SYMPTOM-BASED DISEASE CLASSIFICATION USING TF-IDF AND KNN

A MINI PROJECT REPORT

Submitted by

LINDA CHRISTINA F 312322205091

PAVITHRA G R 312322205123

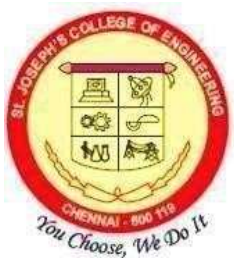
in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY



St. JOSEPH'S COLLEGE OF ENGINEERING

(An Autonomous Institution)

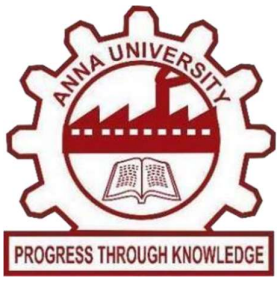
St. Joseph's Group of Institution

OMR, Chennai 600 119

ANNA UNIVERSITY: CHENNAI 600 025

April - 2025

ANNA UNIVERSITY: CHENNAI 600 025



BONAFIDE CERTIFICATE

Certified that this mini project report **"Symptom-Based Disease Classification Using TF-IDF and KNN"** is the bonafide work of **LINDA CHRSTINA F (312322205091)** and **PAVITHRA G R (312322205123)** who carried out the phase I project under my supervision, for the partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Information Technology.

SIGNATURE

SUPERVISOR

Dr. J. DIVYA, M.Tech., Ph.D.

Assistant Professor,

Department of Information Technology,

St. Joseph's College of Engineering,

OMR, Chennai- 600119.

SIGNATURE

HEAD OF THE DEPARTMENT

Dr. HELTIN GENITA C, M.E., Ph.D.,

Professor,

Department of Information Technology,

St. Joseph's College of Engineering,

OMR, Chennai- 600119.

CERTIFICATE OF EVALUATION

College name : St. Joseph's College of Engineering

Branch : Information Technology

Semester : VI

SL.NO	NAME OF THE STUDENTS	TITLE OF THE PROJECT	NAME OF THE SUPERVISOR
1. 2.	LINDA CHRISTINA F (312322205091) PAVITHRA G R (312322205123)	Symptom-Based Disease Classification Using TF-IDF and KNN	Dr. DIVYA J

The report of the Phase I project work submitted by the above students in partial fulfillment for the award of Bachelor of Technology Degree in Information Technology of Anna University were evaluated and confirmed to be reports of the work done by the above students.

Submitted for mini project and Viva Examination held on_____.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

At the outset we would like to express our sincere gratitude to the beloved **Chairman, Dr. Babu Manoharan, M.A., M.B.A., Ph.D.,** for his constant guidance and support.

We would like to express our heartfelt thanks to our respected **Managing Director,,Mr. B. Shashi Sekar, M.Sc.,** for her kind encouragement and blessings.

We wish to express our sincere thanks to our **Executive Director Mrs. S. Jessie Priya, M.Com.,** for providing ample facilities in the institution.

We express our deepest gratitude and thanks to our beloved **Principal Dr. Vaddi Seshagiri Rao, B.E., M.E., M.B.A., Ph.D., F.I.E.,** for his inspirational ideas during the project.

We are immensely grateful to our esteemed **Dean School of Computing, Mrs. G. Lathaselvi, B.E., M.E., (Ph.D.),** for her invaluable support and academic guidance throughout this endeavor.

We wish to express our sincere thanks and gratitude to **Dr.Heltin Genitha M.E., Ph.D., Head of the department,** Department of Information Technology, St. Joseph's College of Engineering for her guidance and assistance in solving the various intricacies involved in the project.

It is with deep sense of gratitude that we acknowledge our supervisor,
Dr. Divya J M.Tech , Ph.D. for her expert guidance and connoisseur suggestion.

Finally, we thank our department staff members who helped us in the successful completion of this project.

ABSTRACT

Timely and accurate disease identification based on patient-reported symptoms is critical for effective medical intervention, particularly in resource-limited settings. However, many existing diagnostic systems rely on structured data or manual feature engineering, which may not effectively capture the nuances of natural language symptom descriptions. This research proposes a machine learning-based framework that leverages text mining and classification techniques to map unstructured symptom inputs to their corresponding diseases. The system employs Term Frequency–Inverse Document Frequency (TF-IDF) for vectorizing symptom text, transforming it into meaningful numerical features. The K-Nearest Neighbors (KNN) algorithm is then applied to classify the input based on similarity to labeled symptom-disease datasets. To address challenges such as class imbalance, data sparsity, and semantic overlap between symptom descriptions, preprocessing steps including stop-word removal, stemming, and normalization are performed. Additionally, oversampling techniques are explored to enhance model generalization. The model's performance is evaluated using metrics such as accuracy, precision, recall, and F1-score to ensure reliability across diverse conditions. The proposed system demonstrates promising results in predicting diseases from free-text symptom inputs, offering a lightweight, explainable, and accessible decision-support tool for preliminary diagnosis and healthcare triage.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ABBREVIATION	DEFINITION
ML	Machine Learning
TF-IDF	Term Frequency–Inverse Document Frequency
NLP	Natural Language Processing
KNN	K-Nearest Neighbors

CHAPTER 1
INTRODUCTION

In modern healthcare, early and accurate diagnosis based on patient's descriptions of symptoms is vital to initiate timely treatment and improve patient outcomes. However, a major challenge in clinical diagnosis is that patients often describe their symptoms in free-form, unstructured text, which cannot be easily interpreted by traditional automated systems. Most existing diagnostic models are designed to work with structured data such as predefined symptom checklists, numeric test results, or categorical values. While structured data provides consistency, it fails to capture the subtle variations and nuances present in natural language symptom descriptions, leading to potential loss of information during the diagnostic process.

To bridge this gap, this project adopts a text classification approach that utilizes Natural Language Processing (NLP) and machine learning techniques to interpret unstructured symptom text and classify it into one of several disease categories. The two core techniques employed in this system are Term Frequency–Inverse Document Frequency (TF-IDF) and K-Nearest Neighbors (KNN).

TF-IDF is a statistical method that transforms raw text into meaningful numerical vectors. It works by evaluating how important a word is in a document relative to the entire dataset. Words that frequently occur in a specific document but are rare across the entire corpus are assigned higher weights. This helps the system focus on disease-relevant symptoms while downplaying generic or commonly occurring words that do not contribute significantly to classification.

Once the symptom text has been vectorized using TF-IDF, the K-Nearest Neighbors algorithm is applied for classification. KNN is a simple, non-parametric algorithm that classifies new input based on the majority label among its 'k' closest neighbors in the feature space. This makes the approach highly interpretable and effective, especially for datasets where the relationships between samples are more meaningful than the parameters of a learned model.

CHAPTER 2
LITERATURE SURVEY

In recent years, machine learning has played an increasingly prominent role in healthcare, particularly in automating diagnosis and symptom classification tasks. A wide range of algorithms and models have been explored for disease prediction based on patient input, especially where data is in textual form. Traditional healthcare diagnostic systems often rely on manual feature engineering, rule-based decision trees, or structured data input. These systems are limited by their rigid data formats and an inability to handle the variability and richness found in natural language symptom descriptions provided by patients.

To overcome these limitations, researchers have increasingly turned to Natural Language Processing (NLP) techniques to process and understand free-text input. One such method is TF-IDF (Term Frequency–Inverse Document Frequency), a statistical approach that assigns weights to words based on their frequency in a document relative to the entire dataset. This technique has proven effective in emphasizing disease-specific terms while downplaying commonly used non-informative words. As a result, TF-IDF has become a widely used tool for converting symptom descriptions into numerical feature vectors suitable for machine learning algorithms.

Alongside TF-IDF, other representation methods such as word embeddings (e.g., Word2Vec, GloVe) have also been used to capture semantic meaning and context in symptom narratives. These techniques map words into dense vectors based on their contextual similarity and co-occurrence, enabling models to understand relationships between symptoms that are not explicitly identical but have similar meanings.

Among the classifiers employed, K-Nearest Neighbors (KNN) stands out for its simplicity, transparency, and effectiveness in text classification tasks. KNN does not require a complex training phase and classifies new instances based on their proximity to labelled data in the feature space. When combined with a robust feature representation such as TF-IDF, KNN has shown high accuracy in predicting diseases from textual symptom data, making it an attractive option for low-resource implementations.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

Traditional diagnostic systems in healthcare often rely on structured data inputs, such as checkboxes, dropdown menus, or manually crafted features. These systems require users to conform to predefined symptom lists, which limits flexibility and fails to capture the real-world way in which patients describe their health issues. Such models are primarily rule-based or depend heavily on manual feature engineering, which can be time-consuming and may not generalize well across diverse patient populations or evolving language patterns.

Moreover, existing systems are typically built for specific conditions or contexts and lack adaptability. They do not support free-text input, which is essential for capturing the full depth and variability of patient symptom narratives. This rigid structure reduces diagnostic accuracy and user engagement, especially in environments where users may not be familiar with medical terminology.

These systems also suffer from scalability issues. When deployed in real-time applications—such as chatbots, telemedicine, or mobile diagnostic apps—they may fail to interpret non-standard language inputs or variations in phrasing. In resource-limited settings, where access to high-end computational infrastructure and expert-designed systems is minimal, traditional models often become unusable. Additionally, the interpretability of many black-box models can be limited, making them less suitable for use in critical domains like healthcare, where trust and transparency are essential.

3.2 PROPOSED SYSTEM

To address the limitations of traditional systems, the proposed system introduces a machine learning-based framework that uses TF-IDF vectorization and K-Nearest Neighbors (KNN) classification for disease prediction based on unstructured symptom descriptions. The goal is to provide a lightweight, interpretable, and effective method for mapping free-text symptom inputs to likely disease categories.

The system begins with text preprocessing, a crucial step that includes the removal of stopwords (commonly used but non-informative words), lemmatization (reducing words to their base form), punctuation removal, and conversion to lowercase. This ensures the input data is clean, consistent, and ready for vectorization.

Once preprocessed, the symptom descriptions are transformed into numerical vectors using the Term Frequency–Inverse Document Frequency (TF-IDF) technique. TF-IDF weighs words based on their importance in a document relative to a collection of documents, effectively capturing the relevance of each term in the context of the dataset.

The vectorized input is then classified using the K-Nearest Neighbors (KNN) algorithm. KNN is an instance-based learning method that classifies new samples by analyzing the 'k' most similar data points from the training set. It is particularly useful for its simplicity and effectiveness in text classification tasks, especially when combined with meaningful feature representations like TF-IDF.

To enhance the model's performance, Grid Search and Cross-Validation are applied for hyperparameter tuning, ensuring the most suitable value of 'k' is selected. This helps in avoiding overfitting and underfitting, improving the model's generalization capability on unseen data.

CHAPTER 4

SYSTEM DESIGN

4.1 ARCHITECTURE DIAGRAM

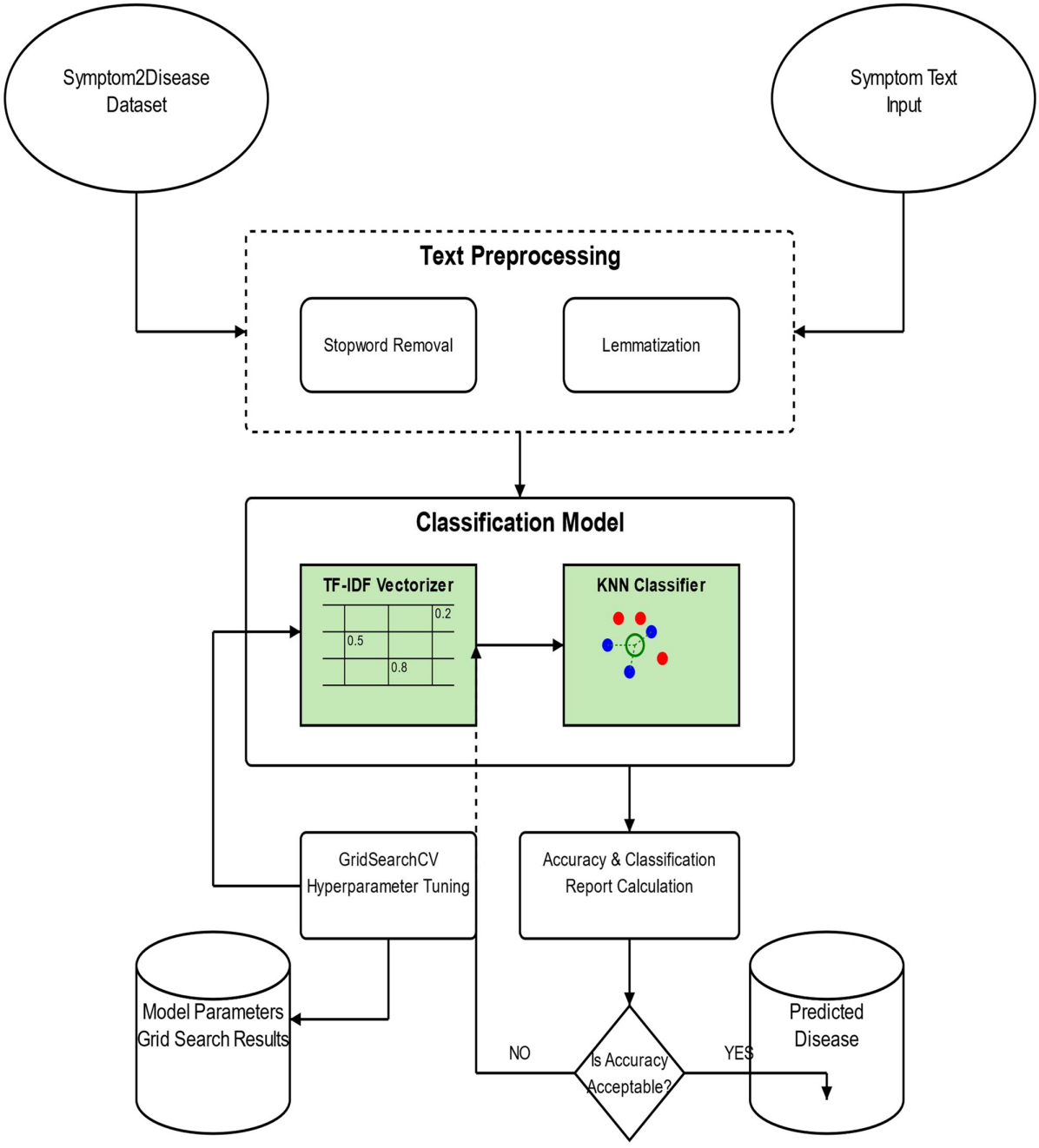


Figure 4.1 Architecture diagram

4.2 SYSTEM FLOW DIAGRAM

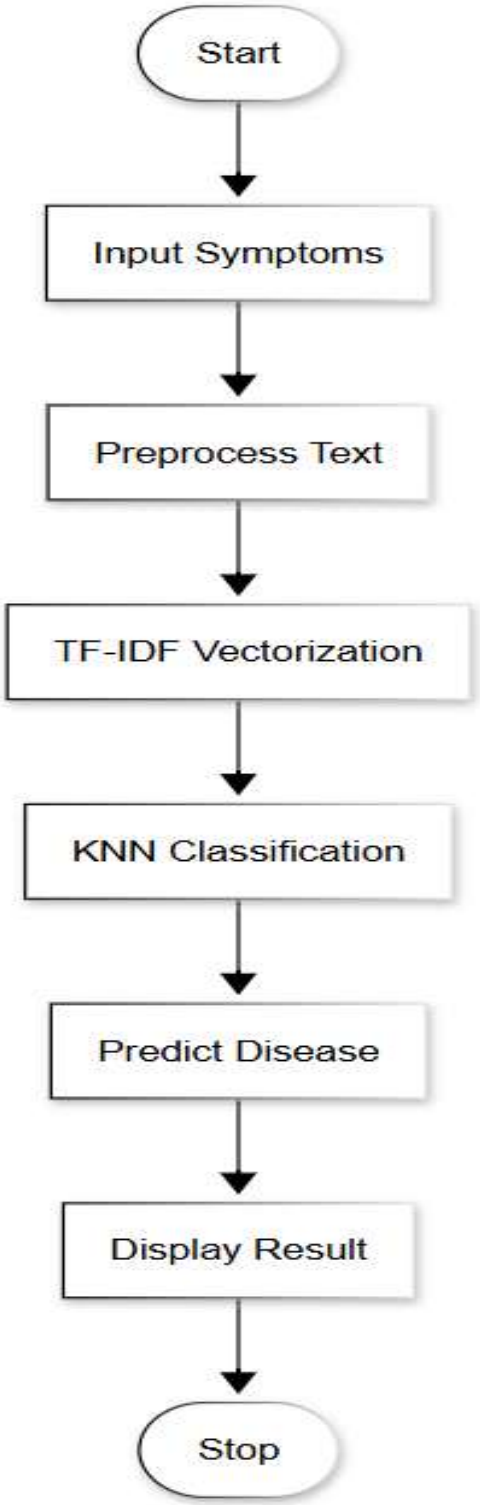


Fig: 4.2: System Flow Diagram

4.3 USE CASE DIAGRAM

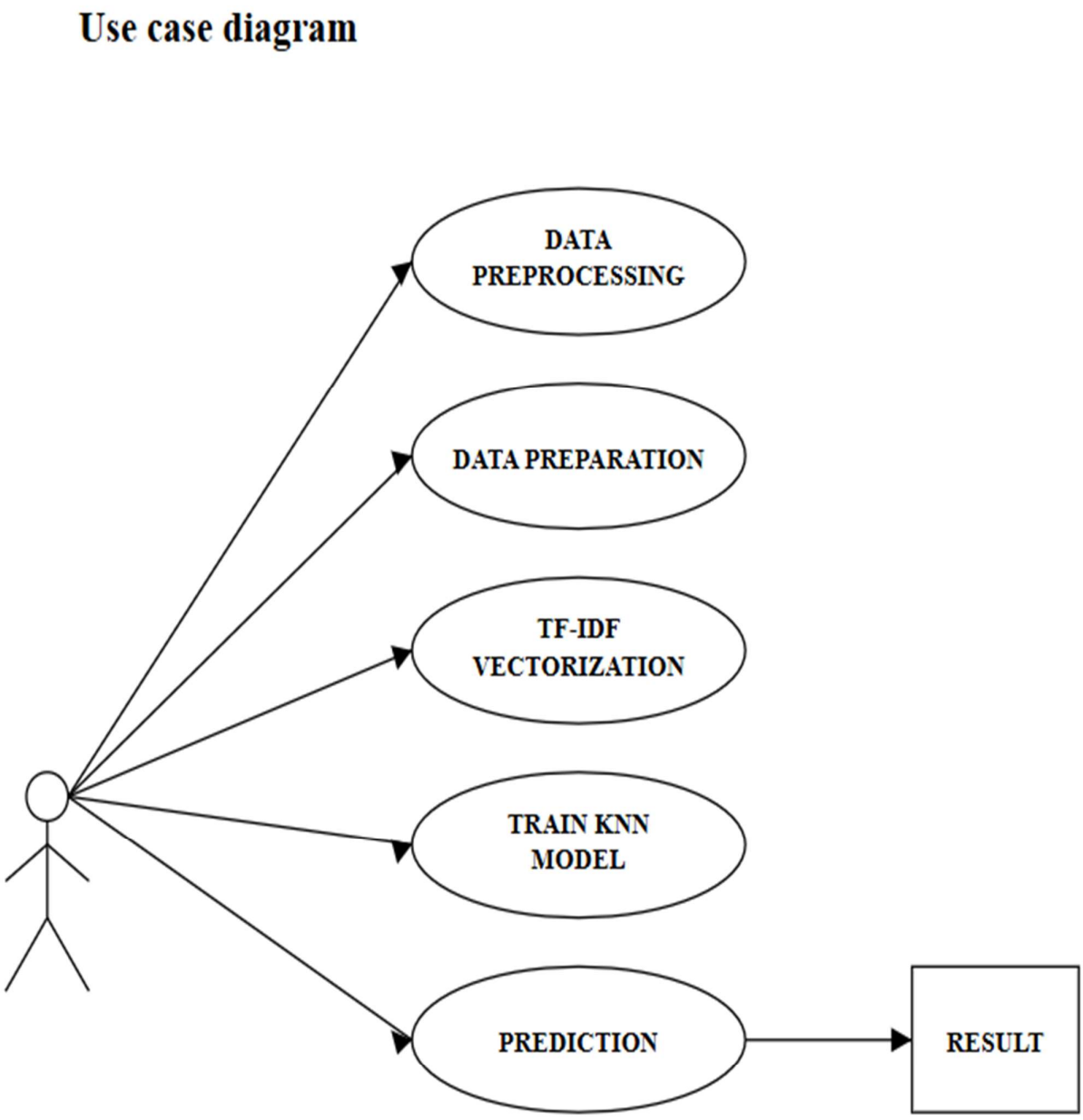


Figure 4.3 Use case diagram

4.4 SEQUENCE DIAGRAM

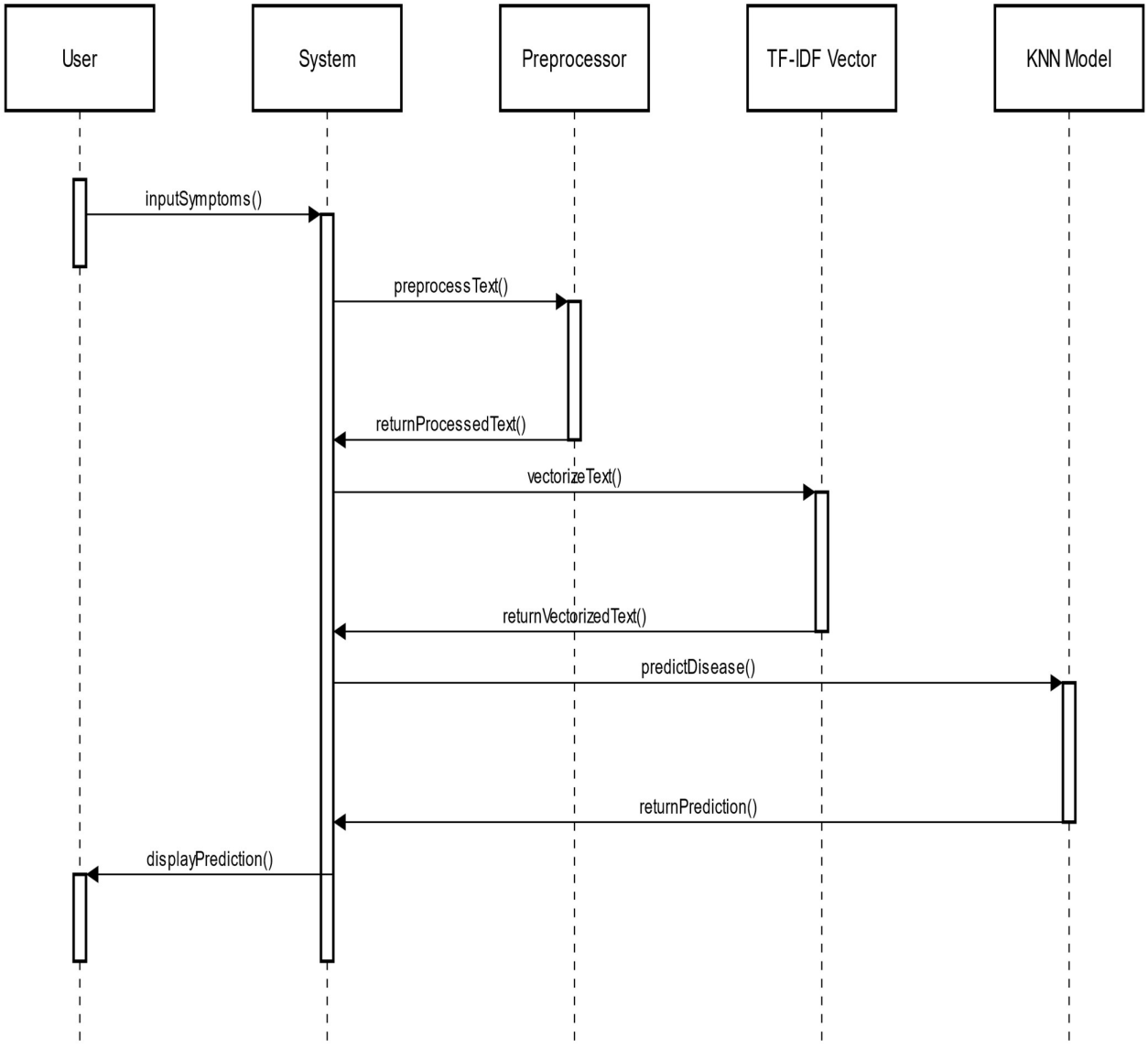


Figure 4.4 Sequence diagram

4.5 ACTIVITY DIAGRAM

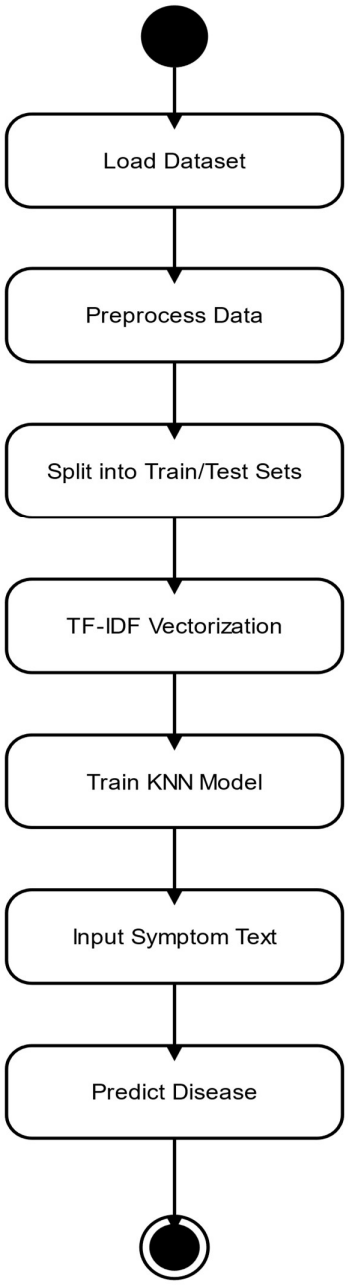


Figure 4.5 Activity diagram

CHAPTER 5

SYSTEM IMPLEMENTATION

5.1 MODEL DESCRIPTION

The model developed in this project aims to predict diseases based on natural language symptom descriptions using a combination of TF-IDF vectorization and the K-Nearest Neighbors (KNN) algorithm. The core strength of this system lies in its ability to process unstructured textual data—such as free-form patient symptom inputs—and transform it into structured numerical representations that a machine learning model can understand and classify effectively.

The dataset used for this project comprises 1200 records, stored in a CSV file format. Each record contains:

- **text**: a free-text symptom description written in natural language.
- **label**: the associated disease diagnosis for that symptom description.

The dataset includes 24 unique diseases, such as Migraine, Diabetes, Typhoid, Chickenpox, Pneumonia, Fungal Infection, Psoriasis, Dengue, and more. The goal of the model is to correctly predict the disease (label) when given a new, unseen symptom description (text).

5.2 METHODOLOGIES

5.2.1 PREPROCESSING AND FEATURE ENGINEERING

The pipeline begins with text preprocessing, which is crucial to ensuring consistent, clean, and informative inputs to the model. This involves:

- **Stopword Removal**: Commonly used but semantically insignificant words are removed.
- **Punctuation Removal**: Symbols and special characters are stripped from the text to reduce noise.
- **Lowercasing**: All characters are converted to lowercase to maintain uniformity.
- **Lemmatization**: Words are reduced to their base or root form to avoid duplication and preserve semantic meaning.

This preprocessing pipeline ensures that all symptom descriptions are represented in a standard format, improving the quality of feature extraction.

5.2.2 FEATURE EXTRACTION USING TF-IDF

After preprocessing, the cleaned text is transformed into numerical vectors using the Term Frequency–Inverse Document Frequency (TF-IDF) method. TF-IDF helps quantify how important a word is in a given symptom description relative to the entire dataset. This ensures that more relevant and disease-indicative terms are given higher importance, while generic terms receive lower weights.

The TF-IDF vectorizer converts each symptom description into a high-dimensional sparse vector, where each dimension represents a unique word from the corpus. These vectors are then used as input features for classification.

5.2.3 CLASSIFICATION USING K-NEAREST NEIGHBORS (KNN)

The classification task is handled by the K-Nearest Neighbors (KNN) algorithm. KNN is a distance-based, non-parametric method that classifies new instances based on the majority class among its 'k' closest neighbors in the training data. It is particularly well-suited for text classification when combined with meaningful feature representations like TF-IDF.

To optimize the model's performance, Grid Search is employed to determine the best value of k, and Cross-Validation is used to ensure the model generalizes well across different folds of the dataset.

MISCLASSIFICATION RATE CALCULATION:

The rate of misclassification for any classification that arises from a dataset can be found by observing confusion matrix metrics. Normally, the confusion matrix, which summarizes performances of the classifications done by a model, is used in this regard. Put differently, the misclassification rate, sometimes also called an error rate. Or, to be more precise, the misclassification rate is simply the sum of false negative and false positive which is divided by the total number of samples. The formula for the misclassification rate looks as follows:

$$MCC = \frac{TN \times TP - FN \times FP}{\sqrt{(FN \times TN)}\big((FP \times TN)(TN \times FN)(FP \times TP)\big)}$$

CHAPTER 6

CONCLUSION AND FUTURE SCOPE

This project demonstrates a simple yet effective machine learning approach for disease prediction using natural language symptom descriptions. By leveraging TF-IDF for feature extraction and K-Nearest Neighbors (KNN) for classification, the system successfully transforms unstructured symptom text into accurate disease predictions. Preprocessing techniques such as stopword removal, lemmatization, punctuation removal, and lowercasing were crucial in standardizing the input data and enhancing model performance.

The model was trained and evaluated on a dataset of 1200 entries spanning 24 diseases, achieving a high classification accuracy of 93.611%, with a corresponding misclassification rate of only 6.389%. Performance metrics such as precision, recall, and F1-score further validate the system's reliability and robustness. Grid Search and cross-validation ensured the model was optimally tuned for generalization.

In future enhancements, the system can be extended to support multi-label classification, which is crucial in real-world scenarios where a single symptom description may correspond to multiple possible diseases. This would increase the model's practical applicability in clinical settings. Additionally, replacing or augmenting the TF-IDF vectorization with more context-aware word embedding techniques such as Word2Vec, GloVe, or BERT could enable the model to capture deeper semantic relationships within symptom descriptions, thereby improving classification accuracy and robustness. Another important direction is the inclusion of larger and more diverse real-world datasets. Expanding the dataset with clinical records from varied sources would enhance the model's ability to generalize across different patient populations and symptom expressions, making it more reliable for deployment in real healthcare environments.

APPENDICES

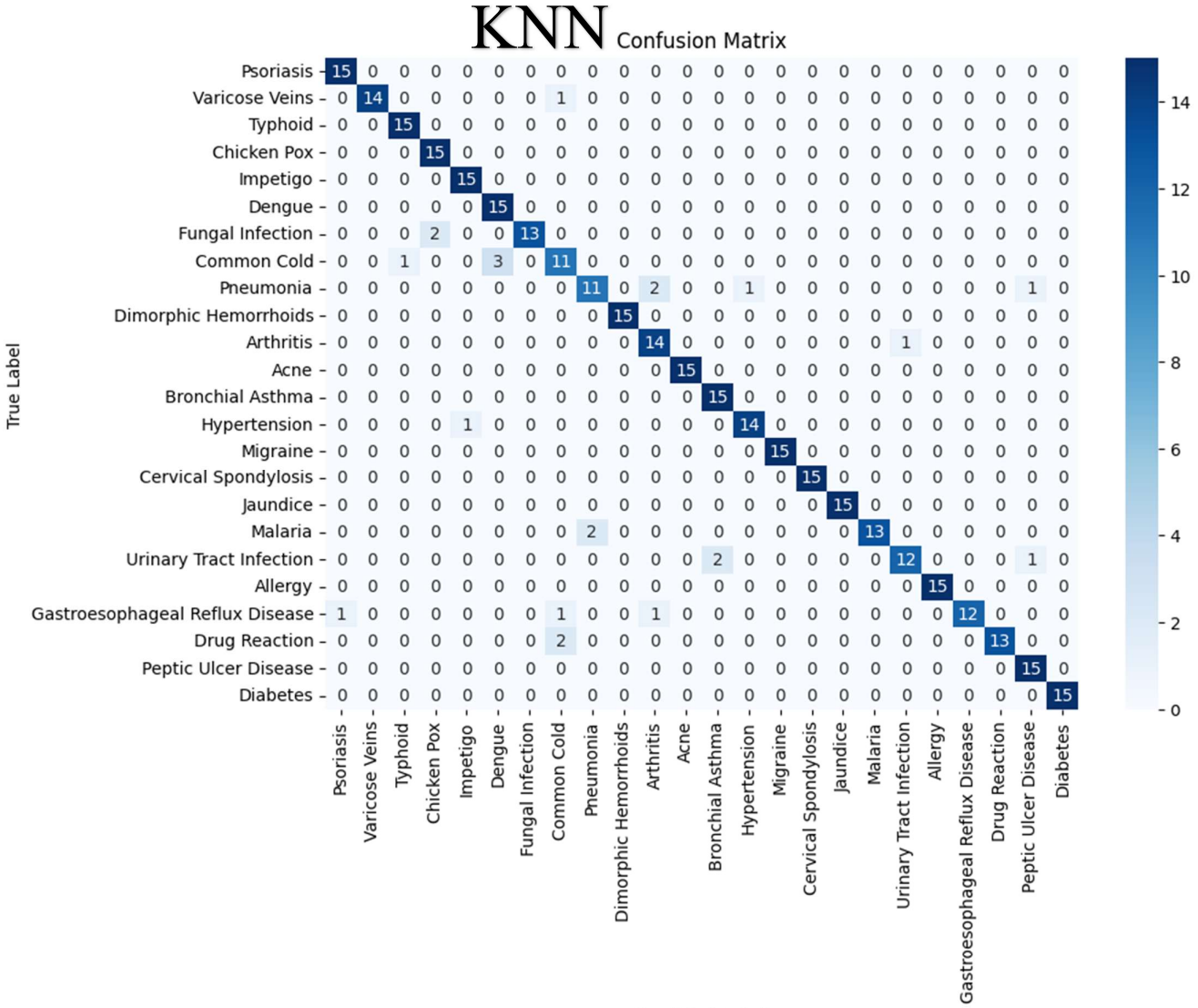
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
   for filename in filenames:
      print(os.path.join(dirname, filename))
symptom_df = pd.read_csv('/content/Symptom2Disease.csv')
symptom_df['label'] = symptom_df['label'].apply(str.title)
symptom_df.head(10)
symptom_df = symptom_df.drop('Unnamed: 0', axis=1)
symptom_df
# Exploratory Data Analysis
# Distribution Frequency of Diseases

disease_counts = symptom_df['label'].value_counts()

plt.figure(figsize=(12, 6))

sns.barplot(x=disease_counts.index, y=disease_counts.values,
palette='viridis')

plt.title('Distribution of Diseases', fontsize=16)
plt.xlabel('Disease', fontsize=12)
plt.ylabel('Number of Symptom Descriptions', fontsize=12)

plt.xticks(rotation=90)

plt.show()
# Word Cloud of Different Diseases

disease = 'Psoriasis'
disease_text = " ".join(symptom_df[symptom_df['label'] ==
disease]['text'])

print(f"Text for {disease}: {disease_text[:200]}...")  # Print first 200
characters

# Generate the word cloud
wordcloud = WordCloud(width=800, height=400,
background_color='white').generate(disease_text)

# Plot the word cloud
plt.figure(figsize=(10, 5))
plt.imshow(wordcloud, interpolation='bilinear')
```

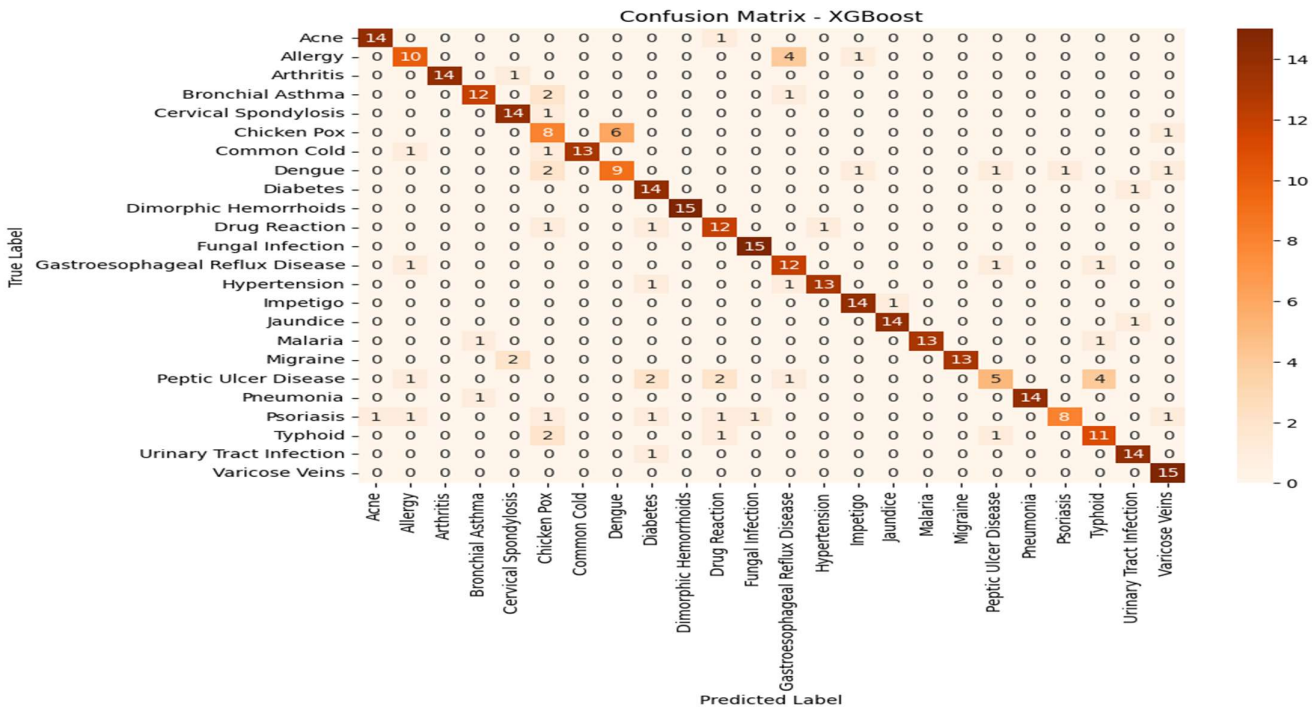
```
plt.axis('off')  # No axes for word cloud
plt.title(f'Word Cloud for {disease}', fontsize=16)
plt.show()
# Preprocessing
from nltk.stem import WordNetLemmatizer
from nltk.corpus import wordnet
from nltk.tokenize import word_tokenize
from nltk import pos_tag
from nltk.corpus import stopwords

from textblob import TextBlob
import nltk

!unzip /usr/share/nltk_data/corpora/wordnet.zip -d
/usr/share/nltk_data/corpora/
import string
import nltk

nltk.download('stopwords')
stop_words = set(stopwords.words('english'))
lemmatizer = WordNetLemmatizer()

# Function to get the WordNet POS tag from NLTK POS tag
def get_wordnet_pos(treebank_tag):
   if treebank_tag.startswith('J'):
      return wordnet.ADJ
   elif treebank_tag.startswith('V'):
      return wordnet.VERB
   elif treebank_tag.startswith('N'):
      return wordnet.NOUN
   elif treebank_tag.startswith('R'):
      return wordnet.ADV
   else:
      return wordnet.NOUN

# Function to lemmatize a single text
def lemmatize_text(text):

   tokens = word_tokenize(text)
   tagged_tokens = pos_tag(tokens)
   lemmatized_tokens = [lemmatizer.lemmatize(word, get_wordnet_pos(pos))
for word, pos in tagged_tokens]

   return " ".join(lemmatized_tokens)

# Function to remove common stopwords
def remove_stopwords(text):

   words = word_tokenize(text)
   filtered_words = [word for word in words if word.lower() not in
```

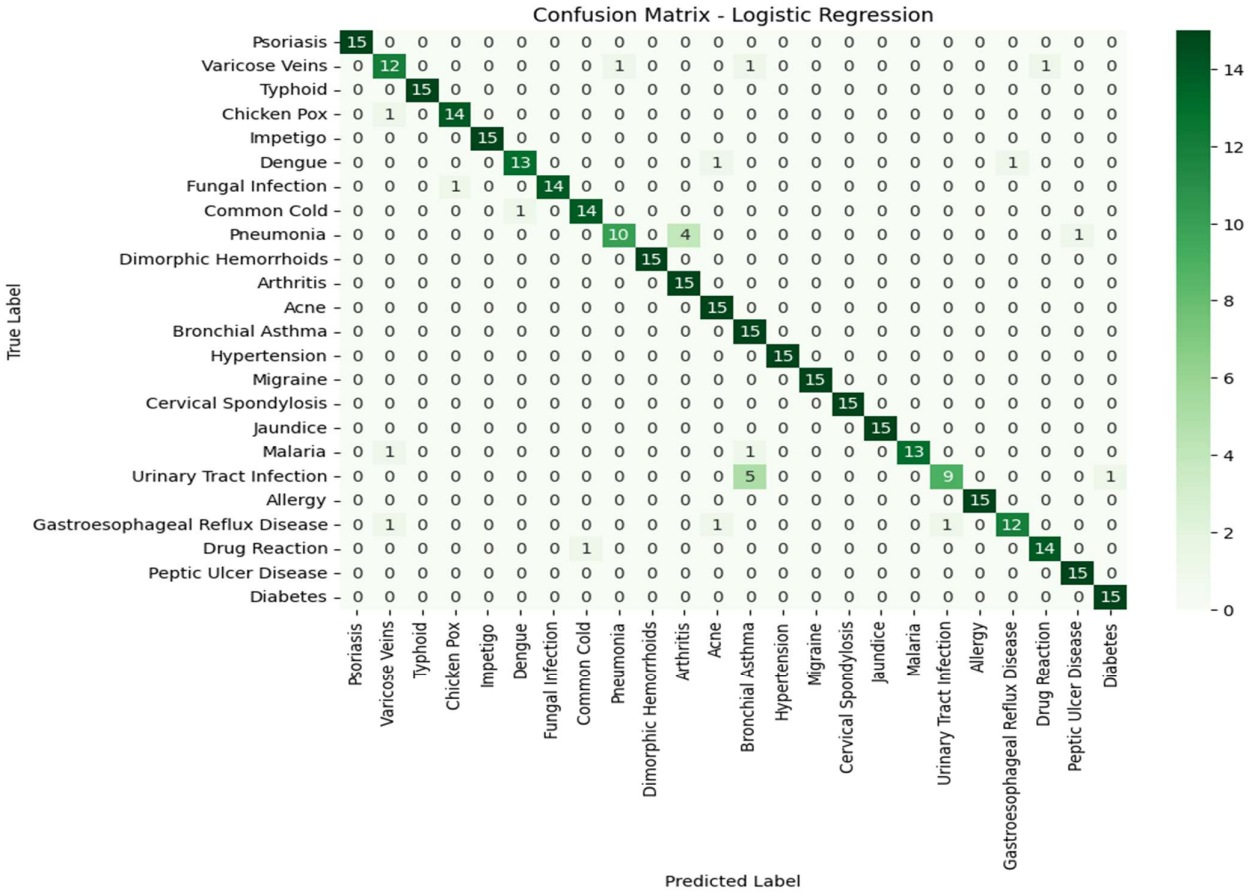
```
stop_words]

   return " ".join(filtered_words)

# Function to remove punctuations and make texts lowercase
def remove_punctuations_and_lowercase(text):
   # Convert text to lowercase
   text = text.lower()
   # Remove punctuation
   text = text.translate(str.maketrans('', '', string.punctuation))
   return text

# Function that encapsulates the other preprocessing techniques
def preprocessing(text):
   removed_stopwords = remove_stopwords(text)
   lemmatized = lemmatize_text(removed_stopwords)
   finalised_text = remove_punctuations_and_lowercase(lemmatized)

   return finalised_text
import nltk
nltk.download('punkt_tab')
nltk.download('averaged_perceptron_tagger_eng')
nltk.download('wordnet')
nltk.download('omw-1.4')

symptom_df_w2v = symptom_df.copy()
symptom_df_w2v['text'] = symptom_df['text'].apply(preprocessing)
symptom_df_w2v
# NLP Models Creation
#TF-IDF Vectorization with KNN Classification
from sklearn.neighbors import KNeighborsClassifier
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import train_test_split

tfidf_vectorizer = TfidfVectorizer(max_features=1500)

X_train_text, X_test_text, y_train, y_test =
train_test_split(symptom_df_w2v['text'],
                                              symptom_df['label'],
                                              stratify=symptom_df['label']
,
                                              test_size=0.3,
                                              random_state=42)

X_train = tfidf_vectorizer.fit_transform(X_train_text)
X_test = tfidf_vectorizer.transform(X_test_text)
# Cross Validation with Grid Search to Determine the Best Model
Parameters
```

```
param_grid = {
   'n_neighbors': [3, 5, 7, 10],  # Number of neighbors to test
   'weights': ['uniform', 'distance'],  # Weight function used in
prediction
   'algorithm': ['auto', 'ball_tree', 'kd_tree', 'brute']  # Algorithm
to compute the nearest neighbors
}

knn_classifier = KNeighborsClassifier()

grid_search = GridSearchCV(estimator=knn_classifier,
                   param_grid=param_grid,
                   scoring='accuracy',
                   cv=5,
                   verbose=4,
                   n_jobs=-1)

grid_search.fit(X_train, y_train)

best_params = grid_search.best_params_
best_score = grid_search.best_score_

print("Best Parameters:", best_params)
print("Best Score:", best_score)

best_knn_classifier = grid_search.best_estimator_
# Predict on test set
y_pred = best_knn_classifier.predict(X_test)
from sklearn.metrics import confusion_matrix, classification_report,
accuracy_score

cm = confusion_matrix(y_test, y_pred)

# Plot the confusion matrix using Seaborn
plt.figure(figsize=(10, 7))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
         xticklabels=symptom_df['label'].unique(),
         yticklabels=symptom_df['label'].unique())
plt.xlabel('Predicted Label')
plt.ylabel('True Label')
plt.title('Confusion Matrix')
plt.show()
# Generate classification report
report = classification_report(y_test, y_pred,
target_names=symptom_df['label'].unique())

# Print classification report
print("Classification Report:")
print(report)
```

```
# Calculate and print accuracy
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.5f}")
symptom = "I have a lot of trouble breathing, my chest feels heavy and
i cough mucus"
preprocessed_symptom = preprocessing(symptom)
symptom_tfidf = tfidf_vectorizer.transform([preprocessed_symptom])
predicted_disease = best_knn_classifier.predict(symptom_tfidf)
print(f'Predicted Disease: {predicted_disease[0]}')
```

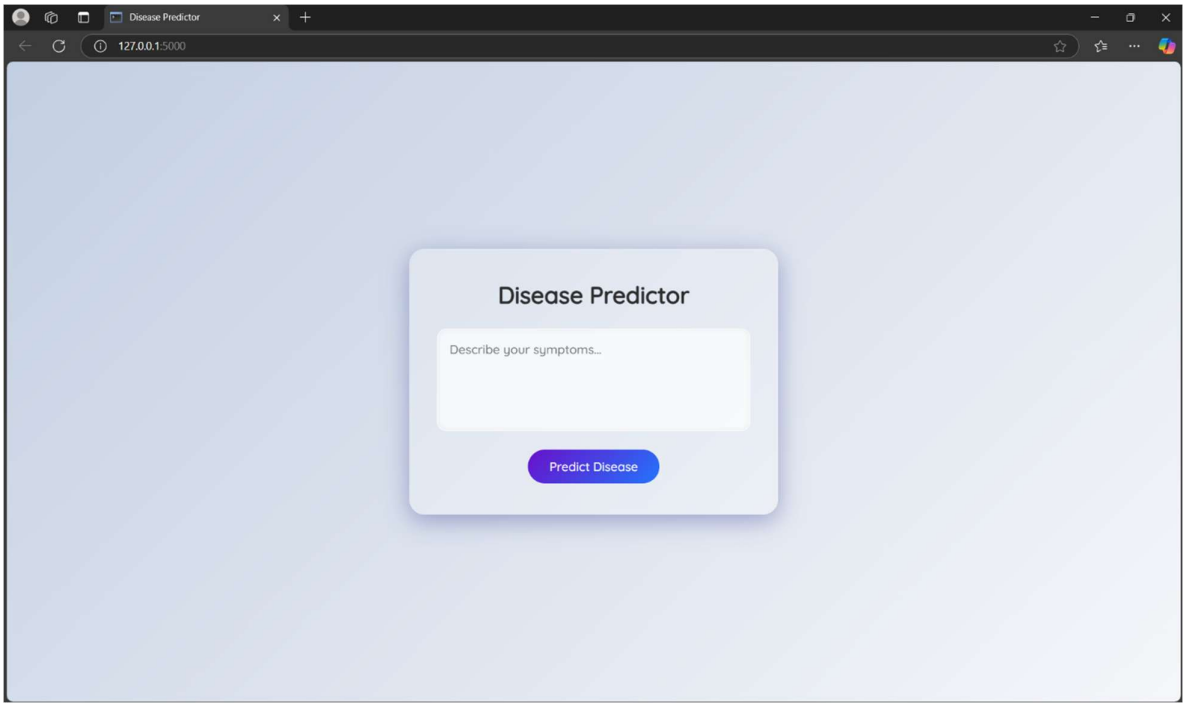

KNN Confusion Matrix
Psoriasis 15 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Varicose Veins 0 14 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Typhoid 0 0 15 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Chicken Pox 0 0 0 15 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Impetigo 0 0 0 0 15 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Dengue 0 0 0 0 0 15 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Fungal Infection 0 0 0 2 0 0 13 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Common Cold 0 0 1 0 0 3 0 11 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Pneumonia 0 0 0 0 0 0 0 0 11 0 2 0 0 1 0 0 0 0 0 0 0 0 0 1 0
Dimorphic Hemorrhoids 0 0 0 0 0 0 0 0 0 15 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Arthritis 0 0 0 0 0 0 0 0 0 0 14 0 0 0 0 0 0 0 1 0 0 0 0 0 0
Acne 0 0 0 0 0 0 0 0 0 0 0 15 0 0 0 0 0 0 0 0 0 0 0 0 0
Bronchial Asthma 0 0 0 0 0 0 0 0 0 0 0 0 15 0 0 0 0 0 0 0 0 0 0 0 0
Hypertension 0 0 0 0 1 0 0 0 0 0 0 0 0 14 0 0 0 0 0 0 0 0 0 0 0
Migraine 0 0 0 0 0 0 0 0 0 0 0 0 0 0 15 0 0 0 0 0 0 0 0 0 0
Cervical Spondylosis 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 15 0 0 0 0 0 0 0 0 0
Jaundice 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 15 0 0 0 0 0 0 0 0
Malaria 0 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 13 0 0 0 0 0 0 0
Urinary Tract Infection 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 0 12 0 0 0 1 0
Allergy 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 15 0 0 0 0 0
Gastroesophageal Reflux Disease 1 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 12 0 0 0 0
Drug Reaction 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 13 0 0 0
Peptic Ulcer Disease 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 15 0 0
Diabetes 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 15 0
Psoriasis Varicose Veins Typhoid Chicken Pox Impetigo Dengue Fungal Infection Common Cold Pneumonia Dimorphic Hemorrhoids Arthritis Acne Bronchial Asthma Hypertension Migraine Cervical Spondylosis Jaundice Malaria Urinary Tract Infection Allergy Gastroesophageal Reflux Disease Drug Reaction Peptic Ulcer Disease Diabetes
True Label
Predicted Label
0 2 4 6 8 10 12 14

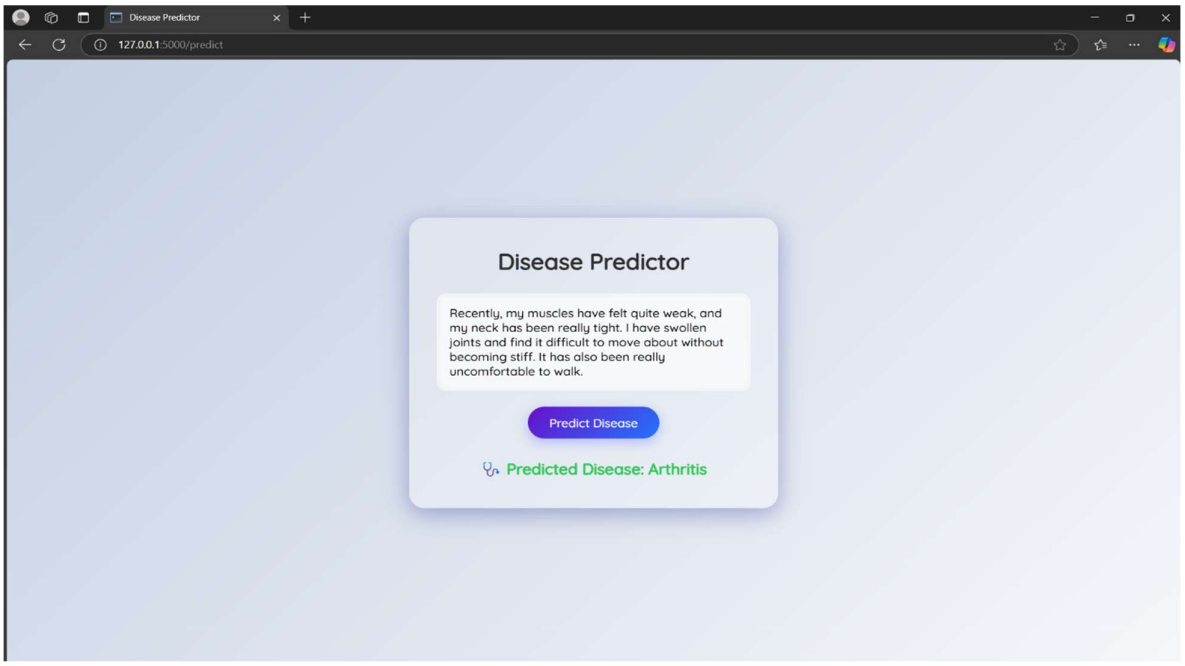

Confusion Matrix - XGBoost
Acne 14 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
Allergy 0 10 0 0 0 0 0 0 0 0 0 0 4 0 1 0 0 0 0 0 0 0 0 0
Arthritis 0 0 14 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Bronchial Asthma 0 0 0 12 0 2 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
Cervical Spondylosis 0 0 0 0 14 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Chicken Pox 0 0 0 0 0 8 0 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
Common Cold 0 1 0 0 0 1 13 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Dengue 0 0 0 0 0 2 0 9 0 0 0 0 0 0 1 0 0 0 1 0 1 0 0 1
Diabetes 0 0 0 0 0 0 0 0 14 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
Dimorphic Hemorrhoids 0 0 0 0 0 0 0 0 0 15 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Drug Reaction 0 0 0 0 0 1 0 0 1 0 12 0 0 1 0 0 0 0 0 0 0 0 0 0
Fungal Infection 0 0 0 0 0 0 0 0 0 0 0 15 0 0 0 0 0 0 0 0 0 0 0 0
Gastroesophageal Reflux Disease 0 1 0 0 0 0 0 0 0 0 0 0 12 0 0 0 0 0 1 0 0 1 0 0
Hypertension 0 0 0 0 0 0 0 0 1 0 0 0 1 13 0 0 0 0 0 0 0 0 0 0
Impetigo 0 0 0 0 0 0 0 0 0 0 0 0 0 0 14 1 0 0 0 0 0 0 0 0
Jaundice 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 14 0 0 0 0 0 0 1 0
Malaria 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 13 0 0 0 0 1 0 0
Migraine 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 13 0 0 0 0 0 0
Peptic Ulcer Disease 0 1 0 0 0 0 0 0 2 0 2 0 1 0 0 0 0 0 5 0 0 4 0 0
Pneumonia 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 14 0 0 0 0
Psoriasis 1 1 0 0 0 1 0 0 1 0 1 1 0 0 0 0 0 0 0 0 8 0 0 1
Typhoid 0 0 0 0 0 2 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 11 0 0
Urinary Tract Infection 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 14 0
Varicose Veins 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 15
Acne Allergy Arthritis Bronchial Asthma Cervical Spondylosis Chicken Pox Common Cold Dengue Diabetes Dimorphic Hemorrhoids Drug Reaction Fungal Infection Gastroesophageal Reflux Disease Hypertension Impetigo Jaundice Malaria Migraine Peptic Ulcer Disease Pneumonia Psoriasis Typhoid Urinary Tract Infection Varicose Veins
True Label
Predicted Label
0 2 4 6 8 10 12 14



APPENDIX 2

Approach	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Support (%)
KNN	93.61	93.67	94.05	94.3	93.6
Logistic Regression	93.06	94.00	93.00	93.00	93.00
XGBoost	82.22	83.00	82.00	82.00	82.00


Disease Predictor
127.0.0.1:5000
Disease Predictor
Describe your symptoms...
Predict Disease


Disease Predictor
127.0.0.1:5000/predict
Disease Predictor
Recently, my muscles have felt quite weak, and my neck has been really tight. I have swollen joints and find it difficult to move about without becoming stiff. It has also been really uncomfortable to walk.
Predict Disease
Predicted Disease: Arthritis

This project presents a machine learning-based system designed to predict diseases from natural language symptom descriptions using TF-IDF vectorization and K-Nearest Neighbors (KNN) classification. The model is trained on a dataset of 1200 records, each containing free-text symptom descriptions and corresponding disease labels across 24 distinct medical conditions.

To ensure clean and consistent input for vectorization, the text undergoes a thorough preprocessing pipeline, including stopword removal, lemmatization, punctuation removal, and lowercasing. The TF-IDF technique is used to convert the preprocessed text into numerical feature vectors that highlight term relevance. The KNN algorithm then classifies each symptom input based on the majority label among its nearest neighbors.
The model achieved a high accuracy of 93.611%, with performance validated using precision, recall, and F1-score metrics. Techniques such as Grid Search, cross-validation, and oversampling were used to fine-tune the model and handle class imbalance.
To ensure robustness, the system was also evaluated using two additional classifiers: Logistic Regression and XGBoost. Logistic Regression achieved an accuracy of 93.06%, while XGBoost lagged behind with 82.22%. The KNN model outperformed both alternatives across most evaluation metrics. The performance difference highlights the strength of instance-based learning methods like KNN when applied to high-dimensional, sparse representations like TF-IDF vectors. XGBoost, which typically excels on structured datasets, was less effective here—indicating its limitations when dealing with sparse textual features.

This comparative analysis confirms KNN as the most suitable model for this symptom-based disease classification task, owing to its simplicity, non-parametric nature, and strong alignment with TF-IDF features.
This project demonstrates a lightweight, interpretable, and efficient solution for symptom-based disease classification, offering potential applications in clinical triage, telemedicine, and low-resource healthcare environments. Future enhancements include supporting multi-label classification, integrating word embeddings such as Word2Vec or BERT to improve semantic understanding, and expanding the dataset with more diverse, real-world patient inputs to improve generalizability and accuracy in practical deployment.

REFERENCES

[1] Ramos, J. (2003). *Using TF-IDF to Determine Word Relevance in Document Queries*. Proceedings of the First Instructional Conference on Machine Learning.

[2] Cover, T., & Hart, P. (1967). *Nearest neighbor pattern classification*. IEEE Transactions on Information Theory, 13(1), 21–27.

[3] Gupta, D., & Khanna, A. (2017). *A Study of Applications of Machine Learning in Healthcare*. International Journal of Computer Applications, 162(8), 1–5.

[4] Liu, L., Tang, L., Dong, W., Yao, S., & Zhou, W. (2016). *An overview of topic modeling and its current applications in bioinformatics*. SpringerPlus, 5(1), 1608.

[5] Tiwari, R., & Pant, M. (2019). *Disease Prediction System Based on Symptoms Using Machine Learning*. In Proceedings of the 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (pp. 229–234). IEEE.

[6] Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013). *Distributed Representations of Words and Phrases and their Compositionality*. Advances in Neural Information Processing Systems (NeurIPS).

[7] Kaggle Dataset: Niyar Barman. (2022). *Symptom to Disease Dataset*. https://www.kaggle.com/datasets/niyarrbarman/symptom2disease