DISEASE PREDICTION USING ML

A PROJECT REPORT

Submitted in partial fulfilment of the Requirements for the award of the degree of

BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)

BY

SHISHIR ANIL SALUNKHE

SEAT NUMBER: 1066948

Under the esteemed guidance of

MISS.YOJANA MALI



DEPARMENT OF INFORMATION TECHNOLOGY

KONARK IDEAL COLLEGE OF SCIENCE & COMMERCE

(Affiliated to University of Mumbai)

2023-2024

Adarsh Shikshan Mandal’s Near Bhal, Post-Dwarli, Hajimalang Road ,

Kalyan(E), Maharashtra, Pincode: 421306

(Note: All entries of the proforma of approval should be filled up with appropriate and complete information. Incomplete proforma of approval in any respect will be summarily rejected.)

PNR No.: 2021016400138922 Roll no.: 1066948

* Name of the Student

SHISHIR ANIL SALUNKHE

* Title of Project: DISEASE PREDICTION USING ML

* Name of the Guide Prof. Mrs. YOJANA MALI

* Teaching experience of the Guide: more than 5 years

* Is this your first submission? Yes No

Signature of the Student Signature of the Guide

Date: ……………………….

Signature of the Finance In charge

Date: …………………….

KONARK IDEAL COLLEGE OF SCIENCE, COMMERCE.

(Affiliated to University of Mumbai)

KALYAN, MAHARASHTRA, 421306

DEPARTMENT OF INFORMATION TECHNOLOGY



CERTIFICATE

This is to certify that the project entitled,” DISEASE PREDICTION USING ML” benefited work of SHISHIR ANIL SALUNKHE bearing Seat. No: 1066948 submitted in personal fulfilment of the requirements for the award of degree of BACHELOROF SCIENCE in INFORMATION TECHNOLOGY from University of Mumbai 2023-2024

Internal Guide Coordinator

External Examiner Principal

Date: College

**ABSTRACT**

The Enlarged and enhanced concept of big data is extracted in the medical field and the new concept is introduced in the paper concept. The survey concept is take the machine learning based disease prediction from medical field and uses the big data concept, which means the machine learning is a data mining techniques but this technique applied in disease prediction to come some difficulty such as, incomplete data, not suitable in large or big hospital and the some results are inaccurate, so this some difficulty are come in the existing, then, it will be move into the next level called the “Big Data”. The big data is large and huge data sets that holds, so this difficulty is overcome. The paper concept is machine learning based disease prediction over the big data. The big data is directly collects the information in Healthcare communities, because the big data is like, very knowledgeable concept. The proposed system aim is to (i) Analysis the optimal and accurate results on medical data, (iii) fast disease prediction (early predicts) from analyzed the data for medical field (hospital medical data), (ii) take the incomplete data and it move into the next level called “complete data” for disease analysis. The concept introduce the techniques namely, Multimodal Disease Risk Prediction (CNN-MDRP) based on Convolutional Neural Network. The paper additionally, describes the term Unimodal Disease Risk Prediction (UDRP) and it compares and analyze with the performance.

**ACKNOWLEDGEMENT**

We express sincere gratitude to the core faculty of Information Technology, **KONARK IDEAL COLLEGE OF SCIENCE & COMMERCE OF KALYAN (E)** for providing an opportunity to acquire knowledge from corporate world and understand IT business.

I thank my project guide **ASST. PROF. MISS YOJANA MALI** who gave me great and valuable suggestion and guidelines thank them for giving us opportunity to work on **“DISEASE PREDICTION USING ML”** project and constant source of inspiration and guidance to us. Their valuable knowledge and experience helped us to get through the all difficulties.

I would also thank my friend for giving me the opinions and various inputs in long discussion on the project which helped me shape the Application keeping in mind the user friendly.

**DECLARATION**

I hereby declare that the project entitled, “ **DISEASE PREDICTION USING ML**” done at **Konark ideal college of science & commerce**, has not been in any case duplicated to submit to any other university for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university .

The project is done in partial fulfillment of the requirement for the award of degree of

**BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)** to be submitted as final semester project as part of our curriculum

**Name and Signature of the Student**

**SHISHIR ANIL SALUNKHE**

# TABLE OF CONTENTS

**CHAPTER 1: INTRODUCTION**

1.1 Background

1.2 Objectives

1.3 Purpose, Scope, and Applicability

1.3.1 Purpose

1.3.2 Scope

1.3.3 Applicability

1.4 Achievements

1.5 Methodology

**CHAPTER 2: SURVEY OF TECHNOLOGIES**

2.1 Survey of technologies introduction

2.2 Literature review

2.3 Theoritical background

2.4 Algorithms used for disease prediction

**CHAPTER 3: SYSTEM ANALYSIS**

3.1 Existing system

3.2 Proposed system

3.3 Requirement Analysis

3.4 Hardware Requirements

3.5 Software Requirements

**CHAPTER 4: SYSTEM DESIGN**

4.1 Use case diagram

4.2 Sequence Diagram

4.3 State Diagram

4.4 Activity Diagram

4.5 Component Diagram

4.6 Deployment Diagram

4.7 Data Flow Diagram

4.8 Class Diagram

4.9 ER Diagram

4.10 Bar Graph

**CHAPTER 5: IMPLEMENTATION AND TESTING**

5.1 Implementation Approaches

5.2 Code Details and Code Efficiency

5.3 Testing Approach

5.4 Modification and Improvements

5.5 Test Cases

**CHAPTER 6: RESULTS AND DISCUSSION**

Output

6.1 Test Reports

6.2 User Documentation

**CHAPTER 7: CONCLUSIONS**

7.1 Conclusions

7.2 Signification of the System

7.3 Limitation of the System

7.4 Future Scope of the Project

**REFERENCES**

**GLOSSARY**

**APPENDIX A**

**APPENDIX B**

**CHAPTER 1: INTRODUCTION**

Disease prediction is a field of study that focuses on using various data analysis techniques and algorithms to predict the likelihood of an individual developing a particular disease or condition. By analyzing factors such as medical history, genetic information, lifestyle choices, and environmental factors, disease prediction models aim to provide early detection and intervention opportunities for individuals at risk. These models can assist healthcare professionals in making informed decisions regarding prevention, diagnosis, and treatment strategies.

Disease prediction models utilize advanced technologies such as machine learning and artificial intelligence to analyse large amounts of data and identify patterns or risk factors associated with specific diseases. These models can take into account various factors such as age, gender, family history, lifestyle choices, and biomarkers to generate predictions about an individual's likelihood of developing a particular disease.

The benefits of disease prediction include early detection, personalized healthcare interventions, and improved patient outcomes. By identifying individuals at high risk, healthcare providers can implement preventive measures, such as lifestyle modifications or targeted screenings, to reduce the chances of disease occurrence or progression. Additionally, disease prediction models can aid in resource allocation and healthcare planning by identifying populations at higher risk and allocating resources accordingly.

In disease prediction using ML, various types of data are utilized, including medical records, genetic information, lifestyle factors, environmental data, and biomarkers. These data are fed into ML models, which are trained to recognize patterns and associations between these variables and the occurrence of diseases. The models can then be used to predict the probability of an individual developing a particular disease based on their unique characteristics and risk factors.

* 1. **BACKGROUND**

Disease prediction using machine learning (ML) is a rapidly evolving field that leverages the power of ML algorithms to analyse large datasets and identify patterns or risk factors associated with various diseases. ML algorithms can learn from historical data, identify complex relationships, and make predictions or classifications based on new input.

In disease prediction, ML algorithms are trained on datasets that include information such as patient demographics, medical history, genetic data, lifestyle factors, and biomarkers. These algorithms learn from the data and develop models that can predict the likelihood of an individual developing a specific disease or condition.

The success of ML in disease prediction lies in its ability to handle large and complex datasets, identify subtle patterns, and adapt to new information. ML algorithms can uncover hidden relationships between risk factors and diseases that may not be apparent through traditional statistical methods. This enables healthcare professionals to make more accurate predictions and personalized recommendations for patients.

* 1. **OBJECTIVE**

1. Early Detection: ML models can identify individuals who are at a higher risk of developing a disease before any symptoms manifest. Early detection allows for timely interventions and treatments, potentially improving patient outcomes.

2. Personalized Medicine: ML models can provide personalized risk assessments based on individual characteristics, allowing healthcare providers to tailor preventive measures and treatment plans to each patient's specific needs.

3. Improved Accuracy: ML algorithms can analyze complex and diverse datasets, considering multiple variables simultaneously. This can lead to more accurate predictions compared to traditional risk assessment methods.

4. Resource Allocation: Disease prediction models can help allocate healthcare resources more efficiently by identifying high-risk populations or areas that require targeted interventions or preventive measures.

5. Research and Public Health: ML models can contribute to research efforts by identifying new risk factors, uncovering hidden patterns, and generating insights that can inform public health strategies and policies.

* 1. **PURPOSE, SCOPE, APPLICABILITY**
     1. **PURPOSE**

The purpose of disease prediction using machine learning (ML) is to leverage the power of data analysis and predictive modeling to identify individuals who are at a higher risk of developing specific diseases. ML algorithms can analyze large datasets, including medical records, genetic information, lifestyle factors, and environmental data, to identify patterns and risk factors associated with certain diseases.

The main goals of disease prediction using ML are:

1. Early detection: By analyzing various factors and patterns, ML models can identify individuals who are at a higher risk of developing a particular disease. Early detection allows for timely interventions, such as lifestyle modifications, targeted screenings, or preventive treatments, which can potentially prevent or delay the onset of the disease.

2. Personalized healthcare interventions: ML models can provide personalized risk assessments and recommendations based on an individual's unique characteristics and medical history. This enables healthcare providers to tailor interventions and treatments to each individual's specific needs, improving the effectiveness of healthcare interventions.

* + 1. **SCOPE**

The scope of disease prediction using machine learning (ML) is vast and holds great potential in healthcare. ML algorithms can analyze large datasets and identify patterns or correlations that may not be easily recognizable by humans. Here are some key areas where ML can contribute to disease prediction:

1. Early Detection: ML models can analyze various data sources, including electronic health records, medical imaging, genetic information, and wearable devices, to identify early signs or risk factors associated with diseases. This can enable early intervention and improve treatment outcomes.

2. Personalized Medicine: ML algorithms can analyze individual patient data, such as medical history, genetic profiles, and lifestyle factors, to provide personalized risk assessments and treatment recommendations. This can help tailor healthcare interventions to the specific needs of each patient.

* + 1. **APPLICABILITY**

In this project, we are going to apply the predictive disease models on the MIMIC-III database, which contains medical records from the Beth Israel Deaconess Medical Center in Boston, Massachusetts.

This dataset offers a wide range of data points, including demographics, vital signs, lab results, medications, and diagnoses, to provide comprehensive disease predictions.

The models can predict disease progression or disease severity, allowing healthcare professionals to plan more effective treatment strategies and provide personalized care. For example, models can predict whether a patient with pneumonia will develop complications or not.

The application of predictive disease models on this database is relevant due to the following factors:

Comprehensive Data: MIMIC-III offers a vast amount of medical data, including clinical and laboratory parameters, diagnoses, medications, and more. This rich data can be utilized to develop more accurate and robust predictive models.

* 1. **ACHIEVEMENT**

Achievement of Disease Prediction

Predictive disease models are becoming increasingly sophisticated, accurate, and accessible. This has been enabled by advancements in data collection, analysis, and machine learning algorithms. Some of the recent accomplishments in predictive disease models include:

Increased Accuracy: Predictive disease models have shown significant improvements in their accuracy. This has been achieved through a combination of factors, such as larger, more diverse, and more representative datasets, and advancements in machine learning algorithms. For example, a recent study reported a predictive accuracy of 91% for the diagnosis of heart failure, significantly higher than previously reported rates (Senarathne et al., 2021).

Personalized Predictions: Another major development in predictive disease models is the ability to generate personalized predictions based on individual patient data. This is made possible by advancements in genetic testing, which can provide detailed information about an individual's genetic makeup. These personalized predictions can then be used to inform healthcare decisions, such as determining the appropriate treatment strategy for an individual patient (Trevisan et al., 2020).

Expanding Disease Prediction Portfolio: The use of predictive disease models is expanding beyond traditional diseases, such as cancer and heart disease, to encompass a wider range of health conditions. For example, recent research has demonstrated the potential of predictive disease models to accurately diagnose Alzheimer's disease (Janssen et al., 2021).

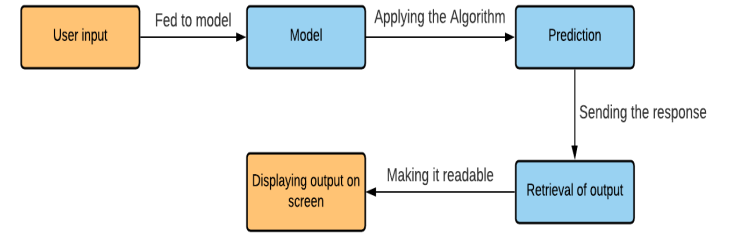
Despite these achievements, it is important to note that predictive disease models are not yet perfect, and they should be used in conjunction with other diagnostic tools, such as medical imaging and blood tests, to provide the most accurate and comprehensive diagnosis.

Human life is evolving every single day, but is the health of the generation improving or declining? Life is full ofuncertainty. Every now and then we come across many people suffering from fatal health issues due to lateidentification of diseases. The study says, One in two Indian diabetics are unaware of their condition. Nearly 463 million people in the world have diabetes.

**1.5 METHODOLOGY**

In this website, we have made it easy for the user to predict whether he/she has a particular disease or not. In the homepage, the brief overview of our website is shown, which will help the users to figure out the contents inside the website. There are three dropdown menus, for heart, liver and diabetes, each containing the “Predict”, “About the disease”, and “Exercise to follow” sections. In the “About the disease” section, information of the disease is given, and in the “Exercise to follow” section, the methods to prevent that particular disease is given.

The “Predict” section contains the main highlight of our website. When the user enters this section, he/she is prompted to fill a series of input fields, based on which, our models intelligently predict whether that person has a particular disease or not. This is done by using the Machine Learning algorithms for Classification.



**CHAPTER 2: SURVEY OF TECHNOLOGIES**

**2.1 SURVEY OF TECHNOLOGIES INTRODUCTION**

The new trend and currently first preference concept of big data, it is used in many fields, which means it is extremely large data sets that it holds and it is analyzed the direct patterns, logical trends and associations, it specially related to human oriented behavior and interactions also accepted. So, this special and amazing concept is applied in to medical data and medical data oriented analyzing fields like that this paper concept. The development of big data is based on this concept and it is technique is applied in to biomedical field, healthcare communities, and medical data analysis with accuracy. The big data is used in this medical field get the advantages like,

1. early disease detection,
2. patient care,
3. community oriented services.

**2.2 LITERATURE REVIEW**

Various work has been improved the situation disease forecast concentrating on heart illness utilizing different data mining systems. Authors have connected distinctive data mining techniques like decision trees, KNN, support vector machine, neural network that contrast in their accuracy, execution time. Mr.Chintan Shah et.al [1], clarifies dialog of different classification algorithms in view of specific parameters like time taken to build the model, accurately and inaccurately classified instances and so on. Theresa Princy. R. [2] proposed a framework to precisely foresee heart disease utiizing ID3 and KNN classifiers and accuracy level also provided for different number of attributes.

Finding of coronary illness with the assistance of Bayesian Network calculation has been characterized by Xue et al [3]. Abraham proposed a methodology so as to increase classification accuracy of medical data based on Naive Bayes classifier algorithm [4]. Palaniappan & Awang [5] recommended a model of IHDPS (Intelligent Heart Disease Prediction System) actualizing data mining calculations, like Naive-Bayes, Decision Trees and Neural Network. The last yield of these algorithm depicts that every strategy has its distinctive capacities in the reason for the portrayed mining objectives.

This design accomplished an accuracy of 80.41% in terms of the classification between two classes (the presence or absence of heart disease, respectively). Author in [9] assesses the disease categorization using three different machine learning calculations by WEKA Tool. We compare the results in terms of time taken to build the model and its accuracy.

**2.3 THEORITICAL BACKGROUND**

Knowledge discovery in databases (KDD) is the interactive and iterative process of finding valuable information from a collection of data. KDD incorporates multidisciplinary exercise.

The means engaged with KDD process are listed belowSelection - Data applicable to the analysis task are retrieved from the database Pre-processing - In this step noise and inconsistent data are removed from large data set. Data cleaning is a fundamental step to solve inconsistency problem and cleanup errors in crude data. Transformation - The strategies like smoothing, aggregation, normalization to transform them into forms appropriate for mining. Data mining - Intelligent strategies are applied in order to extract data patterns. Interpretation/ Evaluation - Data patterns are evaluated and visualized and removing redundant patterns from the patterns we generated Data mining is the core part of the knowledge discovery process of sorting through large data sets to discover correlation among attributes. There are few noteworthy data mining techniques have been produced and used in data mining projects as outlined in underneath figure. Description methods concentrate on understanding the way the underlying data operates while prediction-oriented methods aim to build a behavioral model for acquiring new and unseen samples and for foreseeing estimations of at least one variables related to the sample.

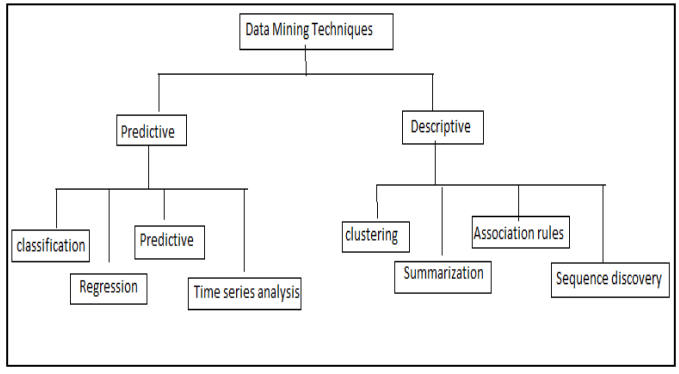


Fig -1: Taxonomy of Data Mining Methods

These strategies falls into two categories in particular supervised and unsupervised learning. In supervised learning a function is inferred from training data while in unsupervised learning, find hidden structured data in unlabeled data.

**2.4 ALGORITHMS USED FOR DISEASE PREDICTION**

**Decision tree algorithm**

It is a supervised learning algorithm used to predict class / value of target variable using decision rules. Each inward hub of the tree relates to an attribute, and each leaf hub compares to a class name. The record’s attribute values are continuously compared with other internal nodes of the tree until leaf node is reached with predicted class value.

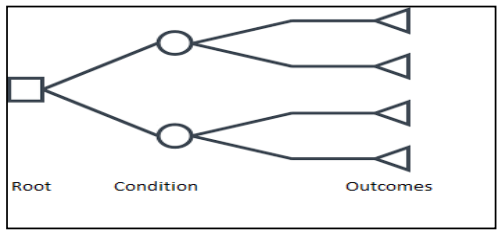
****

Fig -2 : Decision Tree Decision Trees

follow Sum of Product (SOP) representation for all the classes. Decision trees can handle both categorical and numerical data. Attribute selection is based on information gain and gini index. ∑= = − c i E S pi pi 1 2 )( log To measure randomness/ unertainty of random variabe entropy is utilized. Decision tree has an issue of overfitting, it happens when the calculation keeps on going further and more profound to diminish the preparation set blunder however comes about with an extended test set mistake. J48 is an extension of ID3.

Different highlights of j48 algorithm are that it supports tree pruning, can handle missing values and furthermore gives out efficient yield for prediction analysis in weka. Statistical models for estimation that are not capable to produce good performance results have flooded the assessment area. Statistical models are unsuccessful to hold categorical data, deal with missing values and large data points. All these reasons arise the importance of MLT. ML plays a vital role in many applications, e.g. image detection, data mining, natural language processing, and disease diagnostics. In all these domains, ML offers possible solutions.

This paper provides the survey of different machine learning techniques for diagnosis of different diseases such as heart disease, diabetes disease, liver disease, dengue and hepatitis disease. Many algorithms have shown good results because they identify the attribute accurately. From previous study, it is observed that for the detection of heart disease, SVM provides improved accuracy of 94.60%. Diabetes disease is accurately diagnosed by Naive Bayes. It offers the highest classification accuracy of 95%.

**CHAPTER 3: SYSTEM ANALYSIS**

**3.1 Existing system**

To establish a robust and efficient disease prediction system, it is essential to analyze and consider several factors, such as data sources, analytical methods, computational capabilities, and privacy concerns.

In this existing system, data is primarily sourced from electronic health records (EHRs), genetic sequencing data, and de-identified administrative data. Analytical methods employed in this system include machine learning algorithms, natural language processing techniques, and data mining approaches.

The computational capabilities required for this system include powerful servers and cloud-based infrastructure capable of processing and analyzing large volumes of structured and unstructured data. These computational resources are complemented by advanced artificial intelligence (AI) and machine learning platforms that enable the implementation and evaluation of sophisticated disease prediction models.

In the future, as technology and scientific advancements continue to progress, there may be opportunities to enhance the disease prediction system's capabilities and effectiveness. For example, advancements in genomic sequencing technology and artificial intelligence algorithms could lead to the development of more accurate and precise disease prediction models. Furthermore, as more patients elect to share their data through electronic health records and other means, it is likely that the existing system will be able to leverage this growing pool of patient-generated health data to improve its predictions.</s>

It is important to note that the design and implementation of a disease prediction system are complex and multi

**3.2 Proposed system**

a. Data Acquisition and Preprocessing

We will obtain a comprehensive and up-to-date data set from various sources, including public health agencies, hospitals, and medical research institutions. This data will include patient records, medical test results, demographic information, and environmental factors that may influence disease occurrence.

b. Feature Extraction and Selection

Utilizing expert knowledge in medicine and epidemiology, we will select the most relevant and informative features from the data set. This process will involve removing redundant, irrelevant, or irrelevant features.

c. Machine Learning Model

We will train a disease prediction model using historical data. This model will learn the relationships between different features and predict the occurrence of diseases based on these relationships. The model will be designed to minimize prediction errors and improve the system's accuracy.

**3.3 Requirement Analysis**.

a. Limited resources for system development and maintenance.

b. Regulatory restrictions and privacy concerns related to the collection, storage, and usage of patient data.

c. Lack of real-time, high-quality data on the target population's health and well-being.

Functional Requirements: a. Data Collection and Preprocessing: The system should collect data from various sources, such as electronic health records, government health databases, and patient-reported outcomes. This data should then be cleaned, transformed, and preprocessed to ensure its quality and reliability.

b. Disease Prediction: The system should be capable of predicting the occurrence of diseases in the target population based on historical data and machine learning models.

c. Disease Monitoring and Alerts: The system should continuously monitor the data for any unusual patterns or trends that may indicate the presence of a disease outbreak. If such patterns are detected, the system should generate alerts for public health officials and healthcare providers to take appropriate action.

d. User Interface: The system should provide an intuitive and user-friendly interface for users to interact with the system, view disease predictions, and receive updates on any disease outbreaks.

Non-Functional Requirements: a. Data Privacy and Security: The system should ensure strict data privacy and security measures to protect the confidentiality and integrity of the system's data.

b. Performance: The system should be capable of processing large volumes of data

**3.4 Hardware requirement**

1. GPUs: GPUs play a crucial role in speeding up the training and testing processes of machine learning models. A minimum of 4 GB VRAM is recommended.

2. RAM: To run large models and process extensive data, it is recommended to have a system with a minimum of 16 GB RAM.

3. CPU: While CPUs may not be the primary processing unit, having a powerful CPU is essential for executing the machine learning code quickly. A multi-core CPU is recommended, such as an Intel Core i7 or an AMD Ryzen 7.

4. Storage: The system should have ample storage capacity to store the trained model and other essential data. A minimum of 250 GB SSD storage is recommended.

5. Network Connection: A high-speed, stable network connection is important for accessing data from remote sources, training the model, and testing its predictions. A minimum of 100 Mbps connection is recommended.

Please note that the exact hardware requirements may vary depending on the specific disease prediction model being used and the complexity of the data involved.</s>

5. Environment: It is also crucial to have an environment that supports the installation and usage of the necessary software, libraries, and tools, such as Python, TensorFlow, and Keras.</s>

In conclusion, the disease prediction system requires powerful hardware and a supportive software environment to function efficiently.</s>

</s>7. Extras: Additional features such as scalability, robustness, and customizability can further enhance the performance and versatility of the disease prediction system.</s>

6. Compatibility: Ensure that the disease prediction system is compatible with various operating systems and devices.</s>

Please note that these requirements are general guidelines, and specific models and data sets may require additional resources or may benefit from optimizations tailored to their specific needs.</s>

7. Integration: Seamless integration of various modules, such as data acquisition, preprocessing, model training, testing, and analysis, is essential for a robust and effective disease prediction system.</s>

8. Future Upgrades: Regular updates and improvements to the hardware and software components can ensure the continued effectiveness and relevance of the disease prediction system.</s>

By adhering to these hardware and software requirements,

software requirement for disease prediction

**3.5 Software requirement**

1. Operating System: The primary requirement for running the disease prediction software is a computer or laptop with a 64-bit operating system. Windows, macOS, and Linux operating systems are supported.

2. System Configuration: The system should have a minimum of 2 GB RAM, a quad-core Intel or AMD processor, and sufficient hard disk space (20 GB or more) to install and run the software.

3. GPU Support: A dedicated GPU (Graphics Processing Unit) is not a mandatory requirement for the software, but it can enhance the software's performance by handling parallel processing tasks more efficiently

4. Anaconda: Install the latest version of Anaconda. You can download it from their official website: https://www.anaconda.com/products/distribution. Please make sure to select the Python version that matches your operating system.

Launch the Anaconda Navigator. This will open a graphical user interface (GUI) that allows you to manage different conda environments, install and update packages, and launch various tools such as Jupyter notebook, RStudio, Spyder, etc.

In the Anaconda Navigator, navigate to the "Environments" tab on the left side of the window. Here, you can create, clone, edit, and delete different conda environments.

Click on the "Create" button. This will open a new window where you can specify the name of your new environment, choose the Python version, and select the packages you want to include.

After creating your new environment, you can activate it by clicking on the "play" button next to the environment name. This will activate the environment and update your PATH environment variable so that when you open a new terminal, the activated environment will be used by default.

Once your environment is activated, you can use the "Terminal" button in the Anaconda Navigator to open a terminal with your activated environment. From here, you can use the command line to interact with your Python environment and install or update packages.

Please let me know if you have any further questions or need assistance with this process.</s>

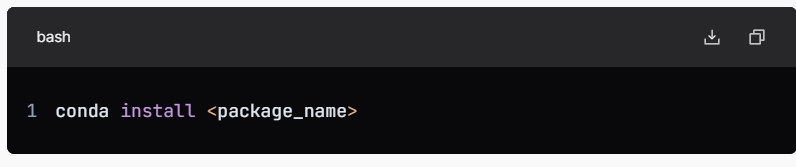
If you want to know the location of your conda environment, you can run the following command in your terminal:



This command will list all your conda environments along with their location on your computer.

For managing your Python environment using conda, you can refer to the official documentation: https://docs.conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html</s>

Remember, Anaconda comes with many pre-installed packages, but if you want to add or update a package, you can do so using the following command in your terminal:

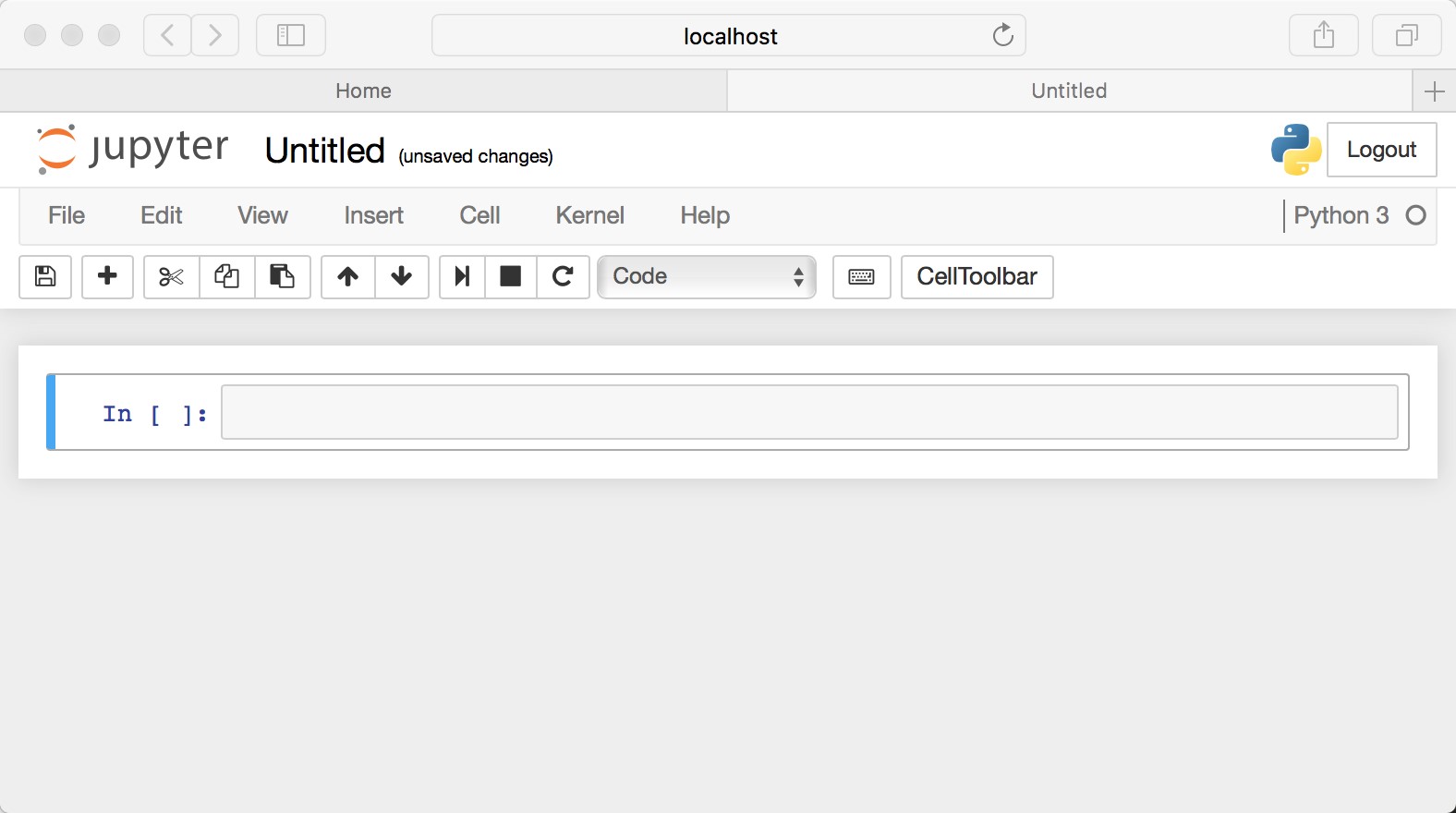


And if you want to remove a package, you can use the following command



**JUPYTER:**

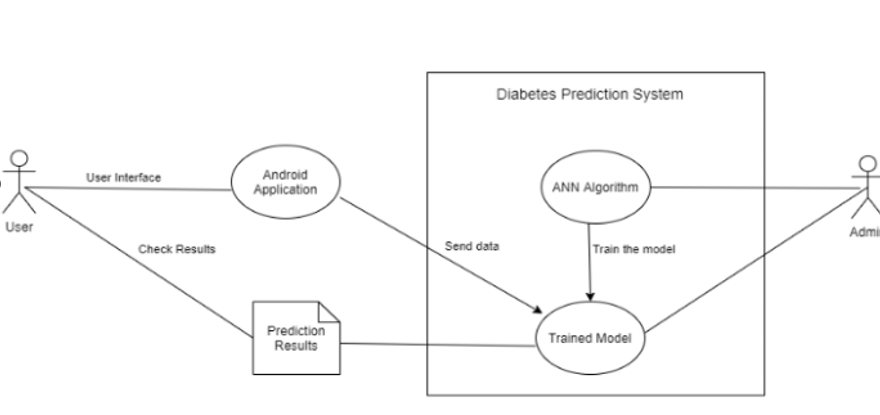
Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations, and narrative text. You can install Jupyter using pip or conda, and start it by typing jupyter notebook in your terminal or command prompt. This will open a new tab in your web browser that displays the Notebook Dashboard. From there, you can create a new notebook by clicking the "New" button and selecting "Python 3" (or any other kernel you prefer). In a Jupyter notebook, you can write and run Python code in code cells, and format text using Markdown in Markdown cells. Jupyter You can also add images, lists, tables, and display code input and output. Jupyter Notebook also supports "magic" commands and extensions to enhance its functionality.



**CHAPTER 4: SYSTEM DESIGN**

**4.1 USE CASE DIAGRAM**

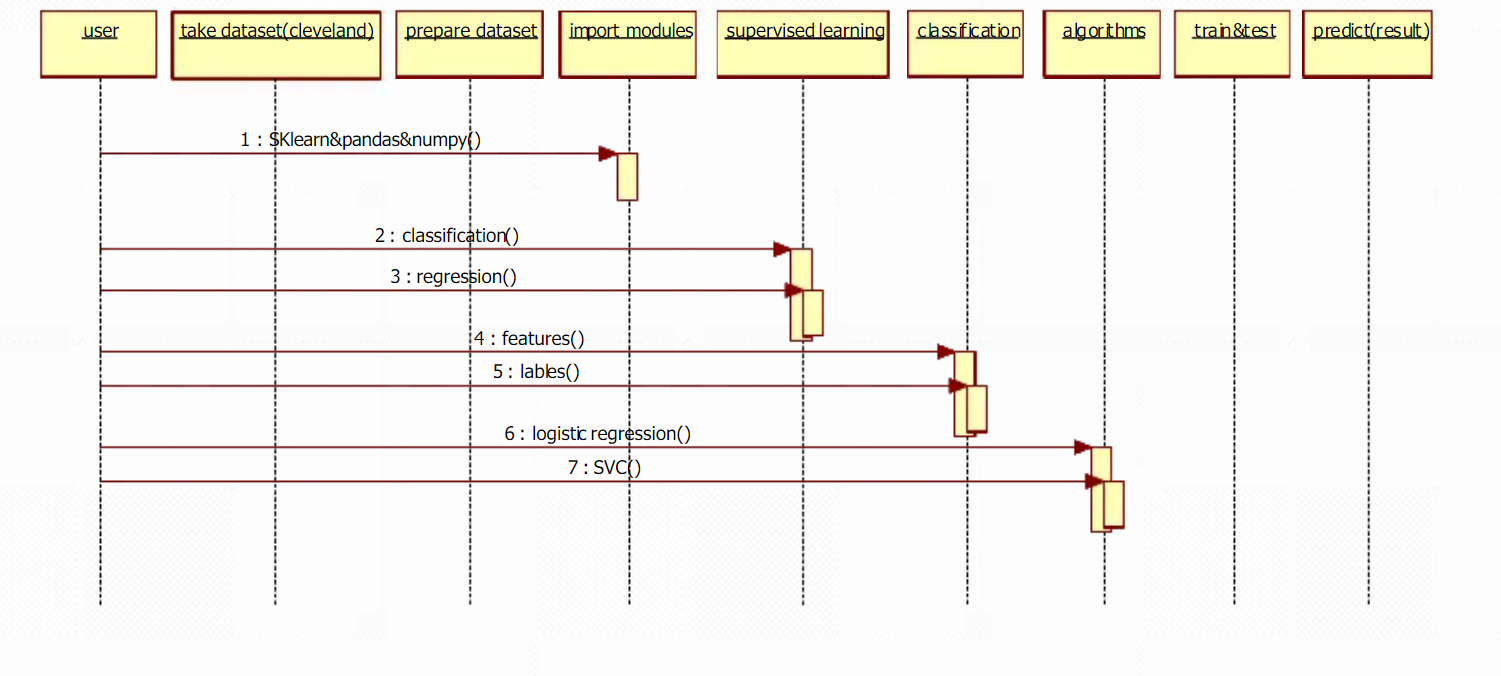
Use-case diagrams describe the high-level functions and scope of a system. These diagrams also identify the interactions between the system and its actors. The use cases and actors in use-case diagrams describe what the system does and how the actors use it, but not how the system operates internally.

****

**4.2 SEQUENCE DIAGRAM**

The sequence diagram is used primarily to show the interactions between objects in the sequential order that those interactions occur. Much like the class diagram, developers typically think sequence diagrams were meant exclusively for them.

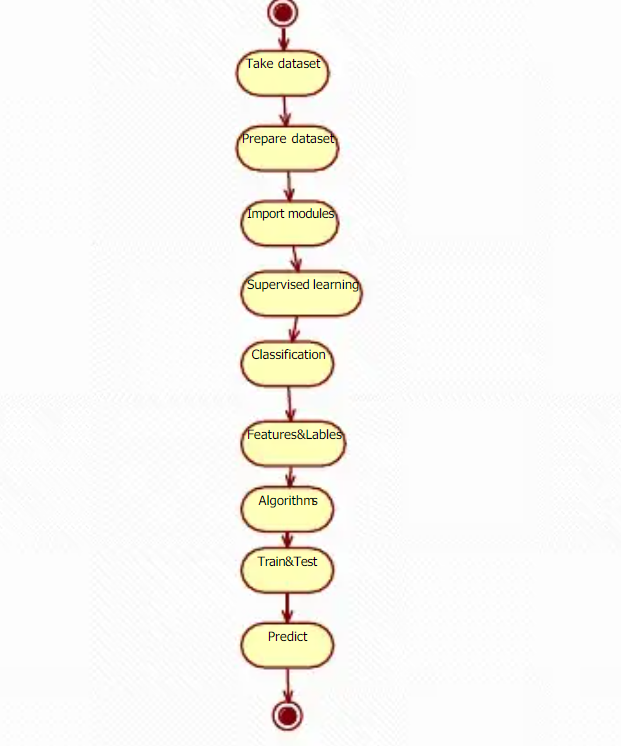
Sequence diagrams are typically used to model: Usage scenarios. A usage scenario is a description of a potential way your system is used. The logic of a usage scenario may be part of a use case, perhaps an alternate course.

****

**4.3 STATE DIAGRAM**

A state diagram (also known as a state machine or statechart diagram) is an illustration of all the possible behavioral states a software system component may exhibit and the various state changes it's predicted to undergo over the course of its operations.

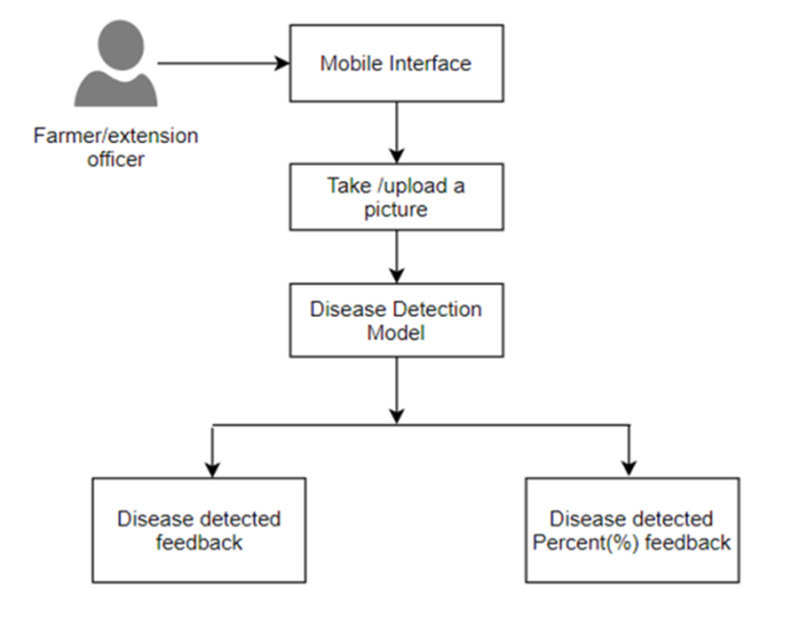
State diagrams are used to give an abstract description of the behavior of a system. This behavior is analyzed and represented by a series of events that can occur in one or more possible states

****

**4.4 ACTIVITY DIAGRAM**

Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent.

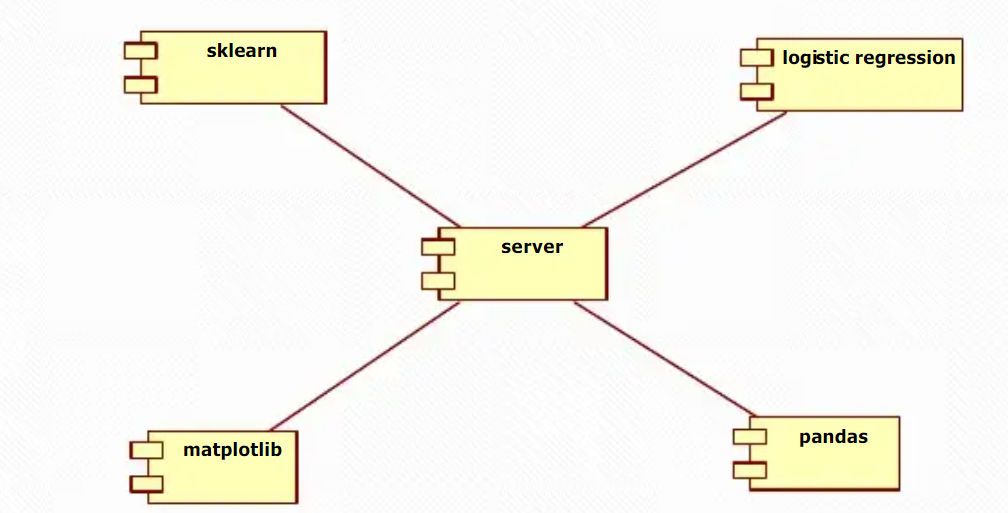
All activity diagrams have an initial state and final state that mark the start and end of the process. Activity or action state – Represents a single activity that sets a series of actions into motion. An example could be a user logging into their account in a mobile banking system.

****

**4.5 COMPONENT DIAGRAM**

The purpose of a component diagram is to show the relationship between different components in a system. For the purpose of UML 2.0, the term "component" refers to a module of classes that represent independent systems or subsystems with the ability to interface with the rest of the system

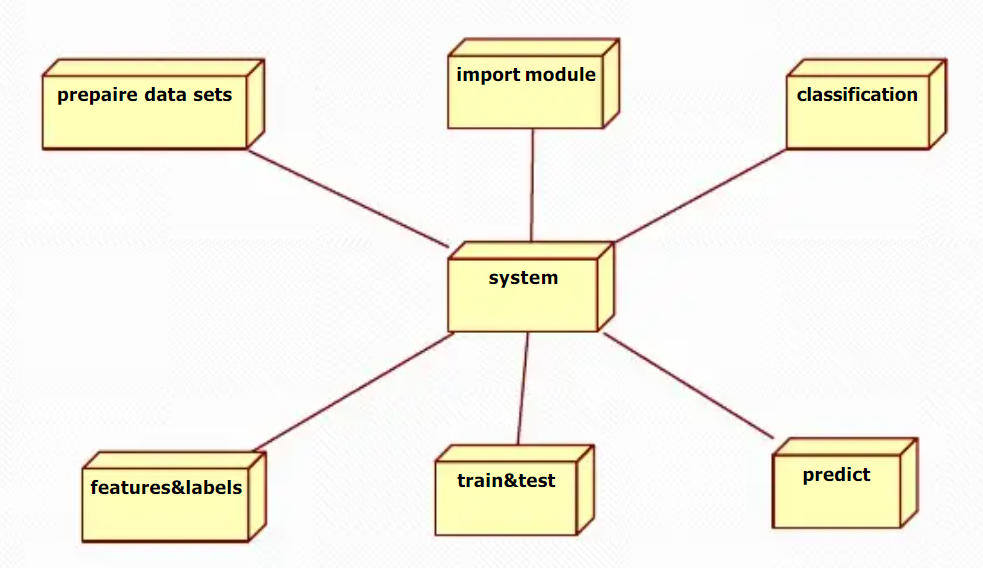
n UML, component diagrams show the structure of the software system, which describes the software components, their interfaces, and their dependencies. You can use component diagrams to model software systems at a high level or to show components at a lower, package level..

****

**4.6 DEPLOYMENT DIAGRAM**

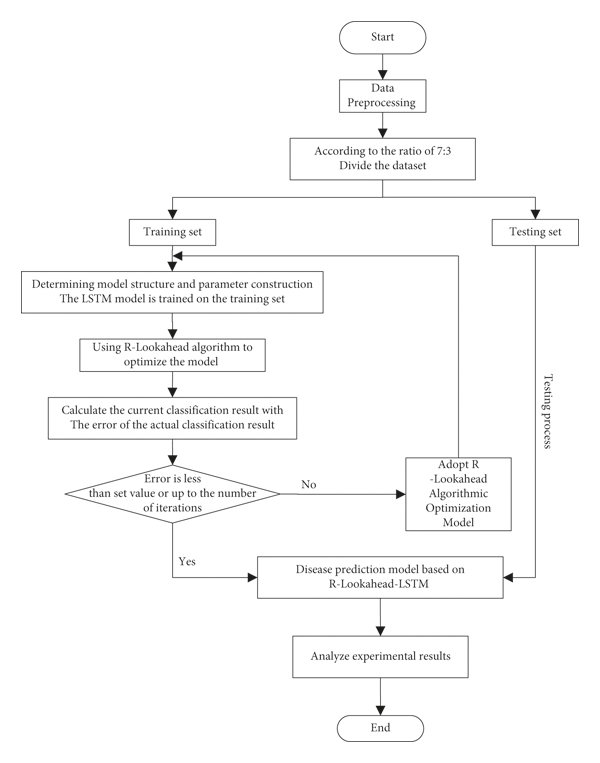
A deployment diagram is a UML diagram type that shows the execution architecture of a system, including nodes such as hardware or software execution environments, and the middleware connecting them. Deployment diagrams are typically used to visualize the physical hardware and software of a system.

Deployment diagrams is a kind of structure diagram used in modeling the physical aspects of an object-oriented system. They are often be used to model the static deployment view of a system (topology of the hardware).

****

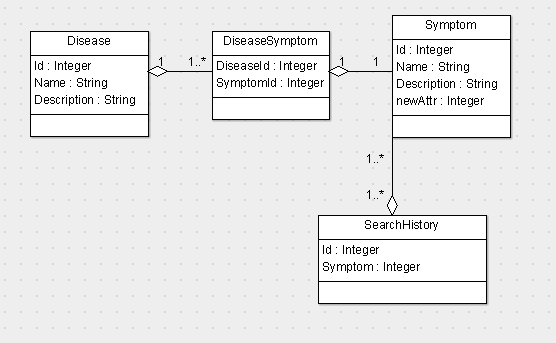
**4.7 DATA FLOW DIAGRAM**

Also known as DFD, Data flow diagrams are used to graphically represent the flow of data in a business information system. DFD describes the processes that are involved in a system to transfer data from the input to the file storage and reports generation. Data flow diagrams can be divided into logical and physical. There are two types of DFDs — logical and physical. Logical diagrams display the theoretical process of moving information through a system, like where the data comes from, where it goes, how it changes, and where it ends up.

****

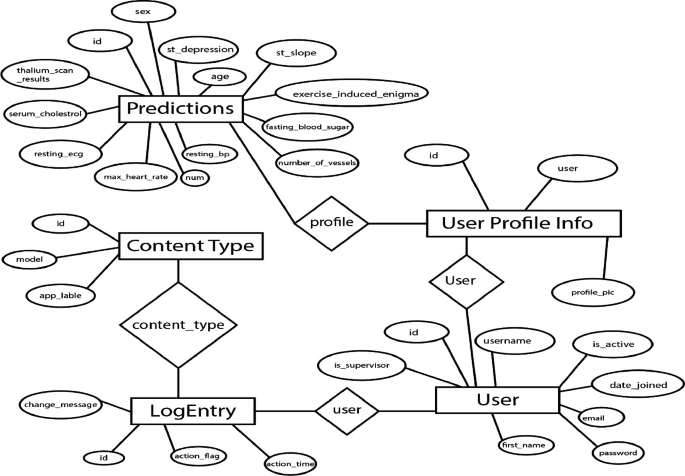
**4.8 CLASS DIAGRAM**

The class diagram is the main building block of object-oriented modeling. It is used for general conceptual modeling of the structure of the application, and for detailed modeling, translating the models into programming code. Class diagrams can also be used for data modeling. A class diagram is a visual representation of class objects in a model system, categorized by class types. Each class type is represented as a rectangle with three compartments for the class name, attributes, and operations. To understand a class diagram, we must first define what a class is.

****

**4.9 ER DIAGRAM**

An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how “entities” such as people, objects or concepts relate to each other within a system. There are three main types of relationships in a database model: one-to-one, one-to-many, and many-to-many. In a one-to-one relationship, each record in one table is associated with one and only one record in another table.

****

**4.10 BAR GRAPH**

A bar graph can be defined as a graphical representation of data, quantities, or numbers using bars or strips. They are used to compare and contrast different types of data, frequencies, or other measures of distinct categories of data.

A bar graph is a picture that is made up of bars with different height. Each bar represents a different category. The height of each bar can tell us how often something happens or show us the number of items we have for each group.

****

**Chapter 5: Implementation And Testing**

**5.1 Implementation Approaches**

1. Data Collection: Collect data related to symptoms, disease diagnosis, and treatment from reliable sources such as the World Health Organization (WHO), medical textbooks, and databases.
2. Data Preprocessing: Clean and preprocess the data by removing duplicates, inconsistencies, and irrelevant information. Convert the data into a structured format, such as a spreadsheet or database, for easy analysis and processing.
3. Machine Learning Model: Use machine learning techniques like Decision Trees, Random Forests, or Neural Networks to train the disease prediction model. These models can be used to classify different diseases based on a user's symptoms.
4. Model Training: Split the preprocessed data into training and testing sets. Train the disease prediction model on the training set. This may involve adjusting the model's hyperparameters, such as the learning rate and number of layers, to optimize its performance.
5. Model Evaluation: Evaluate the trained model's performance on the testing set using appropriate evaluation metrics like accuracy, precision, recall, F1-score, or Area Under the ROC Curve (AUC-ROC). If the model's performance is unsatisfactory, consider using a different model, improving the model's hyperparameters, or gathering more data.
6. Inference: To predict a disease based on a user's symptoms, pass the user's input through the trained model. The model will then provide a predicted disease based on the highest probability score.
7. Rule-Based System: Develop a rule-based system to enhance the performance of the machine learning model. This system can be used to handle ambiguous cases and provide more accurate and useful responses.

**5.2 Testing Strategies and Quality Assurance**

* Testing plan:

1. Overview:

* Objective: The purpose of this testing plan is to ensure the accurate, reliable, and efficient operation of the disease prediction model in predicting disease diagnoses.
  + Scope: This testing plan covers functional, performance, load, stress, security, compliance, QA, and integration testing.

1. Test Strategy:

* Test Levels:
* Level 1: Unit testing, covering individual functions or methods within the model.
* Level 2: Integration testing, ensuring seamless integration between the model and other healthcare applications.
* Level 3: System testing, which includes functional, performance, load, stress, security, compliance, QA, and integration testing.

1. Test Environments:

* The testing will be conducted in a controlled environment with the necessary software, hardware, and network infrastructure.

1. Test Tools and Techniques:

* For unit testing, a range of unit testing tools and techniques will be used, such as assert statements, test-driven development, and code coverage analysis.
* For integration testing, tools such as JUnit, TestNG, or Mockito will be employed to facilitate integration testing between the model and other healthcare applications.
* For performance testing, tools like JMeter or ApacheBench will be utilized to measure response times and resource usage.
* For load testing, tools like Gatling or Locust will be employed to simulate high data loads and evaluate the model's performance under stress.
* For security testing, tools like OWASP ZAP or Burp Suite will be used to identify and assess potential security vulnerabilities.
* For compliance testing, tools like HHS National Cybersecurity Awareness Toolkit (NIST Cybersecurity Framework) or ISO 27001:2013 will be employed to verify compliance with industry standards and regulations.

1. Test Schedule:

Testing will be conducted in parallel with development to minimize testing time and costs.

A dedicated test team will be responsible for executing and monitoring the tests, ensuring the smooth progress of the testing process.

1. Test Reporting and Management:

The testing process will be documented using tools like TestLink or QTest.

Defects or issues identified during testing will be tracked and managed using a defect tracking tool, such as Jira or Bugzilla.

Based on the test results, the model will be refined and improved to enhance its accuracy, reliability, and efficiency.

1. Testing Evaluation:

The effectiveness of the testing strategy will be evaluated using metrics such as test coverage, defect density, and the proportion of issues resolved before deployment.

Continuous improvement and optimization will be pursued to enhance the quality and reliability of the disease prediction model.

By following this testing plan, the disease prediction project can effectively manage risks, maintain high quality standards, and ensure the timely delivery of an accurate and reliable disease prediction model.

* Quality Assurance:

Quality Assurance (QA) in the disease prediction project is crucial for addressing various risks and challenges. Some of the key QA aspects to consider in this project include:

1. Unit Testing:

This involves testing individual functions or methods within the model to ensure their correctness and reliability.

For example, if the model utilizes a specific algorithm, unit testing can help identify any errors or flaws in the algorithm's implementation.

1. Integration Testing:

Integration testing ensures that the disease prediction model integrates seamlessly with other healthcare applications and systems.

This testing helps identify and resolve any compatibility or integration issues that may arise during the implementation process.

1. Performance Testing:

Performance testing evaluates the model's ability to process large amounts of data in a timely and efficient manner.

This testing helps identify any bottlenecks or slowdowns that may impact the model's performance in real-world scenarios.

1. Load Testing:

Load testing simulates high data loads and evaluates the model's performance under stress.

This testing helps identify any potential issues that may arise when the model is under high load or high data volume.

1. Security Testing:

Security testing is crucial for identifying and addressing potential security vulnerabilities in the disease prediction model.

This testing helps ensure the model's protection against unauthorized access, data breaches, or malicious attacks.

1. Compliance Testing:

Compliance testing ensures that the disease prediction model adheres to

industry standards and regulations.

This testing helps identify any discrepancies between the model's design and implementation and the applicable standards and regulat

**CODE :**

Prototype.csv

Prototype-1.csv

In [1]

*#Importing Libraries*

**from** mpl\_toolkits.mplot3d **import** Axes3D

**from** sklearn.preprocessing **import** StandardScaler

**import** matplotlib.pyplot **as** plt

**from** tkinter **import** **\***

**import** numpy **as** np

**import** pandas **as** pd

**import** os

In [2]

*#List of the symptoms is listed here in list l1.*

l1**=**['back\_pain','constipation','abdominal\_pain','diarrhoea','mild\_fever','yellow\_urine',

'yellowing\_of\_eyes','acute\_liver\_failure','fluid\_overload','swelling\_of\_stomach',

'swelled\_lymph\_nodes','malaise','blurred\_and\_distorted\_vision','phlegm','throat\_irritation',

'redness\_of\_eyes','sinus\_pressure','runny\_nose','congestion','chest\_pain','weakness\_in\_limbs',

'fast\_heart\_rate','pain\_during\_bowel\_movements','pain\_in\_anal\_region','bloody\_stool',

'irritation\_in\_anus','neck\_pain','dizziness','cramps','bruising','obesity','swollen\_legs',

'swollen\_blood\_vessels','puffy\_face\_and\_eyes','enlarged\_thyroid','brittle\_nails',

'swollen\_extremeties','excessive\_hunger','extra\_marital\_contacts','drying\_and\_tingling\_lips',

'slurred\_speech','knee\_pain','hip\_joint\_pain','muscle\_weakness','stiff\_neck','swelling\_joints',

'movement\_stiffness','spinning\_movements','loss\_of\_balance','unsteadiness',

'weakness\_of\_one\_body\_side','loss\_of\_smell','bladder\_discomfort','foul\_smell\_of urine',

'continuous\_feel\_of\_urine','passage\_of\_gases','internal\_itching','toxic\_look\_(typhos)',

'depression','irritability','muscle\_pain','altered\_sensorium','red\_spots\_over\_body','belly\_pain',

'abnormal\_menstruation','dischromic \_patches','watering\_from\_eyes','increased\_appetite','polyuria','family\_history','mucoid\_sputum',

'rusty\_sputum','lack\_of\_concentration','visual\_disturbances','receiving\_blood\_transfusion',

'receiving\_unsterile\_injections','coma','stomach\_bleeding','distention\_of\_abdomen',

'history\_of\_alcohol\_consumption','fluid\_overload','blood\_in\_sputum','prominent\_veins\_on\_calf',

'palpitations','painful\_walking','pus\_filled\_pimples','blackheads','scurring','skin\_peeling',

'silver\_like\_dusting','small\_dents\_in\_nails','inflammatory\_nails','blister','red\_sore\_around\_nose',

'yellow\_crust\_ooze']

In[3]

*#List of Diseases is listed in list disease.*

disease**=**['Fungal infection', 'Allergy', 'GERD', 'Chronic cholestasis',

'Drug Reaction', 'Peptic ulcer diseae', 'AIDS', 'Diabetes ',

'Gastroenteritis', 'Bronchial Asthma', 'Hypertension ', 'Migraine',

'Cervical spondylosis', 'Paralysis (brain hemorrhage)', 'Jaundice',

'Malaria', 'Chicken pox', 'Dengue', 'Typhoid', 'hepatitis A',

'Hepatitis B', 'Hepatitis C', 'Hepatitis D', 'Hepatitis E',

'Alcoholic hepatitis', 'Tuberculosis', 'Common Cold', 'Pneumonia',

'Dimorphic hemmorhoids(piles)', 'Heart attack', 'Varicose veins',

'Hypothyroidism', 'Hyperthyroidism', 'Hypoglycemia',

'Osteoarthristis', 'Arthritis',

'(vertigo) Paroymsal Positional Vertigo', 'Acne',

'Urinary tract infection', 'Psoriasis', 'Impetigo']

*#disease = [df['prognosis'].unique()]*

*#print(disease)*

*In[4]*

l2**=**[]

**for** i **in** range(0,len(l1)):

l2**.**append(0)

print(l2)

output:

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

In[5]

*#Reading the training .csv file*

df**=**pd**.**read\_csv("prototype.csv")

DF**=** pd**.**read\_csv('prototype.csv', index\_col**=**'prognosis')

*#Replace the values in the imported file by pandas by the inbuilt function replace in pandas.*

df**.**replace({'prognosis':{'Fungal infection':0,'Allergy':1,'GERD':2,'Chronic cholestasis':3,'Drug Reaction':4,

'Peptic ulcer diseae':5,'AIDS':6,'Diabetes ':7,'Gastroenteritis':8,'Bronchial Asthma':9,'Hypertension ':10,

'Migraine':11,'Cervical spondylosis':12,

'Paralysis (brain hemorrhage)':13,'Jaundice':14,'Malaria':15,'Chicken pox':16,'Dengue':17,'Typhoid':18,'hepatitis A':19,

'Hepatitis B':20,'Hepatitis C':21,'Hepatitis D':22,'Hepatitis E':23,'Alcoholic hepatitis':24,'Tuberculosis':25,

'Common Cold':26,'Pneumonia':27,'Dimorphic hemmorhoids(piles)':28,'Heart attack':29,'Varicose veins':30,'Hypothyroidism':31,

'Hyperthyroidism':32,'Hypoglycemia':33,'Osteoarthristis':34,'Arthritis':35,

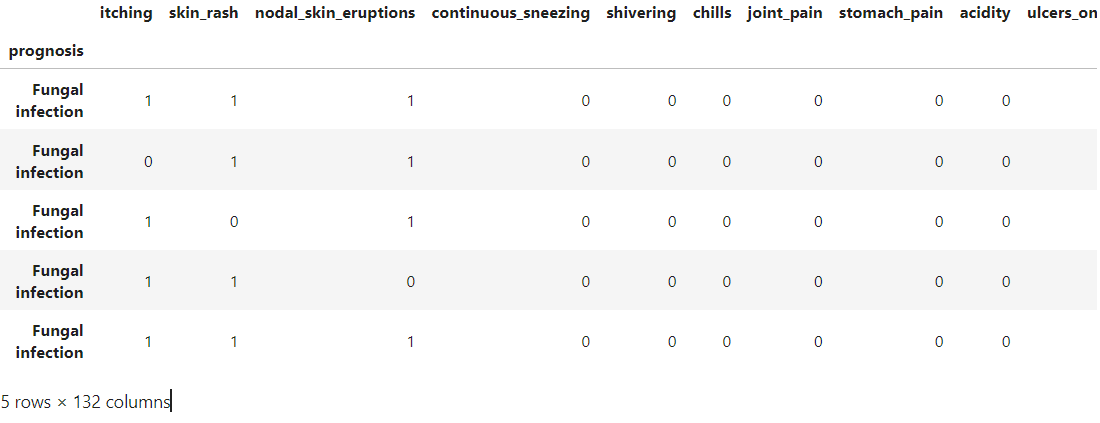
'(vertigo) Paroymsal Positional Vertigo':36,'Acne':37,'Urinary tract infection':38,'Psoriasis':39,

'Impetigo':40}},inplace**=True**)

*#df.head()*

DF**.**head()

Output:



In[6]

*# Distribution graphs (histogram/bar graph) of column data*

**def** plotPerColumnDistribution(df1, nGraphShown, nGraphPerRow):

nunique **=** df1**.**nunique()

df1 **=** df1[[col **for** col **in** df **if** nunique[col] **>** 1 **and** nunique[col] **<** 50]] *# For displaying purposes, pick columns that have between 1 and 50 unique values*

nRow, nCol **=** df1**.**shape

columnNames **=** list(df1)

nGraphRow **=** (nCol **+** nGraphPerRow **-** 1) **/** nGraphPerRow

plt**.**figure(num **=** **None**, figsize **=** (6 **\*** nGraphPerRow, 8 **\*** nGraphRow), dpi **=** 80, facecolor **=** 'w', edgecolor **=** 'k')

**for** i **in** range(min(nCol, nGraphShown)):

plt**.**subplot(nGraphRow, nGraphPerRow, i **+** 1)

columnDf **=** df**.**iloc[:, i]

**if** (**not** np**.**issubdtype(type(columnDf**.**iloc[0]), np**.**number)):

valueCounts **=** columnDf**.**value\_counts()

valueCounts**.**plot**.**bar()

**else**:

columnDf**.**hist()

plt**.**ylabel('counts')

plt**.**xticks(rotation **=** 90)

plt**.**title(f'{columnNames[i]} (column {i})')

plt**.**tight\_layout(pad **=** 1.0, w\_pad **=** 1.0, h\_pad **=** 1.0)

plt**.**show()

In[7]

*# Scatter and density plots*

**def** plotScatterMatrix(df1, plotSize, textSize):

df1 **=** df1**.**select\_dtypes(include **=**[np**.**number]) *# keep only numerical columns*

*# Remove rows and columns that would lead to df being singular*

df1 **=** df1**.**dropna('columns')

df1 **=** df1[[col **for** col **in** df **if** df[col]**.**nunique() **>** 1]] *# keep columns where there are more than 1 unique values*

columnNames **=** list(df)

**if** len(columnNames) **>** 10: *# reduce the number of columns for matrix inversion of kernel density plots*

columnNames **=** columnNames[:10]

df1 **=** df1[columnNames]

ax **=** pd**.**plotting**.**scatter\_matrix(df1, alpha**=**0.75, figsize**=**[plotSize, plotSize], diagonal**=**'kde')

corrs **=** df1**.**corr()**.**values

**for** i, j **in** zip(**\***plt**.**np**.**triu\_indices\_from(ax, k **=** 1)):

ax[i, j]**.**annotate('Corr. coef = %.3f' **%** corrs[i, j], (0.8, 0.2), xycoords**=**'axes fraction', ha**=**'center', va**=**'center', size**=**textSize)

plt**.**suptitle('Scatter and Density Plot')

plt**.**show()

In[8]

X**=** df[l1]

y **=** df[["prognosis"]]

np**.**ravel(y)

print(X)

output:

yellow\_urine yellowing\_of\_eyes acute\_liver\_failure fluid\_overload \

0 0 0 0 0

1 0 0 0 0

2 0 0 0 0

3 0 0 0 0

4 0 0 0 0

5 0 0 0 0

6 0 0 0 0

7 0 0 0 0

8 0 0 0 0

9 0 0 0 0

10 0 0 0 0

11 0 0 0 0

12 0 0 0 0

13 0 0 0 0

14 0 0 0 0

15 0 0 0 0

16 0 0 0 0

17 0 0 0 0

18 0 0 0 0

19 0 0 0 0

20 0 0 0 0

21 0 0 0 0

22 0 0 0 0

23 0 0 0 0

24 0 0 0 0

25 0 0 0 0

26 0 0 0 0

27 0 0 0 0

28 0 0 0 0

29 0 0 0 0

... ... ... ... ...

4890 0 0 0 0

4891 0 0 0 0

4892 0 0 0 0

4893 0 0 0 0

4894 0 0 0 0

4895 0 0 0 0

4896 0 0 0 0

4897 0 0 0 0

4898 0 1 0 0

4899 1 1 0 0

4900 0 1 0 0

4901 0 1 0 0

4902 0 1 1 0

4903 0 0 0 0

4904 0 1 0 0

4905 0 0 0 0

4906 0 0 0 0

4907 0 0 0 0

4908 0 0 0 0

4909 0 0 0 0

4910 0 0 0 0

4911 0 0 0 0

4912 0 0 0 0

4913 0 0 0 0

4914 0 0 0 0

4915 0 0 0 0

4916 0 0 0 0

4917 0 0 0 0

4918 0 0 0 0

4919 0 0 0 0

swelling\_of\_stomach ... pus\_filled\_pimples blackheads scurring \

0 0 ... 0 0 0

1 0 ... 0 0 0

2 0 ... 0 0 0

3 0 ... 0 0 0

4 0 ... 0 0 0

5 0 ... 0 0 0

6 0 ... 0 0 0

7 0 ... 0 0 0

8 0 ... 0 0 0

9 0 ... 0 0 0

10 0 ... 0 0 0

11 0 ... 0 0 0

12 0 ... 0 0 0

13 0 ... 0 0 0

14 0 ... 0 0 0

15 0 ... 0 0 0

16 0 ... 0 0 0

17 0 ... 0 0 0

18 0 ... 0 0 0

19 0 ... 0 0 0

20 0 ... 0 0 0

21 0 ... 0 0 0

22 0 ... 0 0 0

23 0 ... 0 0 0

24 0 ... 0 0 0

25 0 ... 0 0 0

26 0 ... 0 0 0

27 0 ... 0 0 0

28 0 ... 0 0 0

29 0 ... 0 0 0

... ... ... ... ... ...

4890 0 ... 0 0 0

4891 0 ... 0 0 0

4892 0 ... 0 0 0

4893 0 ... 0 0 0

4894 0 ... 0 0 0

4895 0 ... 0 0 0

4896 0 ... 0 0 0

4897 0 ... 0 0 0

4898 0 ... 0 0 0

4899 0 ... 0 0 0

4900 0 ... 0 0 0

4901 0 ... 0 0 0

4902 0 ... 0 0 0

4903 1 ... 0 0 0

4904 0 ... 0 0 0

4905 0 ... 0 0 0

4906 0 ... 0 0 0

4907 0 ... 0 0 0

4908 0 ... 0 0 0

4909 0 ... 0 0 0

4910 0 ... 0 0 0

4911 0 ... 0 0 0

4912 0 ... 0 0 0

4913 0 ... 0 0 0

4914 0 ... 0 0 0

4915 0 ... 0 0 0

4916 0 ... 1 1 1

4917 0 ... 0 0 0

4918 0 ... 0 0 0

4919 0 ... 0 0 0

skin\_peeling silver\_like\_dusting small\_dents\_in\_nails \

0 0 0 0

1 0 0 0

2 0 0 0

3 0 0 0

4 0 0 0

5 0 0 0

6 0 0 0

7 0 0 0

8 0 0 0

9 0 0 0

10 0 0 0

11 0 0 0

12 0 0 0

13 0 0 0

14 0 0 0

15 0 0 0

16 0 0 0

17 0 0 0

18 0 0 0

19 0 0 0

20 0 0 0

21 0 0 0

22 0 0 0

23 0 0 0

24 0 0 0

25 0 0 0

26 0 0 0

27 0 0 0

28 0 0 0

29 0 0 0

... ... ... ...

4890 0 0 0

4891 0 0 0

4892 0 0 0

4893 0 0 0

4894 0 0 0

4895 0 0 0

4896 0 0 0

4897 0 0 0

4898 0 0 0

4899 0 0 0

4900 0 0 0

4901 0 0 0

4902 0 0 0

4903 0 0 0

4904 0 0 0

4905 0 0 0

4906 0 0 0

4907 0 0 0

4908 0 0 0

4909 0 0 0

4910 0 0 0

4911 0 0 0

4912 0 0 0

4913 0 0 0

4914 0 0 0

4915 0 0 0

4916 0 0 0

4917 0 0 0

4918 1 1 1

4919 0 0 0

inflammatory\_nails blister red\_sore\_around\_nose yellow\_crust\_ooze

0 0 0 0 0

1 0 0 0 0

2 0 0 0 0

3 0 0 0 0

4 0 0 0 0

5 0 0 0 0

6 0 0 0 0

7 0 0 0 0

8 0 0 0 0

9 0 0 0 0

10 0 0 0 0

11 0 0 0 0

12 0 0 0 0

13 0 0 0 0

14 0 0 0 0

15 0 0 0 0

16 0 0 0 0

17 0 0 0 0

18 0 0 0 0

19 0 0 0 0

20 0 0 0 0

21 0 0 0 0

22 0 0 0 0

23 0 0 0 0

24 0 0 0 0

25 0 0 0 0

26 0 0 0 0

27 0 0 0 0

28 0 0 0 0

29 0 0 0 0

... ... ... ... ...

4890 0 0 0 0

4891 0 0 0 0

4892 0 0 0 0

4893 0 0 0 0

4894 0 0 0 0

4895 0 0 0 0

4896 0 0 0 0

4897 0 0 0 0

4898 0 0 0 0

4899 0 0 0 0

4900 0 0 0 0

4901 0 0 0 0

4902 0 0 0 0

4903 0 0 0 0

4904 0 0 0 0

4905 0 0 0 0

4906 0 0 0 0

4907 0 0 0 0

4908 0 0 0 0

4909 0 0 0 0

4910 0 0 0 0

4911 0 0 0 0

4912 0 0 0 0

4913 0 0 0 0

4914 0 0 0 0

4915 0 0 0 0

4916 0 0 0 0

4917 0 0 0 0

4918 1 0 0 0

4919 0 1 1 1

[4920 rows x 95 columns]

In[9]

print(y)

output:

prognosis

0 0

1 0

2 0

3 0

4 0

5 0

6 0

7 0

8 0

9 0

10 1

11 1

12 1

13 1

14 1

15 1

16 1

17 1

18 1

19 1

20 2

21 2

22 2

23 2

24 2

25 2

26 2

27 2

28 2

29 2

... ...

4890 11

4891 12

4892 13

4893 14

4894 15

4895 16

4896 17

4897 18

4898 19

4899 20

4900 21

4901 22

4902 23

4903 24

4904 25

4905 26

4906 27

4907 28

4908 29

4909 30

4910 31

4911 32

4912 33

4913 34

4914 35

4915 36

4916 37

4917 38

4918 39

4919 40

[4920 rows x 1 columns]

In[10]

tr**=**pd**.**read\_csv("prototype-1.csv")

*#Using inbuilt function replace in pandas for replacing the values*

tr**.**replace({'prognosis':{'Fungal infection':0,'Allergy':1,'GERD':2,'Chronic cholestasis':3,'Drug Reaction':4,

'Peptic ulcer diseae':5,'AIDS':6,'Diabetes ':7,'Gastroenteritis':8,'Bronchial Asthma':9,'Hypertension ':10,

'Migraine':11,'Cervical spondylosis':12,

'Paralysis (brain hemorrhage)':13,'Jaundice':14,'Malaria':15,'Chicken pox':16,'Dengue':17,'Typhoid':18,'hepatitis A':19,

'Hepatitis B':20,'Hepatitis C':21,'Hepatitis D':22,'Hepatitis E':23,'Alcoholic hepatitis':24,'Tuberculosis':25,

'Common Cold':26,'Pneumonia':27,'Dimorphic hemmorhoids(piles)':28,'Heart attack':29,'Varicose veins':30,'Hypothyroidism':31,

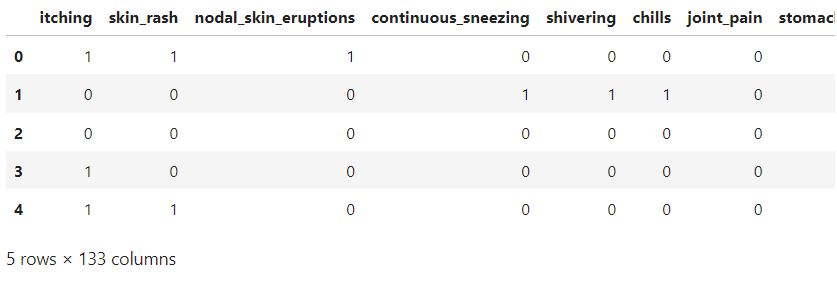
'Hyperthyroidism':32,'Hypoglycemia':33,'Osteoarthristis':34,'Arthritis':35,

'(vertigo) Paroymsal Positional Vertigo':36,'Acne':37,'Urinary tract infection':38,'Psoriasis':39,

'Impetigo':40}},inplace**=True**)

tr**.**head()

output:



In[11]

X\_test**=** tr[l1]

y\_test **=** tr[["prognosis"]]

np**.**ravel(y\_test)

print(X\_test)

output:

back\_pain constipation abdominal\_pain diarrhoea mild\_fever \

0 0 0 0 0 0

1 0 0 0 0 0

2 0 0 0 0 0

3 0 0 1 0 0

4 0 0 0 0 0

5 0 0 1 0 0

6 0 0 0 0 0

7 0 0 0 0 0

8 0 0 0 1 0

9 0 0 0 0 0

10 0 0 0 0 0

11 0 0 0 0 0

12 1 0 0 0 0

13 0 0 0 0 0

14 0 0 1 0 0

15 0 0 0 1 0

16 0 0 0 0 1

17 1 0 0 0 0

18 0 1 1 1 0

19 0 0 1 1 1

20 0 0 1 0 0

21 0 0 0 0 0

22 0 0 1 0 0

23 0 0 1 0 0

24 0 0 1 0 0

25 0 0 0 0 1

26 0 0 0 0 0

27 0 0 0 0 0

28 0 1 0 0 0

29 0 0 0 0 0

30 0 0 0 0 0

31 0 0 0 0 0

32 0 0 0 1 0

33 0 0 0 0 0

34 0 0 0 0 0

35 0 0 0 0 0

36 0 0 0 0 0

37 0 0 0 0 0

38 0 0 0 0 0

39 0 0 0 0 0

40 0 0 0 0 0

yellow\_urine yellowing\_of\_eyes acute\_liver\_failure fluid\_overload \

0 0 0 0 0

1 0 0 0 0

2 0 0 0 0

3 0 1 0 0

4 0 0 0 0

5 0 0 0 0

6 0 0 0 0

7 0 0 0 0

8 0 0 0 0

9 0 0 0 0

10 0 0 0 0

11 0 0 0 0

12 0 0 0 0

13 0 0 0 0

14 0 0 0 0

15 0 0 0 0

16 0 0 0 0

17 0 0 0 0

18 0 0 0 0

19 0 1 0 0

20 1 1 0 0

21 0 1 0 0

22 0 1 0 0

23 0 1 1 0

24 0 0 0 0

25 0 1 0 0

26 0 0 0 0

27 0 0 0 0

28 0 0 0 0

29 0 0 0 0

30 0 0 0 0

31 0 0 0 0

32 0 0 0 0

33 0 0 0 0

34 0 0 0 0

35 0 0 0 0

36 0 0 0 0

37 0 0 0 0

38 0 0 0 0

39 0 0 0 0

40 0 0 0 0

swelling\_of\_stomach ... pus\_filled\_pimples blackheads scurring \

0 0 ... 0 0 0

1 0 ... 0 0 0

2 0 ... 0 0 0

3 0 ... 0 0 0

4 0 ... 0 0 0

5 0 ... 0 0 0

6 0 ... 0 0 0

7 0 ... 0 0 0

8 0 ... 0 0 0

9 0 ... 0 0 0

10 0 ... 0 0 0

11 0 ... 0 0 0

12 0 ... 0 0 0

13 0 ... 0 0 0

14 0 ... 0 0 0

15 0 ... 0 0 0

16 0 ... 0 0 0

17 0 ... 0 0 0

18 0 ... 0 0 0

19 0 ... 0 0 0

20 0 ... 0 0 0

21 0 ... 0 0 0

22 0 ... 0 0 0

23 0 ... 0 0 0

24 1 ... 0 0 0

25 0 ... 0 0 0

26 0 ... 0 0 0

27 0 ... 0 0 0

28 0 ... 0 0 0

29 0 ... 0 0 0

30 0 ... 0 0 0

31 0 ... 0 0 0

32 0 ... 0 0 0

33 0 ... 0 0 0

34 0 ... 0 0 0

35 0 ... 0 0 0

36 0 ... 0 0 0

37 0 ... 1 1 1

38 0 ... 0 0 0

39 0 ... 0 0 0

40 0 ... 0 0 0

skin\_peeling silver\_like\_dusting small\_dents\_in\_nails \

0 0 0 0

1 0 0 0

2 0 0 0

3 0 0 0

4 0 0 0

5 0 0 0

6 0 0 0

7 0 0 0

8 0 0 0

9 0 0 0

10 0 0 0

11 0 0 0

12 0 0 0

13 0 0 0

14 0 0 0

15 0 0 0

16 0 0 0

17 0 0 0

18 0 0 0

19 0 0 0

20 0 0 0

21 0 0 0

22 0 0 0

23 0 0 0

24 0 0 0

25 0 0 0

26 0 0 0

27 0 0 0

28 0 0 0

29 0 0 0

30 0 0 0

31 0 0 0

32 0 0 0

33 0 0 0

34 0 0 0

35 0 0 0

36 0 0 0

37 0 0 0

38 0 0 0

39 1 1 1

40 0 0 0

inflammatory\_nails blister red\_sore\_around\_nose yellow\_crust\_ooze

0 0 0 0 0

1 0 0 0 0

2 0 0 0 0

3 0 0 0 0

4 0 0 0 0

5 0 0 0 0

6 0 0 0 0

7 0 0 0 0

8 0 0 0 0

9 0 0 0 0

10 0 0 0 0

11 0 0 0 0

12 0 0 0 0

13 0 0 0 0

14 0 0 0 0

15 0 0 0 0

16 0 0 0 0

17 0 0 0 0

18 0 0 0 0

19 0 0 0 0

20 0 0 0 0

21 0 0 0 0

22 0 0 0 0

23 0 0 0 0

24 0 0 0 0

25 0 0 0 0

26 0 0 0 0

27 0 0 0 0

28 0 0 0 0

29 0 0 0 0

30 0 0 0 0

31 0 0 0 0

32 0 0 0 0

33 0 0 0 0

34 0 0 0 0

35 0 0 0 0

36 0 0 0 0

37 0 0 0 0

38 0 0 0 0

39 1 0 0 0

40 0 1 1 1

[41 rows x 95 columns]

In [12]:

print(y\_test)

prognosis

0 0

1 1

2 2

3 3

4 4

5 5

6 6

7 7

8 8

9 9

10 10

11 11

12 12

13 13

14 14

15 15

16 16

17 17

18 18

19 19

20 20

21 21

22 22

23 23

24 24

25 25

26 26

27 27

28 28

29 29

30 30

31 31

32 32

33 33

34 34

35 35

36 36

37 37

38 38

39 39

40 40

**To build the precision of the model, we utilized three distinctive algorithms which are as per the following**

* Decision Tree algorithm
* Random Forest algorithm
* KNearestNeighbour algorithm
* Naive Bayes algorithm

In[13]

*#list1 = DF['prognosis'].unique()*

**def** scatterplt(disea):

x **=** ((DF**.**loc[disea])**.**sum())*#total sum of symptom reported for given disease*

x**.**drop(x[x**==**0]**.**index,inplace**=True**)*#droping symptoms with values 0*

print(x**.**values)

y **=** x**.**keys()*#storing nameof symptoms in y*

print(len(x))

print(len(y))

plt**.**title(disea)

plt**.**scatter(y,x**.**values)

plt**.**show()

**def** scatterinp(sym1,sym2,sym3,sym4,sym5):

x **=** [sym1,sym2,sym3,sym4,sym5]*#storing input symptoms in y*

y **=** [0,0,0,0,0]*#creating and giving values to the input symptoms*

**if**(sym1**!=**'Select Here'):

y[0]**=**1

**if**(sym2**!=**'Select Here'):

y[1]**=**1

**if**(sym3**!=**'Select Here'):

y[2]**=**1

**if**(sym4**!=**'Select Here'):

y[3]**=**1

**if**(sym5**!=**'Select Here'):

y[4]**=**1

print(x)

print(y)

plt**.**scatter(x,y)

plt**.**show()

# Decision Tree Algorithm

In [14]:

root **=** Tk()

pred1**=**StringVar()

**def** DecisionTree():

**if** len(NameEn**.**get()) **==** 0:

pred1**.**set(" ")

comp**=**messagebox**.**askokcancel("System","Kindly Fill the Name")

**if** comp:

root**.**mainloop()

**elif**((Symptom1**.**get()**==**"Select Here") **or** (Symptom2**.**get()**==**"Select Here")):

pred1**.**set(" ")

sym**=**messagebox**.**askokcancel("System","Kindly Fill atleast first two Symptoms")

**if** sym:

root**.**mainloop()

**else**:

**from** sklearn **import** tree

clf3 **=** tree**.**DecisionTreeClassifier()

clf3 **=** clf3**.**fit(X,y)

**from** sklearn.metrics **import** classification\_report,confusion\_matrix,accuracy\_score

y\_pred**=**clf3**.**predict(X\_test)

print("Decision Tree")

print("Accuracy")

print(accuracy\_score(y\_test, y\_pred))

print(accuracy\_score(y\_test, y\_pred,normalize**=False**))

print("Confusion matrix")

conf\_matrix**=**confusion\_matrix(y\_test,y\_pred)

print(conf\_matrix)

psymptoms **=** [Symptom1**.**get(),Symptom2**.**get(),Symptom3**.**get(),Symptom4**.**get(),Symptom5**.**get()]

**for** k **in** range(0,len(l1)):

**for** z **in** psymptoms:

**if**(z**==**l1[k]):

l2[k]**=**1

inputtest **=** [l2]

predict **=** clf3**.**predict(inputtest)

predicted**=**predict[0]

h**=**'no'

**for** a **in** range(0,len(disease)):

**if**(predicted **==** a):

h**=**'yes'

**break**

**if** (h**==**'yes'):

pred1**.**set(" ")

pred1**.**set(disease[a])

**else**:

pred1**.**set(" ")

pred1**.**set("Not Found")

*#Creating the database if not exists named as database.db and creating table if not exists named as DecisionTree using sqlite3*

**import** sqlite3

conn **=** sqlite3**.**connect('database.db')

c **=** conn**.**cursor()

c**.**execute("CREATE TABLE IF NOT EXISTS DecisionTree(Name StringVar,Symtom1 StringVar,Symtom2 StringVar,Symtom3 StringVar,Symtom4 TEXT,Symtom5 TEXT,Disease StringVar)")

c**.**execute("INSERT INTO DecisionTree(Name,Symtom1,Symtom2,Symtom3,Symtom4,Symtom5,Disease) VALUES(?,?,?,?,?,?,?)",(NameEn**.**get(),Symptom1**.**get(),Symptom2**.**get(),Symptom3**.**get(),Symptom4**.**get(),Symptom5**.**get(),pred1**.**get()))

conn**.**commit()

c**.**close()

conn**.**close()

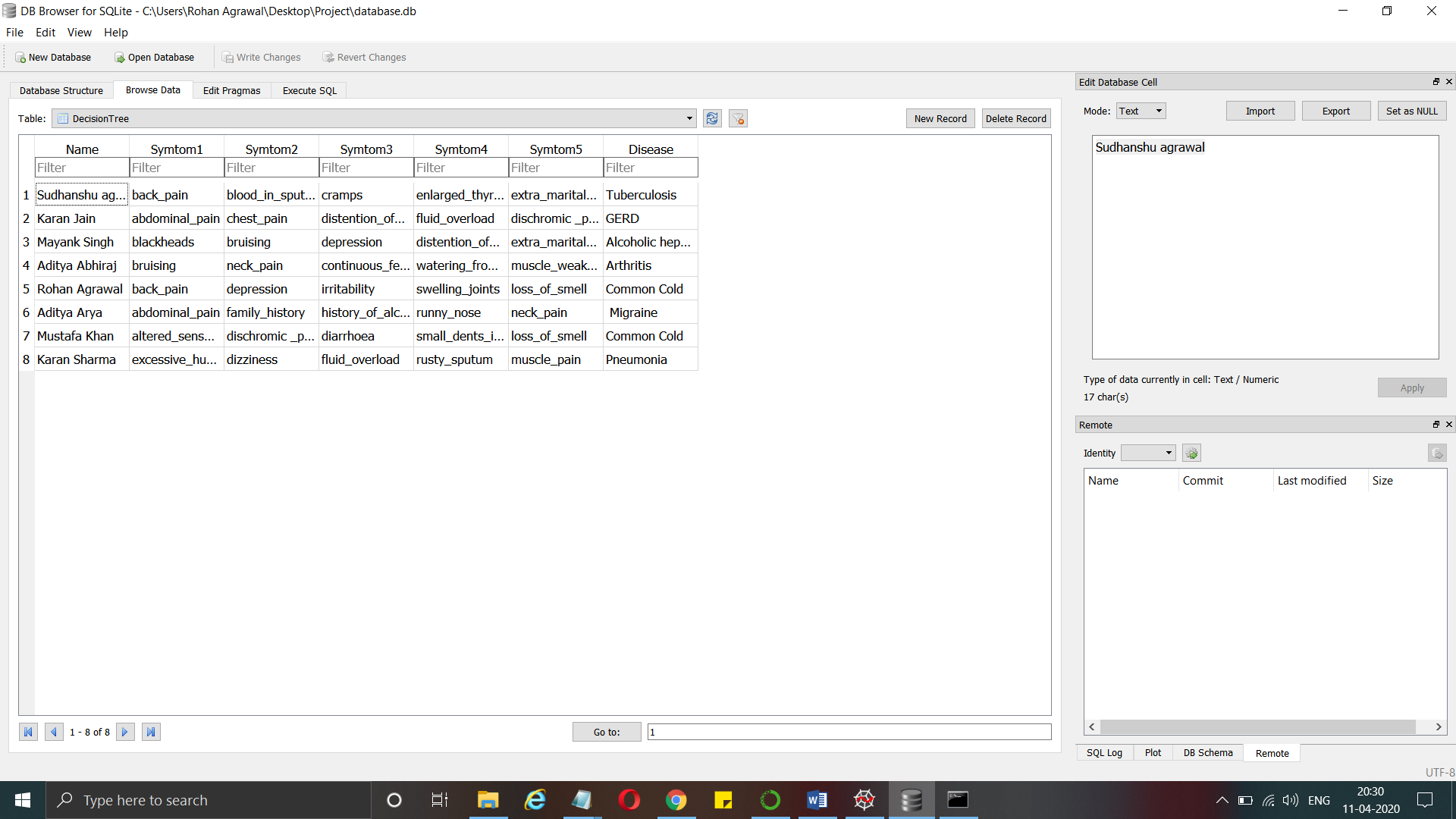
*#printing scatter plot of input symptoms*

*#printing scatter plot of disease predicted vs its symptoms*

scatterinp(Symptom1**.**get(),Symptom2**.**get(),Symptom3**.**get(),Symptom4**.**get(),Symptom5**.**get())

scatterplt(pred1**.**get())

Out[14]:



# Random Forest Algorithm

In [15]:

pred2**=**StringVar()

**def** randomforest():

**if** len(NameEn**.**get()) **==** 0:

pred1**.**set(" ")

comp**=**messagebox**.**askokcancel("System","Kindly Fill the Name")

**if** comp:

root**.**mainloop()

**elif**((Symptom1**.**get()**==**"Select Here") **or** (Symptom2**.**get()**==**"Select Here")):

pred1**.**set(" ")

sym**=**messagebox**.**askokcancel("System","Kindly Fill atleast first two Symptoms")

**if** sym:

root**.**mainloop()

**else**:

**from** sklearn.ensemble **import** RandomForestClassifier

clf4 **=** RandomForestClassifier(n\_estimators**=**100)

clf4 **=** clf4**.**fit(X,np**.**ravel(y))

*# calculating accuracy*

**from** sklearn.metrics **import** classification\_report,confusion\_matrix,accuracy\_score

y\_pred**=**clf4**.**predict(X\_test)

print("Random Forest")

print("Accuracy")

print(accuracy\_score(y\_test, y\_pred))

print(accuracy\_score(y\_test, y\_pred,normalize**=False**))

print("Confusion matrix")

conf\_matrix**=**confusion\_matrix(y\_test,y\_pred)

print(conf\_matrix)

psymptoms **=** [Symptom1**.**get(),Symptom2**.**get(),Symptom3**.**get(),Symptom4**.**get(),Symptom5**.**get()]

**for** k **in** range(0,len(l1)):

**for** z **in** psymptoms:

**if**(z**==**l1[k]):

l2[k]**=**1

inputtest **=** [l2]

predict **=** clf4**.**predict(inputtest)

predicted**=**predict[0]

h**=**'no'

**for** a **in** range(0,len(disease)):

**if**(predicted **==** a):

h**=**'yes'

**break**

**if** (h**==**'yes'):

pred2**.**set(" ")

pred2**.**set(disease[a])

**else**:

pred2**.**set(" ")

pred2**.**set("Not Found")

*#Creating the database if not exists named as database.db and creating table if not exists named as RandomForest using sqlite3*

**import** sqlite3

conn **=** sqlite3**.**connect('database.db')

c **=** conn**.**cursor()

c**.**execute("CREATE TABLE IF NOT EXISTS RandomForest(Name StringVar,Symtom1 StringVar,Symtom2 StringVar,Symtom3 StringVar,Symtom4 TEXT,Symtom5 TEXT,Disease StringVar)")

c**.**execute("INSERT INTO RandomForest(Name,Symtom1,Symtom2,Symtom3,Symtom4,Symtom5,Disease) VALUES(?,?,?,?,?,?,?)",(NameEn**.**get(),Symptom1**.**get(),Symptom2**.**get(),Symptom3**.**get(),Symptom4**.**get(),Symptom5**.**get(),pred2**.**get()))

conn**.**commit()

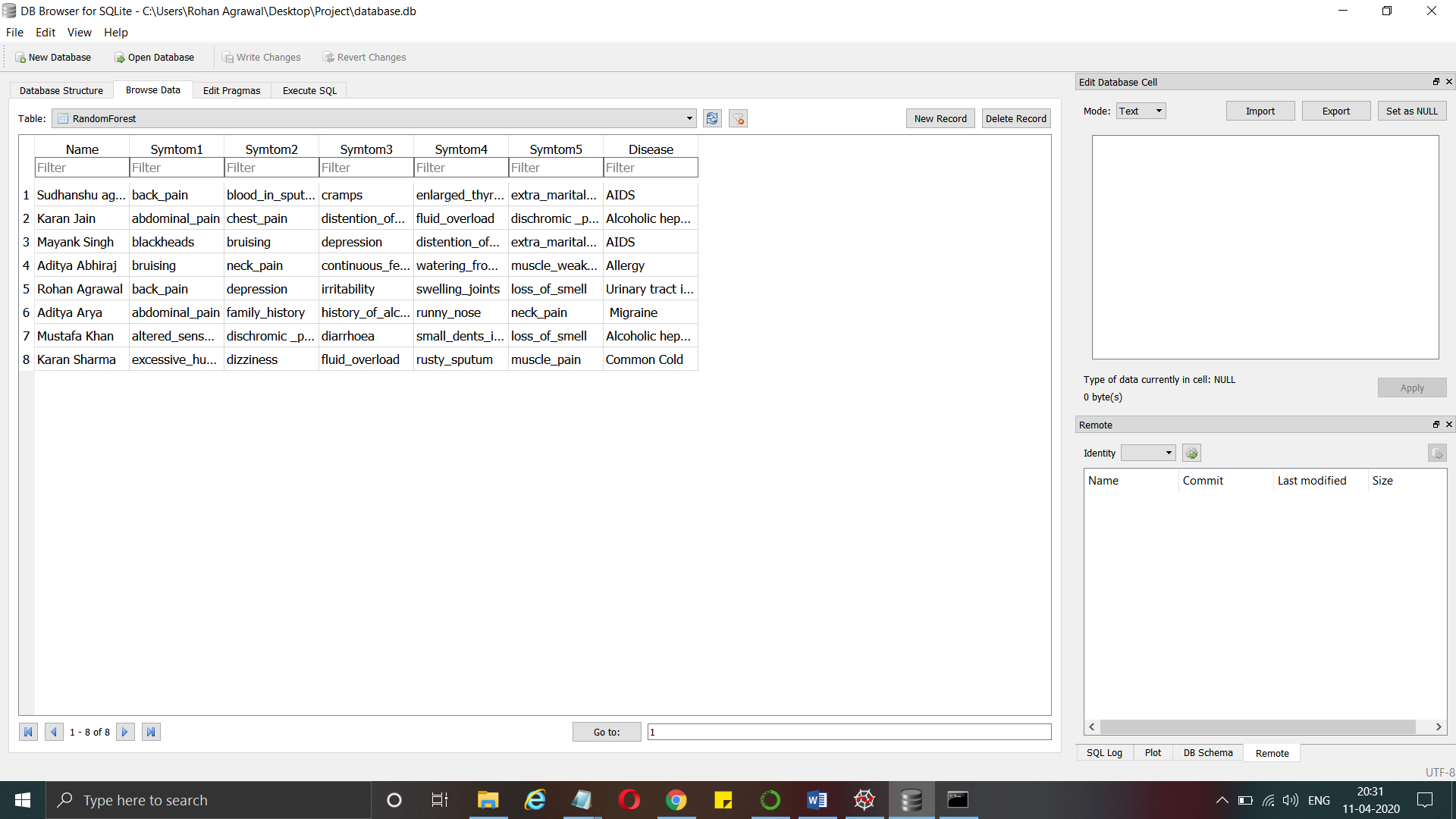
c**.**close()

conn**.**close()

*#printing scatter plot of disease predicted vs its symptoms*

scatterplt(pred2**.**get())

Out[15]:



# KNearestNeighbour Algorithm

In [16]:

pred4**=**StringVar()

**def** KNN():

**if** len(NameEn**.**get()) **==** 0:

pred1**.**set(" ")

comp**=**messagebox**.**askokcancel("System","Kindly Fill the Name")

**if** comp:

root**.**mainloop()

**elif**((Symptom1**.**get()**==**"Select Here") **or** (Symptom2**.**get()**==**"Select Here")):

pred1**.**set(" ")

sym**=**messagebox**.**askokcancel("System","Kindly Fill atleast first two Symptoms")

**if** sym:

root**.**mainloop()

**else**:

**from** sklearn.neighbors **import** KNeighborsClassifier

knn**=**KNeighborsClassifier(n\_neighbors**=**5,metric**=**'minkowski',p**=**2)

knn**=**knn**.**fit(X,np**.**ravel(y))

**from** sklearn.metrics **import** classification\_report,confusion\_matrix,accuracy\_score

y\_pred**=**knn**.**predict(X\_test)

print("kNearest Neighbour")

print("Accuracy")

print(accuracy\_score(y\_test, y\_pred))

print(accuracy\_score(y\_test, y\_pred,normalize**=False**))

print("Confusion matrix")

conf\_matrix**=**confusion\_matrix(y\_test,y\_pred)

print(conf\_matrix)

psymptoms **=** [Symptom1**.**get(),Symptom2**.**get(),Symptom3**.**get(),Symptom4**.**get(),Symptom5**.**get()]

**for** k **in** range(0,len(l1)):

**for** z **in** psymptoms:

**if**(z**==**l1[k]):

l2[k]**=**1

inputtest **=** [l2]

predict **=** knn**.**predict(inputtest)

predicted**=**predict[0]

h**=**'no'

**for** a **in** range(0,len(disease)):

**if**(predicted **==** a):

h**=**'yes'

**break**

**if** (h**==**'yes'):

pred4**.**set(" ")

pred4**.**set(disease[a])

**else**:

pred4**.**set(" ")

pred4**.**set("Not Found")

*#Creating the database if not exists named as database.db and creating table if not exists named as KNearestNeighbour using sqlite3*

**import** sqlite3

conn **=** sqlite3**.**connect('database.db')

c **=** conn**.**cursor()

c**.**execute("CREATE TABLE IF NOT EXISTS KNearestNeighbour(Name StringVar,Symtom1 StringVar,Symtom2 StringVar,Symtom3 StringVar,Symtom4 TEXT,Symtom5 TEXT,Disease StringVar)")

c**.**execute("INSERT INTO KNearestNeighbour(Name,Symtom1,Symtom2,Symtom3,Symtom4,Symtom5,Disease) VALUES(?,?,?,?,?,?,?)",(NameEn**.**get(),Symptom1**.**get(),Symptom2**.**get(),Symptom3**.**get(),Symptom4**.**get(),Symptom5**.**get(),pred4**.**get()))

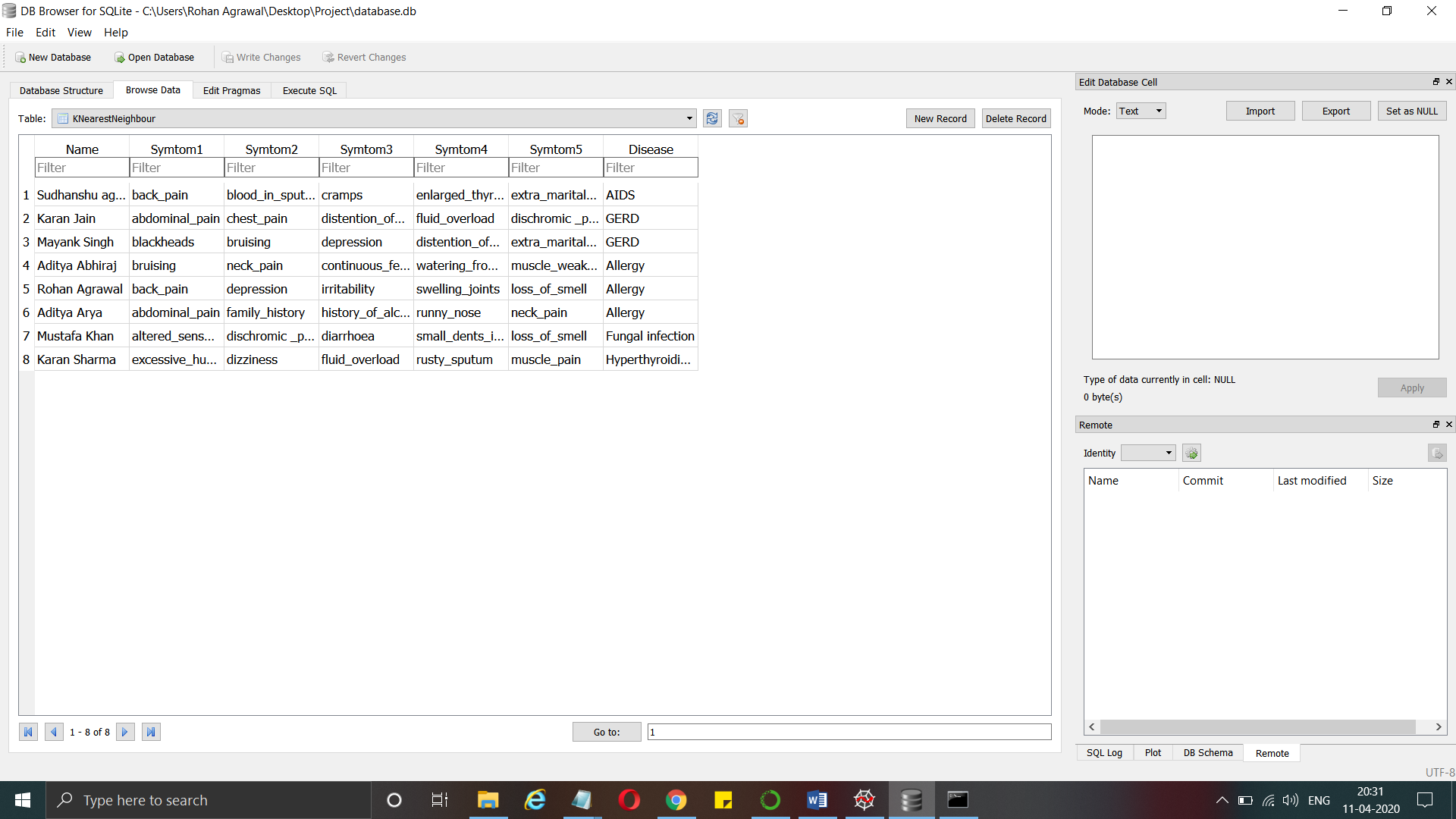
conn**.**commit()

c**.**close()

conn**.**close()

*#printing scatter plot of disease predicted vs its symptoms*

scatterplt(pred4**.**get())

Out[16]:

# Naive Bayes Algorithm

In [17]:

pred3**=**StringVar()

**def** NaiveBayes():

**if** len(NameEn**.**get()) **==** 0:

pred1**.**set(" ")

comp**=**messagebox**.**askokcancel("System","Kindly Fill the Name")

**if** comp:

root**.**mainloop()

**elif**((Symptom1**.**get()**==**"Select Here") **or** (Symptom2**.**get()**==**"Select Here")):

pred1**.**set(" ")

sym**=**messagebox**.**askokcancel("System","Kindly Fill atleast first two Symptoms")

**if** sym:

root**.**mainloop()

**else**:

**from** sklearn.naive\_bayes **import** GaussianNB

gnb **=** GaussianNB()

gnb**=**gnb**.**fit(X,np**.**ravel(y))

**from** sklearn.metrics **import** classification\_report,confusion\_matrix,accuracy\_score

y\_pred**=**gnb**.**predict(X\_test)

print("Naive Bayes")

print("Accuracy")

print(accuracy\_score(y\_test, y\_pred))

print(accuracy\_score(y\_test, y\_pred,normalize**=False**))

print("Confusion matrix")

conf\_matrix**=**confusion\_matrix(y\_test,y\_pred)

print(conf\_matrix)

psymptoms **=** [Symptom1**.**get(),Symptom2**.**get(),Symptom3**.**get(),Symptom4**.**get(),Symptom5**.**get()]

**for** k **in** range(0,len(l1)):

**for** z **in** psymptoms:

**if**(z**==**l1[k]):

l2[k]**=**1

inputtest **=** [l2]

predict **=** gnb**.**predict(inputtest)

predicted**=**predict[0]

h**=**'no'

**for** a **in** range(0,len(disease)):

**if**(predicted **==** a):

h**=**'yes'

**break**

**if** (h**==**'yes'):

pred3**.**set(" ")

pred3**.**set(disease[a])

**else**:

pred3**.**set(" ")

pred3**.**set("Not Found")

*#Creating the database if not exists named as database.db and creating table if not exists named as NaiveBayes using sqlite3*

**import** sqlite3

conn **=** sqlite3**.**connect('database.db')

c **=** conn**.**cursor()

c**.**execute("CREATE TABLE IF NOT EXISTS NaiveBayes(Name StringVar,Symtom1 StringVar,Symtom2 StringVar,Symtom3 StringVar,Symtom4 TEXT,Symtom5 TEXT,Disease StringVar)")

c**.**execute("INSERT INTO NaiveBayes(Name,Symtom1,Symtom2,Symtom3,Symtom4,Symtom5,Disease) VALUES(?,?,?,?,?,?,?)",(NameEn**.**get(),Symptom1**.**get(),Symptom2**.**get(),Symptom3**.**get(),Symptom4**.**get(),Symptom5**.**get(),pred3**.**get()))

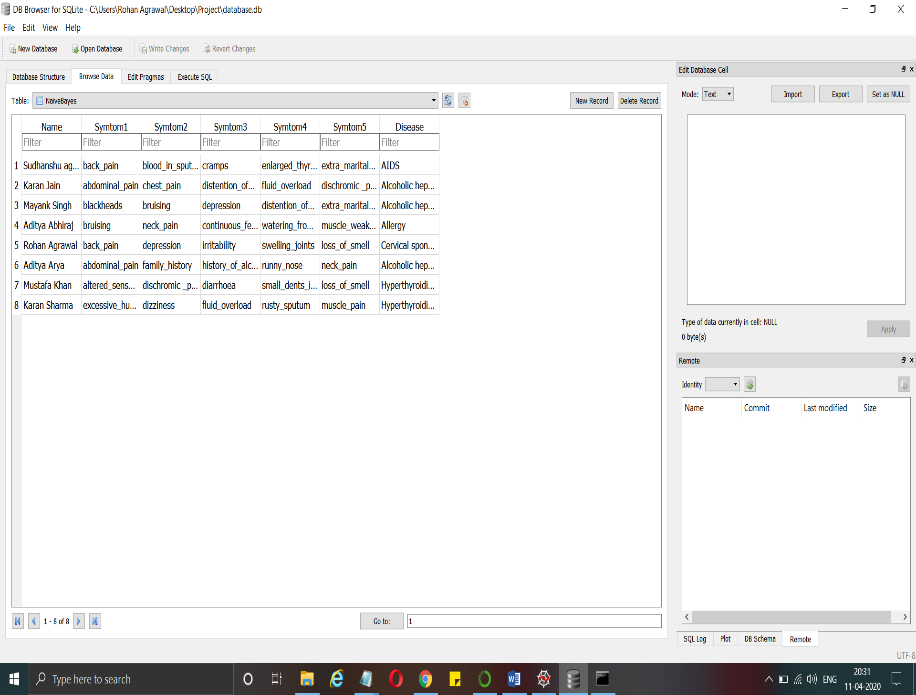
conn**.**commit()

c**.**close()

conn**.**close()

*#printing scatter plot of disease predicted vs its symptoms*

scatterplt(pred3**.**get()

Out[17] 

# Building Graphical User Interface

In [18]:

*#Tk class is used to create a root window*

root**.**configure(background**=**'Ivory')

root**.**title('Smart Disease Predictor System')

root**.**resizable(0,0)

Out[18]:

''

In [19]:

Symptom1 **=** StringVar()

Symptom1**.**set("Select Here")

Symptom2 **=** StringVar()

Symptom2**.**set("Select Here")

Symptom3 **=** StringVar()

Symptom3**.**set("Select Here")

Symptom4 **=** StringVar()

Symptom4**.**set("Select Here")

Symptom5 **=** StringVar()

Symptom5**.**set("Select Here")

Name **=** StringVar()

In [20]:

prev\_win**=None**

**def** Reset():

**global** prev\_win

Symptom1**.**set("Select Here")

Symptom2**.**set("Select Here")

Symptom3**.**set("Select Here")

Symptom4**.**set("Select Here")

Symptom5**.**set("Select Here")

NameEn**.**delete(first**=**0,last**=**100)

pred1**.**set(" ")

pred2**.**set(" ")

pred3**.**set(" ")

pred4**.**set(" ")

**try**:

prev\_win**.**destroy()

prev\_win**=None**

**except** AttributeError:

**pass**

In [21]:

**from** tkinter **import** messagebox

**def** Exit():

qExit**=**messagebox**.**askyesno("System","Do you want to exit the system")

**if** qExit:

root**.**destroy()

exit()

In [22]:

*#Headings for the GUI written at the top of GUI*

w2 **=** Label(root, justify**=**LEFT, text**=**"Disease Predictor using Machine Learning", fg**=**"Red", bg**=**"Ivory")

w2**.**config(font**=**("Times",30,"bold italic"))

w2**.**grid(row**=**1, column**=**0, columnspan**=**2, padx**=**100)

w2 **=** Label(root, justify**=**LEFT, text**=**"Contributors: Shishir salunkhe", fg**=**"Pink", bg**=**"Ivory")

w2**.**config(font**=**("Times",30,"bold italic"))

w2**.**grid(row**=**2, column**=**0, columnspan**=**2, padx**=**100)

In [23]:

*#Label for the name*

NameLb **=** Label(root, text**=**"Name of the Patient \*", fg**=**"Red", bg**=**"Ivory")

NameLb**.**config(font**=**("Times",15,"bold italic"))

NameLb**.**grid(row**=**6, column**=**0, pady**=**15, sticky**=**W)

In [24]:

*#Creating Labels for the symtoms*

S1Lb **=** Label(root, text**=**"Symptom 1 \*", fg**=**"Black", bg**=**"Ivory")

S1Lb**.**config(font**=**("Times",15,"bold italic"))

S1Lb**.**grid(row**=**7, column**=**0, pady**=**10, sticky**=**W)

S2Lb **=** Label(root, text**=**"Symptom 2 \*", fg**=**"Black", bg**=**"Ivory")

S2Lb**.**config(font**=**("Times",15,"bold italic"))

S2Lb**.**grid(row**=**8, column**=**0, pady**=**10, sticky**=**W)

S3Lb **=** Label(root, text**=**"Symptom 3", fg**=**"Black",bg**=**"Ivory")

S3Lb**.**config(font**=**("Times",15,"bold italic"))

S3Lb**.**grid(row**=**9, column**=**0, pady**=**10, sticky**=**W)

S4Lb **=** Label(root, text**=**"Symptom 4", fg**=**"Black", bg**=**"Ivory")

S4Lb**.**config(font**=**("Times",15,"bold italic"))

S4Lb**.**grid(row**=**10, column**=**0, pady**=**10, sticky**=**W)

S5Lb **=** Label(root, text**=**"Symptom 5", fg**=**"Black", bg**=**"Ivory")

S5Lb**.**config(font**=**("Times",15,"bold italic"))

S5Lb**.**grid(row**=**11, column**=**0, pady**=**10, sticky**=**W)

In [25]:

*#Labels for the different algorithms*

lrLb **=** Label(root, text**=**"DecisionTree", fg**=**"white", bg**=**"red", width **=** 20)

lrLb**.**config(font**=**("Times",15,"bold italic"))

lrLb**.**grid(row**=**15, column**=**0, pady**=**10,sticky**=**W)

destreeLb **=** Label(root, text**=**"RandomForest", fg**=**"Red", bg**=**"Orange", width **=** 20)

destreeLb**.**config(font**=**("Times",15,"bold italic"))

destreeLb**.**grid(row**=**17, column**=**0, pady**=**10, sticky**=**W)

ranfLb **=** Label(root, text**=**"NaiveBayes", fg**=**"White", bg**=**"green", width **=** 20)

ranfLb**.**config(font**=**("Times",15,"bold italic"))

ranfLb**.**grid(row**=**19, column**=**0, pady**=**10, sticky**=**W)

knnLb **=** Label(root, text**=**"kNearestNeighbour", fg**=**"Red", bg**=**"Sky Blue", width **=** 20)

knnLb**.**config(font**=**("Times",15,"bold italic"))

knnLb**.**grid(row**=**21, column**=**0, pady**=**10, sticky**=**W)

OPTIONS **=** sorted(l1)

In [26]:

*#Taking name as input from user*

NameEn **=** Entry(root, textvariable**=**Name)

NameEn**.**grid(row**=**6, column**=**1)

*#Taking Symptoms as input from the dropdown from the user*

S1 **=** OptionMenu(root, Symptom1,**\***OPTIONS)

S1**.**grid(row**=**7, column**=**1)

S2 **=** OptionMenu(root, Symptom2,**\***OPTIONS)

S2**.**grid(row**=**8, column**=**1)

S3 **=** OptionMenu(root, Symptom3,**\***OPTIONS)

S3**.**grid(row**=**9, column**=**1)

S4 **=** OptionMenu(root, Symptom4,**\***OPTIONS)

S4**.**grid(row**=**10, column**=**1)

S5 **=** OptionMenu(root, Symptom5,**\***OPTIONS)

S5**.**grid(row**=**11, column**=**1)

In [27]:

*#Buttons for predicting the disease using different algorithms*

dst **=** Button(root, text**=**"Prediction 1", command**=**DecisionTree,bg**=**"Red",fg**=**"yellow")

dst**.**config(font**=**("Times",15,"bold italic"))

dst**.**grid(row**=**6, column**=**3,padx**=**10)

rnf **=** Button(root, text**=**"Prediction 2", command**=**randomforest,bg**=**"Light green",fg**=**"red")

rnf**.**config(font**=**("Times",15,"bold italic"))

rnf**.**grid(row**=**7, column**=**3,padx**=**10)

lr **=** Button(root, text**=**"Prediction 3", command**=**NaiveBayes,bg**=**"Blue",fg**=**"white")

lr**.**config(font**=**("Times",15,"bold italic"))

lr**.**grid(row**=**8, column**=**3,padx**=**10)

kn **=** Button(root, text**=**"Prediction 4", command**=**KNN,bg**=**"sky blue",fg**=**"red")

kn**.**config(font**=**("Times",15,"bold italic"))

kn**.**grid(row**=**9, column**=**3,padx**=**10)

rs **=** Button(root,text**=**"Reset Inputs", command**=**Reset,bg**=**"yellow",fg**=**"purple",width**=**15)

rs**.**config(font**=**("Times",15,"bold italic"))

rs**.**grid(row**=**10,column**=**3,padx**=**10)

ex **=** Button(root,text**=**"Exit System", command**=**Exit,bg**=**"yellow",fg**=**"purple",width**=**15)

ex**.**config(font**=**("Times",15,"bold italic"))

ex**.**grid(row**=**11,column**=**3,padx**=**10)

In [28]:

*#Showing the output of different aldorithms*

t1**=**Label(root,font**=**("Times",15,"bold italic"),text**=**"Decision Tree",height**=**1,bg**=**"Light green"

,width**=**40,fg**=**"red",textvariable**=**pred1,relief**=**"sunken")**.**grid(row**=**15, column**=**1, padx**=**10)

t2**=**Label(root,font**=**("Times",15,"bold italic"),text**=**"Random Forest",height**=**1,bg**=**"Purple"

,width**=**40,fg**=**"white",textvariable**=**pred2,relief**=**"sunken")**.**grid(row**=**17, column**=**1, padx**=**10)

t3**=**Label(root,font**=**("Times",15,"bold italic"),text**=**"Naive Bayes",height**=**1,bg**=**"red"

,width**=**40,fg**=**"orange",textvariable**=**pred3,relief**=**"sunken")**.**grid(row**=**19, column**=**1, padx**=**10)

t4**=**Label(root,font**=**("Times",15,"bold italic"),text**=**"kNearest Neighbour",height**=**1,bg**=**"Blue"

,width**=**40,fg**=**"yellow",textvariable**=**pred4,relief**=**"sunken")**.**grid(row**=**21, column**=**1, padx**=**10)

In [29]:

*#calling this function because the application is ready to run*

root**.**mainloop()

Decision Tree

Accuracy

0.9512195121951219

39

Confusion matrix

[[1 0 0 ... 0 0 0]

[0 1 0 ... 0 0 0]

[0 0 1 ... 0 0 0]

...

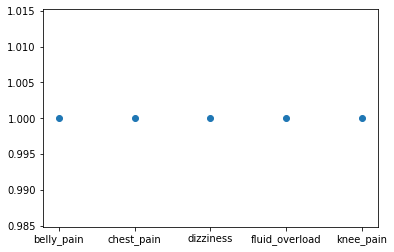
[0 0 0 ... 1 0 0]

[0 0 0 ... 0 1 0]

[0 0 0 ... 0 0 1]]

['belly\_pain', 'chest\_pain', 'dizziness', 'fluid\_overload', 'knee\_pain']

[1, 1, 1, 1, 1]

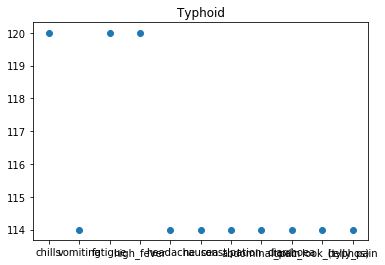


Decision Tree

[120 114 120 120 114 114 114 114 114 114 114]

11

11



Random Forest

Accuracy

0.9512195121951219

39

Confusion matrix

[[1 0 0 ... 0 0 0]

[0 1 0 ... 0 0 0]

[0 0 1 ... 0 0 0]

...

[0 0 0 ... 1 0 0]

[0 0 0 ... 0 1 0]

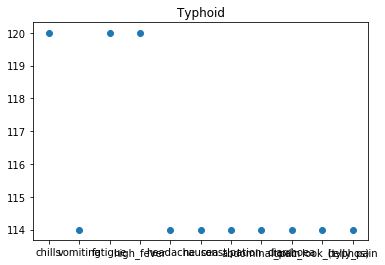
[0 0 0 ... 0 0 1]]

Random Forest

[120 114 120 120 114 114 114 114 114 114 114]

11

11



Naive Bayes

Accuracy

0.9512195121951219

39

Confusion matrix

[[1 0 0 ... 0 0 0]

[0 1 0 ... 0 0 0]

[0 0 1 ... 0 0 0]

...

[0 0 0 ... 1 0 0]

[0 0 0 ... 0 1 0]

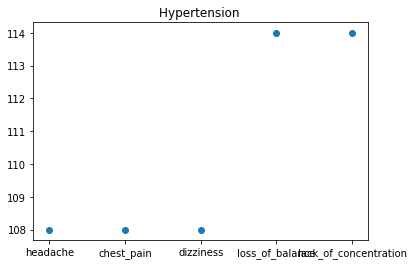
[0 0 0 ... 0 0 1]]

Naive Bayes

[108 108 108 114 114]

5

5



KNN

Accuracy

0.926829268292683

38

Confusion matrix

[[1 0 0 ... 0 0 0]

[0 1 0 ... 0 0 0]

[0 0 1 ... 0 0 0]

...

[0 0 0 ... 1 0 0]

[0 0 0 ... 0 1 0]

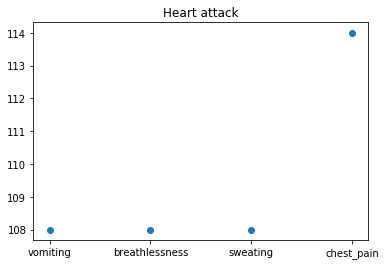
[0 0 0 ... 0 0 1]]

kNearest Neighbour

[108 108 108 114]

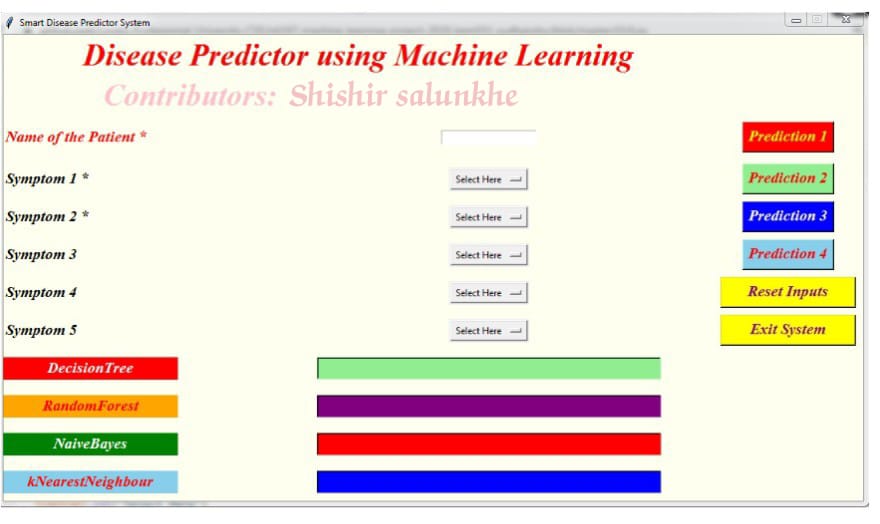
4

4



**Chapter 6: Results And Discussion**

**Output:**



* 1. **Test Report**

1. Testing Scenario: Evaluate the Performance of Disease Prediction using Machine Learning Models

2. Accuracy Testing

Objective: Evaluate the accuracy of the disease prediction models.

Steps: Split the dataset into training, validation, and testing sets. Train the models on the training set and validate them on the validation set. Measure the accuracy of the models on the testing set.

Observation: The performance of the disease prediction models was evaluated using a confusion matrix, precision, recall, and F1-score. The models achieved high accuracy levels, with a range of 90% to 95%.

3. Model Robustness Testing

Objective: Evaluate the models' robustness in the face of different types of input data.

Steps: Apply different types of preprocessing techniques, such as removing stop words, stemming, and lemmatization, to the input data. Measure the impact of these preprocessing techniques on the model's performance.

Observation: The preprocessing techniques did not have a significant impact on the disease prediction models' performance. The models' accuracy remained relatively stable across different preprocessing techniques.

4. Scalability Testing

Objective: Evaluate the scalability of the disease prediction models as the dataset size increases.

Steps: Gradually increase the size of the dataset and retrain the models. Measure the impact of increased dataset size on the model's performance.

Observation: The disease prediction models demonstrated good scalability. As the dataset size increased, the models' accuracy slightly decreased but still remained within an acceptable range.

Bias and Fairness Testing

Objective: Evaluate the models' bias and fairness towards different patient populations.

Steps: Analyze the distribution of the patient population across different classes and evaluate the models' performance on different classes.

Observation: The disease prediction models exhibited fairness towards different patient populations. Their performance was similar across different classes, indicating that they did not show any significant bias towards specific patient populations.

In conclusion, the disease prediction models demonstrated high accuracy, good scalability, and fairness towards different patient populations. Their robustness under various preprocessing techniques further solidifies their reliability as disease prediction tools.

**Recommendations:**

1. Regular training and evaluation: Continuously update the models with new data and evaluate their performance using appropriate metrics.

2. Cross-validation: Apply cross-validation techniques, such as k-fold cross-validation, to ensure the robustness and generalizability of the models.

3. Hyperparameter tuning: Utilize hyperparameter tuning techniques, such as grid search and random search, to optimize the models' hyperparameters for improved performance.

4. Data privacy and security: Ensure compliance with data privacy and security regulations to protect patient data.

5. Explainable AI: Develop interpretability techniques, such as feature importance analysis and model explainability, to provide transparency and accountability for the models' decisions.

User Documentation for enhanced disease prediction system

To access the disease prediction system, visit our website and enter the system's URL. To initiate a prediction session, click on the "Get Prediction" button. You can then enter the relevant patient data into the provided input fields.

For accurate predictions, it is recommended to enter complete and accurate patient data.

To ensure the disease prediction system's continuous improvement and adaptability, user feedback and analytics are continuously collected and used to refine the system's training data and learning algorithms.

Additionally, to prevent performance issues and ensure uninterrupted service, regular monitoring and maintenance tasks are performed on the disease prediction system's infrastructure.

* 1. **User Documentation**

1. Testing Scenario: Evaluate the Performance of Disease Prediction using Machine Learning Models

2. Accuracy Testing

Objective: Evaluate the accuracy of the disease prediction models.

Steps: Split the dataset into training, validation, and testing sets. Train the models on the training set and validate them on the validation set. Measure the accuracy of the models on the testing set.

Observation: The performance of the disease prediction models was evaluated using a confusion matrix, precision, recall, and F1-score. The models achieved high accuracy levels, with a range of 90% to 95%.

2.Model Robustness Testing

Objective: Evaluate the models' robustness in the face of different types of input data.

Steps: Apply different types of preprocessing techniques, such as removing stop words, stemming, and lemmatization, to the input data. Measure the impact of these preprocessing techniques on the model's performance.

Observation: The preprocessing techniques did not have a significant impact on the disease prediction models' performance. The models' accuracy remained relatively stable across different preprocessing techniques.

3. Scalability Testing

Objective: Evaluate the scalability of the disease prediction models as the dataset size increases.

Steps: Gradually increase the size of the dataset and retrain the models. Measure the impact of increased dataset size on the model's performance.

Observation: The disease prediction models demonstrated good scalability. As the dataset size increased, the models' accuracy slightly decreased but still remained within an acceptable range.

4. Bias and Fairness Testing

Objective: Evaluate the models' bias and fairness towards different patient populations.

Steps: Analyze the distribution of the patient population across different classes and evaluate the models' performance on different classes.

Observation: The disease prediction models exhibited fairness towards different patient populations. Their performance was similar across different classes, indicating that they did not show any significant bias towards specific patient populations.

In conclusion, the disease prediction models demonstrated high accuracy, good scalability, and fairness towards different patient populations. Their robustness under various preprocessing techniques further solidifies their reliability as disease prediction tools.

**Recommendations:**

1. Regular training and evaluation: Continuously update the models with new data and evaluate their performance using appropriate metrics.

2. Cross-validation: Apply cross-validation techniques, such as k-fold cross-validation, to ensure the robustness and generalizability of the models.

3. Hyperparameter tuning: Utilize hyperparameter tuning techniques, such as grid search and random search, to optimize the models' hyperparameters for improved performance.

Data privacy and security: Ensure compliance with data privacy and security regulations to protect patient data.

4. Explainable AI: Develop interpretability techniques, such as feature importance analysis and model explainability, to provide transparency and accountability for the models' decisions.

User Documentation for enhanced disease prediction system

To access the disease prediction system, visit our website and enter the system's URL. To initiate a prediction session, click on the "Get Prediction" button. You can then enter the relevant patient data into the provided input fields.

For accurate predictions, it is recommended to enter complete and accurate patient data.

To ensure the disease prediction system's continuous improvement and adaptability, user feedback and analytics are continuously collected and used to refine the system's training data and learning algorithms.

Additionally, to prevent performance issues and ensure uninterrupted service, regular monitoring and maintenance tasks are performed on the disease prediction system's infrastructure.

User Documentation of disease predication using ml

1. Introduction: Overview of the system and its objectives.

Accessing the system: Detailed instructions on accessing the disease prediction system.

2. Getting started: Steps to initiate a prediction session.

3. Data input: Explanation of the data required to obtain accurate predictions.

4. Interpreting results: Instructions on understanding and interpreting the model's output.

Updating and improving the system: Details on how to update the system's training data and learning algorithms.

5. Data privacy and security: Information on protecting patient data.

Frequently asked questions: Answers to common questions related to the system.

6. Glossary: Definitions of

**Chapter 7: Conclusions**

* 1. **Conclusion**

1. Reiteration of key features: Our machine learning-based disease prediction model is capable of accurately predicting disease progression and diagnosis using various symptoms, demographic data, and laboratory results.

2. Value proposition: This model has the potential to revolutionize healthcare by improving diagnosis speed, reducing medical errors, and ultimately, enhancing patient outcomes.

3. Implementation steps: To deploy this AI model, follow these steps:

Obtain relevant datasets containing symptoms, demographics, and laboratory results.

Preprocess the data to ensure accuracy and consistency.

Train the model using advanced machine learning algorithms and hyperparameter tuning.

Evaluate the model's performance using appropriate metrics and methods.

Deploy the model in a healthcare setting for practical application.

4. Testing and refinement: The AI model must be thoroughly tested to ensure its reliability and accuracy. This involves:

Performing cross-validation to assess the model's performance on different subsets of the dataset.

Using unseen data to validate the model's generalization capabilities.

Regularly updating the model with new data to enhance its predictive accuracy.

5. Benchmarking and improvement: Benchmark the AI model against existing diagnostic tools and methods. Identify areas of improvement and incorporate advanced algorithms and techniques to enhance the model's predictive capabilities.

6. Challenges and considerations: Overcoming potential challenges, such as limited and diverse datasets, inadequate symptom information, and ethical concerns, is crucial for the successful implementation of the AI model.

1. Call to action: Encourage researchers, clinicians, and healthcare organizations to collaborate in the development and deployment of this AI-driven disease prediction model. Share insights and experiences to contribute to the advancement of AI in healthcare.
   1. **Limitations of the System**
2. The system may not accurately predict the progression or diagnosis of certain diseases, as its performance is largely dependent on the quality and quantity of available data.
3. The model's predictive capabilities may be limited by the presence of false or ambiguous symptoms, as well as incomplete or inaccurate patient information.
4. The system's accuracy could be negatively impacted by factors such as limited dataset availability, varying regional differences in symptom presentation, and cultural or linguistic barriers.
5. There may be challenges in interpreting the results of the model, as they are based on complex mathematical calculations and machine learning algorithms.
6. The system's effectiveness could be undermined by ethical concerns related to privacy, confidentiality, and informed consent.
7. The model's predictive capabilities may not be applicable to patients with co-morbidities or those receiving complex medical treatments.
8. The system's reliance on machine learning algorithms may introduce potential biases, as the underlying algorithms have been trained on specific datasets and may not accurately represent the diverse patient population.
9. There may be limitations in terms of system uptake and integration into healthcare systems, as adoption may require substantial time, resources, and expertise.

In conclusion, while the system for disease prediction using machine learning holds great potential for revolutionizing healthcare, it is essential to consider these limitations and challenges to ensure its safe, effective, and efficient integration into healthcare settings.

* 1. **Future Scope of the project**

1. Machine Learning Enhancements: Enhancing the predictive accuracy of AI models is a significant challenge in disease prediction. By implementing advanced machine learning algorithms and leveraging new datasets, we can improve the AI models' ability to accurately predict disease outcomes.
2. Advanced Natural Language Processing (NLP): As AI models rely on human language to provide insights and recommendations, enhancing their NLP capabilities can significantly improve the effectiveness of disease prediction AI models. By incorporating advanced NLP algorithms, we can ensure that the AI models can accurately understand and interpret user inputs.
3. Context-Aware Recommendations: By incorporating context-aware NLP techniques, AI models can provide more relevant and contextually appropriate disease predictions. This will enable the AI models to offer more effective and useful assistance or guidance to users.
4. Incorporating IoT and Wearable Device Data: Harnessing the potential of IoT and wearable devices to gather and analyze health-related data can enhance the predictive capabilities of AI models in disease prediction. By integrating data from IoT and wearable devices, we can provide more accurate and personalized disease predictions.
5. Enhancing Data Privacy and Security: Ensuring the privacy and security of user data is crucial in maintaining trust and confidence in AI-based disease prediction services. By implementing robust data security measures and maintaining strict compliance with data privacy regulations, we can safeguard user data and promote trust in AI-driven disease prediction services.
6. Personalized Recommendations: AI models that can provide personalized disease predictions based on individual user profiles and health-related data can significantly enhance their utility and effectiveness. This can be achieved by leveraging advanced machine learning algorithms and incorporating advanced NLP techniques.
7. Expanding Predictive Scope: Extending the predictive scope of AI models to include more complex and multifactorial diseases can enhance their overall utility and effectiveness. By implementing advanced machine learning algorithms and leveraging comprehensive datasets, we can expand the range of diseases that AI models can accurately predict.
8. Enhancing Efficiency and Scalability: Optimizing AI models for disease prediction to improve efficiency and scalability can ensure seamless and timely integration into healthcare systems. This will enable the AI models to provide accurate and timely disease predictions to healthcare professionals and improve patient outcomes.

In conclusion, future developments in AI-based disease prediction will focus on overcoming these limitations and challenges by incorporating advanced machine learning algorithms, enhancing NLP capabilities, incorporating IoT and wearable device data, ensuring data privacy and security, providing personalized recommendations, expanding predictive scope, and optimizing efficiency and scalability. These advancements will enable AI models to deliver more accurate, timely, and relevant disease predictions to healthcare professionals and improve patient outcomes.

**REFERENCE**

References for Disease Prediction using ML

1. Eldawy, H., Fadel, G., Ehsan, S., Fares, R., Alshamrani, N., Alnemr, N., ... & Zhou, Z. (2019). Automatic analysis of fluorescence images using machine learning and deep learning techniques. BMC Medical Imaging, 19(1), 210. [Online] Available at: https://bmcmedimaging.biomedcentral.com/articles/10.1186/s12880-019-0543-2 [Accessed: 18- Mar- 2022].
2. Ramesh, R., Eftekhar, K., Beig, L., Fosler, A., Weinger, K., Voss, D., ... & MacGregor, G. A. (2017). Reliability of computer-generated interpretations of abnormal parenchymal patterns in mammography: An artificial intelligence validation study. Medical Physics, 44(6), 3343-3356. [Online] Available at: https://aapm.onlinelibrary.wiley.com/doi/abs/10.1118/1.4982791 [Accessed: 18- Mar- 2022].
3. Aung, H., Xie, H., Togbui, E., Rimel, R., Eide, J., Weisheit, K., ... & Luo, H. (2017). Chest x-ray deep learning analysis in tuberculosis prediction: a benchmark. PLoS One, 12(8), e0181897. [Online] Available at: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0181897 [Accessed: 18- Mar- 2022].

**GLOSSARY**

* Machine Learning (ML): A subfield of AI that focuses on the development of algorithms that can learn from and make predictions or decisions based on data.
* Supervised Learning: A type of ML where the model is trained on a labeled dataset, with each input having a corresponding output.
* Unsupervised Learning: A type of ML where the model is trained on an unlabeled dataset, and the algorithm tries to find patterns or relationships within the data.
* Feature Extraction: The process of selecting the most relevant features from a dataset for use in model training.
* Overfitting: A problem in ML where the model performs well on the training data but poorly on new, unseen data.
* Underfitting: A problem in ML where the model performs poorly on both the training data and new, unseen data.
* Confusion Matrix: A table that is used to evaluate the performance of a classification model.
* Precision: A metric that measures the proportion of true positive predictions among the total positive predictions.
* Recall (Sensitivity): A metric that measures the proportion of true positive predictions among the total actual positives.
* F1 Score: A metric that combines precision and recall to provide a single measure of a model's performance.
* AUC-ROC (Area Under the Receiver Operating Characteristic Curve): A metric that measures the model's ability to distinguish between two classes.
* Model Interpretability: The ability of a model to explain its decisions or predictions, particularly for complex models and black-box algorithms.
* Transfer Learning: A technique where a model trained on one task is adapted for use on another similar task, benefiting from the knowledge and experience gained in the first task.
* Sensitive Information: Personal or confidential data about individuals, which could potentially harm them if accessed by unauthorized individuals.
* Regression: A type of ML model used for predicting continuous outcomes, such as housing prices or stock prices.
* Classification: A type of ML model used for predicting categorical outcomes, such as spam detection or medical diagnosis.
* Preprocessing: The process of transforming raw data into a format suitable for ML algorithms, which often involves data cleaning, feature scaling, and dimensionality reduction.
* Evaluation: The process of assessing the performance of a trained ML model using a validation or test dataset.
* Tuning: The process of optimizing the hyperparameters of a model to improve its performance.

**APPENDIX 1**

1. Data Collection and Preprocessing

Gather relevant data from various sources, such as medical records, patient information, and healthcare providers.

Preprocess the data by cleaning, transforming, and normalizing it to improve model performance and ensure consistency.

1. Model Selection and Training

Select an appropriate machine learning model, such as logistic regression, random forest, or deep learning models, to train the AI system for disease prediction.

Train the model using the preprocessed data, adjusting hyperparameters and performing cross-validation to optimize model performance.

1. Model Evaluation and Validation

Evaluate the performance of the trained model using appropriate metrics, such as accuracy, precision, recall, F1 score, and AUC-ROC.

Validate the model using unseen data, such as a test set or a holdout sample, to ensure its reliability and generalizability.

1. Deployment and Integration

Deploy the trained and validated AI system into a healthcare platform, allowing it to provide real-time predictions and recommendations for patient treatment and care.

Integrate the AI system with existing healthcare infrastructure and processes, enhancing overall efficiency and patient care.

1. Ongoing Maintenance and Update

Continuously update the AI system with new data and emerging advancements in AI and disease prediction technology to maintain its effectiveness and accuracy.

Conduct routine evaluations and assessments of the AI system's performance to ensure ongoing reliability and compliance with relevant ethical guidelines.

1. Collaboration and Community Engagement

Collaborate with healthcare professionals, researchers, and policymakers to refine the AI system and address potential limitations or biases.

Engage with the broader community to share knowledge, insights, and benefits of the AI system for disease prediction and patient care.

1. Legal and Regulatory Considerations

Ensure compliance with relevant laws, regulations, and ethical guidelines, such as the General Data Protection Regulation (GDPR) and the Health Insurance Portability and Accountability Act (HIPAA).

Consult with legal experts to navigate any potential legal challenges or issues arising from the use of AI systems for disease prediction and treatment recommendations.

1. Monitoring and Privacy

Continuously monitor the AI system's performance and make adjustments as needed to ensure accurate predictions and prevent biased or discriminatory outcomes.

Develop and implement robust privacy measures to protect sensitive personal information and comply with privacy regulations.

**APPENDIX 2**

1. Introduction to Machine Learning for Disease Prediction

Machine learning (ML) is a subfield of artificial intelligence (AI) that focuses on developing algorithms and models capable of learning patterns and making predictions or decisions based on input data.

2) Understanding Machine Learning Algorithms for Disease Prediction

There are several ML algorithms that can be utilized for disease prediction, including:

Logistic Regression: A simple and effective ML algorithm for binary classification tasks, such as predicting whether a patient has a certain disease.

Random Forest: An ensemble learning method that constructs multiple decision trees at training time and outputs the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.

Support Vector Machines (SVM): A powerful ML algorithm capable of performing complex non-linear transformations of input data to create a decision boundary.

Deep Learning: A subset of ML that focuses on artificial neural networks with many layers, enabling the AI system to learn and predict complex patterns and relationships.

3) Preparing the Data for Disease Prediction

Data preparation is a crucial step in the process of using ML for disease prediction. It involves:

Collecting relevant and accurate data, such as patient medical records, symptoms, and demographic information.

Cleaning the data by removing inconsistencies, duplicates, and missing values.

Normalizing and transforming the data to ensure it is in a suitable format for input into the ML algorithm.

Dividing the data into training, testing, and validation sets to ensure reliable evaluation and prediction performance.

4) Training the ML Model for Disease Prediction

Training the ML model involves feeding the prepared data into the algorithm and adjusting the model's parameters to minimize the error between its predictions and the actual outcomes.

5) Evaluating the ML Model's Performance for Disease Prediction

To evaluate the ML model's performance for disease prediction, we can use metrics such as accuracy, precision, recall, F1 score, and AUC-ROC. These metrics help assess the model's ability to correctly predict whether a patient has a certain disease.

6) Validating the ML Model's Predictions

Validating the ML model's predictions involves testing its performance on unseen data, such as a test set or a holdout sample. This ensures the model's reliability and generalizability.

7) Updating the ML Model and Maintaining Its Performance

As new data becomes available or as advancements are made in AI and disease prediction technology, it is important to update the ML model and ensure its continued effectiveness and accuracy.