them are dynamically calculated based upon their connection.

Historically, language models could only read input text sequentially – either left-to-right or right-to-left – but couldn't do both at the same time. BERT is different because it's designed to read in both directions at once. The introduction of transformer models enabled this capability, which is known as bidirectionality. Using bidirectionality, BERT is pretrained on two different but related NLP tasks: masked language modeling (MLM) and next sentence prediction (NSP).

The objective of MLM training is to hide a word in a sentence and then have the program predict what word has been hidden based on the hidden word's context. The objective of NSP training is to have the program predict whether two given sentences have a logical, sequential connection or whether their relationship is simply random.

BERT also relies on a self-attention mechanism that captures and understands relationships among words in a sentence. The bidirectional transformers at the center of BERT's design make this possible. This is significant because often, a word may change meaning as a sentence develops. Each word added augments the overall meaning of the word the NLP algorithm is focusing on. The more words that are present in each sentence or phrase, the more ambiguous the word in focus becomes. BERT accounts for the augmented meaning by reading bidirectionally, accounting for the effect of all other words in a sentence on the focus word and eliminating the left-to-right momentum that biases words towards a certain meaning as a sentence progresses.[10][10]

[10] S. Pyysalo, T. Ohta, R. Rak, D. Sullivan, C. Mao, C. Wang, , and S. Ananiadou, *Overview of the ID, EPI and REL tasks of BioNLP Shared Task 2011*. BMC bioinformatics, 2012

The model sschet/biobert disease ner token classifier which we will be using in our third experiment is based on DistilBERT tokenizer which uses WordPiece subword segmentation. The DistilBERT model was proposed in the blog post Smaller, faster, cheaper, lighter: Introducing DistilBERT, a distilled version of BERT, and the paper DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. DistilBERT is a small, fast, cheap and light Transformer model trained by distilling BERT base. It has 40% less parameters than google-bert/bert-base-uncased, runs 60% faster while preserving over 95% of BERT's performances as measured on the GLUE language understanding benchmark[11]. Below Table 6.9 describes the default settings of the Hugging Face Biobert model parameters.

[11] V. Sanh, "Smaller, faster, cheaper, lighter: Introducing distilbert, a distilled version of bert," *medium.com*, 2019

Parameter	**Value**
Type	Token Classification
Model-Type	DistilBERT
Dropout	0.1
Activation	GELU
Max Position Embeddings	512
Layers	6
Attention Heads	12
Vocab Size	30522
Dimension	768

Table 6.9: Bert Model Parameters.

Here our wordPiece tokenizer will check whether the word is present in our vocabulary. If the word is present then it will used as a token but if not then our word is split into subwords recursively until the subwords are found in our corpus. This process is effective in handling the out of vocabulary words. Below Table 6.10 shows how the words from the sentence are split into subwords.

Word Piece Tokenizer
'ha'
'##r'
'##von'
'##i'
'for'
'treatment'
'of'
'his'
'he'
'##pa'
'##titis'
'c'
'virus'
'g'
'##eno'
'##type'
'1'
'##a'
'.'

Table 6.10: Word Piece Tokenizers

6.9 *Description of the Metrics.*

[12]

[12] A. Zheng, *Evaluatiing Machine Learning Models*. Oreilly Media, 2015

In machine learning, various metrics are employed to assess the performance and effectiveness of models. One fundamental metric is accuracy, which measures the proportion of correctly predicted instances over the total number of instances. While accuracy provides a broad overview, it might not be sufficient in scenarios where class distribution is imbalanced. Precision and recall offer a more nuanced evaluation. Precision gauges the accuracy of positive predictions, emphasizing the relevance of identified instances. On the other hand, recall, also known as sensitivity or true positive rate, assesses the model's ability to capture all relevant instances within the dataset. Striking a balance between precision and recall is crucial and is often visualized using the F1 score, which combines both metrics into a single value, especially beneficial when there's a need for a comprehensive performance assessment.

Additionally, in binary classification problems, metrics like the area under the receiver operating characteristic curve (AUC-ROC) provide insights into a model's ability to discriminate between classes across various thresholds. The confusion matrix, detailing true positives, true negatives, false positives, and false negatives, is a cornerstone for deriving these metrics and gaining a more granular understanding of model behavior. These metrics collectively contribute to a comprehensive evaluation framework, aiding practitioners in fine-tuning models for optimal performance in diverse real-world applications.

6.9.1 Confusion Matrix

A confusion matrix is a tabular representation that provides a detailed breakdown of a machine learning model's performance by categorizing predictions into four categories: true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). True positives are instances where the model correctly predicts the positive class, while true negatives are instances correctly predicted as the negative class. False positives represent instances incorrectly predicted as positive, and false negatives are instances incorrectly predicted as negative.

The confusion matrix serves as a foundation for deriving various performance metrics, such as accuracy, precision, recall, and the F1 score. These metrics are essential for evaluating different aspects of a model's performance. For instance, accuracy measures overall correctness, precision assesses the accuracy of positive predictions, recall evaluates the ability to capture all relevant instances, and the F1 score strikes a balance between precision and recall. By analyzing the confusion matrix, practitioners can gain insights into the strengths and weaknesses of a model, identify areas of improvement, and make informed decisions about adjusting the model's parameters or employing different algorithms for enhanced performance. Overall, the confusion matrix provides a comprehensive and granular perspective on a model's behavior, aiding in the iterative process of refining machine learning models for optimal results.

Precision, recall, and accuracy are key metrics used to evaluate the performance of classification models.

6.9.2 Precision.

Precision is a measure of the accuracy of positive predictions made by the model. It is calculated as the ratio of true positives (correctly predicted positive instances) to the sum of true positives and false positives (instances incorrectly predicted as positive).

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

A high precision value indicates that when the model predicts the positive class, it is often correct, minimizing false positives.

6.9.3 Recall.

Recall assesses the model's ability to capture all relevant instances of the positive class. It is calculated as the ratio of true positives to the sum of true positives and false negatives (instances incorrectly predicted as negative).

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

A high recall value signifies that the model can effectively identify most positive instances, reducing false negatives.

6.9.4 *Accuracy.*

Accuracy provides an overall measure of the model's correctness. It is calculated as the ratio of the sum of true positives and true negatives to the total number of instances.

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

While accuracy is a fundamental metric, it may be misleading in imbalanced datasets, as it does not consider the distribution of classes. It's crucial to interpret accuracy alongside precision and recall for a comprehensive evaluation.

These metrics, derived from the confusion matrix, collectively offer insights into different aspects of a model's performance, allowing practitioners to fine-tune models based on specific objectives and requirements.

6.9.5 *F1 Score*

The F1 score is a metric that combines precision and recall into a single value, providing a balanced measure of a classification model's performance. It is particularly useful when there is a need to strike a balance between minimizing false positives and false negatives.The F1 score is calculated using the following formula

$$F1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

It takes into account both false positives and false negatives, making it a suitable metric when there is an inherent trade-off between precision and recall. A high F1 score implies a model that performs well in terms of both precision and recall.

The F1 score is especially beneficial in scenarios where there is an uneven class distribution or when false positives and false negatives carry different degrees of importance. By considering both aspects of the model's performance, the F1 score provides a more comprehensive evaluation, guiding practitioners in optimizing their models for a balanced performance across various applications.

7 Results and Discussion

7.1 Experiment 1

Label	Precision	Recall	F1-score	Support
0	0.00	0.00	0.00	2
1	0.88	0.19	0.31	160
2	0.00	0.00	0.00	37
3	0.71	0.83	0.76	476
4	0.94	0.39	0.55	304
5	0.83	0.36	0.50	483
6	0.00	0.00	0.00	35
7	0.00	0.00	0.00	30
8	0.93	0.19	0.31	222
9	0.00	0.00	0.00	120
10	0.84	0.51	0.64	219
11	0.00	0.00	0.00	55
12	0.00	0.00	0.00	28
13	0.00	0.00	0.00	115
14	0.00	0.00	0.00	112
15	0.97	0.81	0.88	321
16	0.61	0.97	0.75	697
17	0.86	0.77	0.81	253
18	0.00	0.00	0.00	14
19	0.59	0.23	0.33	234
20	0.00	0.00	0.00	2
21	0.38	0.98	0.54	1090
22	0.90	0.42	0.57	219
23	0.00	0.00	0.00	25
24	0.00	0.00	0.00	89
25	1.00	0.08	0.14	159
26	1.00	0.01	0.03	149
27	0.00	0.00	0.00	43
28	0.00	0.00	0.00	62
29	0.00	0.00	0.00	19

Table 7.1: Naive Bayes Classifier(imbalanced data)(Experiment 1).

Precision	Recall	F1-score	Accuracy
0.62	0.56	0.5	0.56

Table 7.2: Classification Report (imbalanced data)(Experiment 1).

We got low classification score on experiment 1 for imbalanced data, after applying SMOTE analysis to all the data we got better classification score, so we applied SMOTE analysis for the other two experiments.

Label	Precision	Recall	F1-score	Support
0	0.94	1.00	0.97	1045
1	0.75	0.87	0.81	1021
2	0.91	0.88	0.90	1023
3	0.82	0.73	0.78	1064
4	0.87	0.75	0.81	1056
5	0.74	0.57	0.65	1037
6	0.69	0.86	0.77	1066
7	0.97	0.97	0.97	1079
8	0.87	0.69	0.77	1067
9	0.76	0.80	0.78	1073
10	0.85	0.76	0.80	1051
11	0.88	0.92	0.90	1042
12	0.95	0.82	0.88	1035
13	0.90	0.84	0.87	1035
14	0.81	0.75	0.78	1030
15	0.95	0.87	0.91	1026
16	0.85	0.83	0.84	1083
17	0.74	0.92	0.82	1031
18	0.83	0.83	0.83	1028
19	0.82	0.61	0.70	1033
20	0.97	1.00	0.99	1034
21	0.77	0.65	0.71	1058
22	0.83	0.74	0.78	1027
23	0.73	0.89	0.80	1056
24	0.66	0.82	0.73	993
25	0.85	0.79	0.82	1040
26	0.91	0.72	0.81	1048
27	0.68	0.94	0.79	996
28	0.94	0.92	0.93	995
29	0.48	0.65	0.55	1049

Table 7.3: Naive Bayes Classifier(balanced data)(Experiment 1).

Precision	Recall	F1-score	Accuracy
0.82	0.81	0.81	0.81

Table 7.4: Classification Report(balanced data)(Experiment 1).

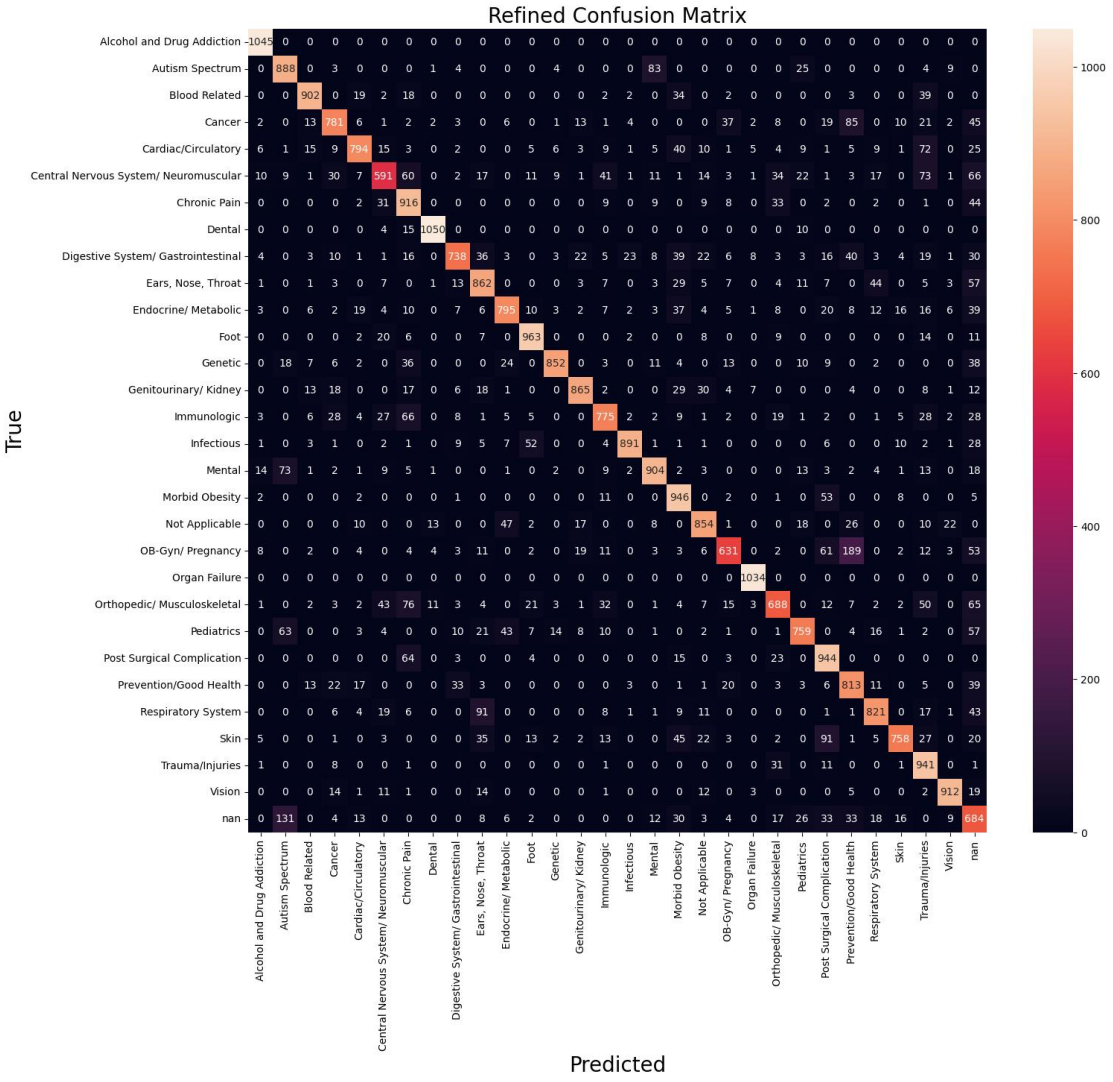

Figure 7.1: Confusion Matrix (Experiment 1)

7.2 Experiment 2

Label	Precision	Recall	F1-score	Support
0	0.83	1.00	0.91	1045
1	0.73	0.89	0.80	1021
2	0.86	0.82	0.84	1023
3	0.70	0.69	0.69	1064
4	0.72	0.76	0.74	1056
5	0.71	0.59	0.65	1037
6	0.46	0.88	0.61	1066
7	0.93	0.89	0.91	1079
8	0.81	0.68	0.74	1067
9	0.82	0.72	0.77	1073
10	0.88	0.70	0.78	1051
11	0.85	0.78	0.81	1042
12	0.89	0.78	0.83	1035
13	0.90	0.79	0.84	1035
14	0.81	0.66	0.72	1030
15	0.83	0.88	0.85	1026
16	0.81	0.79	0.80	1083
17	0.67	0.89	0.77	1031
18	0.86	0.64	0.73	1028
19	0.75	0.58	0.65	1033
20	0.92	0.98	0.95	1034
21	0.67	0.54	0.60	1058
22	0.80	0.65	0.72	1027
23	0.73	0.54	0.62	1056
24	0.54	0.75	0.63	993
25	0.75	0.81	0.78	1040
26	0.77	0.70	0.73	1048
27	0.76	0.83	0.79	996
28	0.92	0.87	0.89	995
29	0.58	0.68	0.63	1049

Table 7.5: Naive Bayes Classifier(balanced data)(Experiment 2).

Precision	Recall	F1-score	Accuracy
0.78	0.76	0.76	0.76

Table 7.6: Classification Report(balanced data)(Experiment 2).

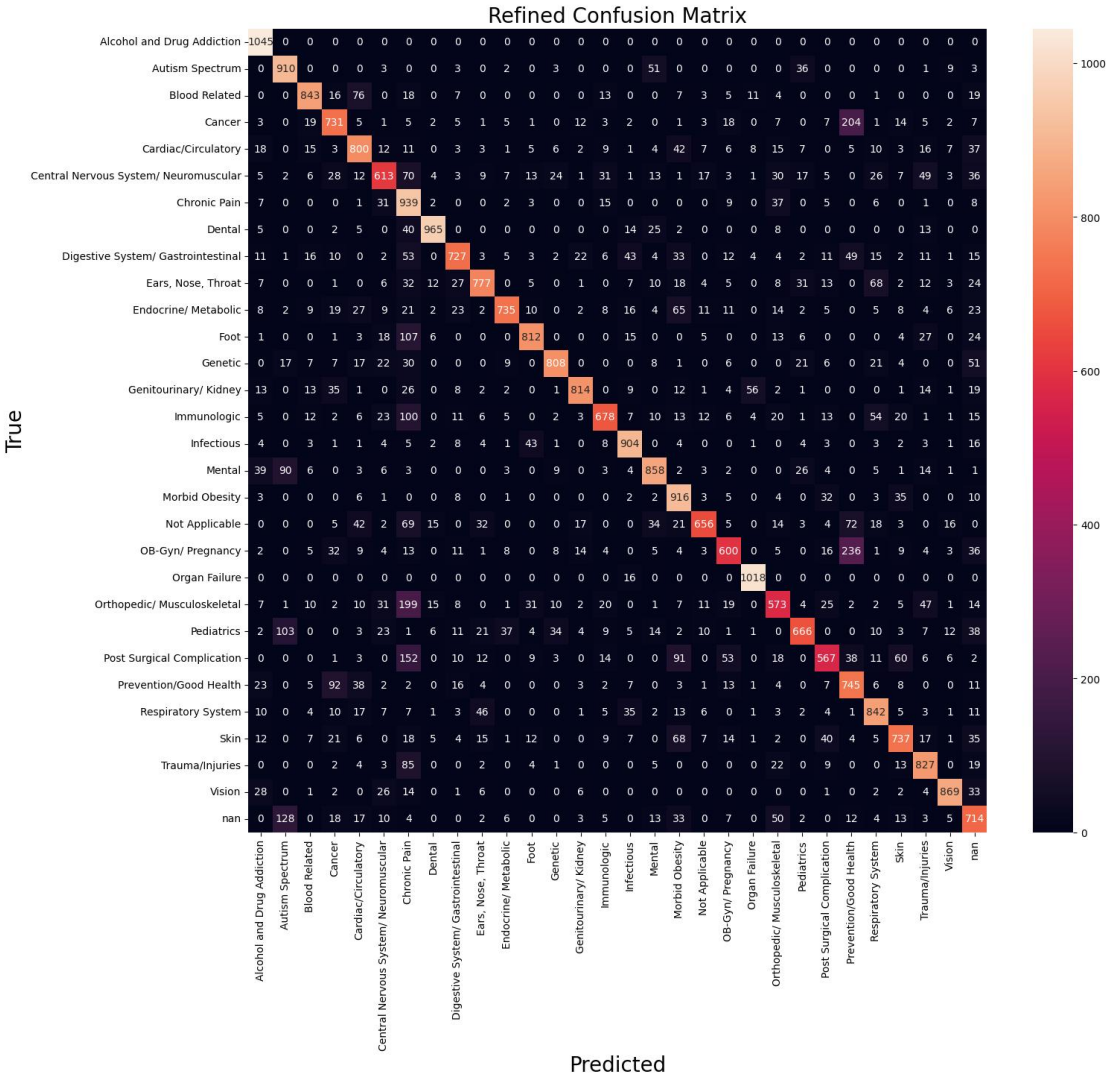

Figure 7.2: Confusion Matrix (Experiment 2)

7.3 Experiment 3

Label	Precision	Recall	F1-score	Support
0	0.96	1.00	0.98	1045
1	0.77	0.89	0.82	1021
2	0.81	0.68	0.74	1023
3	0.69	0.65	0.67	1064
4	0.67	0.76	0.72	1056
5	0.65	0.64	0.64	1037
6	0.68	0.73	0.70	1066
7	0.90	0.78	0.84	1079
8	0.70	0.65	0.67	1067
9	0.78	0.74	0.76	1073
10	0.78	0.73	0.75	1051
11	0.86	0.80	0.83	1042
12	0.87	0.74	0.80	1035
13	0.84	0.75	0.79	1035
14	0.72	0.65	0.68	1030
15	0.87	0.86	0.86	1026
16	0.79	0.79	0.79	1083
17	0.67	0.83	0.74	1031
18	0.82	0.58	0.68	1028
19	0.69	0.54	0.60	1033
20	0.92	1.00	0.96	1034
21	0.56	0.65	0.60	1058
22	0.82	0.74	0.78	1027
23	0.74	0.59	0.66	1056
24	0.29	0.78	0.42	993
25	0.69	0.77	0.73	1040
26	0.80	0.72	0.76	1048
27	0.78	0.78	0.78	996
28	0.83	0.85	0.84	995
29	0.81	0.23	0.36	1049

Table 7.7: Naive Bayes Classifier(balanced data)(Experiment 3).

Precision	Recall	F1-score	Accuracy
0.76	0.73	0.73	0.73

Table 7.8: Classification Report(balanced data)(Experiment 3).

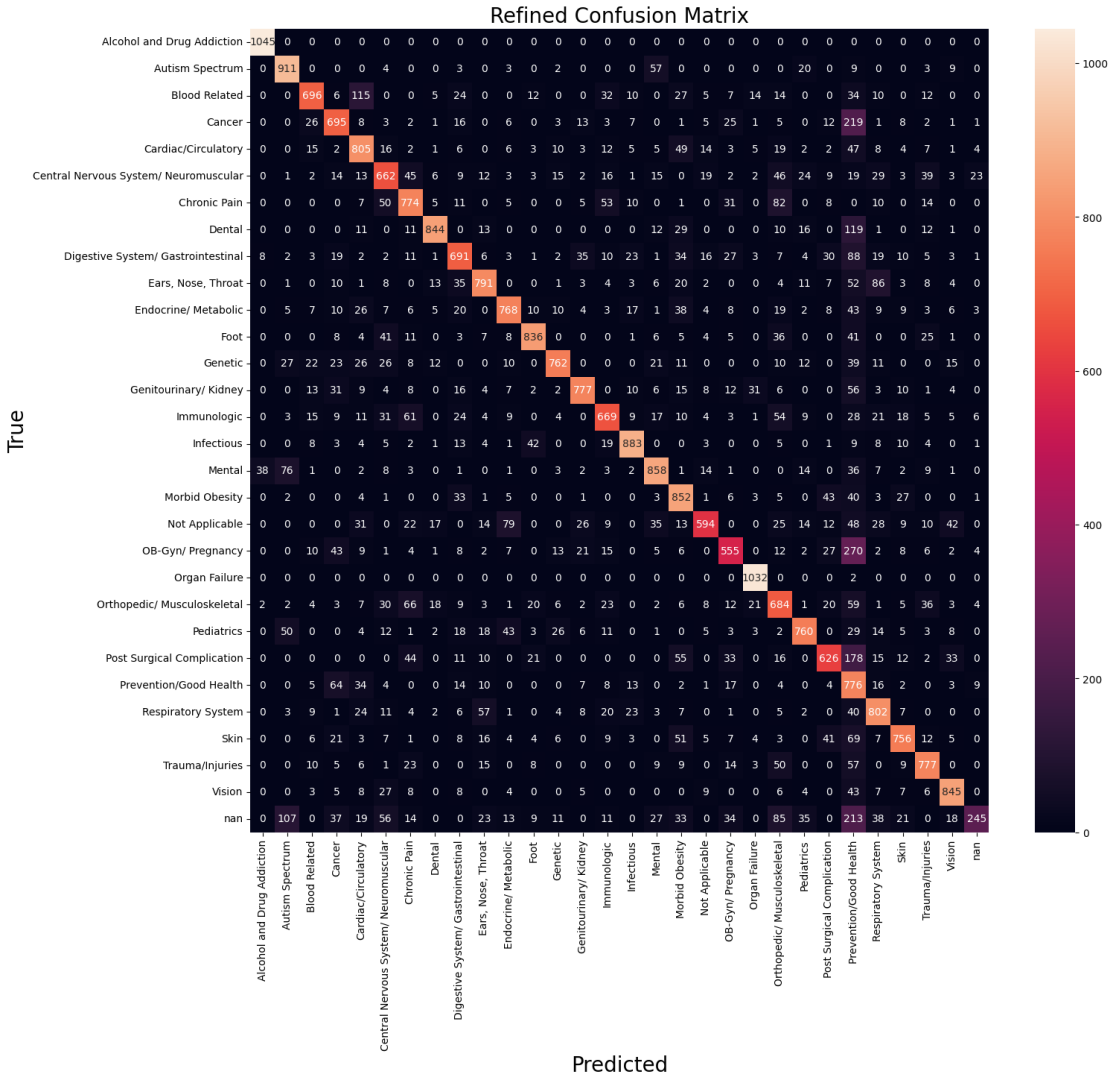

Figure 7.3: Confusion Matrix (Experiment 3)

Experiment	Time Taken
Experiment 1	13min 33s
Experiment 2	33min 18s
Experiment 3	4h 59min 50s

Table 7.9: Time taken for each experiments.

7.4 *Analysis and Discussion:*

- From the above experiments performed, the performance metrics suggests that model from experiment 1 has better classification than all other experiments.
- The time taken for training a model increases as we go from one experiment to another.

Maximum time is taken by experiment 3 which is almost 5 hours, while minimum time is taken by experiment 1 which is around 14 minutes.

- Hugging Face Model(Experiment 3) uses LLM model to predict the classification. This LLM model has already been trained over a large dataset however it takes lot of time to train the model over new dataset and also it gives lower classification scores than the other two experiments.
- In the experiments Diagnosis category values are converted to numbers from 0 to 29 based on their type. We have noticed that higher diagnosis value have lower prediction rate. We can do another experiment of taking 9 lower predicted values for Diagnosis category in previous experiments and convert them in range from 0 to 8 and see if we get better prediction for them.

7.4.1 *Processing time for Vectorization of medical findings*

Time taken in each of the experiments depends on time it takes to create the TF-IDF vectors of the Medical findings. Experiment 3 decreases the number of column vectors by significant amount however the time taken to create these vectors is significantly high. It takes 13 minutes to create the vectors in experiment 1 while it takes 5 hours to creates vectors in experiment 3 and it takes 30 minutes to create the vectors in experiment 2. Below Table 7.10 show the processing time of each row in the dataset for all the three experiments.

Experiment	**Processing Time**
Experiment 1	59.8ms
Experiment 2	118 ms
Experiment 3	1.39s

Table 7.10: Processing Time for calculating TF-IDF vectors.

In case of experiment 1 we just remove stop words and take all the words in the sentence for tokenization due to which there is less processing time. In experiment 2 we filter out words relevant to medical terms like cancer and organ due to which it takes littler more time than experiment 1. In experiment 3 WordPiece tokenizer is used to generate the tokens. Wordpiece Tokenizer for each row takes about 1.4 due to which convertion of tf-idf vectors for about 19000 rows takes about 5 hours which is significantly higher than the first two experiments.

Saving these vectors to a file and loading them to be used whenever we want to create a model seems to a good approach to save time. Python libraries like pickle can be used to save the vectors to a file.

7.4.2 *Experiment with different label encoded values*

In this experiment, we are taking all the label encoded values together for which we got classification less than 0.7 to evaluate the classification Report. The previous label encoded values are given as below in Table 7.11. Since there are 9 values whose classification was less than 0.7 they would be encoded from 0 to 8.

Diagnosis Category	Previous value	New value
Cancer	3	0
Cardiac/Circulatory	4	1
Central Nervous System/ Neuromuscular	5	2
Chronic Pain	6	3
Morbid Obesity	17	4
OB-Gyn/ Pregnancy	19	5
Orthopedic/ Musculoskeletal	21	6
Prevention/Good Health	24	7
Respiratory System	25	8

Table 7.11: New label encoding values for diagnosis category

We will create another dataset with only the above selected diagnosis category values as shown below. We will then apply the same process of SMOTE analysis and Naive Bayes classification to get the classification Report.

Label	Precision	Recall	F1-score	Support
0	0.81	0.71	0.76	1015
1	0.88	0.82	0.85	1097
2	0.85	0.82	0.84	1048
3	0.81	0.81	0.81	1013
4	0.68	0.96	0.79	1012
5	0.81	0.60	0.69	1038
6	0.77	0.77	0.77	1051
7	0.63	0.70	0.66	1055
8	0.91	0.88	0.89	1038

Table 7.12: Naive Bayes Classifier(balanced data).

Precision	Recall	F1-score	Accuracy
0.79	0.79	0.79	0.78

Table 7.13: Classification Report(balanced data).

Label encoding is a common technique used to convert categorical data into numerical data. However it can result in artificial ordering and hence it is not the best option for nominal data[1]. For example it can consider the variable with label encoded value of 2 is average of variables with label encoded values of 1 and 3. We have proved this from the above experiment where label encoded values with 0 to 9 had better results than being encoded with different random values like 3 4 5 6 17 19 22 25. We have to use models like One Hot encoding or Decision Tree classification which works for nominal data to see if we can better classification score.

[1] B. Wohlwend, "Converting categorical data into numerical form: A practical guide for data science," *medium.com*, 2023

7.5 Decision Tree Classification with TF-IDF

In case of Decision tree classifier we will not label encode the Diagnosis category from 0 to 29, instead we will use hiearchial classification with the original column. We however have to convert the null values to 'Not Applicable' String before passing to the classifier. We will get the TF-IDF vector values from the three experiments performed before but will use Decision Tree Classifier to classify and results for three experiments are shown as in below Tables.

7.5.1 Experiment 1

Diagnosis Category	Precision	Recall	F1-score	Support
Alcohol and Drug Addiction	1.00	1.00	1.00	1037
Autism Spectrum	0.95	0.96	0.96	1015
Blood Related	0.97	0.97	0.97	1021
Cancer	0.89	0.83	0.86	1061
Cardiac/Circulatory	0.87	0.88	0.87	1020
Central Nervous System/ Neuromuscular	0.78	0.73	0.75	1038
Chronic Pain	0.93	0.99	0.96	1065
Dental	0.98	0.99	0.99	1104
Digestive System/ Gastrointestinal	0.88	0.85	0.86	1050
Ears, Nose, Throat	0.93	0.93	0.93	1058
Endocrine/ Metabolic	0.90	0.89	0.89	1023
Foot	0.95	0.98	0.97	1031
Genetic	0.98	0.99	0.99	1059
Genitourinary/ Kidney	0.96	0.95	0.95	1011
Immunologic	0.93	0.91	0.92	1036
Infectious	0.95	0.95	0.95	1074
Mental	0.88	0.86	0.87	1076
Morbid Obesity	0.90	0.94	0.92	1010
Not Applicable	0.96	0.97	0.97	1104
OB-Gyn/ Pregnancy	0.85	0.86	0.85	992
Organ Failure	1.00	1.00	1.00	1034
Orthopedic/ Musculoskeletal	0.71	0.66	0.69	1068
Pediatrics	0.92	0.93	0.92	1040
Post Surgical Complication	0.97	0.98	0.97	1002
Prevention/Good Health	0.92	0.95	0.93	1003
Respiratory System	0.91	0.93	0.92	1060
Skin	0.89	0.92	0.91	1039
Trauma/Injuries	0.97	0.97	0.97	969
Vision	0.97	0.98	0.98	1081

Table 7.14: Decision Tree Classifier(balanced data)(Experiment 1).

Precision	Recall	F1-score	Accuracy
0.92	0.92	0.92	0.92

Table 7.15: Classification Report with DT(Experiment 1).

7.5.2 *Experiment 2*

Diagnosis Category	Precision	Recall	F1-score	Support
Alcohol and Drug Addiction	1.00	1.00	1.00	1037
Autism Spectrum	0.84	0.96	0.90	1015
Blood Related	0.96	0.97	0.96	1021
Cancer	0.88	0.72	0.79	1061
Cardiac/Circulatory	0.86	0.87	0.87	1020
Central Nervous System/ Neuromuscular	0.81	0.75	0.78	1038
Chronic Pain	0.94	0.86	0.90	1065
Dental	0.97	0.99	0.98	1104
Digestive System/ Gastrointestinal	0.91	0.87	0.89	1050
Ears, Nose, Throat	0.95	0.90	0.93	1058
Endocrine/ Metabolic	0.90	0.87	0.89	1023
Foot	0.96	0.87	0.91	1031
Genetic	0.98	0.99	0.98	1059
Genitourinary/ Kidney	0.95	0.93	0.94	1011
Immunologic	0.92	0.91	0.91	1036
Infectious	0.93	0.95	0.94	1074
Mental	0.90	0.84	0.87	1076
Morbid Obesity	0.82	0.92	0.87	1010
Not Applicable	0.96	0.95	0.96	1104
OB-Gyn/ Pregnancy	0.76	0.73	0.74	992
Organ Failure	1.00	1.00	1.00	1034
Orthopedic/ Musculoskeletal	0.55	0.69	0.61	1068
Pediatrics	0.92	0.89	0.90	1040
Post Surgical Complication	0.89	0.93	0.91	1002
Prevention/Good Health	0.73	0.88	0.80	1003
Respiratory System	0.90	0.92	0.91	1060
Skin	0.90	0.85	0.88	1039
Trauma/Injuries	0.94	0.92	0.93	969
Vision	0.97	0.97	0.97	1081

Table 7.16: Decision Tree Classifier(balanced data)(Experiment 2).

Precision	Recall	F1-score	Accuracy
0.9	0.89	0.89	0.89

Table 7.17: Classification Report with DT(Experiment 2).

7.5.3 *Experiment 3*

Decision Category	Precision	Recall	F1-score	Support
Alcohol and Drug Addiction	1.00	1.00	1.00	1037
Autism Spectrum	0.92	0.94	0.93	1015
Blood Related	0.94	0.92	0.93	1021
Cancer	0.89	0.73	0.80	1061
Cardiac/Circulatory	0.85	0.82	0.84	1020
Central Nervous System/Neuromuscular	0.79	0.72	0.76	1038
Chronic Pain	0.87	0.93	0.90	1065
Dental	0.97	0.87	0.92	1104
Digestive System/ Gastrointestinal	0.86	0.80	0.83	1050
Ears, Nose, Throat	0.93	0.88	0.91	1058
Endocrine/ Metabolic	0.88	0.83	0.85	1023
Foot	0.94	0.91	0.92	1031
Genetic	0.97	0.94	0.96	1059
Genitourinary/ Kidney	0.93	0.88	0.91	1011
Immunologic	0.88	0.85	0.87	1036
Infectious	0.94	0.94	0.94	1074
Mental	0.90	0.79	0.84	1076
Morbid Obesity	0.81	0.88	0.84	1010
Not Applicable	0.91	0.83	0.87	1104
OB-Gyn/ Pregnancy	0.83	0.64	0.72	992
Organ Failure	1.00	1.00	1.00	1034
Orthopedic/ Musculoskeletal	0.73	0.61	0.66	1068
Pediatrics	0.92	0.88	0.90	1040
Post Surgical Complication	0.43	0.97	0.59	1002
Prevention/Good Health	0.72	0.84	0.77	1003
Respiratory System	0.91	0.87	0.89	1060
Skin	0.92	0.86	0.89	1039
Trauma/Injuries	0.92	0.90	0.91	969
Vision	0.98	0.95	0.96	108

Table 7.18: Decision Tree Classifier(balanced data)(Experiment 3).

Precision	Recall	F1-score	Accuracy
0.88	0.86	0.87	0.86

Table 7.19: Classification Report with DT(Experiment 3).

7.6 *Hyperparameters for Decision Tree Model*

Decision Trees are prone to over-fitting. A decision tree will always overfit the training data if we allow it to grow to its max depth. Overfitting in decision trees occurs when the tree becomes too complex and captures the noise in the training data, rather than the underlying pattern[2]. Let us check the difference between train and test accuracy score.

[2] R. Ravindran, "Overfitting and pruning in decision trees - improving model's accuracy," *medium.com*, 2023

Train accuracy	Test accuracy
0.99	0.65

Table 7.20: Train and test accuracy before SMOTE analysis

We don't see much difference between train and test data so it is not that much prone to overfitting. It is not candidate for hyperparameter tuning, However we would like to check the same without SMOTE analysis.

Train accuracy	Test accuracy
0.99	0.92

Table 7.21: Train and test accuracy after SMOTE analysis

In this case we see there is significant difference between test and train accuracy, so this suggests we have to prune the branches of tree model to get a more stable model. We would use GridSearchCV to get the optimum tree which is pruned to decrease the difference between test and train accuracy. Below are the parameters we will use for hypertuning the tree model using GridSearchCV [14].

- CRITERION: Decides the measure of the quality of a split based on criteria like "gini" for the Gini impurity and "entropy" for the information gain.
- MAX_DEPTH: The maximum depth of the tree, the more depth of tree generally it overfits the data.
- MIN_SAMPLES_SPLIT: The minimum number of samples that are required to split an internal node.
- MIN_SAMPLES_LEAF: The minimum number of samples that are required to be at a leaf node of the tree.

Parameter	Best value
Criterion	Gini
Max Depth	100
Min Samples Leaf	5
Min Samples Split	70

Table 7.22: Best Parameters after hyperparameters tuning

We had to manually tune to best parameters by repeatedly modifying the above parameters to get the optimum model.

Train accuracy	Test accuracy
0.77	0.68

Table 7.23: Train and Test Accuracy after hypertuning.

7.7 *Industrial Implementation.*

Extracting tokens from the medical data to classify the diagnosis category plays an important role in short term and long term findings. In case of urgency the tf-idf vectors which only take medical words from the document rather than all the words provide faster solution as they have less column vectors. The medical scispacy model used in experiment 2 can provide tokens for extraction of chemicals and diseases from medical text and will be useful for biomedical text mining. As we have used the above to get the Diagnosis category, we can use the same methodology to get the Treatment for the patient.

Similarly Fine tuning data with Deep learning BIOBERT LLM models can also provide text classification of medical findings without the use of tf-idf vectors. This however requires infrastructure with large amount of GPUs to provide high computational power to train the model.

8 Prototype

Our Prototype would be methodology as defined in Figure 8.1 for our experiments. Medical Findings is a text data and needs to be converted to numeric using TF-IDF(Term Frequency, Inverse Document Frequency) before we can send it to classifier model. The TF-IDF will use different tokenization methods based on the experiments performed. SMOTE analysis will be performed on the TF-IDF vectors to oversample minority classes for getting better classification scores. The Classifier Model used can be Naives Bayes or Decision Tree Classifier. After training the model we will check the Classification Report to test our hypothesis based on metrics generated.

In our scenario, since we are dependent on TF-IDF vectors which depends on the whole document, we have to train the model each time when new data is added. We can however change our prototype to bot if we ignore TF-IDF vector convertions and directly use the text to fine tune text classification with BIOBERT models.

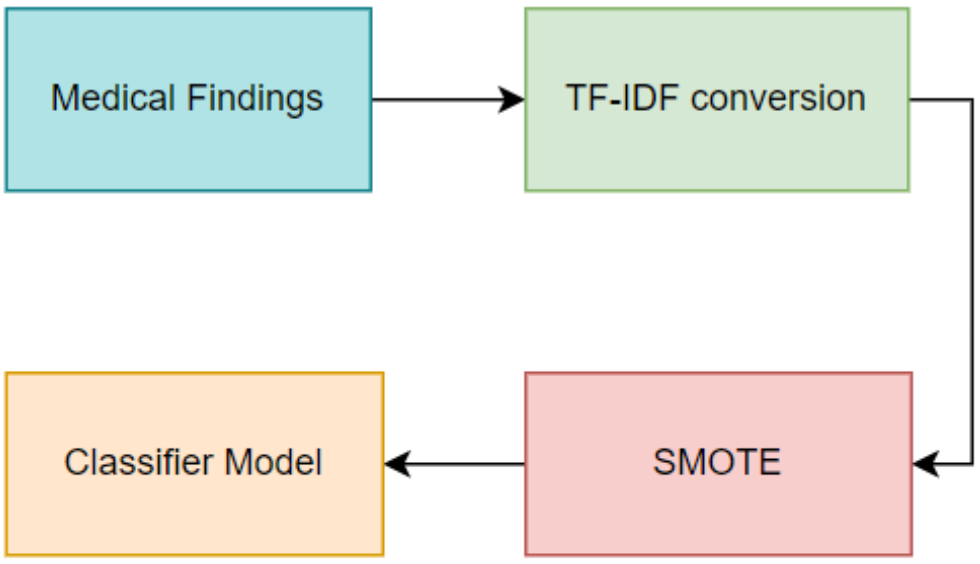

Figure 8.1: Prototype

9 Conclusions and Future work

9.1 Conclusions

- Label encoding of output parameters results in artificial ordering of nominal values and may result in poor classification scores. We can avoid that by using one hot encoding or using Decision Tree classification without Label encoding.
- Medical Scispacy models help us to use only text related to medical terminology instead of all the words in the text like spacy. This decreases the time consumed in creating the model however the time in creating tf-idf vectors is increased.
- The tokens generated from Biobert LLM Models are encoded before generating tf-idf vectors, the processing time is increased significantly for creating vectors from these encoding but it results in lesser output variables for final classification Model. The classification scores of both Medical Scispacy and BIOBERT LLM models are not far apart than the spacy models.
- The Decision Tree gives us the best classification score but it is subject to overfitting. We found that there was not much difference between test and train classification score with SMOTE analysis but without SMOTE analysis we see there is significant difference between test and train classification score which suggests overfitting.

9.2 Future Work

We would want to use LLM BIOBERT Deep Learning model to generate a fine tuned text classification model directly without converting it into TF-IDF vectors. This would require us large amout of GPU memory for training as it is very time consuming with normal CPU memory. It is not freely avaiable but we may have to use various cloud resources which offer them at a particular rate.

We will also look into eliminating the "Non Applicable" and "Preventive/Good Health" type of Diagnosis category from the dataset as they don't help us diagnosing a sick patient. Also we will be trying to predict other parameters like Treatment Category and type of medical findings using same methodology used in the experiments.

10 Appendix A

10.1 Language Bias

The HuggingFace model we used for creating tokenizations are trained only for english language. So this model would not be able to create proper tokens for other languages. There are however BERT models in foreign languages available and which can help us remove this bias. Another approach is use multilingual models like mBERT, XLM, and XLM Roberta. XLM Roberta. There is another workaround we can use is to translate the medical text from non-english language to english before they are trained.[1]

[1] nlpcloud.com, "multilingual-nlp-how-to-perform-nlp-in-non-english-languages.html," *nlpcloud.com*, 2022

10.2 Mental illness Bias

The medical scispacy model we used for creating medical tokens depends on finding diseases from the text. But there can be many other psychological diseases like stress and anxiety which it would not be able to detect. In this case the model may detect the person as healthy as it could not find releavant medical terms in the document.

10.3 Bias in pre-trained Models

Since the medical data has been trained without looking into other parameters like sex,ethinicity or region of the patient, they would not have any human biases. There can however be bias in already trained LLM model we used to generate tokens. This bias comes from the process of how they were trained and what dataset they used

An example to this is how Amazon used resume samples of job candidates from a 10-year period to train its recruitment models. This supervised downstream NLP application learned how to score candidates by computing the patterns in previous resume samples from Amazon and respective information regarding the success level of the job candidate. As a result, the trained model learned the historical trends associated with employment at Amazon by discovering linguistic patterns on resumes. Women were underrepresented in the training set collected from employees. Consequently, the resume screening model associated men and the linguistic signals on their resumes with successful employment at Amazon, whereas resumes of candidates which contained words associated with women were frequently discarded by the algorithm. The biased patterns learned by the model led to discrimination against female job candidates.[2]

[2] A. Caliskan, *AI and Bias*. 2021

11 Appendix B

11.1 Steps for installing python packages in Google Colab

Step 1: pip install spacy

Step 2: pip install –upgrade nltk

Step 3: pip install imblearn

Step 4: pip install ipython-autotime

Step 5: pip install https://s3-us-west-2.amazonaws.com/ai2-s2-scispacy/releases/v0.5.3/en_ner_bc5cdr_md-0.5.3.tar.gz

Step 6: pip install https://s3-us-west-2.amazonaws.com/ai2-s2-scispacy/releases/v0.5.3/en_ner_bionlp13cg_md-0.5.3.tar.gz

Step 7: pip install scispacy

Step 8: pip install seaborn

Step 9: pip install transformers

11.2 Useful python functions for creation of model

1. get_tf_idf_vectors
2. get_smote_data
3. eng_tokenizer
4. label_encoding
5. get_naive_bayes_classification_report
6. get_confusion_matrix
7. medical_tokenizer
8. biobert_tokenizer
9. get_decision_tree_classification_report
10. get_best_hyperparameters

12 Appendix C

12.1 Generating TF-IDF vectors

```
from sklearn.feature_extraction.text import TfidfVectorizer


def get_tf_idf_vectors(tokenizer, data):

    # Parameters
        # tokenizer: tokenizer used based on experiment(list)
        # data: medical findings data (array-like, dataframe, sparse matrix)

    # Returns
       # vectorized_findings: tf-idf vectors(array-like, dataframe, sparse matrix)

    tfidf = TfidfVectorizer(tokenizer = tokenizer)
    vectorized_findings = tfidf.fit_transform(data)

    return vectorized_findings
```

12.2 Loading and saving picked data

```
import pickle
def get_pickled_output(type, file):
    # Parameters:
       # type: 'load' or 'dump'
       # file: File path (string)
    # Returns:
       # pickle object

    if type == 'load':
        return pickle.load(open(file, "rb"))
    elif type == 'dump':
        return pickle.dump(file, open(file, "wb"))
```

12.3 Experiment 1 tokenizer

```
# Initial imports

import nltk
from nltk.stem import WordNetLemmatizer
import spacy
import string
```

```
from spacy.lang.en.stop_words import STOP_WORDS
from spacy.lang.en import English
import re

# Initial load

nlp = spacy.load("en_core_web_sm")
nltk.download("wordnet")
nltk.download("omw-1.4")


def eng_tokenizer(text):

    # Parameters
       # text: Medical text (string)

    # Returns
       # tokens: List of processed string from the medical text (list)


    text = text.replace('Nature of Statutory Criteria/Case Summary: ','').replace('An enrollee has requested','').replace('Findings:', '')
    text = text.lower()
    text = re.sub(" +", " ", text)
    text = re.sub("[^a-zA-Z\s\']", " ", text)

    punctuations = string.punctuation
    stop_words = spacy.lang.en.stop_words.STOP_WORDS
    tokens = nlp(text)

    from spacy.lang.en.stop_words import STOP_WORDS
    from spacy.lang.en import English
    # Lemmatizer
    tokens = [ word.lemma_.lower().strip() if word.lemma_ != "-PRON-" else word.lower_ for word in tokens ]
    # Removing stop words and punctuation
    tokens = [ word for word in tokens if word not in stop_words and word not in punctuations ]


    return tokens
```

12.4 *Experiment 2 tokenizer*

```
import string
from spacy.lang.en.stop_words import STOP_WORDS
from spacy.lang.en import English
import re

nlp_med = spacy.load('en_ner_bc5cdr_md')
nlp_organ = spacy.load('en_ner_bionlp13cg_md')

def medical_tokenizer(data):
    # Parameters
       # text: Medical text (string)

    # Returns
       # tokens: List of processed string from the medical text (list)

```

```
    text = text.replace('Nature of Statutory Criteria/Case Summary: ','').replace('An enrollee has requested','').replace('Findings:', '')
    text = text.lower()
    text = re.sub(" +", " ", text)
    text = re.sub("[^a-zA-Z\s\']", " ", text)

    punctuations = string.punctuation
    stop_words = spacy.lang.en.stop_words.STOP_WORDS
    tokens = nlp(text)

    from spacy.lang.en.stop_words import STOP_WORDS
    from spacy.lang.en import English
    # Lemmatizer
    tokens = [ word.lemma_.lower().strip() if word.lemma_ != "-PRON-" else word.lower_ for word in tokens ]
    # Removing stop words and punctuation
    tokens = [ word for word in tokens if word not in stop_words and word not in punctuations ]

    # Adding back tokens to add bc5cdr parser
    medical_string = ' '.join(tokens)
    med_tokens = nlp_med(medical_string)
    organ_tokens = nlp_organ(medical_string)

    # Applying medical model
    med_tokens = [str(w) for word in med_tokens.ents for w in word.text.split(' ') if word.label_ == 'DISEASE']
    organ_tokens = [str(w) for word in organ_tokens.ents for w in word.text.split(' ') if word.label_ in ['ORGAN', 'CANCER']]

    token_original = tokens
    if med_tokens == []:
        if organ_tokens == []:
            return token_original
        return organ_tokens

    return med_tokens
```

12.5 *Experiment 3 tokenizer*

```
from transformers import pipeline
pipeline = pipeline(model="sschet/biobert_diseases_ner")

def biobert_tokenizer(text, pipeline):

    # Parameters
       # text: Medical text (string)
       # pipeline: HuggingFace pipleine for token classification

    # Returns
       # tokens: List of processed string from the medical text (list)


    text = text.replace('Nature of Statutory Criteria/Case Summary: ','')\
    .replace('An enrollee has requested','').replace('Findings:', '')
    text = text.lower()


    #tokens = list(set([i['word'] for i in pipeline(text) if i['entity'] != 'o' ]))
    tokens = [i['word'] for i in pipeline(text)]
```

```
21
22      return tokens
```

12.6 Label Encoding

```
1  from sklearn.preprocessing import LabelEncoder
2
3  def label_encoding(data):
4      # Parameters:
5         # data: diagnosis category data (array-like, dataframe, sparse matrix)
6      # Returns:
7         # transform_data: Label encoded values (list(int))
8
9      le = LabelEncoder()
10     le.fit(data)
11     return  le.transform(data)
```

12.7 Generating Naive Bayes Classification Report

```
1  from sklearn.naive_bayes import MultinomialNB
2  from sklearn.metrics import classification_report
3  from sklearn.model_selection import train_test_split
4
5
6  def get_naive_bayes_classification_report(input_data, output_data):
7      # Parameters:
8         # input_data: tf-idf vectors after smote (array-like, dataframe, sparse matrix)
9         # output_data: label encoded values (list(int))
10     # Returns:
11        # classificaton_report: Metrics report of prediction (array-like, dataframe,
       sparse matrix))
12
13
14     X_train, X_test, y_train, y_test = train_test_split(input_data, output_data,
       test_size=0.3, random_state=42)
15     clf_nb = MultinomialNB()
16     clf_nb.fit(X_train, y_train)
17     y_pred = clf_nb.predict(X_test)
18     return classification_report(y_test, y_pred)
```

12.8 Generating Seaborn Confusion Matrix

```
1  import matplotlib.pyplot as plt
2  import seaborn as sns
3  from sklearn.metrics import confusion_matrix
4
5  cm = confusion_matrix(y_test, y_pred)
6
7  def get_confusion_matrix(cm, class_names, file):
8
9    # Parameters:
10
11       # cm: confusion matrix (array-like, dataframe, sparse matrix)
12       # class_names: names of the classes from label encoder. (list)
```

```
        # file: file where the image of confusion matrix is saved

    # Returns:
        # None

    # Plot confusion matrix in a beautiful manner
    fig = plt.figure(figsize=(16, 14))
    ax= plt.subplot()
    sns.heatmap(cm, annot=True, ax = ax, fmt = 'g'); #annot=True to annotate cells
    # labels, title and ticks
    ax.set_xlabel('Predicted', fontsize=20)
    ax.xaxis.set_label_position('bottom')
    plt.xticks(rotation=90)
    ax.xaxis.set_ticklabels(class_names, fontsize = 10)
    ax.xaxis.tick_bottom()

    ax.set_ylabel('True', fontsize=20)
    ax.yaxis.set_ticklabels(class_names, fontsize = 10)
    plt.yticks(rotation=0)

    plt.title('Refined Confusion Matrix', fontsize=20)
    plt.savefig(file, bbox_inches='tight')
```

12.9 Generating Decision Tree Classification Report

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report
from sklearn.model_selection import train_test_split


def get_decision_tree_classification_report(input_data, output_data):

    # Parameters:
        # input_data: tf-idf vectors after smote (array-like, dataframe, sparse matrix)
        # output_data: label encoded values (list(int))
    # Returns:
        # classificaton_report: Metrics report of prediction (array-like, dataframe, sparse matrix))


    X_train, X_test, y_train, y_test = train_test_split(input_data, output_data, test_size=0.3, random_state=42)
    decision_tree = DecisionTreeClassifier()
    decision_tree.fit(X_train, y_train)
    y_pred = decision_tree.predict(X_test)
    return classification_report(y_test, y_pred)
```

12.10 Generating Grid Search CV Best Hyperparameters

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import train_test_split


def get_best_hyperparameters(input_data, output_data, max_depth, min_samples_leaf, min_samples_split, creterion clf):

```

```
    # Parameters:
        # input_data: tf-idf vectors after smote (array-like, dataframe, sparse matrix)
        # output_data: label encoded values (list(int))
        # max_depth: Maximum depth of the tree
        # min_samples_leaf: Minimum number of samples required to be at a leaf node
        # min_samples_split: Minimum number of samples required to split an internal
    node
        # creterion: Criterion to measure the quality of a split
        # clf: DecisionTreeClassifier object
    # Returns:
        # best_params: Best hyperparameters (dict)
        # best_score: Best score (float)

X_train, X_test, y_train, y_test = train_test_split(input_data, output_data,
    test_size=0.2, random_state=42)
# Define the parameter grid
param_grid = {'criterion': criterion,
'max_depth': max_depth,
'min_samples_leaf': min_samples_leaf,
'min_samples_split': min_samples_split}


# Create an instance of the GridSearchCV
grid_search = GridSearchCV(clf, param_grid, cv=5)

# Fit the GridSearchCV to the data
grid_search.fit(X_train, y_train)

# Print the best set of hyperparameters
return (grid_search.best_params_, grid_search.best_score_)
```

Bibliography

[1] R. K. Gupta, "An introduction to tf-idf," *medium.com*, 2019.

[2] S. satpathy, "overcoming-class-imbalance-using-smote-techniques," *Analytics Vidya*, 2023.

[3] D. Nadeau and S. Sekine, *A survey of named entity recognition and classification*. Lingvisticae Investigationes, 2007.

[4] Özlem Uzuner, B. R. South, S. Shen, and S. L. DuVall2, *challenge on concepts, assertions, and relations in clinical text*. Journal of the American Medical Informatics Association,, 2011.

[5] G. K. Savova, Masanz, J. J., P. V. Ogren, J. Zheng, K.-S. Sohn, S., K. C., and C. G. Chute, *Mayo clinical Text Analysis and Knowledge Extraction System (cTAKES): architecture, component evaluation and applications*. Journal of the American Medical Informatics Association,, 2010.

[6] S.-H. Tsang, "Biobert: a pre-trained biomedical language representation model for biomedical text mining," *medium.com*, 2023.

[7] E. Kavlakoglu, "Classifying data using the multinomial naive bayes algorithm," *ibm.com*, 2024.

[8] G. H. dos Santos, "A brief introduction to decision trees," *Analytics Vidya*, 2021.

[9] E. Lázaro, J.-C. Yepez, P. Marín-Maicas, P. López-Masés, T. Gimeno, S. de Paúl, and V. Moscardó, "Computers in human behavior reports," *ScienceDirect*, 2023.

[10] A. Klein, H. Cai, D. Weissenbacher, L. Levine, and G. Gonzalez-Hernandez, "A natural language processing pipeline to advance the use of twitter data for digital epidemiology of adverse pregnancy outcomes," *Journal of Biomedical Informatics*, 2020.

[11] L. Guamán, L. Armando, A. Flores, and D. Omar, "Trabajo de titulación previo a la obtención del título de ingeniero en electrónica y telecomunicaciones," *Google Scholar*, 2017.

[12] J. Sancho, C. Fanjul, M. D. la Iglesia Vayá, J. Montell, and M. Escartí, "Aplicación de la inteligencia artificial con procesamiento del lenguaje natural para textos de investigación cualitativa en la relación médico-paciente con enfermedad mental mediante el uso de tecnologías móviles," *Revista de Comunicación y Salud*, 2020.

[13] V. Bhise, S. S. R. F. Sitting, R. O. Morgan, P. Chaudhary, and H. Singh, "Defining and measuring diagnostic uncertainty in medicine: A systematic review," *Journal of General Internal Machine*, 2017.

[14] S. Maiti, "extracting-medical-information-from-clinical-text-with-nlp," *Analytics Vidya*, 2023.

[15] G. Salton and C. Buckley, *Term-weighting approaches in automatic text retrieval.* Information processing and management, 1988.

[16] A. Fernández, S. García, F. Herrera, and N. V. Chawla, *SMOTE for learning from imbalanced data: progress and challenges, marking the 15-year 31 anniversary.* Journal of artificial intelligence research,, 2018.

[17] A. McCallum and K. Nigam, *A comparison of event models for naive Bayes text classification.* AAAI-98 workshop on learning for text categorization, 1998.

[18] N. Japkowicz and S. Stephen, *The class imbalance problem: A systematic study.* Intelligent data analysis, 2002.

[19] I. Rish, *An empirical study of the naive Bayes classifier.* IJCAI 2001 workshop on empirical methods in artificial intelligence, 2001.

[20] J. R. Zech, M. A. Badgeley, M. Liu, A. B. Costa, Titano, J. J., and E. K. Oermann, *Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: A cross-sectional study.* PLOS Medicine, 2018.

[21] M. Neumann, D. King, I. Beltagy, and W. Ammar, *ScispaCy: Fast and Robust Models for Biomedical Natural Language Processing*. arXiv preprint arXiv, 2019.

[22] J. Li, Y. Sun, R. J. Johnson, D. Sciaky, C. H. Wei, R. Leaman, and Z. Lu, *BioCreative V CDR task corpus: a resource for chemical disease relation extraction.* Database, 2016.

[23] S. Pyysalo, T. Ohta, R. Rak, D. Sullivan, C. Mao, C. Wang, , and S. Ananiadou, *Overview of the ID, EPI and REL tasks of BioNLP Shared Task 2011.* BMC bioinformatics, 2012.

[24] V. Sanh, "Smaller, faster, cheaper, lighter: Introducing distilbert, a distilled version of bert," *medium.com*, 2019.

[25] A. Zheng, *Evaluatiing Machine Learning Models*. Oreilly Media, 2015.

[26] B. Wohlwend, "Converting categorical data into numerical form: A practical guide for data science," *medium.com*, 2023.

[27] R. Ravindran, "Overfitting and pruning in decision trees - improving model's accuracy," *medium.com*, 2023.

[28] nlpcloud.com, "multilingual-nlp-how-to-perform-nlp-in-non-english-languages.html," *nlpcloud.com*, 2022.

[29] A. Caliskan, *AI and Bias*. 2021.