

PROFESSIONAL TRAINING REPORT
at
Sathyabama Institute of Science and Technology
(Deemed to be University)

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering Degree in Computer Science and Engineering

By

Deepa Jothi K
REG. NO. 38119005



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING
SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119, TAMILNADU

March 2022



SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade "A" by NAAC
(Established under Section 3 of UGC Act, 1956)
JEPPIAAR NAGAR, RAJIV GANDHI SALAI
CHENNAI- 600119
www.sathyabamauniversity.ac.in**



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Deepa Jothi K (Reg. No: 38119005)** who carried out the project entitled "**Flawless Multi Perspective Vision For Prediction Of Disease Using Machine Learning**" under my supervision from June 2021 to November 2021.

Internal Guide

Dr. R.M.Gomathi, M.Tech., Ph.D.,

Head of the Department

**Dr. L. LAKSHMANAN M.E., Ph.D.,
Dr.S.VIGNESHWARI, M.E., Ph.D.,**

Submitted for Viva voce Examination held on

Internal Examiner

External Examiner

DECLARATION

I Deepa Jothi K hereby declare that the Project Report entitled **Flawless Multi Perspective Vision For Prediction Of Disease Using Machine Learning** done by me under the guidance of **Dr. R.M.Gomathi, M.Tech., Ph.D., and Dr.L.LAKSHMANAN M.E., Ph.D., and Dr.S.VIGNESHWARI M.E.,Ph.D.,** at **Sathyabama Institute of Science and Technology** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE:

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph.D., Dean**, School of Computing , **and Dr.L.Lakshmanan M.E., Ph.D., Dr.S.Vigneshwari M.E., Ph.D.**, Heads of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr.R.M.Gomathi M.Tech., Ph.D.**, for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

This framework predicts the disease based on the inputs provided by the patients in the form of symptoms. It gives an idea of predicting multiple disease using machine learning algorithms. It is very useful for predictive analysis in healthcare. Early prediction of disease helps in best diagnosis. Over the years the use of disease prediction tools along with concerning the health of the patient has been increased due to a variety of diseases and less doctor patient interaction. The aim of developing this model is to help to solve the health related issues by assisting the physicians to predict and diagnose the disease at an early stage. Machine learning algorithms like random forest, decision tree classifiers, support vector machine and naïve bayes classifier gives maximum accuracy in prediction purpose. This project shows the performance of these algorithms in predicting the disease based on symptoms.

Keywords— Disease prediction, Decision tree, Naïve bayes, Random forest, Support vector machine, Machine learning, Symptoms.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	Abstract	v
	List of figures	viii
	List of tables	ix
1	INTRODUCTION	1
	1.1 Introduction	1
	1.2 Introduction to Python and Machine learning	2
	1.2.1 Python	2
	1.2.2 Python Tkinter	3
	1.2.2 Machine learning	4
2	LITERATURE SURVEY	8
3	PROBLEM STATEMENT, PROJECT PURPOSE AND PROJECT FEATURES	11
	3.1 Project Statement	11
	3.2 Project Purpose	11
	3.3 Project Features	12
4	METHODOLOGY	13
	4.1 Existing System	13
	4.2 Proposed System	13
	4.3 System Architecture	14
	4.3.1 Dataset	14
	4.3.2 Data Preprocessing	14
	4.3.3 Machine Learning Algorithms	18
	4.3.3.1 Decision Tree	19
	4.3.3.2 Random Forest	22
	4.3.3.3 Naïve Bayes	24

	4.3.3.4 Support Vector Machine	25
5	REQUIREMENT ANALYSIS	30
	5.1 Functional Requirement	30
	5.2 Non Functional Requirement	30
	5.3 Hardware Requirements	30
	5.4 Software Requirements	30
6	SYSTEM DESIGN	31
	6.1 Data Flow Diagram	31
	6.2 Use Case Diagram	32
	6.3 Class Diagram	33
	6.4 Sequence Diagram	34
	6.5 Collaboration Diagram	35
	6.6 Component Diagram	36
7	TESTING	37
	7.1 Types of Testing	37
	7.1.1 Unit Testing	37
	7.1.2 Integration Testing	37
	7.1.3 Functional Testing	37
	7.1.4 Validation Testing	38
	7.1.3 Acceptance Testing	38
	7.2 Test Report	38
8	CONCLUSION AND FUTURE WORK	40
	8.1 Conclusion	40
	8.2 Future Work	40
9	APPENDIX	41
	9.1 Source Code	41
	9.2 Screenshots	62
10	REFERENCES	65

LIST OF FIGURES

FIGURE NO.	FIGURE NAME	PAGENO.
1	Architecture Diagram	14
2	Data Preprocessing	15
3	Splitting of dataset	18
4	Structure of Decision Tree	20
5	Working of Random Forest	23
6	Boundaries of SVM	26
7	Data Flow Diagram	31
8	Use Case Diagram	32
9	Class Diagram	33
10	Sequence Diagram	34
11	Collaboration Diagram	35
12	Component Diagram	36

LIST OF TABLES

TABLE NO	TABLE NAME	PAGE NO
1	Tkinter Widgets	3
2	Test Report	38
3	Test Report	

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

Disease Prediction using Machine Learning is a system which predicts the disease based on the information provided by the user. It also predicts the disease of the patient or the user based on the information or the symptoms he/she enter into the system and provides the accurate results based on that information. If the patient is not much serious and the user just wants to know the type of disease, he/she has been through. It is a system which provides the user the tips and tricks to maintain the health system of the user and it provides a way to find out the disease using this prediction. Now a day's health industry plays major role in curing the diseases of the patients so this is also some kind of help for the health industry to tell the user and also it is useful for the user in case he/she doesn't want to go to the hospital or any other clinics, so just by entering the symptoms and all other useful information the user can get to know the disease he/she is suffering from and the health industry can also get benefit from this system by just asking the symptoms from the user and entering in the system and in just few seconds they can tell the exact and up to some extent the accurate diseases. This DPURL is previously done by many other organizations but our intention is to make it different and beneficial for the users who are using this system. This Disease Prediction Using Machine Learning is completely done with the help of Machine Learning and Python Programming language with Tkinter Interface for it and also using the dataset that is available previously by the hospitals using that we will predict the disease. Now a day's doctors are adopting many scientific technologies and methodology for both identification and diagnosing not only common disease, but also many fatal diseases. The successful treatment is always attributed by right and accurate diagnosis. Doctors may sometimes fail to take accurate decisions while diagnosing the disease of a patient, therefore disease prediction systems which use machine learning algorithms assist in such cases to get accurate results. The project disease prediction using machine learning is developed to overcome general disease in earlier stages as we all know in competitive environment of economic development the mankind has involved so much that he/she is not concerned about health according to research there are 40% peoples how ignores about general disease which leads to harmful disease later. The main reason of ignorance is laziness to consult a doctor and time concern the peoples have involved themselves so much that they have no time to take an appointment and consult the doctor which later results into fatal disease.

According to research there are 70% peoples in India suffers from general disease and 25%of peoples face death due to early ignorance the main motive to develop[p this project is that a user can sit at their convenient place and have a check-up of their health the UI is designed in such a simple way that everyone can easily operate on it and can have a check-up.

1.2 INTRODUCTION TO PYTHON AND MACHINE LEARNING

1.2.1 PYTHON

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. It was created by Guido van Rossum during 1985- 1990. Like Perl, Python source code is also available under the GNU General Public License (GPL). This tutorial gives enough understanding on Python programming language. Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

Python is a MUST for students and working professionals to become a great Software Engineer especially when they are working in Web Development Domain. I will list down some of the key advantages of learning Python:

- Python is Interpreted – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- Python is Interactive – You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- Python is Object-Oriented – Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- Python is a Beginner's Language – Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

Following are important characteristics of Python Programming –

- It supports functional and structured programming methods as well as OOP.

- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

1.2.2 PYTHON TKINTER

Python has a lot of GUI frameworks, but Tkinter is the only framework that's built into the Python standard library. Tkinter has several strengths. It's **cross-platform**, so the same code works on Windows, macOS, and Linux. Visual elements are rendered using native operating system elements, so applications built with Tkinter look like they belong on the platform where they're run. Although Tkinter is considered the de-facto Python GUI framework, it's not without criticism. One notable criticism is that GUIs built with Tkinter look outdated. If you want a shiny, modern interface, then Tkinter may not be what you're looking for. However, Tkinter is lightweight and relatively painless to use compared to other frameworks. This makes it a compelling choice for building GUI applications in Python, especially for applications where a modern sheen is unnecessary, and the top priority is to build something that's functional and cross-platform quickly.

To create a tkinter app:

- Importing the module – tkinter
- Create the main window (container)
- Add any number of widgets to the main window
- Apply the event Trigger on the widgets.

There are various widgets like button, canvas, checkbutton, entry, etc. that are used to build the python GUI applications.

Table 1 : TKinter widgets

S N	Widget	Description
1	Button	The Button is used to add various kinds of buttons to the python application.
2	Canvas	The canvas widget is used to draw the canvas on the window.
3	Checkbutton	The Checkbutton is used to display the CheckButton on the window.

4	Entry	The entry widget is used to display the single-line text field to the user. It is commonly used to accept user values.
5	Frame	It can be defined as a container to which, another widget can be added and organized.
6	Label	A label is a text used to display some message or information about the other widgets.
7	ListBox	The ListBox widget is used to display a list of options to the user.
8	Menubutton	The Menubutton is used to display the menu items to the user.
9	Menu	It is used to add menu items to the user.
10	Message	The Message widget is used to display the message-box to the user.
11	Radiobutton	The Radiobutton is different from a checkbutton. Here, the user is provided with various options and the user can select only one option among them.
12	Scale	It is used to provide the slider to the user.
13	Scrollbar	It provides the scrollbar to the user so that the user can scroll the window up and down.
14	Text	It is different from Entry because it provides a multi-line text field to the user so that the user can write the text and edit the text inside it.
14	Toplevel	It is used to create a separate window container.
15	Spinbox	It is an entry widget used to select from options of values.
16	PanedWindow	It is like a container widget that contains horizontal or vertical panes.
17	LabelFrame	A LabelFrame is a container widget that acts as the container
18	MessageBox	This module is used to display the message-box in the desktop based applications.

1.2.3 MACHINE LEARNING

The term Machine Learning was coined by Arthur Samuel in 1959, an American pioneer in the field of computer gaming and artificial intelligence, and stated that “it gives computers the ability to learn without being explicitly programmed”. And in 1997, Tom Mitchell gave a “well-posed” mathematical and relational definition that “A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E.

Machine Learning is the latest buzzword floating around. It deserves to, as it is one of the most interesting subfields of Computer Science. So what does Machine Learning really mean?

Let's try to understand Machine Learning in layman's terms. Consider you are trying to toss a paper into a dustbin.

After the first attempt, you realize that you have put too much force into it. After the second attempt, you realize you are closer to the target but you need to increase your throw angle. What is happening here is basically after every throw we are learning something and improving the end result. We are programmed to learn from our experience.

This implies that the tasks in which machine learning is concerned to offer a fundamentally operational definition rather than defining the field in cognitive terms. This follows Alan Turing's proposal in his paper "Computing Machinery and Intelligence", in which the question "Can machines think?" is replaced with the question "Can machines do what we (as thinking entities) can do?" Within the field of data analytics, machine learning is used to devise complex models and algorithms that lend themselves to prediction; in commercial use, this is known as predictive analytics. These analytical models allow researchers, data scientists, engineers, and analysts to "produce reliable, repeatable decisions and results" and uncover "hidden insights" through learning from historical relationships and trends in the data set(input).

Suppose that you decide to check out that offer for a vacation. You browse through the travel agency website and search for a hotel. When you look at a specific hotel, just below the hotel description there is a section titled "You might also like these hotels". This is a common use case of Machine Learning called "Recommendation Engine". Again, many data points were used to train a model in order to predict what will be the best hotels to show you under that section, based on a lot of information they already know about you. So if you want your program to predict, for example, traffic patterns at a busy intersection (task T), you can run it through a machine learning algorithm with data about past traffic patterns (experience E) and, if it has successfully "learned", it will then do better at predicting future traffic patterns (performance measure P). The highly complex nature of many real-world problems, though, often means that inventing specialized algorithms that will solve them perfectly every time is impractical, if not impossible. Examples of machine learning problems include, "Is this cancer?", "Which of these people are good friends with each other?", "Will this person like this movie?" such problems are excellent targets for Machine Learning, and in fact, machine learning has been applied to such problems with great success.

Machine learning implementations are classified into three major categories, depending on the nature of the learning "signal" or "response" available to a learning system which is as

follows:-

- **Supervised learning:** When an algorithm learns from example data and associated target responses that can consist of numeric values or string labels, such as classes or tags, in order to later predict the correct response when posed with new examples comes under the category of Supervised learning. This approach is indeed similar to human learning under the supervision of a teacher. The teacher provides good examples for the student to memorize, and the student then derives general rules from these specific examples.
- **Unsupervised learning:** Whereas when an algorithm learns from plain examples without any associated response, leaving to the algorithm to determine the data patterns on its own. This type of algorithm tends to restructure the data into something else, such as new features that may represent a class or a new series of un-correlated values. They are quite useful in providing humans with insights into the meaning of data and new useful inputs to supervised machine learning algorithms.

As a kind of learning, it resembles the methods humans use to figure out that certain objects or events are from the same class, such as by observing the degree of similarity between objects. Some recommendation systems that you find on the web in the form of marketing automation are based on this type of learning.

- **Reinforcement learning:** When you present the algorithm with examples that lack labels, as in unsupervised learning. However, you can accompany an example with positive or negative feedback according to the solution the algorithm proposes comes under the category of Reinforcement learning, which is connected to applications for which the algorithm must make decisions (so the product is prescriptive, not just descriptive, as in unsupervised learning), and the decisions bear consequences. In the human world, it is just like learning by trial and error.

Errors help you learn because they have a penalty added (cost, loss of time, regret, pain, and so on), teaching you that a certain course of action is less likely to succeed than others. An interesting example of reinforcement learning occurs when computers learn to play video games by themselves.

In this case, an application presents the algorithm with examples of specific situations, such as having the gamer stuck in a maze while avoiding an enemy. The application lets the algorithm know the outcome of actions it takes, and learning occurs while trying to avoid what it discovers to be dangerous and to pursue

survival. You can have a look at how the company Google DeepMind has created a reinforcement learning program that plays old Atari's video games. When watching the video, notice how the program is initially clumsy and unskilled but steadily improves with training until it becomes a champion.

- **Semi-supervised learning:** where an incomplete training signal is given: a training set with some (often many) of the target outputs missing. There is a special case of this principle known as Transduction where the entire set of problem instances is known at learning time, except that part of the targets are missing.

Another categorization of machine learning tasks arises when one considers the desired output of a machine-learned system:

- **Classification:** When inputs are divided into two or more classes, and the learner must produce a model that assigns unseen inputs to one or more (multi-label classification) of these classes. This is typically tackled in a supervised way. Spam filtering is an example of classification, where the inputs are email (or other) messages and the classes are “spam” and “not spam”.
- **Regression:** Which is also a supervised problem, A case when the outputs are continuous rather than discrete.
- **Clustering:** When a set of inputs is to be divided into groups. Unlike in classification, the groups are not known beforehand, making this typically an unsupervised task.

CHAPTER 2

2. LITERATURE SURVEY

Sehrish aqueel's [1] report depicts the arranged tests in medication. Investigation of clinical data is viewed as a transformation apparatus to make the treatment interaction more proficient and powerful. It centers around the investigation of how medical clinics settle on top to bottom choices in an investigation of an assortment of inside and out preparing procedures and practice instruments. as a learning apparatus.

Arwinder Dhillon and Ashima Singh's [2] motivation behind this article is to sum up the underlying examination and inspect the hindrances, uncontrolled learning techniques, and

issues that lead to inside and out investigation of medication. For AI to be a simple apparatus in our day to day routines, we want to follow the standards of uncontrolled figuring out how to work on the presentation and exactness of the medical services framework.

Hasnuhana Mazumder and Sayantan Saha [3] proposed a web model for people of all ages to cure health related problems. Their new prototype model uses ID3 algorithm to train the dataset and identify the disease based on the allergies.

Min Chen [4] depicted a model to identify disease over massive data from various healthcare sources. It defines an efficient way for data analytics and introduced convolution neural network model for prediction of disease. It compares several algorithms and suggest the best accuracy results.

K.Gomathi and D.Shanmuga [5] fostered a prediction model utilizes data mining techniques which is well suited for disease prediction. It uses naïve bayes for classification purposes

Majd A. M. Alhaj and Ashraf Y. A. Maghari [6] inspects the existences of malignant growth patients utilizing straightforward memory calculations. The review was led utilizing the Rapid Miner device, which is utilized to perform stage by-stage estimations and to gauge their presentation while at the same time with the exactness of the model.

Mamatha Sai Yarabarla and Lakshmi Kavya Ravi [7] proposed an article is to decide whether an individual has bosom malignant growth with the assistance of the data gave. The fundamental thought is to involve the most recent advancements in the improvement of the CAD framework and related innovations.

Chinmai Tallam and Aarsha Peruboyina [8] Predicts cellular breakdown in the lungs and the data gave. It analyzes bunch models to various classes, for example, support for vector machines, KNNs, non-repeating memory, craftsmanship pipelines, and Hybrid models. The exhibition of the different strategies is looked at and assessed in actuality. Along these lines, it is not difficult to recognize patients with cellular breakdown in the lungs early utilizing an assortment of present day innovations.

Sneha Grampurohit and Chetan Sagarnal [9] suggests an outline of sickness mindfulness through understanding assortment. The exploration shows the sickness utilizing an assortment of AI calculations, like exceptional memory, guileless base

classifier, and tree certificate. This paper presents a review contrasting the consequences of the above calculation.

Prasan Kumar Saho, Suvendu Kumar Mohapatra [10] investigates life and future wellbeing. In this paper, a potential information assortment strategy has been produced for the examination of the gathered information. Mistaken models are expected for the accompanying life thoughts.

Roger Lee and ChinmaiChitnis [11] tells the best way to deal with your wellbeing early rather than after treatment. It looks at five forecast calculations. The outcomes currently show that the muscle network gives genuine data subsequent to analyzing the model.

AhelamTicoticar and Mallikarjun Kodabag [12] presents this exploration paper in an assortment of ways of separating data used to anticipate infection. Media mining has turned into a positive routine for decision-production through PC supported independent direction.

Y.Bengio [13] discuss the motivations and principles regarding learning algorithms for deep architectures and exploiting as building blocks unsupervised learning of single layer models such as restricted boltman machines, used to construct deep models like deep belief networks.

Qiulin Wang [14] investigates how to decide the heading of a lower prosthesis dependent on a CNN research calculation. A top to bottom investigation of programmed object acknowledgment and decompression calculations can further develop low-level clever interaction the board execution and identify changes in wise cycle structure in an assortment of ways.

Haijian Shi's [15] proposed paper inspects whether it is feasible to decide the legitimate planning of trees dependent on the genuine data set up to blend the enormous numbers that have been made dependent on the approval of the great tree development of the main choice.

CHAPTER 3

3. PROBLEM STATEMENT, PROJECT PURPOSE AND PROJECT FEATURES

3.1 PROBLEM STATEMENT

Now a day's in Health Industry there are various problems related to machines or devices which will give wrong or unaccepted results, so to avoid those results and get the correct and desired results we are building a program or project which will give the accurate predictions based on information provided by the user and also based on the datasets that are available in that machine. The health industry in information yet and knowledge poor and this industry is very vast industry which has lot of work to be done. So, with the help of all those algorithms, techniques and methodologies we have done this project which will help the peoples who are in the need. So the problem here is that many people goes to hospitals or clinic to know how is their health and how much they are improving in the given days, but they have to travel to get to know there answers and sometimes the

patients may or may not get the results based on various factors such as doctor might be on leave or some whether problem so he might not have come to the hospital and many more reasons will be there so to avoid all those reasons and confusion we are making a project which will help all those person's and all the patients who are in need to know the condition of their health, and at sometimes if the person has been observing few symptoms and he/she is not sure about the disease he/she is encountered with so this will lead to various diseases in future. So, to avoid that and get to know the disease in early stages of the symptoms this disease prediction will help a lot to the various people's ranging from children to teenagers to adults and also the senior citizens.

3.2. PROJECT PURPOSE

The purpose of making this project called "Disease Prediction Using Machine Learning" is to predict the accurate disease of the patient using all their general information's and also the symptoms. Using this information, there we will compare with our previous datasets of the patients and predicts the disease of the patient he/she is been through. If this Prediction is done at the early stages of the disease with the help of this project and all other necessary measure the disease can be cured and in general this prediction system can also be very useful in health industry. If health industry adopts this project then the work of the doctors can be reduced and they can easily predict the disease of the patient. The general purpose of this Disease prediction is to provide prediction for the various and generally occurring diseases that when unchecked and sometimes ignored can turns into fatal disease and cause lot of problem to the patient and as well as their family members. This system will predict the most possible disease based on the symptoms. The health industry in information yet and knowledge poor and this industry is very vast industry which has lot of work to be done. So, with the help of all those algorithms, techniques and methodologies we have done this project which will help the peoples who are in the need.

3.3. PROJECT FEATURES

The features of Disease Prediction Using Machine Learning are as follows.

- This Project will predict the diseases of the patients based on the symptoms and other general information using the datasets.
- This is done based on the previous datasets of the hospitals so after comparing it can provide up to 80% of accurate results, and the project is still developing further to get the 100% accurate results.
- With the help of Disease prediction, it can predict the disease of the patient and can solve various problems and prevents from various aspects.
- It provides security for the system so that no one can break into that and no one

can make any changes in the system.

- The disease is predicted using the algorithms and the user has to enter the symptoms from the given drop-down menu, in order to get correct accuracy, the user has to enter all the symptoms.
- Here we can easily prepare the data and transform that data into algorithm, which will reduce the overall work of the project.
- To make user more application friendly rather than discussing with others for their disease.
- It provides the necessary options to choose from the types and attributes

CHAPTER 4

4. METHODOLOGY

4.1. EXISTING SYSTEM

Hospital mostly uses a manual system for the management and maintenance of critical information. Prediction of disease using traditional methods involves various risk factors. Paper medical record has the problems such as storage and lack of back up, security issues and no clear audit trails and version history. But these models are only valuable in clinical situations and not in big industry sector. In this evolving world, the symptoms of diseases also evolved over a period of time. The user will have to wait for long periods to get the medical results in hospitals. So, by considering the patients time and health the proposed work utilizing AI ideas and learning techniques will surely prove to be an excellent method to expose the disease.

4.2. PROPOSED SYSTEM

The proposed system of disease prediction using machine learning is that we have used many techniques and algorithms and all other various tools to build a system which predicts the disease of the patient using the symptoms and by taking those symptoms we are comparing with the system's dataset that is previously available. By taking those datasets and comparing with the patient's disease we will predict the accurate percentage disease of the patient. The dataset and symptoms go to the prediction model of the system

where the data is pre-processed for the future references and then the feature selection is done by the user where he will enter the various symptoms. Then the classification of those data is done with the help of various algorithms and techniques such as Decision Tree, Support vector machine, Naïve Bayes, Random Forest and etc. Then the data goes in the recommendation model, there it shows the risk analysis that is involved in the system and it also provides the probability estimation of the system such that it shows the various probability like how the system behaves when there are n number of predictions are done and it also does the recommendations for the patients from their final result and also from their symptoms like it can show what to use and what not to use from the given datasets and the final results. Here we have combined the overall structure and unstructured form of data for the overall risk analysis that is required for doing the prediction of the disease. Using the structured analysis, we can identify the chronic types of disease in a particular region and particular community. In unstructured analysis we select the features automatically with the help of algorithms and techniques. This system takes symptoms from the user and predicts the disease accordingly based on the symptoms that it takes and also from the previous datasets, it also helps in continuous evaluation of viral diseases, heart rate, blood pressure, sugar level and much more which is in the system and along with other external symptoms its predicts the appropriate and accurate disease.

4.3 System Architecture

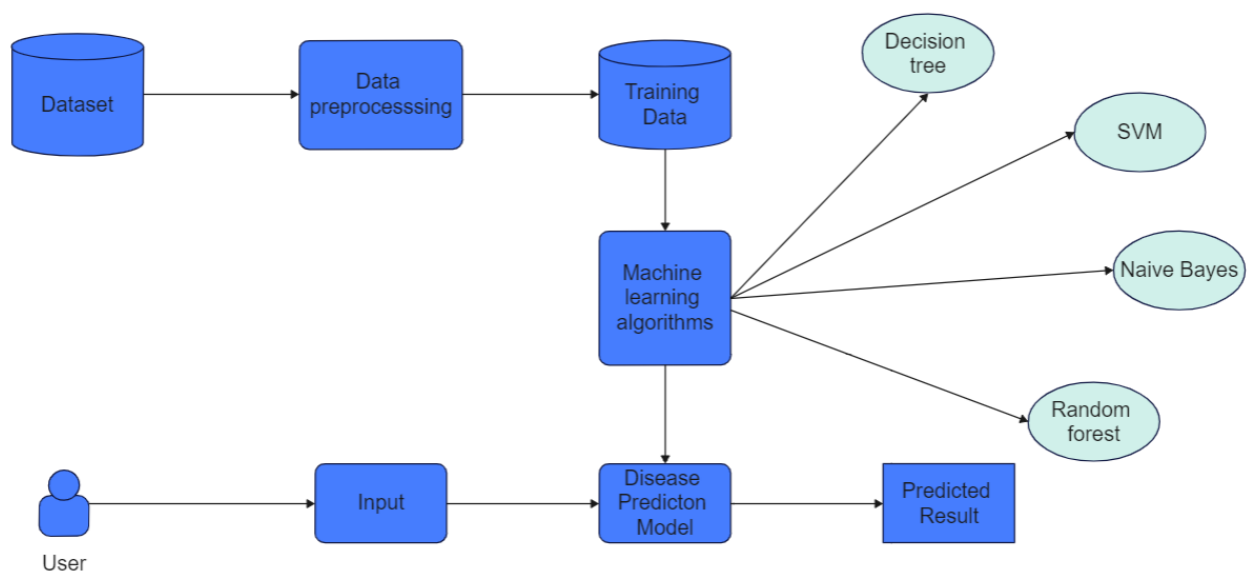


Fig 1 : Architecture Diagram

Modules:

- 1) Dataset Collection
- 2) Data Preprocessing
- 3) Applying Machine Learning Algorithms

4.3.1 Dataset

Dataset which is used for our project contains 132 kinds of symptoms which is for 41 types of diseases. It contains 4920 patient's record.

4.3.2 Data Preprocessing

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model. When creating a machine learning project, it is not always a case that we come across the clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put in a formatted way. So for this, we use data preprocessing task. A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data preprocessing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

It involves below steps:

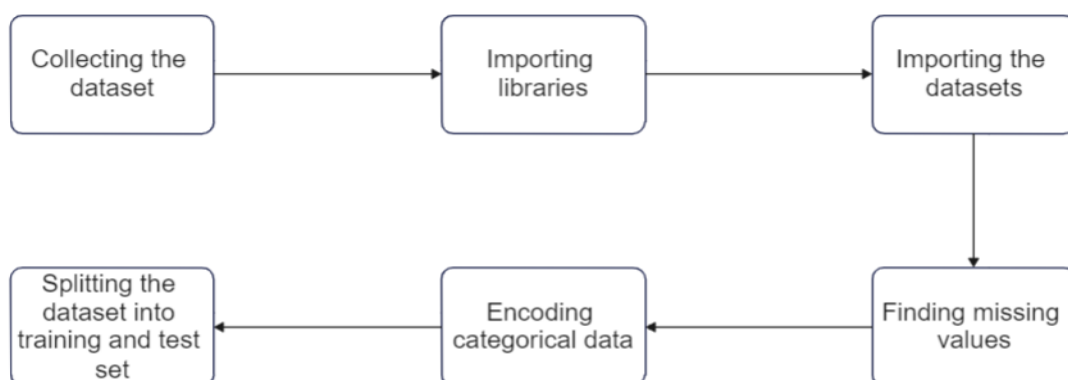


Fig 2 : Data Preprocessing

1) Collecting the dataset:

To create a machine learning model, the first thing we required is a dataset as a machine learning model completely works on data. The collected data for a particular problem in a proper format is known as the dataset.

Dataset may be of different formats for different purposes, such as, if we want to create a machine learning model for business purpose, then dataset will be different with the dataset required for a liver patient. So each dataset is different from another dataset. To use the dataset in our code, we usually put it into a CSV file. However, sometimes, we may also need to use an HTML or xlsx file.

CSV stands for "Comma-Separated Values" files; it is a file format which allows us to save the tabular data, such as spreadsheets. It is useful for huge datasets and can use these datasets in programs. We can also create our dataset by gathering data using various API with Python and put that data into a .csv file.

2) Importing libraries:

In order to perform data preprocessing using Python, we need to import some predefined Python libraries. These libraries are used to perform some specific jobs. There are three specific libraries that we will use for data preprocessing, which are:

Numpy: NumPy library is an important foundational tool for studying Machine Learning. Many of its functions are very useful for performing any mathematical or scientific calculation. As it is known that mathematics is the foundation of machine learning, most of the mathematical tasks can be performed using NumPy. NumPy stands for '*Numerical Python*'. It is an open-source Python library used to perform various mathematical and scientific tasks. It contains multi-dimensional arrays and matrices, along with many high-level mathematical functions that operate on these arrays and matrices.

Matplotlib: Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002. One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc.

Pandas: Pandas is an open-source library that is made mainly for working with relational or labeled data both easily and intuitively. It provides various data structures and operations for manipulating numerical data and time series. This library is built on top of the NumPy library. Pandas is fast and it has high performance & productivity for users.

3) Importing the datasets:

Now we need to import the datasets which we have collected for our machine learning project. To import the dataset, we will use `read_csv()` function of pandas library, which is used to read a csv file and performs various operations on it. Using this function, we can read a csv file locally as well as through an URL.

We can use `read_csv` function as below:

```
data_set= pd.read_csv('Dataset.csv')
```

Here, `data_set` is a name of the variable to store our dataset, and inside the function, we have passed the name of our dataset. Once we execute the above line of code, it will successfully import the dataset in our code.

Extracting dependent and independent variables: In machine learning, it is important to distinguish the matrix of features (independent variables) and dependent variables from dataset.

Extracting independent variable: To extract an independent variable, we will use `iloc[ ]` method of Pandas library. It is used to extract the required rows and columns from the dataset.

```
x= data_set.iloc[:, :-1].values
```

In the above code, the first colon(:) is used to take all the rows, and the second colon(:) is for all the columns. Here we have used `:-1`, because we don't want to take the last column as it contains the dependent variable. So by doing this, we will get the matrix of features.

Extracting dependent variable: To extract dependent variables, again, we will use Pandas `.iloc[ ]` method.

```
y= data_set.iloc[:, 3].values
```

Here we have taken all the rows with the last column only. It will give the array of dependent variables.

4) Finding missing values:

The next step of data preprocessing is to handle missing data in the datasets. If our dataset contains some missing data, then it may create a huge problem for our machine learning model. Hence it is necessary to handle missing values present in the dataset.

There are mainly two ways to handle missing data, which are:

By deleting the particular row: The first way is used to commonly deal with null values. In this way, we just delete the specific row or column which consists of null values. But this way is not so efficient and removing data may lead to loss of information which will not give the accurate output.

By calculating the mean: In this way, we will calculate the mean of that column or row which contains any missing value and will put it on the place of missing value. This strategy is useful for the features which have numeric data such as age, salary, year, etc. Here, we will use this approach.

5) Encoding categorical data:

Categorical data is data that contains categorical variables. Since machine learning model completely works on mathematics and numbers, but if our dataset would have a categorical variable, then it may create trouble while building the model. So it is necessary to encode these categorical variables into numbers.

6) Splitting the dataset into training and test set:

In machine learning data preprocessing, we divide our dataset into a training set and test set. This is one of the crucial steps of data preprocessing as by doing this, we can enhance the performance of our machine learning model. Suppose, if we have given training to our machine learning model by a dataset and we test it by a completely different dataset. Then, it will create difficulties for our model to understand the correlations between the models.

If we train our model very well and its training accuracy is also very high, but we provide a new dataset to it, then it will decrease the performance. So we always try to make a machine learning model which performs well with the training set and also with the test dataset. Here, we can define these datasets as:

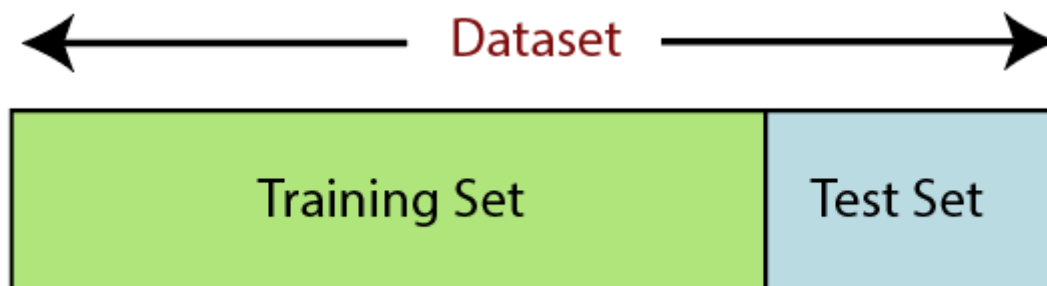


Fig 3 : Splitting of Dataset

Training Set: A subset of dataset to train the machine learning model, and we already know the output.

Test set: A subset of dataset to test the machine learning model, and by using the test set, model predicts the output.

4.3.3 Machine Learning Models

It is vital to look at the presentation of various AI calculations, and it will be helpful to make a device to think about various AI machines in Python with scikit-learn. It can involve this

test as a format for your AI issues and add more to look at. Each model will have an alternate capacity. Utilizing re-choice techniques, for example, cross-approval, you will acquire a comprehension of how each model can match imperceptible numbers. It should have the option to utilize layouts to choose a couple of the best formats from the layout you have made. At the point when new information is free, it is essential to comprehend the data involving various advances to see the data in various ways. One thought is to pick a model. You should utilize a few unique strategies to see the similar components of your AI calculation to choose a couple to finish. The method for doing this is to utilize an assortment of procedures to show accuracy, variance, and different components of the plan dissemination.

We used four different algorithms for developing the model.

- i. Decision tree
- ii. Random forest
- iii. Naïve bayes
- iv. Support vector machine

4.3.3.1 Decision Tree Algorithm:

- Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.
- In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
- The decisions or the test are performed on the basis of features of the given dataset.
- It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.
- It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.
- In order to build a tree, we use the CART algorithm, which stands for Classification and Regression Tree algorithm.

- A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.
- Below diagram explains the general structure of a decision tree:

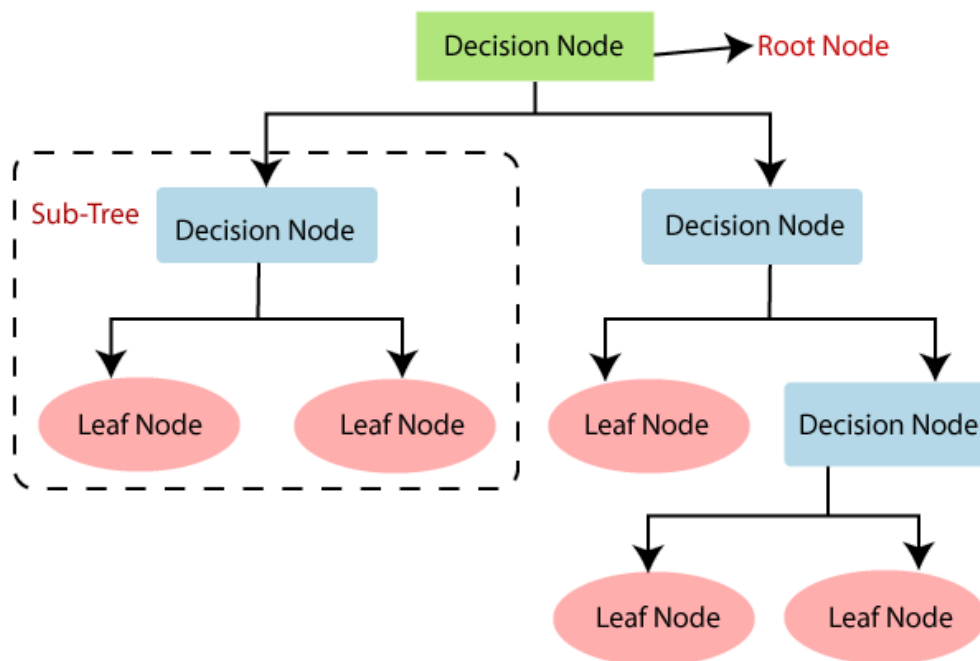


Fig 4 :Structure of decision tree

- **Root Node:** Root node is from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.
- **Leaf Node:** Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.
- **Splitting:** Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.
- **Branch/Sub Tree:** A tree formed by splitting the tree.
- **Pruning:** Pruning is the process of removing the unwanted branches from the tree.
- **Parent/Child node:** The root node of the tree is called the parent node, and other nodes are called the child nodes.

In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the values of root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node. For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further. It continues the process until it reaches the leaf node of the tree.

Steps in decision tree algorithm:

- Step-1: Begin the tree with the root node, says S, which contains the complete dataset.
- Step-2: Find the best attribute in the dataset using Attribute Selection Measure (ASM).
- Step-3: Divide the S into subsets that contains possible values for the best attributes.
- Step-4: Generate the decision tree node, which contains the best attribute.
- Step-5: Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

Attribute Selection Methods:

While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes. So, to solve such problems there is a technique which is called as Attribute selection measure or ASM. By this measurement, we can easily select the best attribute for the nodes of the tree. There are two popular techniques for ASM, which are:

- Information gain
- Gini Index

1) Information gain:

- Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an attribute.
- It calculates how much information a feature provides us about a class.
- According to the value of information gain, we split the node and build the decision tree.
- A decision tree algorithm always tries to maximize the value of information gain, and a node/attribute having the highest information gain is split first. It can be calculated using the below formula:

$$\text{Information Gain} = \text{Entropy}(S) - [(\text{Weighted Avg}) * \text{Entropy}(\text{each feature})]$$

Entropy is a metric to measure the impurity in a given attribute. It specifies randomness in data. Entropy can be calculated as:

$$\text{Entropy}(s) = -P(\text{yes}) \log_2 P(\text{yes}) - P(\text{no}) \log_2 P(\text{no})$$

Where,

- S= Total number of samples
- P(yes)= probability of yes
- P(no)= probability of no

2) Gini Index:

- Gini index is a measure of impurity or purity used while creating a decision tree in the CART(Classification and Regression Tree) algorithm.
- An attribute with the low Gini index should be preferred as compared to the high Gini index.
- It only creates binary splits, and the CART algorithm uses the Gini index to create binary splits.
- Gini index can be calculated using the below formula:

$$\text{Gini Index} = 1 - \sum_j P_j^2$$

Advantages:

- It is simple to understand as it follows the same process which a human follow while making any decision in real-life.
- It can be very useful for solving decision-related problems.
- It helps to think about all the possible outcomes for a problem.
- There is less requirement of data cleaning compared to other algorithms.

Disadvantages:

- The decision tree contains lots of layers, which makes it complex.
- It may have an overfitting issue, which can be resolved using the Random Forest algorithm.
- For more class labels, the computational complexity of the decision tree may increase.

4.3.3.2 Random Forest Algorithm:

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of *combining* multiple classifiers to solve a complex problem and to improve the performance of the model. As the name suggests, "Random Forest is a classifier that

contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output. The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

The below diagram explains the working of the Random Forest algorithm:

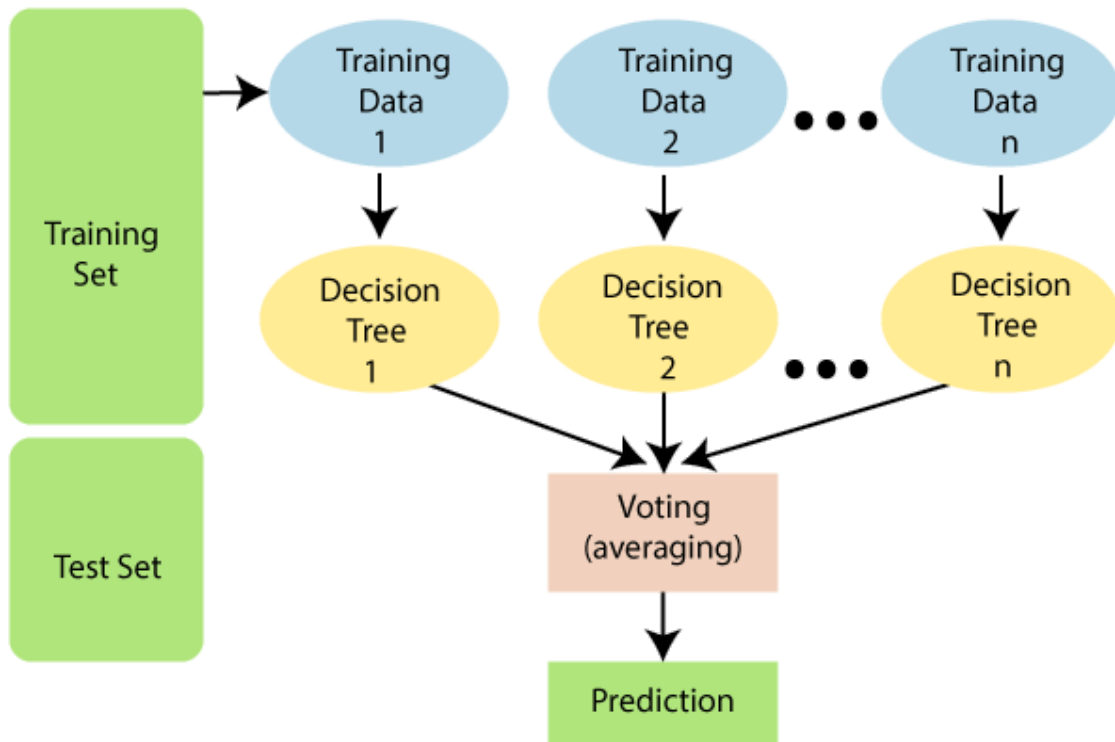


Fig 5 : Working of Random Forest

Since the random forest combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not. But together, all the trees predict the correct output. Therefore, below are two assumptions for a better Random forest classifier:

- There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result.
- The predictions from each tree must have very low correlations.

Below are some points that explain why we should use the Random Forest algorithm:

- It takes less training time as compared to other algorithms.
- It predicts output with high accuracy, even for the large dataset it runs efficiently.
- It can also maintain accuracy when a large proportion of data is missing.

Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase.

The Working process can be explained in the below steps and diagram:

- Step-1: Select random K data points from the training set.
- Step-2: Build the decision trees associated with the selected data points (Subsets).
- Step-3: Choose the number N for decision trees that you want to build.
- Step-4: Repeat Step 1 & 2.
- Step-5: For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

Advantages:

- Random Forest is capable of performing both Classification and Regression tasks.
- It is capable of handling large datasets with high dimensionality.
- It enhances the accuracy of the model and prevents the overfitting issue.

Disadvantages:

- Although random forest can be used for both classification and regression tasks, it is not more suitable for Regression tasks.

4.3.3.3 Naïve Bayes Algorithm:

- Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.
- It is mainly used in text classification that includes a high-dimensional training dataset.
- Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
- It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.
- Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.
- Bayes' theorem is also known as Bayes' Rule or Bayes' law, which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.
- The formula for Bayes' theorem is given as:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Where,

$P(A|B)$ is Posterior probability: Probability of hypothesis A on the observed event B.

$P(B|A)$ is Likelihood probability: Probability of the evidence given that the probability of a hypothesis is true.

$P(A)$ is Prior Probability: Probability of hypothesis before observing the evidence.

$P(B)$ is Marginal Probability: Probability of Evidence.

Working of naïve bayes: Working of Naïve Bayes' Classifier can be understood with the help of the below example:

Suppose we have a dataset of weather conditions and corresponding target variable "Play". So using this dataset we need to decide that whether we should play or not on a particular day according to the weather conditions. So to solve this problem, we need to follow the below steps:

- Convert the given dataset into frequency tables.
- Generate Likelihood table by finding the probabilities of given features.
- Now, use Bayes theorem to calculate the posterior probability.

Advantages:

- Naïve Bayes is one of the fast and easy ML algorithms to predict a class of datasets.
- It can be used for Binary as well as Multi-class Classifications.
- It performs well in Multi-class predictions as compared to the other Algorithms.
- It is the most popular choice for text classification problems.

Disadvantages:

- Naive Bayes assumes that all features are independent or unrelated, so it cannot learn the relationship between features.

4.3.3.4 Support Vector Machine:

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as

Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:

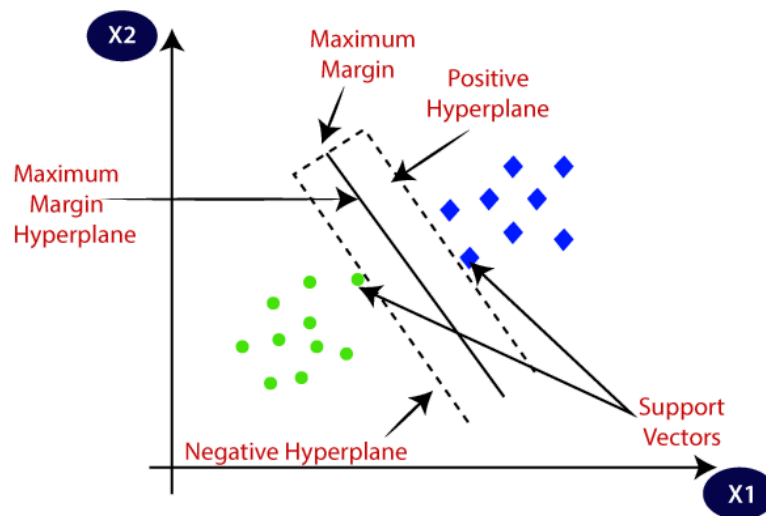


Fig 6 : Boundaries of Support Vector Machine

SVM can be of two types:

- **Linear SVM:** Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.
- **Non-linear SVM:** Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

Hyperplane and Support vectors in SVM:

Hyperplane: There can be multiple lines/decision boundaries to segregate the classes in n-dimensional space, but we need to find out the best decision boundary that helps to classify the data points. This best boundary is known as the hyperplane of SVM.

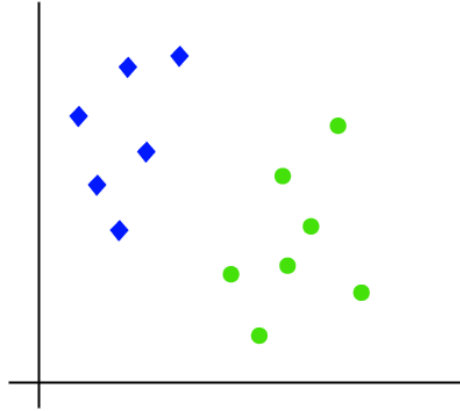
The dimensions of the hyperplane depend on the features present in the dataset, which means if there are 2 features (as shown in image), then hyperplane will be a straight line. And if there are 3 features, then hyperplane will be a 2-dimension plane.

We always create a hyperplane that has a maximum margin, which means the maximum distance between the data points.

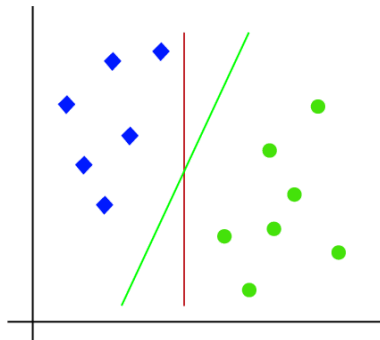
Support Vectors: The data points or vectors that are the closest to the hyperplane and which affect the position of the hyperplane are termed as Support Vector. Since these vectors support the hyperplane, hence called a Support vector.

Working of SVM:

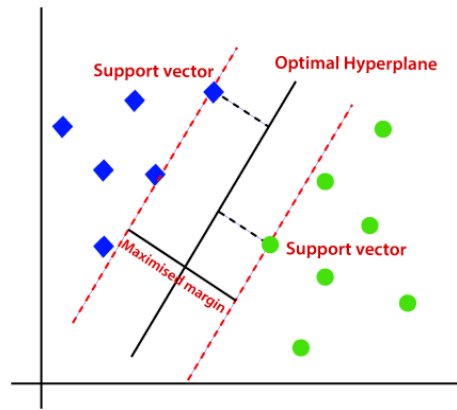
Linear SVM: The working of the SVM algorithm can be understood by using an example. Suppose we have a dataset that has two tags (green and blue), and the dataset has two features x_1 and x_2 . We want a classifier that can classify the pair(x_1 , x_2) of coordinates in either green or blue. Consider the below image:



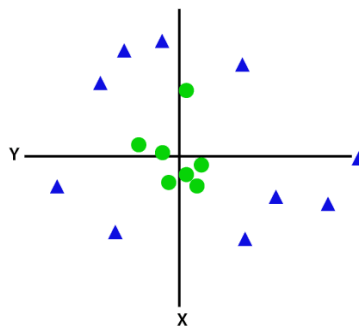
So as it is 2-d space so by just using a straight line, we can easily separate these two classes. But there can be multiple lines that can separate these classes. Consider the below image:



Hence, the SVM algorithm helps to find the best line or decision boundary; this best boundary or region is called as a hyperplane. SVM algorithm finds the closest point of the lines from both the classes. These points are called support vectors. The distance between the vectors and the hyperplane is called as margin. And the goal of SVM is to maximize this margin. The hyperplane with maximum margin is called the optimal hyperplane.



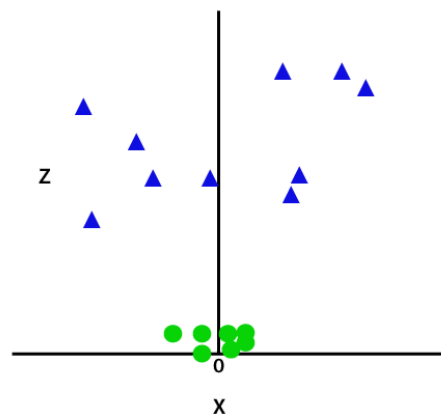
Non Linear SVM: If data is linearly arranged, then we can separate it by using a straight line, but for non-linear data, we cannot draw a single straight line. Consider the below image:



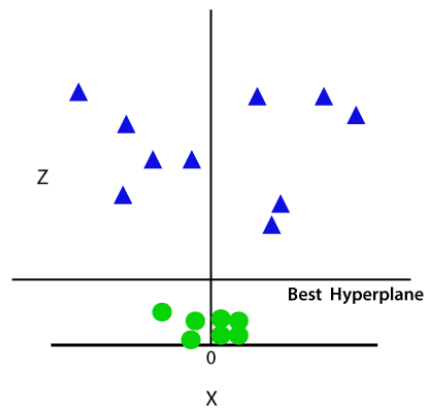
So to separate these data points, we need to add one more dimension. For linear data, we have used two dimensions x and y , so for non-linear data, we will add a third dimension z . It can be calculated as:

$$z = x^2 + y^2$$

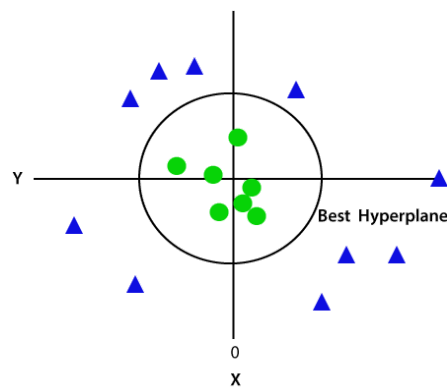
By adding the third dimension, the sample space will become as below image:



So now, SVM will divide the datasets into classes in the following way. Consider the below image:



Since we are in 3-d Space, hence it is looking like a plane parallel to the x-axis. If we convert it in 2d space with $z=1$, then it will become as:



Hence we get a circumference of radius 1 in case of non-linear data.

CHAPTER 5

5.REQUIREMENT ANALYSIS

5.1 FUNCTIONAL REQUIREMENT

Functional Requirement concerns with the specific functions delivered by the system. So, Functional requirements are statements of the services that the system must provide.

- The functional requirements of the system should be both complete and consistent
- Completeness means that all the services required by the user should be defined.
- Consistency means that requirements should not have any contradictory definitions.
- The requirements are usually described in a fairly abstract way. However, functional system requirements describe the system function in details, its inputs and outputs, exceptions and so on.
- Take user id and password match it with corresponding file entries. If a match is found then continue else raise an error message.

5.2 NON FUNCTIONAL REQUIREMENT

Non-functional Requirements refer to the constraints or restrictions on the system. They may relate to emergent system properties such as reliability, response time and store occupancy or the selection of language, platform, implementation techniques and tools.

- Performance requirement
- Platform constraints
- Accuracy and precision
- Modifiability
- Portability
- Usability

5.3 HARDWARE REQUIREMENTS

- Processor : Intel i3
- Speed : 1.6GHz and above
- RAM : 4 GB and above.
- Hard Disk : 120 GB.
- Monitor : 15" LED SVGA

5.4 SOFTWARE REQUIREMENTS

- Platform : Jupyter Notebook (Anaconda)
- Language Used : Python

CHAPTER 6

6 SYSTEM DESIGN

6.1 DATA FLOW DIAGRAM

DFD graphically representing the functions, or processes, which capture, manipulate,

store, and distribute data between a system and its environment and between components of a system. The visual representation makes it a good communication tool between User and System designer. Structures of DFD allows starting from a broad overview and expand it to a hierarchy of detailed diagrams. DFD has often been used due to the following reasons:

- Logical information flow of the system
- Determination of physical system construction requirements
- Simplicity of notation
- Establishment of manual and automated systems requirements

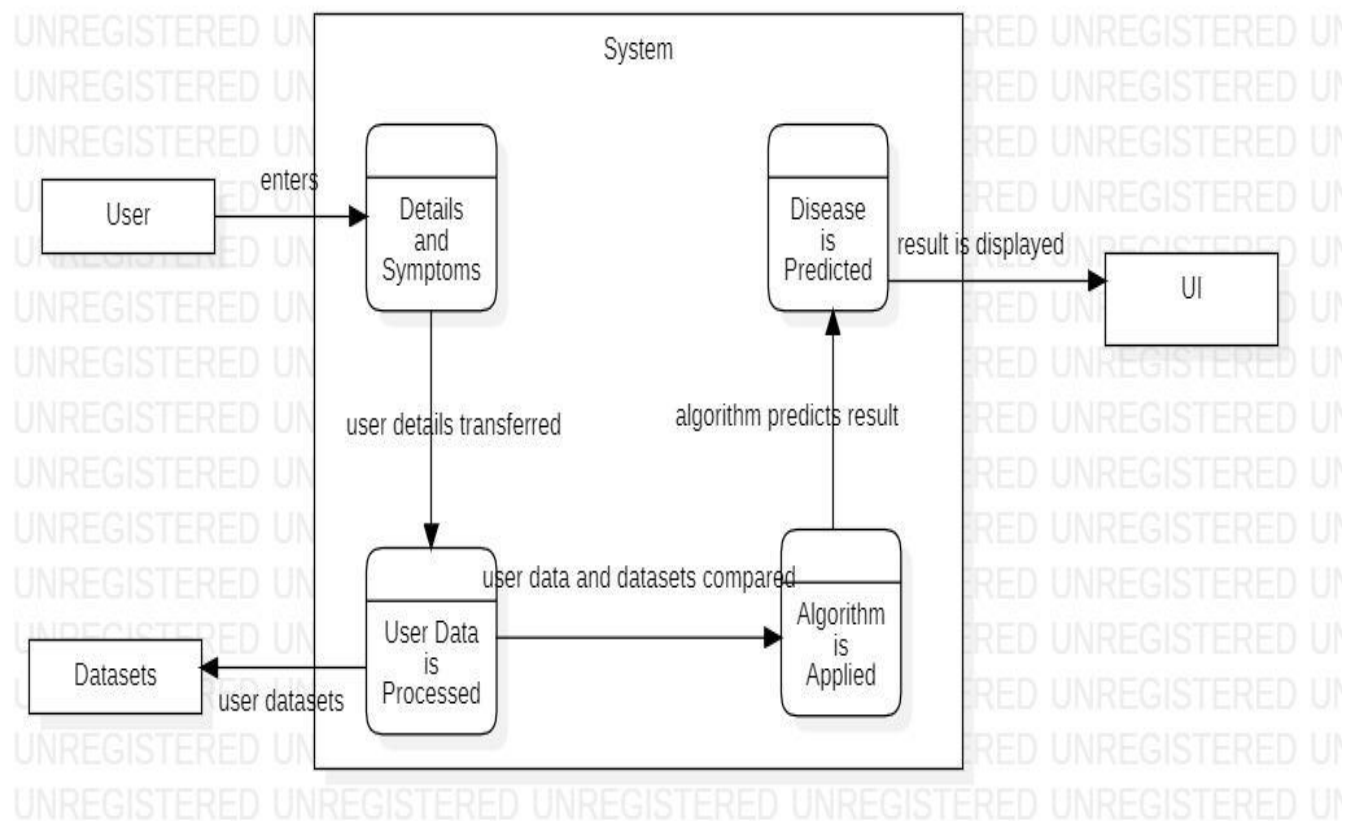


Fig 7 : Data Flow Diagram

6.2 USE CASE DIAGRAM

In the Unified Modeling Language (UML), a use case diagram can summarize the details of your system's users (also known as actors) and their interactions with the system. To build one, you'll use a set of specialized symbols and connectors. An effective use case diagram can help your team discuss and represent:

- Scenarios in which your system or application interacts with people, organizations, or

external systems

- Goals that your system or application helps those entities (known as actors) achieve
- The scope of your system

A use case diagram doesn't go into a lot of detail—for example, don't expect it to model the order in which steps are performed. Instead, a proper use case diagram depicts a high-level overview of the relationship between use cases, actors, and systems. Experts recommend that use case diagrams be used to supplement a more descriptive textual use case.

UML is the modeling toolkit that you can use to build your diagrams. Use cases are represented with a labeled oval shape. Stick figures represent actors in the process, and the actor's participation in the system is modeled with a line between the actor and use case. To depict the system boundary, draw a box around the use case itself.

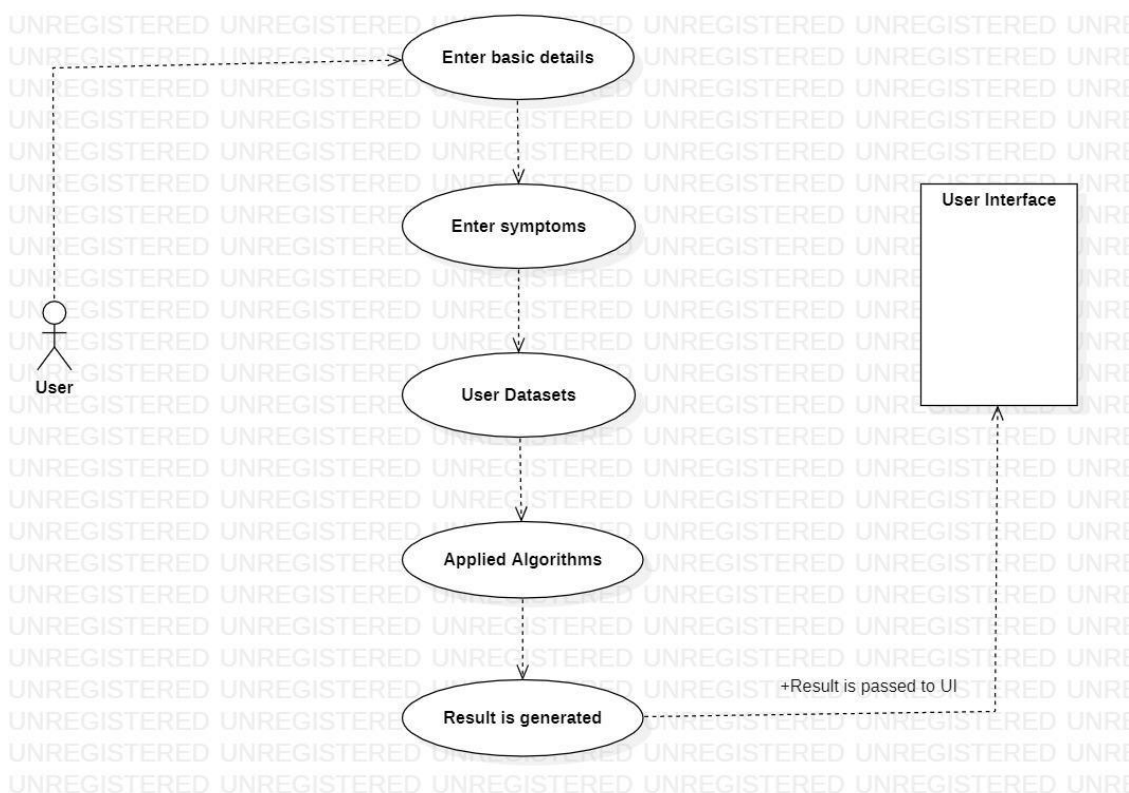


Fig 8: Use Case Diagram

6.3 CLASS DIAGRAM

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application. Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages. Class diagram shows a collection of classes, interfaces,

associations, collaborations, and constraints. It is also known as a structural diagram.

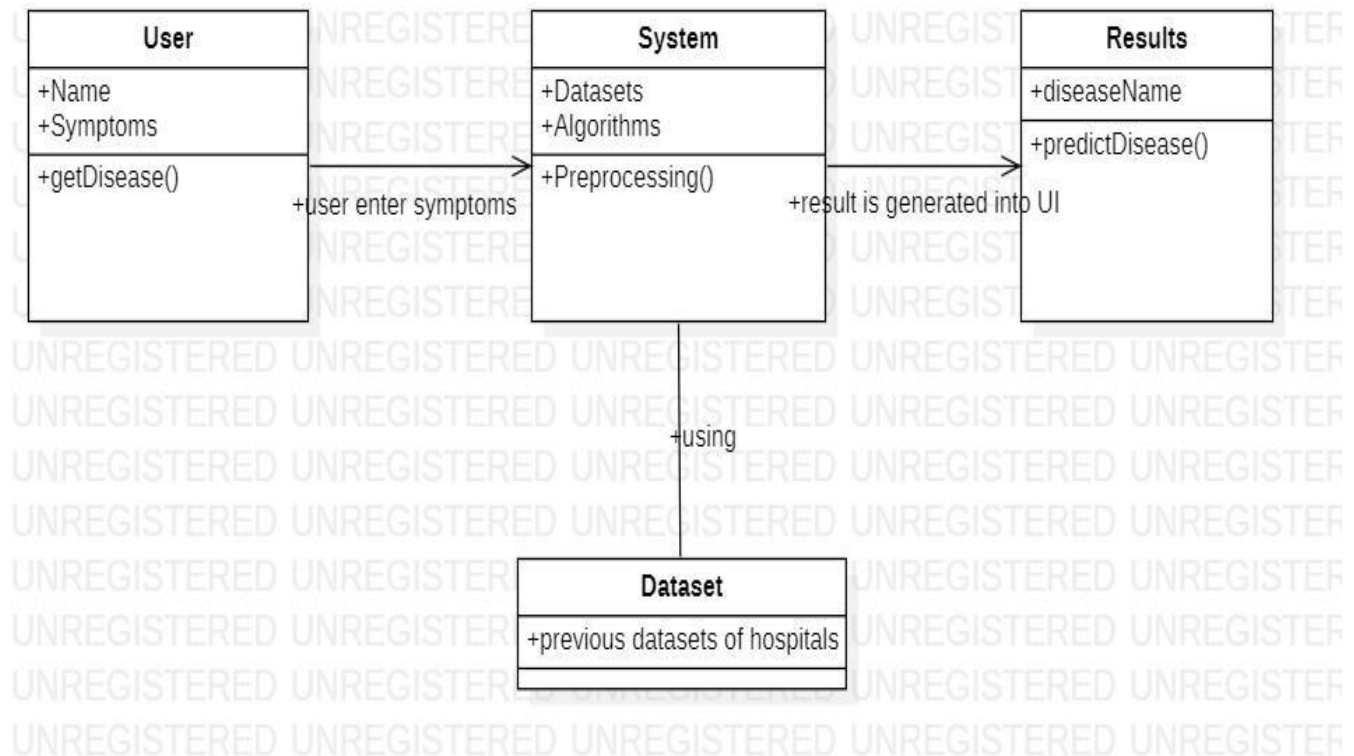


Fig 9 : Class Diagram

6.4 SEQUENCE DIAGRAM

The sequence diagram represents the flow of messages in the system and is also termed as an event diagram. It helps in envisioning several dynamic scenarios. It portrays the communication between any two lifelines as a time-ordered sequence of events, such that these lifelines took part at the run time. In UML, the lifeline is represented by a vertical bar, whereas the message flow is represented by a vertical dotted line that extends across the bottom of the page. It incorporates the iterations as well as branching.

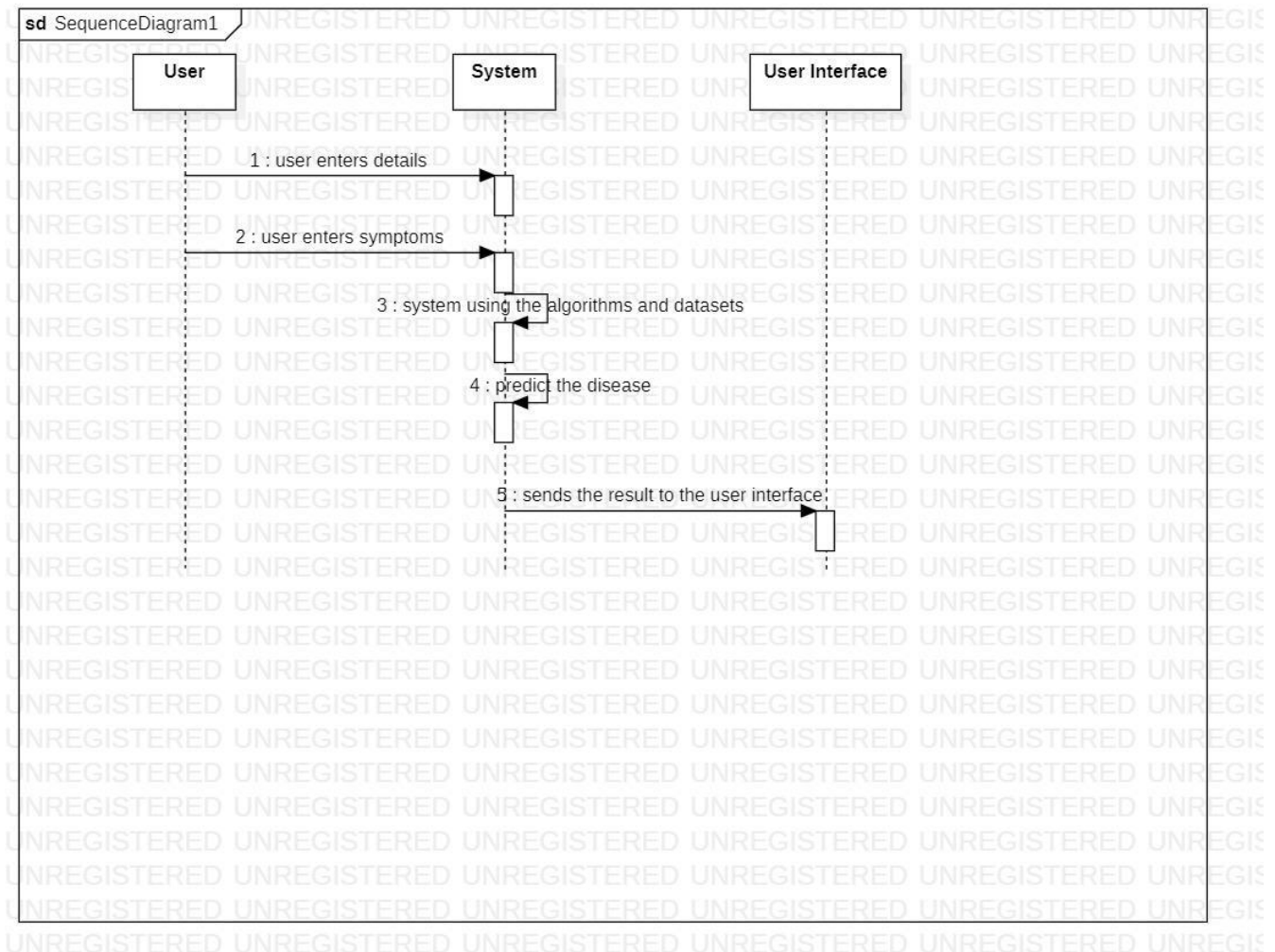


Fig 10: Sequence Diagram

6.5 COLLABORATION DIAGRAM

Collaboration diagrams are like sequence diagrams because of interaction and behavior factors. They are more concerned about object organization rather than sequence diagram that are more focused on a time sequence. They are also known as "Communication Diagram." These are used to represent the flow of messages between the objects.

- The collaboration diagram and sequence diagram shows similar information but in a distinct form.
- It can portray the architecture of an object inside the system.
- It can be used to depict the relationship among various objects within the system.
- The collaboration diagram shows the nature of a specific use case.
- They are used to determine the interfaces and class responsibilities.
- Objects collaborate through passing messages to each other.

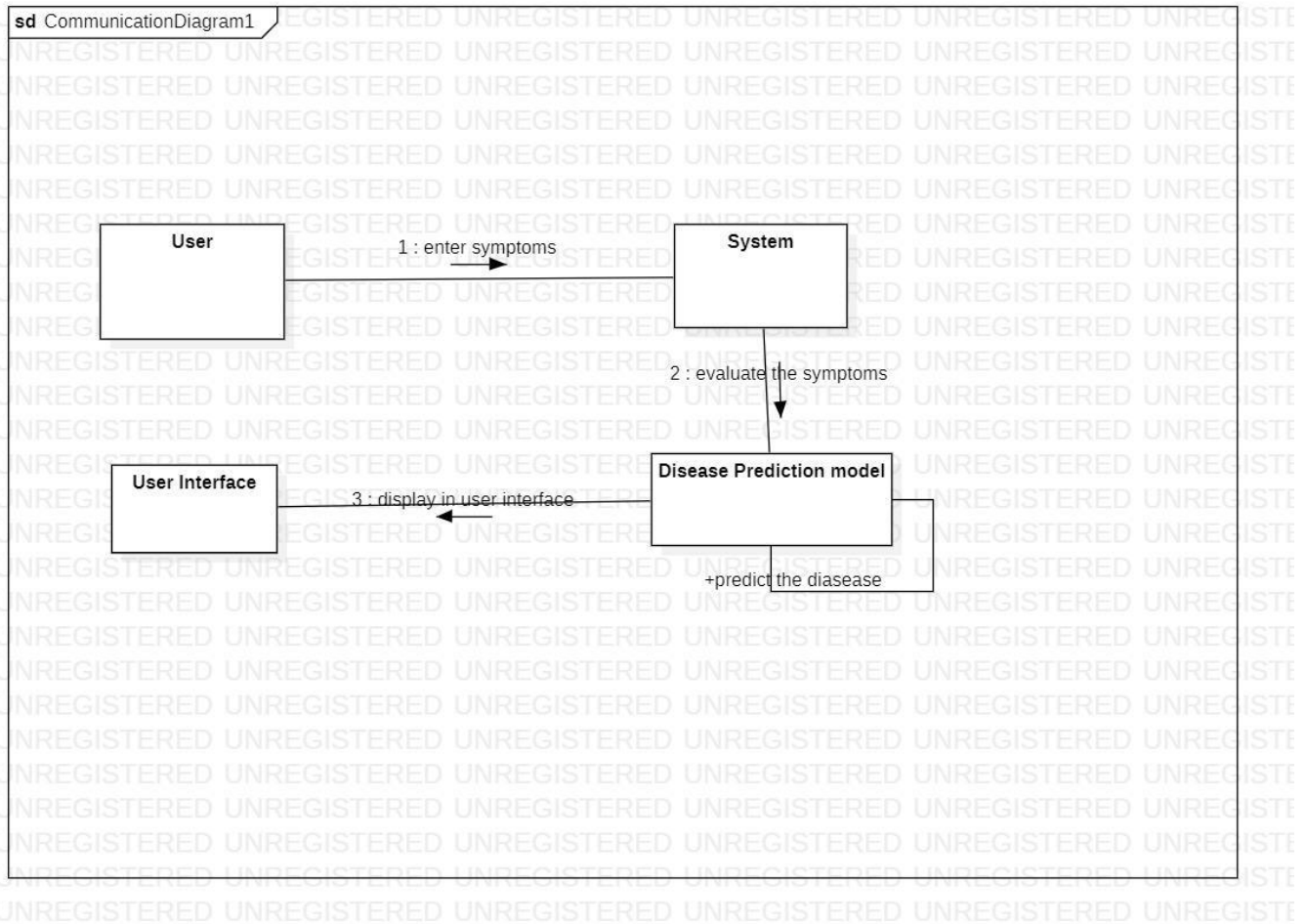


Fig 11 : Collaboration Case Diagram

6.6 COMPONENT DIAGRAM

Component diagrams are different in terms of nature and behavior. Component diagrams are used to model the physical aspects of a system. Physical aspects are the elements such as executables, libraries, files, documents, etc. which reside in a node. Component diagrams are used to visualize the organization and relationships among components in a system. These diagrams are also used to make executable systems. Component diagram is a special kind of diagram in UML. The purpose is also different from all other diagrams discussed so far. It does not describe the functionality of the system but it describes the components used to make those functionalities. Thus from that point of view, component diagrams are used to visualize the physical components in a system. These components are libraries, packages, files, etc. Component diagrams can also be described as a static implementation view of a system. Static implementation represents the organization of the components at a particular moment. A single component diagram cannot represent the

entire system but a collection of diagrams is used to represent the whole.

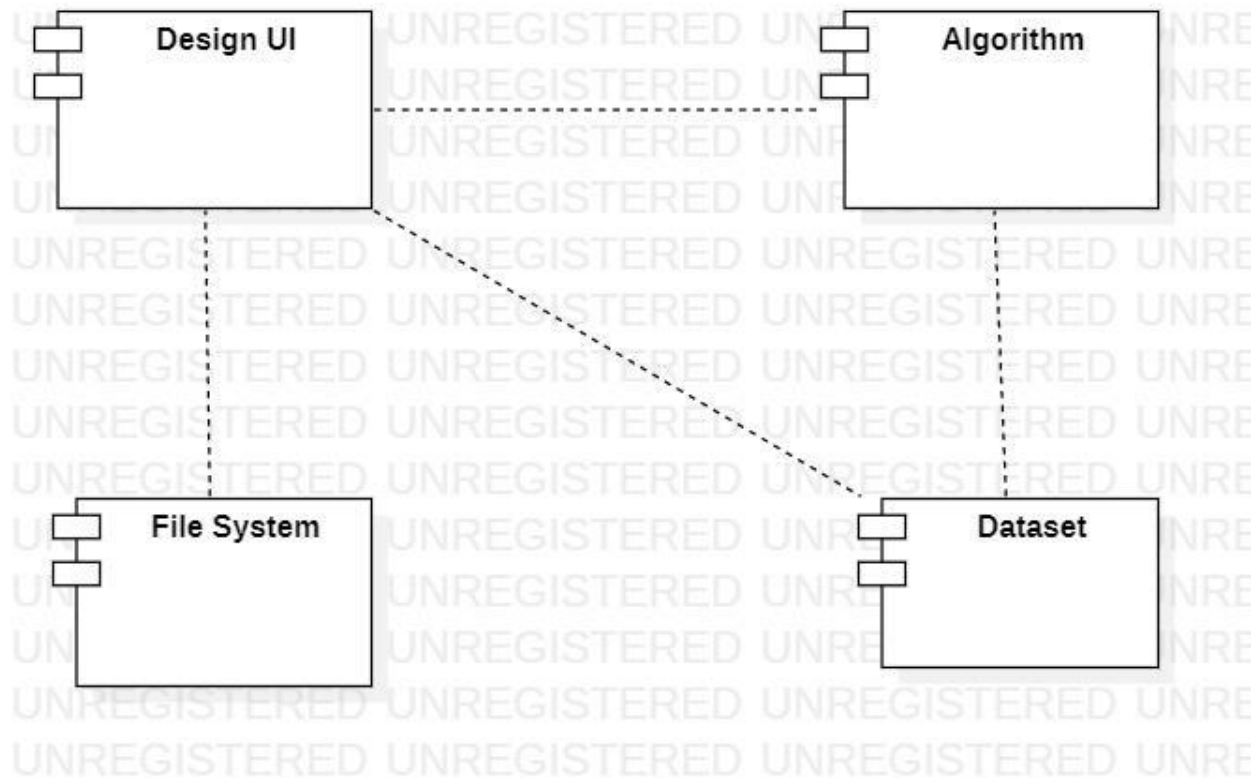


Fig 12 : Component Diagram

CHAPTER 7

7. TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies and or/a finished product. It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations does not fail in unacceptable manner. There are various types of test. Each test type addresses a specific requirement.

7.1 TYPES OF TESTING

7.1.1 UNIT TESTING

Unit testing involves the design of test cases that validate the internal program logic is functioning properly, and that program inputs procedure valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is structural testing that relies on knowledge of its construction and is

invasive. Unit test perform basic test at component level and and test a specific business, application and/or system configuration Unit test ensures that each unique path of princess performs accurately to the documented specifications and contains clearly defined inputs and expected results.

7.1.2 INTEGRATION TESTING

Integration test are designed to test integrated software components to determine if they actually run as one program. Testing in event driven and more concerned with the basic outcome of screens or fields. Integration test demonstrate that although the components were individually satisfaction, as shown successfully by unit testing the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination, of components.

7.1.3 FUNCTIONAL TESTING

Functional test provide systematic demonstrations that function tests are available as specified by the business and technical requirement, system documentation and user manuals.

Functional testing is centered on the following items:

Valid input: identify classes of valid input must be accepted,

Invalid Input: Identify classes of invalid inputs must be rejected,

Functions: Identified be exercised identities function must be exercised,

Output: identify classes of application outputs must be exercised,

Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functions test is focused on requirements, key functions or special test cases. In addition, systematic coverage pertaining to identify business process flow; data fields, predefined process and successive process must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current test is determined.

7.1.4 VALIDATION TESTING

The process of evaluating software during the development process or at the end of the development process to determine whether it satisfies specified business requirements. Validation Testing ensures that the product actually meets the client's needs. It can also be defined as to demonstrate that the product fulfills its intended use when deployed on appropriate environment.

7.1.5 ACCEPTANCE TESTING

Acceptance testing is formal testing based on user requirements and function processing. It determines whether the software is conforming specified requirements and user

requirements or not. It is conducted as a kind of Black Box testing where the number of required users involved testing the acceptance level of the system. User acceptance testing (UAT) is a type of testing, which is done by the customer before accepting the final product. Generally, UAT is done by the customer (domain expert) for their satisfaction, and check whether the application is working according to given business scenarios, real-time scenarios.

7.2 TEST REPORT

Table 2 : Test Report 1

Test case no	1
Module	User
Input	User name and symptoms
Actual Output	Get disease name or error message
Expected Output	Same as expected output
Remarks	Test Successful

Table 3 : Test Report 2

Test case no	2
Module	Disease Prediction
Input	Symptoms and datasets
Actual Output	Display disease in user interface
Expected Output	Same as expected output
Remarks	Test Successful

CHAPTER 8

8 CONCLUSION AND FUTURE WORK

8.1 CONCLUSION

The principle motivation behind this project is to predict the disease based on the symptoms given by the patients with proper implementation of machine learning algorithms. Disease should be predicted automatically with the given dataset by acquiring the maximum accuracy of 95%. So, Finally I conclude by saying that, this project is very much useful in everyone's day to day life and it is mainly more important for the healthcare sector, because they are the one that daily uses these systems to predict the diseases of the patients based on their general information and their symptoms that they have been through. Now a day's health industry plays a major role in curing the diseases of the patients so this is also some kind of help for the health industry to tell the user and also it is useful for the user in case he/she doesn't want to go to the hospital or any other clinics, so just by entering the symptoms and all other useful information the user can get to know the disease he/she is suffering from and the health industry can also get benefit from this system by just asking the symptoms from the user and entering in the system and in just a few seconds they can tell the exact and up to some extent the accurate diseases. If health industry adopts this project then the work of the doctors can be reduced and they can easily predict the disease of the patient. The Disease prediction is to provide prediction for the various and generally occurring diseases that when unchecked and sometimes

ignored can turn into fatal disease and cause a lot of problems for the patient and as well as their family members.

8.2 Future Work

- More interactive user interface
- Can be implemented as mobile or web application
- Include more diseases and symptoms
- Prescribe medicine for those diseases

CHAPTER 9

9. APPENDIX

9.1 SOURCE CODE

```
#Importing Libraries from matplotlib to visualize the data
from mpl_toolkits.mplot3d import Axes3D
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
```

```
#Importing Libraries to create GUI
from tkinter import *
```

```
#Importing Libraries to perform calculations
import numpy as np
import pandas as pd
import os
```

```
#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_extremities','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']

#List of Diseases is listed in list disease.

disease=['Fungal infection', 'Allergy', 'GERD', 'Chronic cholestasis', 'Drug Reaction', 'Peptic ulcer disease', '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)',
'Heartattack', 'Varicoseveins', 'Hypothyroidism', 'Hyperthyroidism', 'Hypoglycemia', 'Osteoarthritis', 'Arthritis', '(vertigo) Parosymal Positional Vertigo', 'Acne', 'Urinary tract

```
infection','Psoriasis',  
    'Impetigo']
```

```
l2=[]  
for i in range(0,len(l1)):  
    l2.append(0)  
print(l2)
```

```
#Reading the training .csv file  
df=pd.read_csv("training.csv")
```

```
#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,'Osteoarthritis':34,'Arthritis':35,  
    '(vertigo) Paroymsal Positional Vertigo':36,'Acne':37,'Urinary tract  
infection':38,'Psoriasis':39,  
    'Impetigo':40}},inplace=True)
```

```
#printing the top 5 rows of the training dataset  
df.head()
```

```
# 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()

```

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()
plotPerColumnDistribution(df, 10, 5)
plotScatterMatrix(df, 20, 10)

```

```

X= df[l1]
y = df[["prognosis"]]
np.ravel(y)
print(X)
print(y)

```

```

#Reading the testing.csv file
tr=pd.read_csv("testing.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 disease':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,'Osteoarthritis':34,'Arthritis':35,
    '(vertigo) Paroymsal Positional Vertigo':36,'Acne':37,'Urinary tract
infection':38,'Psoriasis':39,
    'Impetigo':40}},inplace=True)

```

```
#printing the top 5 rows of the testing data
```

```
tr.head()
```

```
plotPerColumnDistribution(tr, 10, 5)
```

```
plotScatterMatrix(tr, 20, 10)
```

```
X_test= tr[l1]
```

```
y_test = tr[["prognosis"]]
```

```
np.ravel(y_test)
```

```
print(X_test)
```

```
print(y_test)
```

```
#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)#dropping symptoms with values 0
```

```
    print(x.values)
```

```
    y = x.keys()#storing name of 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()
```

```
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:
```

```
        print(NameEn.get())
```

```
        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")
import sqlite3
conn = sqlite3.connect('database.db')
c = conn.cursor()
c.execute("CREATE TABLE IF NOT EXISTS DecisionTree(Name StringVar,Symptom1
StringVar,Symptom2 StringVar,Symptom3 StringVar,Symptom4 TEXT,Symptom5 TEXT,Disease
StringVar)")
c.execute("INSERT INTO
DecisionTree(Name,Symptom1,Symptom2,Symptom3,Symptom4,Symptom5,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.g
et())
    scatterplt(pred1.get())

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")
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())

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("KNN")
        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")

```

```

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())

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")
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())
```

```
#Tk class is used to create a root window  
root.configure(background='Ivory')  
root.title('Smart Disease Predictor System')  
root.resizable(0,0)
```

```
#taking first input as symptom  
Symptom1 = StringVar()  
Symptom1.set("Select Here")
```

```
#taking second input as symptom  
Symptom2 = StringVar()  
Symptom2.set("Select Here")
```

```
#taking third input as symptom  
Symptom3 = StringVar()  
Symptom3.set("Select Here")
```

```
#taking fourth input as symptom  
Symptom4 = StringVar()  
Symptom4.set("Select Here")
```

```
#taking fifth input as symptom  
Symptom5 = StringVar()  
Symptom5.set("Select Here")  
Name = StringVar()
```

```
#function to Reset the given inputs to initial position  
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
```

```
#Exit button to come out of system
```

```
from tkinter import messagebox
```

```
def Exit():
    qExit=messagebox.askyesno("System","Do you want to exit the system")
    if qExit:
        root.destroy()
        exit()
```

```
#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: Sudhanshu,Rohan,Aditya", fg="Pink",
bg="Ivory")
w2.config(font=("Times",30,"bold italic"))
w2.grid(row=2, column=0, columnspan=2, padx=100)
```

```
#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)
```

#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)
```

#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)
```

```
#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)
```

```
#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)
```

```
#Showing the output of different algorithms
```

```
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"
```

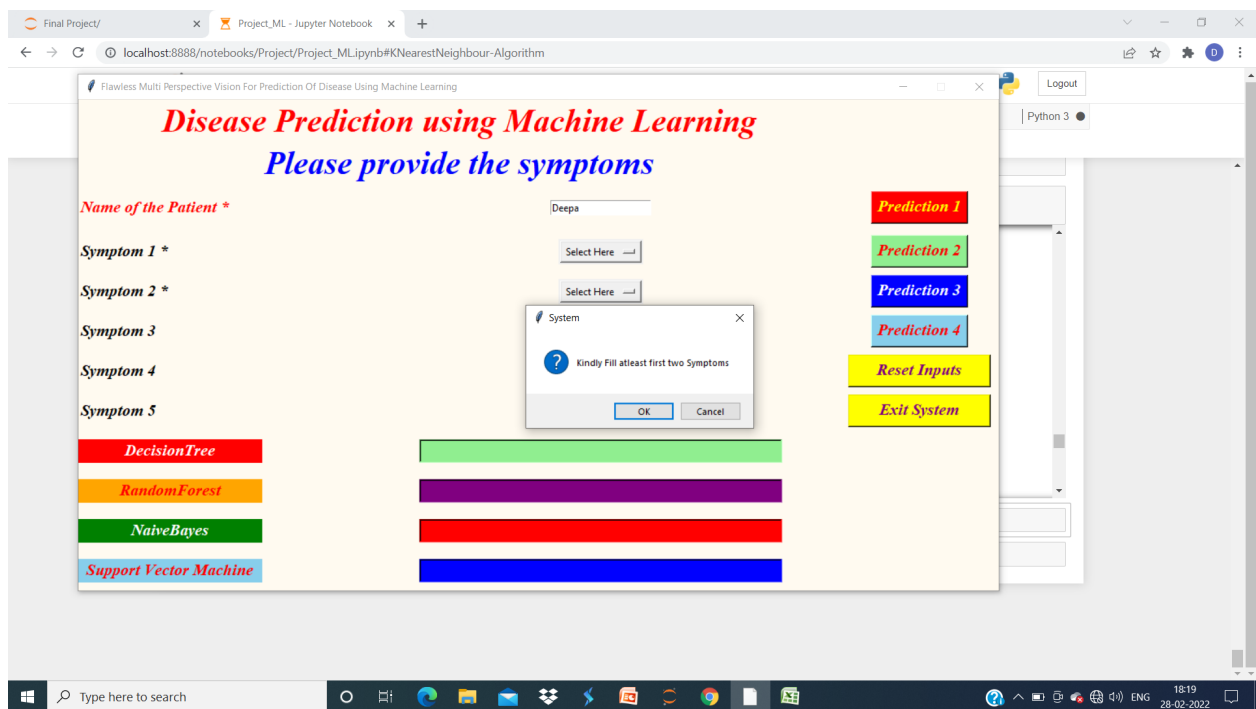
```
,width=40,fg="orange",textvariable=pred3,relief="sunken").grid(row=19, column=1,  
padx=10)
```

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

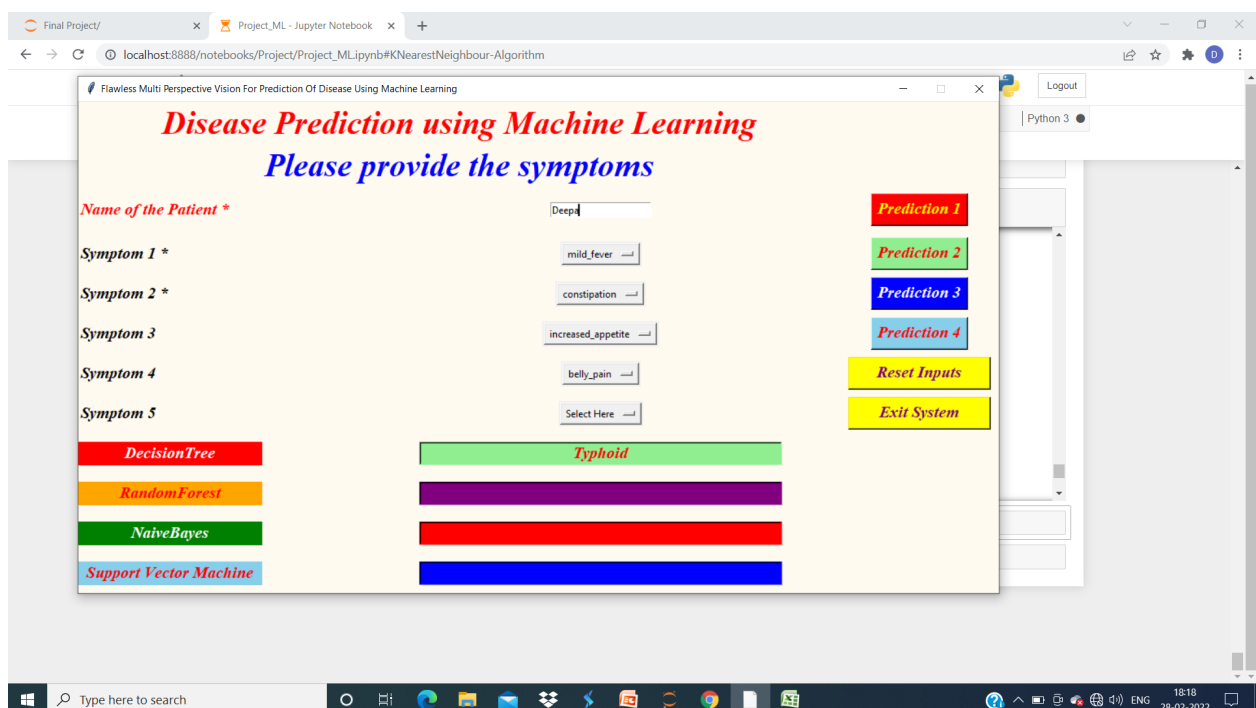
```
,width=40,fg="yellow",textvariable=pred4,relief="sunken").grid(row=21, column=1,  
padx=10)
```

#calling this function because the application is ready to run
root.mainloop()

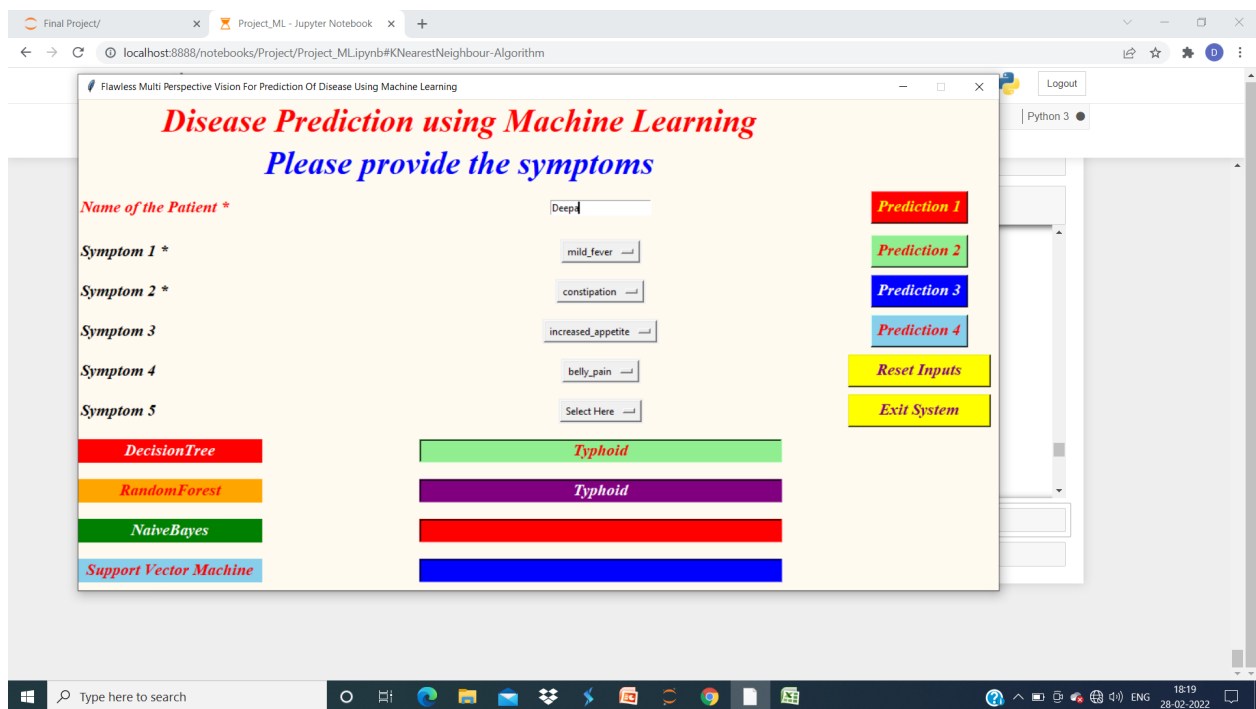
9.2 Screenshots:



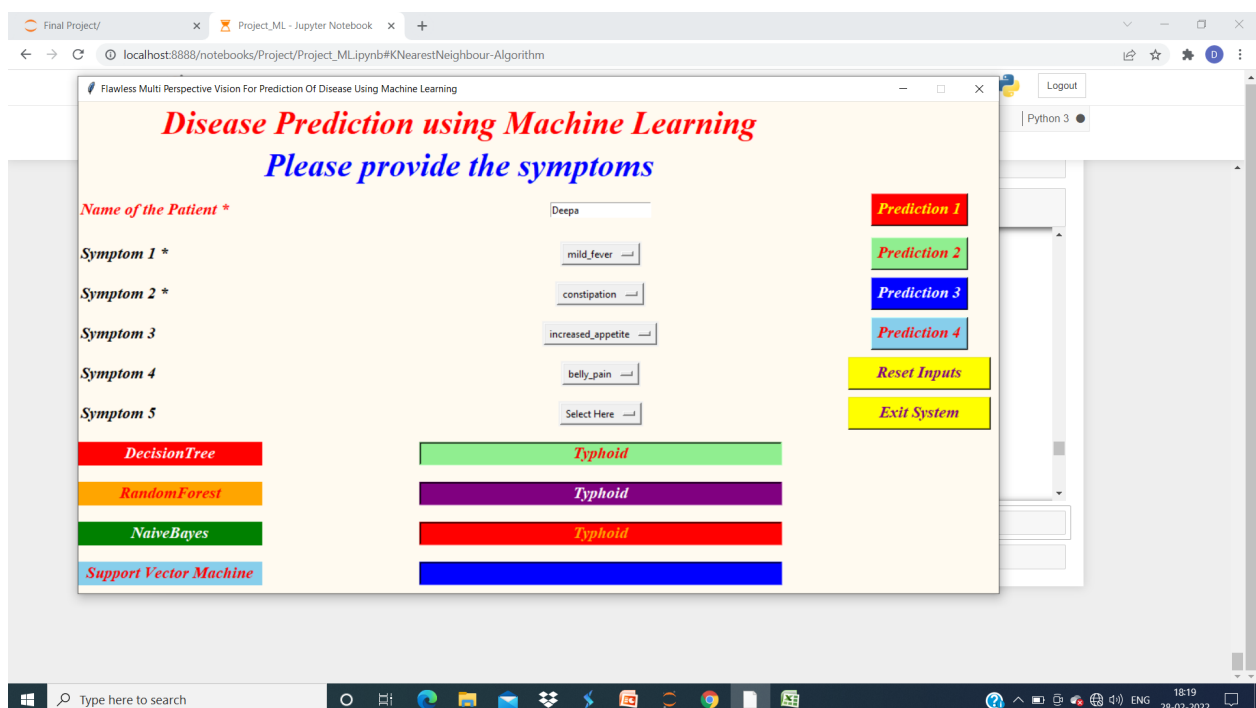
Decision tree output



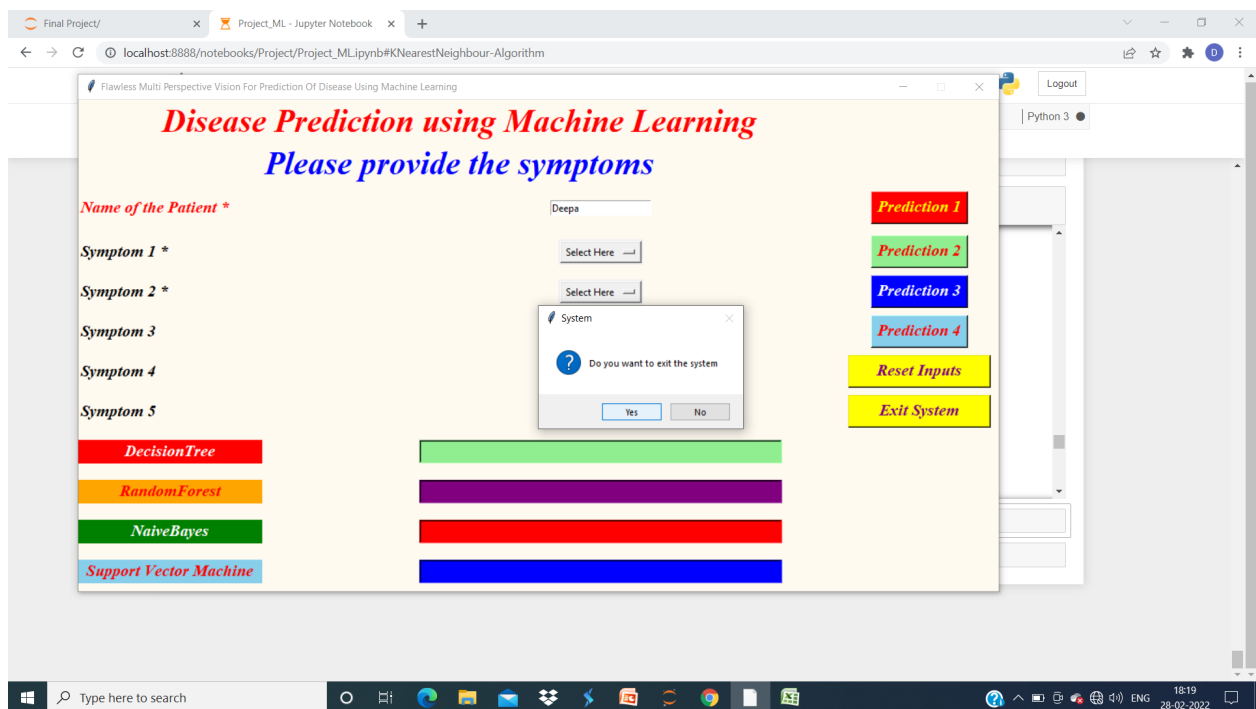
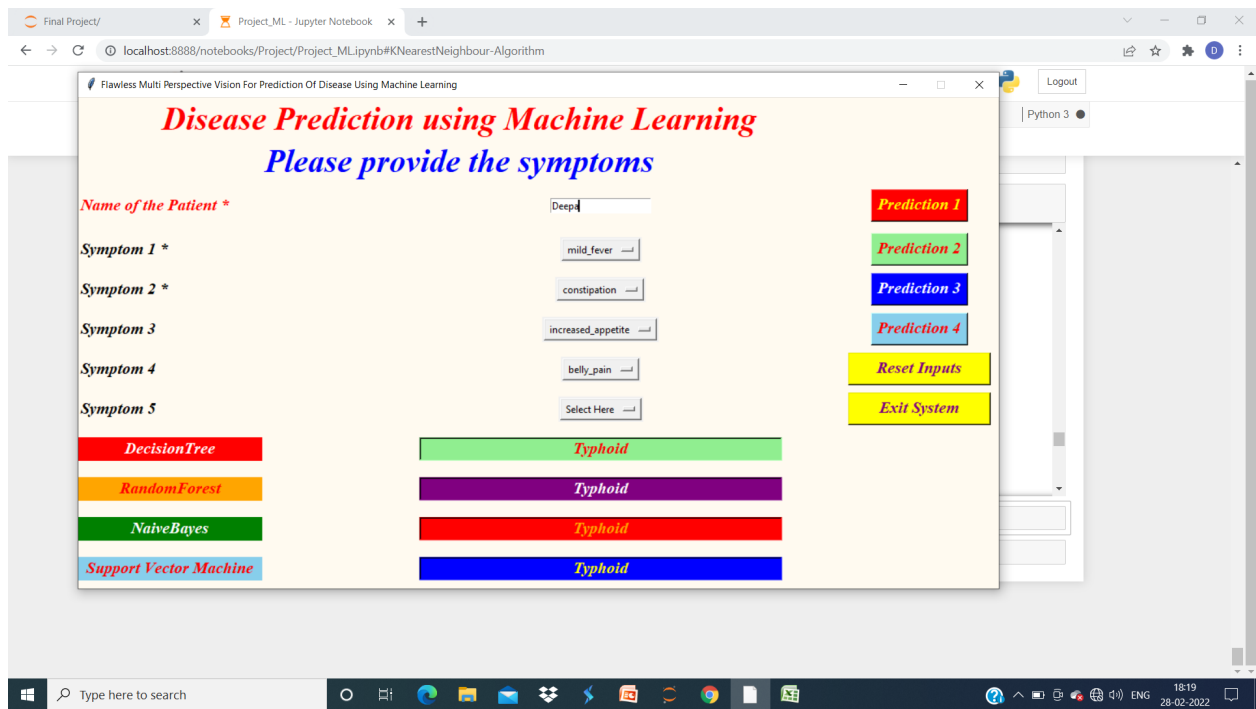
Random forest output



Naïve bayes output



Support vector machine output



CHAPTER 10

10. REFERENCES

- [1] Sehrish aqueel "Deep Learning for Predictive Analytics in Healthcare" The International Conference on Advanced Machine Learning Technologies and Applications (AMLTA2019).
- [2] Arwinder Dhillon and Ashima Singh "Machine Learning in Healthcare Data Analysis: A Survey" Journal of Biology and Today's World 2019 Volume 8, Issue 6
- [3] Hasnuhana Mazumder and Sayantan Saha "Web Based Disease Detection" International Journal Of Engineering Research & Technology(IJERT) Volume 02, Issue 04 (April 2013)
- [4] Min Chen "Disease Prediction by Machine Learning Over Big Data From Healthcare Communities" IEEE Access Volume 5 2017.
- [5] K.Gomathi and D.Shanmuga "Multi Disease Prediction using Data Mining Techniques" Internation Journal of System and Software Engineering 2016
- [6] Majd A. M. Alhaj and ashraf Y. A. Maghari " Cancer Survivability Prediction using Random Forest and Rule Induction Algorithms" 2017 8th international conference on information technology.
- [7] Mamatha Sai Yarabarla and Lakshmi Kavya Ravi " Breast Cancer Prediction via Machine Learning" 2019 3rd International Conference on Trends in electronics and Informatics.
- [8] Chinmayi Thallam and Aarsha Peruboyina "Early stage lung cancer prediction using various machine learning techniques" 2020 4th international conference on Electronics, Communication and Aerospace Technology (ICECA).
- [9] Sneha Grampurohit and Chetan Sagarnal " Disease Prediction using Machine Learning Algorithms" 2020 International Conference for Emerging Technology.
- [10] Prasan Kumar Sahoo and Suwendu Kumar Mohapatra "Analyzing Healthcare Big Data with Prediction for Future Health Condition" IEE Access 2016.
- [11] Roger Lee and Chinmayi Chitnis "Improving Helath Care Systems by Disease Prediction" 2018 International Conference on Computational Science and Computational Intelligence.
- [12] Ahelam tikotikar and Mallikarjun Kodabagi "A Survey on Technique for Prediction of Disease in Medical Data" 2017 International Conference On Smart Technologies For Smart Nation.

- [13] Y.Bengio “Learning Deep Architectures for AI” January 2009.
- [14] Qiulin Wang “Research on the Improved CNN Deep Learning Method for Motion Intention Recognition of Dynamic Lower Limb Prosthesis” December 2019..
- [15] Haijian Shi “Best first Decision Tree Learning” January 2007.