A

Major Project

On

HUMAN DISEASE PREDICTION

(Submitted in partial fulfillment of the requirements for the award of Degree)

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By

P. Krishna Tulasi (177R1A05A0)

B. Harika (177R1A0564)

N. Sindhu (177R1A10595)

Under the Guidance of

A. Mahendar

(Associate Professor)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMRTECHNICALCAMPUS

UGC AUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved byAICTE, New Delhi)

Recognized Under Section 2(f) & 12(B) of the UGCAct.1956,

Kandlakoya (V),MedchalRoad,Hyderabad-501401.

2017-21

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project entitled “HUMAN **DISEASE PREDICTION**” being submitted by **P. KRISHNA TULASI (177R1A05A0), B. HARIKA (177R1A0564) &N. SINDHU (177R1A0595)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by him/her under our guidance and supervision during the year 2020-21.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

Mr. A. Mahendar **Associate Professor** **INTERNALGUIDE**	**Dr. A. Raji Reddy** **DIRECTOR**
Dr. K. Srujan Raju **HoD**	**EXTERNALEXAMINER**

Submitted for viva voice Examination held on ________________

ACKNOWLEGDEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project. We take this opportunity to express my profound gratitude and deep regard to my guide

Mr. A. Mahendar, Associate Professor for his exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by him shall carry us a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to Project Review Committee (PRC) Coordinators: **Mr. J. Narasimha Rao, Mr. B. P. Deepak Kumar, Mr. K. Murali, Dr. Suwarna Gothane and Mr. B. Ramji** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. K. Srujan Raju,** Head, Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy,** Director for being cooperative throughout the course of this project. We would like to express our sincere gratitude to Sri. **Ch. Gopal Reddy,** Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

P. KRISHNA TULASI (177R1A05A0)
B.HARIKA (177R1A0564)
N.SINDHU(177R1A0595)

ABSTRACT

Disease prediction using Machine Learning is a system which predicts the disease based on the information or the symptoms provided by the user based on the that accurate result will be displayed. Now a day's health industry plays major role in curing the diseases of the patients, in case he/she doesn't want to go to the hospital or any 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. Machine Learning algorithms such as Naive Bayes, Decision Tree, KNN and Random forest are employed on the provided dataset and predict the disease. Its implementation is done through the Python Programming Language.

LIST OF FIGURES

LIST OF SCREENSHOTS

TABLE OF CONTENTS

1.INTRODUCTION

1. INTRODUCTION

1.1 PROJECT SCOPE

This project aims to provide a graphical user interface to predict the occurrences of disease on the basis of various symptoms. The user can select various symptoms and can find the diseases.

1.2 PROJECT PURPOSE

The purpose of making this project is to predict the accurate disease of the patient using the symptoms. Using this information, the model 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. 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.

1.3 PROJECT FEATURE

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

- This Project will predict the diseases of the patients based on the symptoms.
- 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 at least two symptoms and user has to enter his/her name.

2. SYSTEMANALYSIS

2. SYSTEM ANALYSIS

SYSTEM ANALYSIS

System Analysis is the important phase in the system development process. The System is studied to the minute details and analyzed. The system analyst plays an important role of an interrogator and dwells deep into the working of the present system. In analysis, a detailed study of these operations performed by the system and their relationships within and outside the system is done. A key question considered here is, “what must be done to solve the problem?” The system is viewed as a whole and the inputs to the system are identified. Once analysis is completed the analysis has a firm understanding of what is to be done.

2.1 PROBLEM DEFINITION

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 .

2.2 EXISTING SYSTEM

In the existing system the data set is typically small, for patients and diseases with specific conditions. These systems are mostly designed for the more colossal diseases such as Heart Disease, Cancer etc. The pre-selected characteristics may sometimes not satisfy the changes in the disease and its influencing factors which could lead to inaccuracy in results. As we live in continuously evolving world, the symptoms of diseases also evolve over a course of time. Also, most of the current systems make the users wait for long periods by making them answer lengthy questionnaires.

2.2.1 LIMITATIONS OF EXISTING SYSTEM

- Only few diseases can be detected.
- Less accuracy
- Weak generalization
- Time consuming

2.3 PROPOSED SYSTEM

We are proposing such a system which will flaunt a simple and elegant User Interface and also be time efficient. In order to make it less time consuming we are aiming at a more specific questionnaire which will be followed by the system. Our aim with this system is to be the connecting bridge between doctors and patients. The main feature will be the machine learning, in which we will be using algorithms such as Naïve Bayes Algorithm, K-Nearest Algorithm, Decision Tree Algorithm, Random Forest Algorithm, which will help us in getting accurate predictions and Also, will find which algorithm gives a faster and efficient result by comparatively-comparing.

2.3.1 ADVANTAGES OF THE PROPOSED SYSTEM

- Data can be stored and used for doctor reference
- Analysis based on multiple algorithms
- More accurate.

2.4 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. Three key considerations involved in the feasibility analysis are

- Economic Feasibility
- Technical Feasibility
- Social Feasibility

2.4.1 ECONOMIC FEASIBILITY

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors, which affect the development of a system, is the cost it would require.

The following are some of the important financial questions asked during preliminary investigation:

- The costs conduct a full system investigation.
- The cost of the hardware and software.
- The benefits in the form of reduced costs or fewer costly errors.

Since the system is developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication of the system is economically possible for development.

2.4.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

2.4.3 BEHAVIORAL FEASIBILITY

This includes the following questions:

- Is there sufficient support for the users?
- Will the proposed system cause harm?

The project would be beneficial because it satisfies the objectives when developed and installed. All behavioral aspects are considered carefully and conclude that the project is behaviorally feasible.

2.5 HARDWARE & SOFTWARE REQUIREMENTS

2.5.1 HARDWARE REQUIREMENTS:

Hardware interfaces specifies the logical characteristics of each interface between the software product and the hardware components of the system. The following are some hardware requirements:

- Device : Laptop
- RAM : 4GB and Above
- Space on Hard Disk : 2GB and Above

2.5.2 SOFTWARE REQUIREMENTS:

Software Requirements specifies the logical characteristics of each interface and software components of the system. The following are some software requirements:

- Operating System : Windows 7 and Above
- Software : Jupiter notebook, Anaconda
- Database : DB browser Sqlite3
- Interface : Python Tkinter
- Language : Python
- Libraries : Numpy, Pandas

3. ARCHITECTURE

3. ARCHITECTURE

3.1 PROJECT ARCITECTURE

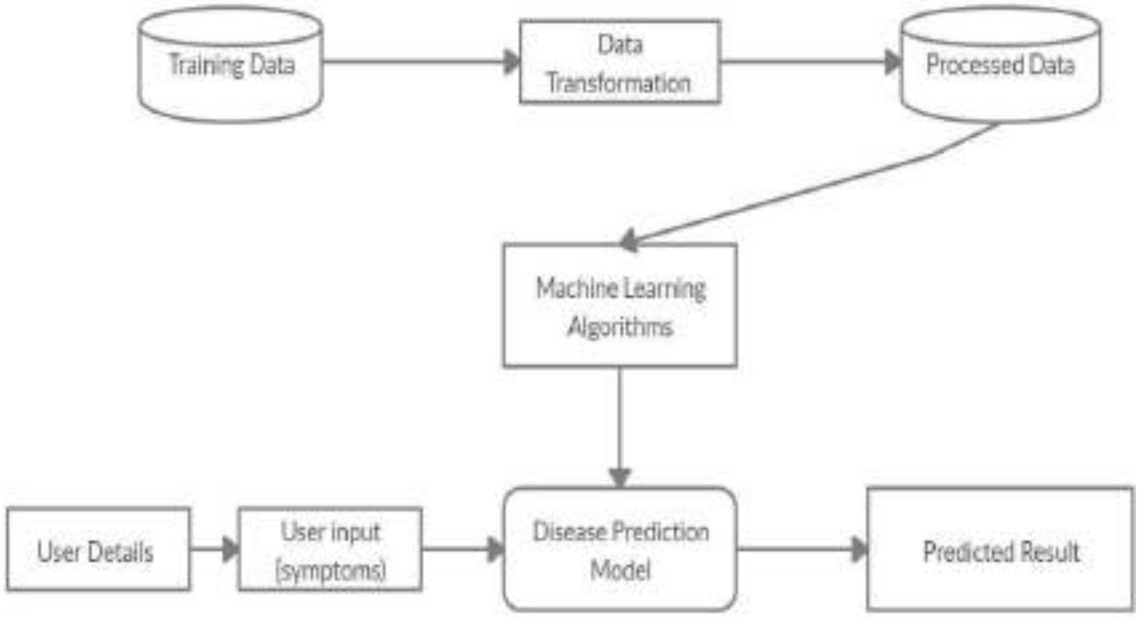


Figure 3.1 : Architecture of the system

Disease prediction using machine learning predicts the presence of the disease for the user based on various symptoms. The architecture of the system disease prediction using machine learning consist of dataset through which we will compare the symptoms of the user and predicts it, then the dataset is transformed into the smaller sets and from there it gets classified basedon the classification algorithms later on the classified data is then processed into the machine learning technologies through which the data gets processed and goes in to the disease prediction model using all the inputs from the user that is mentioned above. Then user entering the above information and overall processed data combines and compares in the prediction model of the system and finally predicts the disease. An architecture diagram is a graphical representation of a set of concepts that are part of architecture, including their principles, elements and components. The above diagram explains about the system software in perception of overview of the system.

3.2 USE CASE DIAGRAM

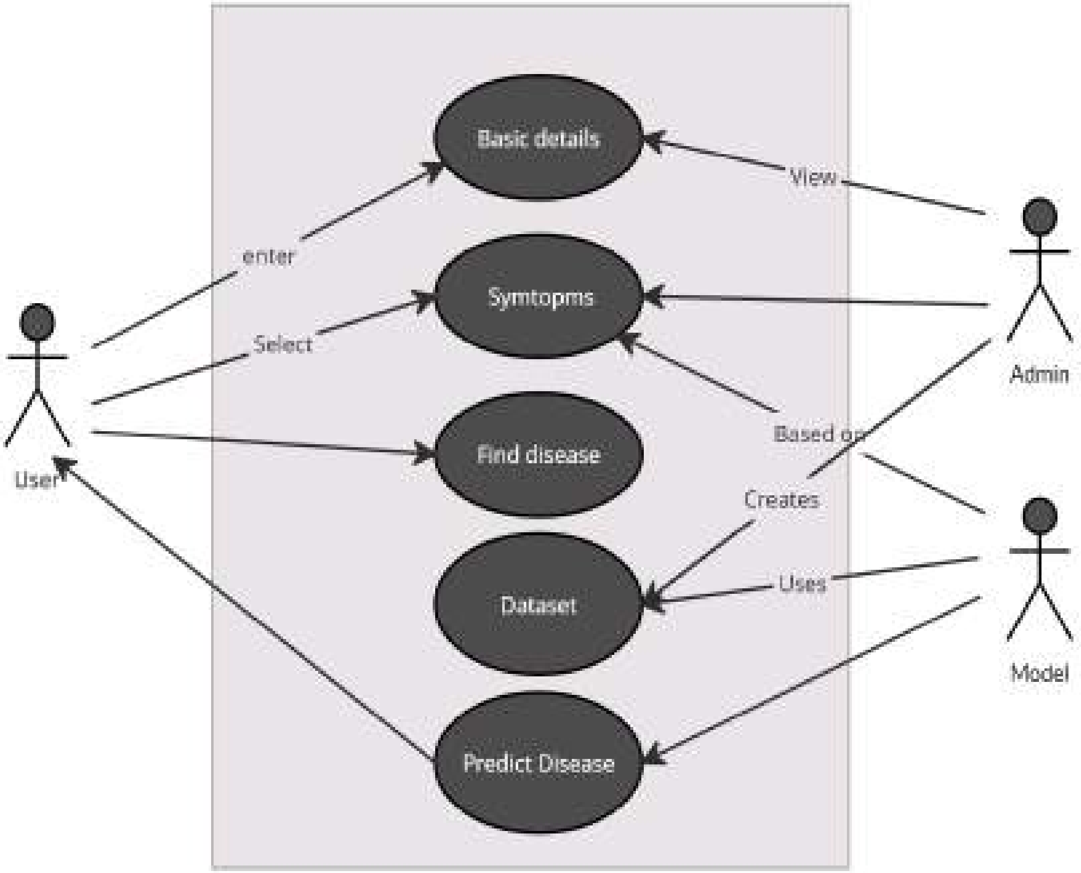


Figure 3.2: Use case diagram

This use case diagram shows how from starting the model flows from one step to another, like he enter into the systemthen enters all the information's and all other general information along with the symptoms that goes into the system, compares with the prediction model and if true is predicts the appropriate results.

3.3 CLASS DIAGRAM

Class Diagram is a collection of classes and objects.

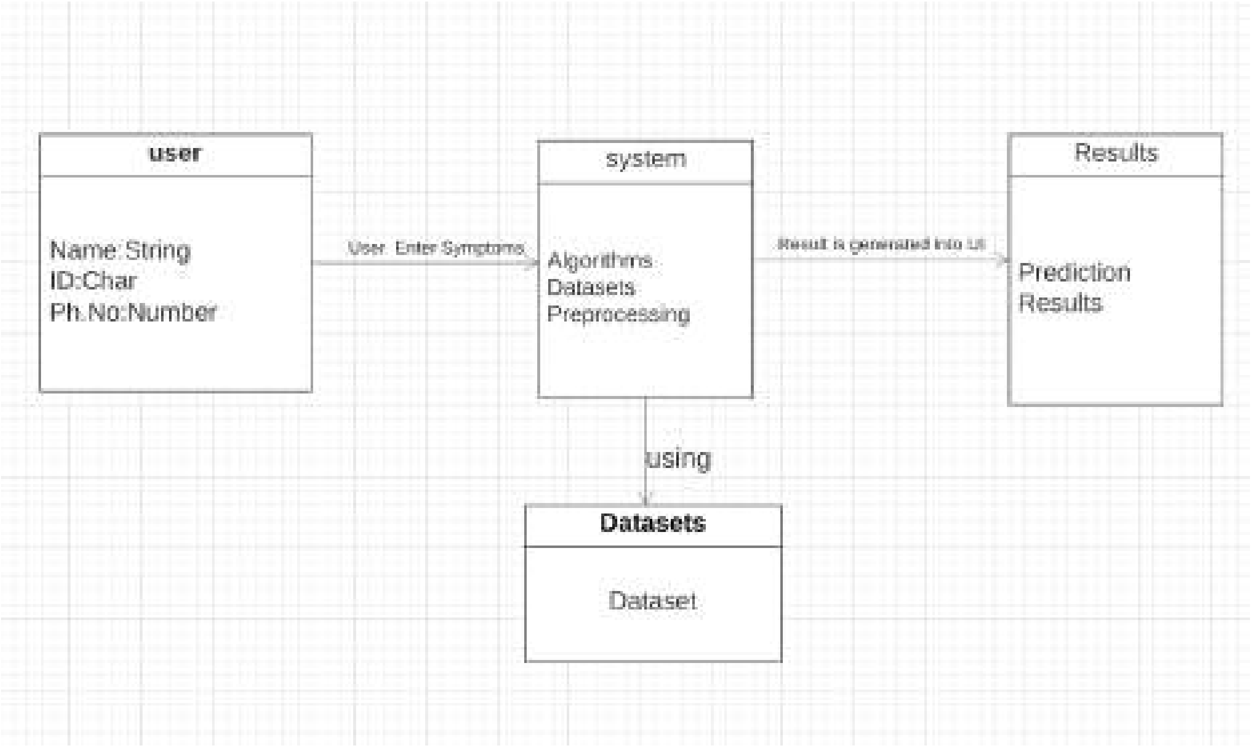


Figure 3.3: Class diagram

Class diagram consist information about all the classes that is used and all the related datasets, and all the other necessary attributes and their relationships with other entities, all these information is necessary in order to use the concept of the prediction, where the user will enter all necessary information such as user name, email, phone number, and many more attributes that is required in order to login into the system and using the files concept we will store the information of the users who are registering into the system and retrieves those information later while logging into the system.

3.4 SEQUENCEDIAGRAM

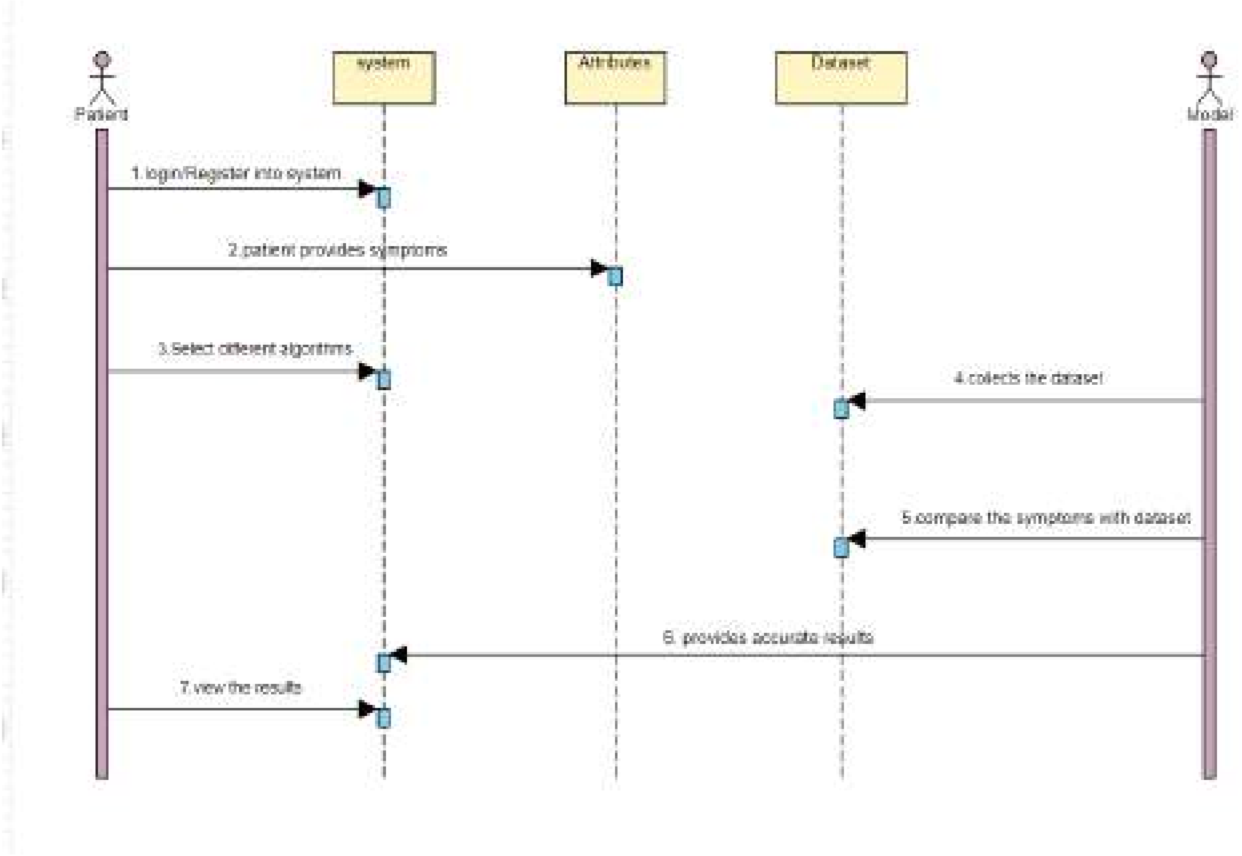

Patient
system
Attributes
Dataset
Model
1.login/Register into system
2.patient provides symptoms
3.Select different algorithms
4.collects the dataset
5.compare the symptoms with dataset
6. provides accurate results
7.view the results

Figure 3.4: Sequence diagram

This sequence diagram shows how from starting the model flows from one step to another, like he enter into the system then enters all the information’s and all other general information along with the symptoms that goes into the system, compares with the prediction model and if true is predicts the appropriate results otherwise it shows the details where the user if gone wrong while entering the information’s and it also shows the appropriate precautionary measure for the user to follow. Here the sequence of all the entities is linked to each other where the user gets started with the system.

3.5 ACTIVITY DIAGRAM

It describes about flow of activity states.

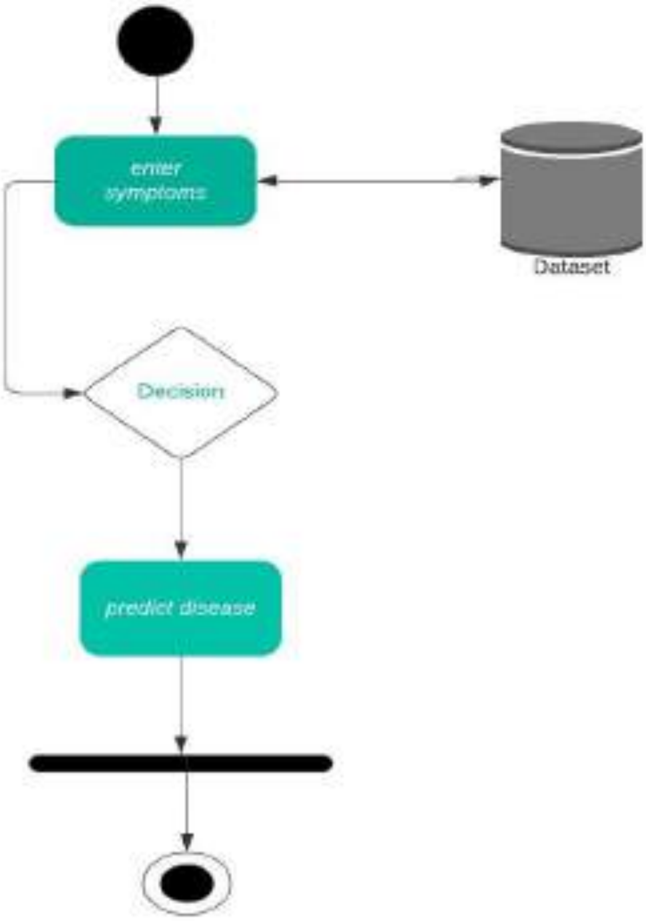


Figure 3.5: Activity diagrams

4. IMPLEMENTATION

4. IMPLEMENTATION

The project Human 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 Project “Human Disease Prediction” is implemented using python completely. Even the interface of this project is done using python’s library interface called Tkinter. Here first the user needs to register into the system in order to use the prediction, user needs to register with username, email-id, phone, age and password. All these values are stored into the file system respectively, then user has option to move forward or leave, then user needs to login to the system using the username and password which he/she provided during the time of registration. If he/she enters incorrect username and correct password then the error message will prompt stating incorrect username and if he/she enters incorrect password and correct username then the error message will prompt stating incorrect password, so both username and password is necessary in order to login to the system. After logging in the user needs to the name and needs to select the symptoms from given drop-down menu, for more accurate result the user needs to enter all the given symptoms, then the system will provide the accurate result. This prediction is basically done with the help of 4 algorithms of machine learning such as Decision Tree, Random Forest Naive Bayes and KNN. When user enters all the symptoms then he needs to press the button.

The project is designed user friendly and also secure to use ever user requires an authentication to enter into the system after which it provides the result based on the user input let me explain the complete implementation and working of project step wise below

- Once user open the system to login user needs to register by clicking on register/signup button
- After which user needs to provide some basic details of signup and then the details of user are saved in system
- Then user needs to login to have a checkup of his/her health
- When user tries to login if he provides wrong user name the system will provide a prompt message stating that the user is not found
- And if user tries to enter the wrong password the system will prompt stating that password is in correct hence the user needs to enter the correct user id and password to get in to the system
- After user enters the system user has to provide the symptoms which he/she is going through based on which we have several algorithms which predict the disease.
- The user needs to enter all the columns of symptoms to get the accurate result.

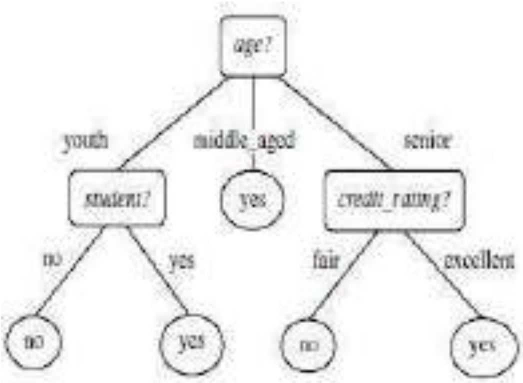
- Training and experimentation on datasets The Disease Prediction model will be trained on the dataset of diseases to do the prediction accurately. In this project 4 different algorithms were used –

  - ➢ Decision Tree
  - ➢ Random Forest
  - ➢ Naive Bayes
  - ➢ KNN

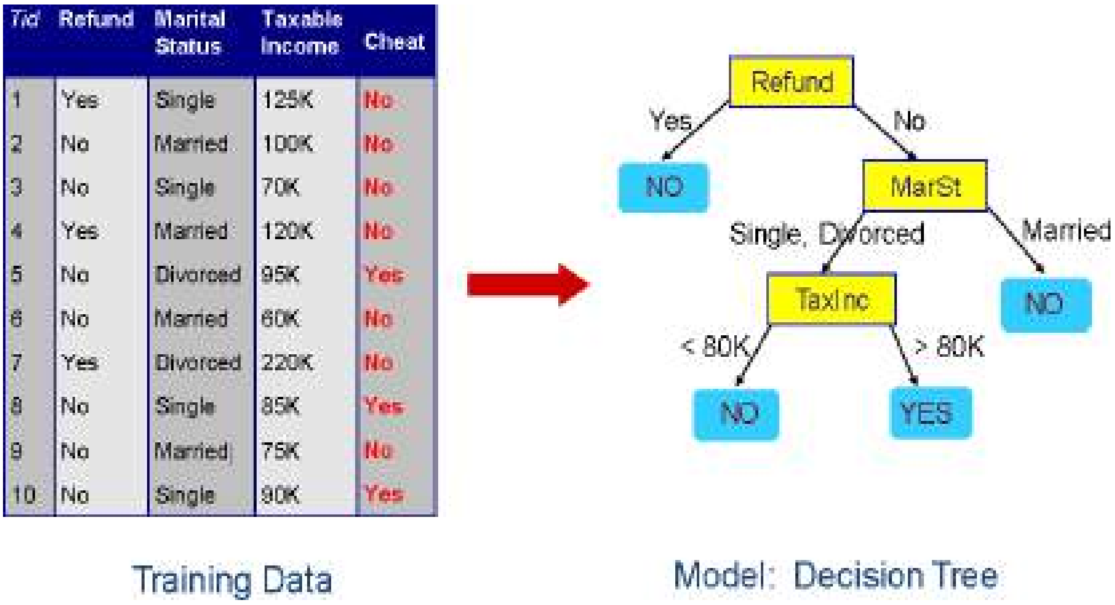
4.1 ALGORITHMS

DECISION TREE ALGORITHM

Decision tree induction is the learning of decision trees from class-labeled training tuples. A decision tree is a flowchart-like tree structure,



Decision Tree algorithm belongs to the family of supervised learning algorithms. Unlike other supervised learning algorithms, the decision tree algorithm can be used for solving regression and classification problems too. The goal of using a Decision Tree is to create a training model that can use to predict the class or value of the target variable by learning simple decision rules inferred from prior data (training data).In Decision Trees, for predicting a class label for a record we start from the root of the tree. We compare the values of the root attribute with the record’s attribute. On the basis of comparison, we follow the branch corresponding to that value and jump to the next node.

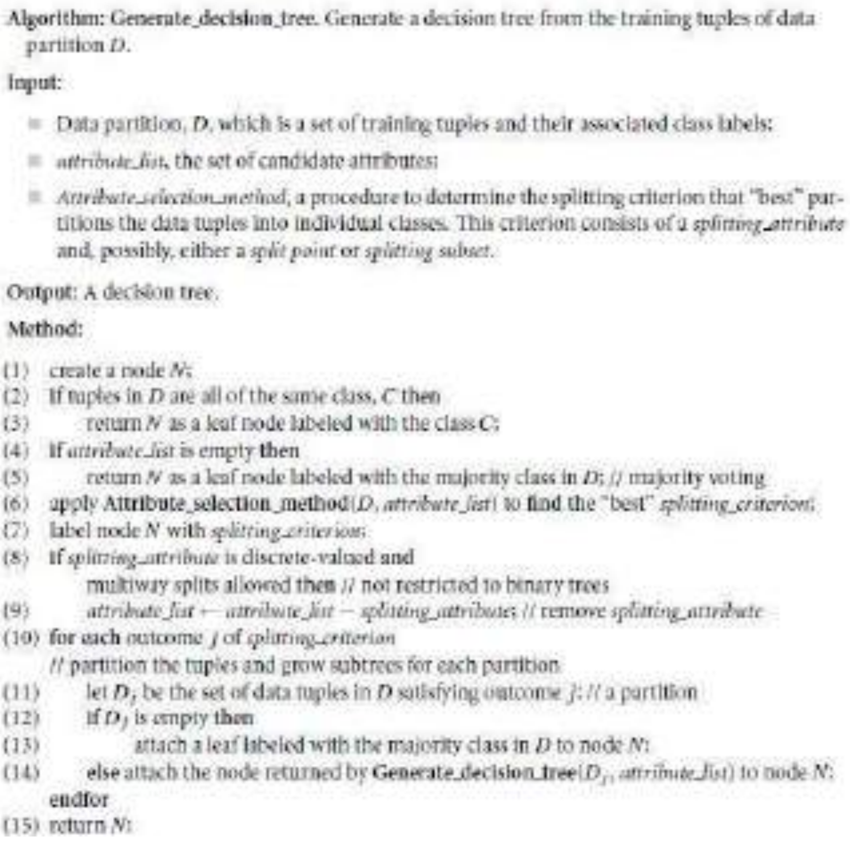


Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Training Data

Model: Decision Tree

Figure 4.1: Example of Decision Tree



Algorithm: Generate_decision_tree. Generate a decision tree from the training tuples of data partition *D*.

Input:

- Data partition, *D*, which is a set of training tuples and their associated class labels;
- *attribute_list*, the set of candidate attributes;
- *Attribute_selection_method*, a procedure to determine the splitting criterion that "best" partitions the data tuples into individual classes. This criterion consists of a *splitting_attribute* and, possibly, either a *split point* or *splitting subset*.

Output: A decision tree.

Method:

```
create a node N;
if tuples in D are all of the same class, C then
    return N as a leaf node labeled with the class C;
if attribute_list is empty then
    return N as a leaf node labeled with the majority class in D; // majority voting
apply Attribute_selection_method(D, attribute_list) to find the "best" splitting_criterion;
label node N with splitting_criterion;
if splitting_attribute is discrete-valued and
    multiway splits allowed then // not restricted to binary trees
    attribute_list ← attribute_list − splitting_attribute; // remove splitting_attribute
for each outcome j of splitting_criterion
// partition the tuples and grow subtrees for each partition
    let $D_j$ be the set of data tuples in D satisfying outcome j; // a partition
    if $D_j$ is empty then
        attach a leaf labeled with the majority class in D to node N;
    else attach the node returned by Generate_decision_tree($D_j$, attribute_list) to node N;
endfor
return N;
```

Figure 4.2: Algorithm of Decision tree

RANDOM FOREST ALGORITHM

Random forest is a supervised learning algorithm. The "forest" it builds, is an ensemble of decision trees, usually trained with the “bagging” method. The general idea of the bagging method is that a combination of learning models increases the overall result. To put it simply, random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction. One big advantage of random forest is that it can be used for both classification and regression problems, which form the majority of current machine learning systems.

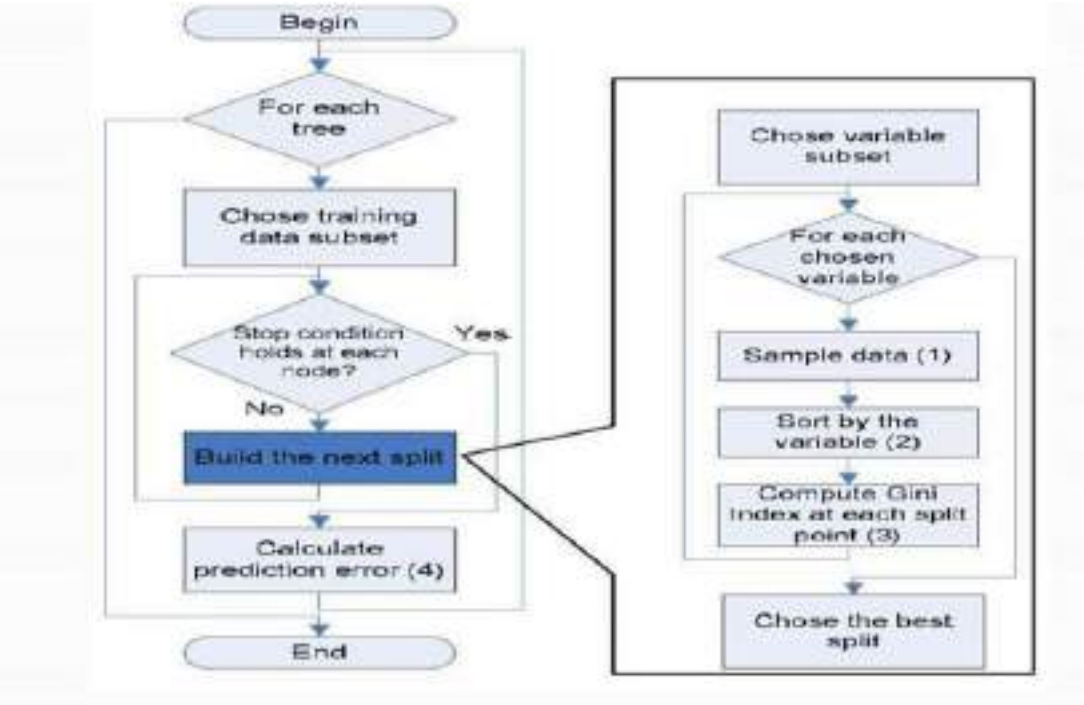


Figure 4.3: Flow Chart of Random Forest

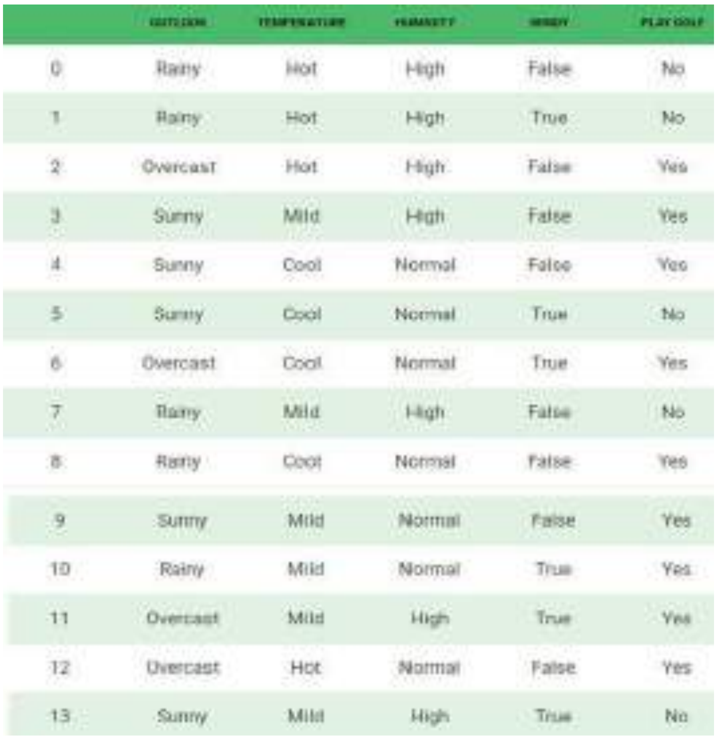
NAIVE BAYES ALGORITHM

Naive Bayes is a supervised learning algorithm used for classification tasks. Hence, it is also called Naive Bayes Classifier. As other supervised learning algorithms, naive bayes uses features to make a prediction on a target variable. The key difference is that naive bayes assumes that features are independent of each other and there is no correlation between features. However, this is not the case in real life. This naive assumption of features being uncorrelated is the reason why this algorithm is called “naive". It uses the basic concepts of probability and show how Bayes’ Theorem, the core of naive bayes classifier, is derived.

- Bayes theorem finds the probability of an event occurring given the probability of another event that has already occurred. Bayes theorem is stated mathematically as the following equation.

P(A/B) = P(B|A) P(A)/P(B)

Below is the example how this algorithm/theorem works with the dataset.



	OUTLOOK	TEMPERATURE	HUMIDITY	WINDY	PLAY GOLF
0	Rainy	Hot	High	False	No
1	Rainy	Hot	High	True	No
2	Overcast	Hot	High	False	Yes
3	Sunny	Mild	High	False	Yes
4	Sunny	Cool	Normal	False	Yes
5	Sunny	Cool	Normal	True	No
6	Overcast	Cool	Normal	True	Yes
7	Rainy	Mild	High	False	No
8	Rainy	Cool	Normal	False	Yes
9	Sunny	Mild	Normal	False	Yes
10	Rainy	Mild	Normal	True	Yes
11	Overcast	Mild	High	True	Yes
12	Overcast	Hot	Normal	False	Yes
13	Sunny	Mild	High	True	No

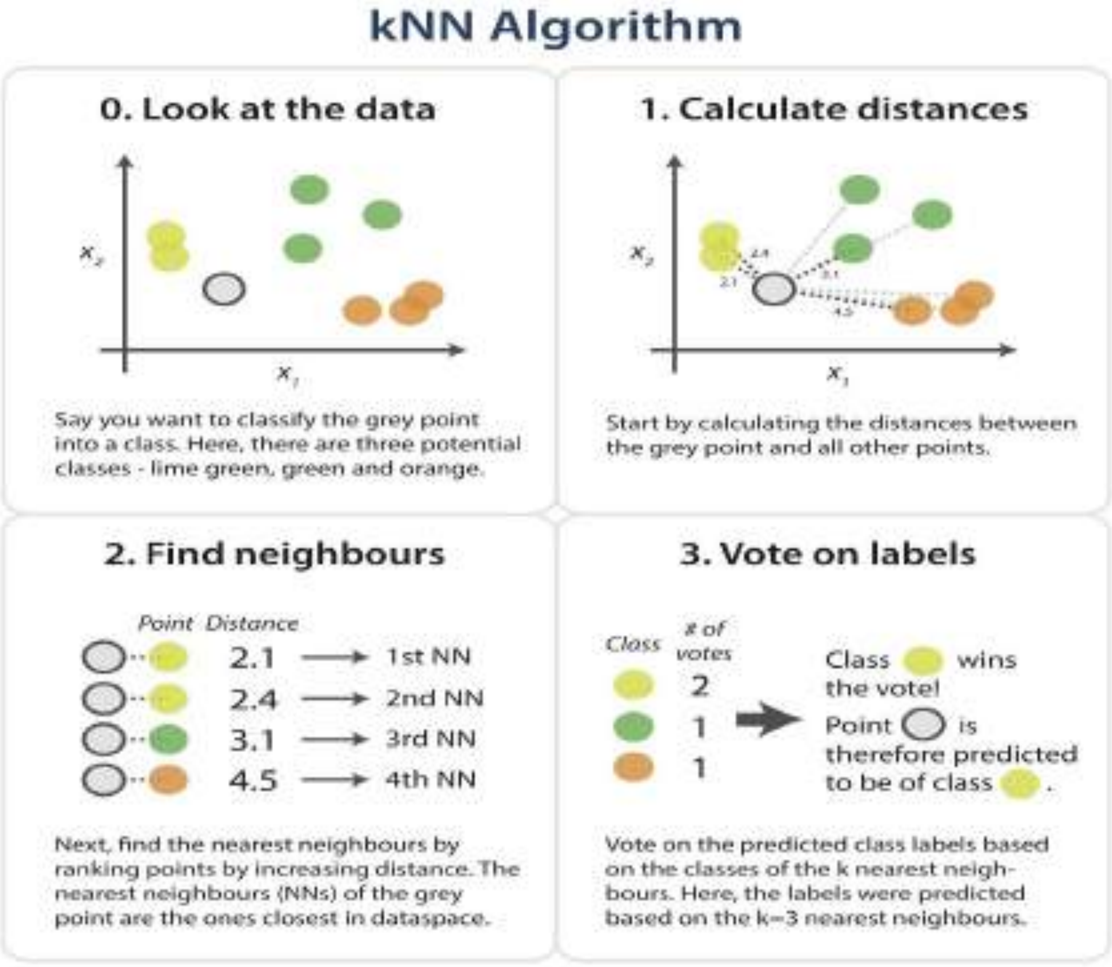
Figure 4.4: Example of Naïve Bayes

K-NEAREST NEIGHBOR

K-Nearest Neighbors (kNN) is a supervised machine learning algorithm that can be used to solve both classification and regression tasks. It is also lazy learning and non-parametric algorithm. However, it is more widely used in classification problems in the industry. To evaluate any technique we generally look at 3 important aspects:

1. Ease to interpret output
2. Calculation time
3. Predictive Power

KNN algorithm fairs across all parameters of considerations. It is commonly used for its easy of interpretation and low calculation time.



Say you want to classify the grey point into a class. Here, there are three potential classes - lime green, green and orange.

Start by calculating the distances between the grey point and all other points.

Next, find the nearest neighbours by ranking points by increasing distance. The nearest neighbours (NNs) of the grey point are the ones closest in dataspace.

Class ● wins the vote! Point ○ is therefore predicted to be of class ●.

Vote on the predicted class labels based on the classes of the k nearest neighbours. Here, the labels were predicted based on the k=3 nearest neighbours.

Figure 4.5: Example of KNN

4.2 SAMPLE CODE

```
import numpy as np
import pandas as pd
import os
from decimal import Decimal
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import accuracy_score
from sklearn.naive_bayes import MultinomialNB
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import cross_val_score

from mpl_toolkits.mplot3d import Axes3D
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt

from tkinter import *
from PIL import ImageTk, Image

def register():
    global register_screen
    register_screen = Toplevel(main_screen)
    register_screen.title("Register")
    register_screen.geometry("1000x750")
    register_screen.configure(bg="lightblue")

    global username
    global password
    global username_entry
    global password_entry
    username = StringVar()
    password = StringVar()
    email=StringVar()
    phone=StringVar()

    Label(register_screen, text="PLEASE ENTER DETAILS BELOW",font=("Times",
24),fg="blue",bg="lightblue").pack()

    username_lable = Label(register_screen, text="USER NAME ",width="300",bg="lightblue",
height="3", font=("Times", 14))
    username_lable.pack()
    username_entry = Entry(register_screen, textvariable=username,width="25")
    username_entry.pack()
    Label(register_screen, text="",bg="lightblue").pack()
   password_lable = Label(register_screen, text="PASSWORD ",bg="lightblue",width="300",
height="3", font=("Times", 14))
```

```
    password_lable.pack()
    password_entry = Entry(register_screen, textvariable=password,show= '*',width="25")
    password_entry.pack()
    Label(register_screen, text="",bg="lightblue").pack()

    email_lable = Label(register_screen, text="EMAIL",width="300",bg="lightblue", height="3",
font=("Times", 14))
    email_lable.pack()
    email_entry = Entry(register_screen, textvariable=email,width="25")
    email_entry.pack()
    Label(register_screen, text="",bg="lightblue").pack()

    phone_lable = Label(register_screen, text="PHONE NUMBER ",width="300",
bg="lightblue",height="3", font=("Times", 14))
    phone_lable.pack()
    phone_entry = Entry(register_screen, textvariable=phone,width="25")
    phone_entry.pack()
    Label(register_screen, text="",bg="lightblue").pack()

    Button(register_screen, text="REGISTER", width=20, height=2,
bg="blue",fg="white",font=("Times", 15), command = register_user).pack()


def login():
    global login_screen
    login_screen = Toplevel(main_screen)
    login_screen.title("Login")
    login_screen.geometry("1000x750")


    login_screen.configure(bg="lightblue")
    Label(login_screen, text="PLEASE DETAILS ENTER BELOW",fg="blue",
bg="lightblue",font=("Times",15)).pack()
    Label(login_screen, text="",bg="lightblue").pack()

    global username_verify
    global password_verify

    username_verify = StringVar()
    password_verify = StringVar()

    global username_login_entry
    global password_login_entry

    Label(login_screen, text="USER NAME  ",width="300", height="2",bg="lightblue",
font=("Times", 13)).pack()


username_login_entry = Entry(login_screen, textvariable=username_verify,width="25")
    username_login_entry.pack()
    Label(login_screen, text="",bg="lightblue").pack()
    Label(login_screen, text="PASSWORD ",width="300", height="2",
bg="lightblue",font=("Times", 13)).pack()

    password_login_entry = Entry(login_screen, textvariable=password_verify, show= '*',width="25")
    password_login_entry.pack()
    Label(login_screen, text="",bg="lightblue").pack()
    Button(login_screen, text="LOGIN", width=20, height=2, fg="white",bg="blue",font=("Times",
13), command = login_verify).pack()

def register_user():

    username_info = username.get()
    password_info = password.get()

    file = open(username_info, "w")
    file.write(username_info + "\n")


    file.write(password_info)
    file.close()

    username_entry.delete(0, END)
    password_entry.delete(0, END)

    Label(register_screen, text="Registration Success", fg="green", font=("Times", 14)).pack()


def login_verify():
    username1 = username_verify.get()
    password1 = password_verify.get()
    username_login_entry.delete(0, END)
    password_login_entry.delete(0, END)

    list_of_files = os.listdir()
    if username1 in list_of_files:
        file1 = open(username1, "r")
        verify = file1.read().splitlines()
        if password1 in verify:
            login_sucess()

        else:
            password_not_recognised()

    else:
        user_not_found()
```

```
def login_sucess():
    global login_success_screen
    login_success_screen = Toplevel(login_screen)
    login_success_screen.title("Success")
    login_success_screen.geometry("500x400")

Label(login_success_screen, text="Login Success",width="30", height="2", font=("Times",
12)).pack()
    Button(login_success_screen, text="OK", command=delete_login_success).pack()
    Label(login_success_screen, text="Exit",width="30", height="2", font=("Times", 12)).pack()
    Button(login_success_screen, text="OK", command=main_screen.destroy).pack()

def password_not_recognised():
    global password_not_recog_screen
    password_not_recog_screen = Toplevel(login_screen)
    password_not_recog_screen.title("Success")
    password_not_recog_screen.geometry("300x300")
    Label(password_not_recog_screen, text="Invalid Password ").pack()
    Button(password_not_recog_screen, text="OK",
command=delete_password_not_recognised).pack()
def user_not_found():
    global user_not_found_screen
    user_not_found_screen = Toplevel(login_screen)
    user_not_found_screen.title("Success")
    user_not_found_screen.geometry("300x300")
    Label(user_not_found_screen, text="User Not Found").pack()
    Button(user_not_found_screen, text="OK", command=delete_user_not_found_screen).pack()

def delete_login_success():
    login_success_screen.destroy()


def delete_password_not_recognised():
    password_not_recog_screen.destroy()

def delete_user_not_found_screen():

user_not_found_screen.destroy()

def main_account_screen():
    global main_screen
    main_screen = Tk()
    main_screen.geometry("1000x800")

    image=Image.open("pred.jpg")
    photo=ImageTk.PhotoImage(image)
    lbl=Label(main_screen,image=photo)
    lbl.place(x=0,y=0,relwidth=1,relheight=1)

    main_screen.title("Account Login")
```

```
        l1= Label(text="HUMAN DISEASE PREDICTION SYSTEM",bg="#8acddd" ,font=("Times",
18)).pack(pady=80,side=TOP,anchor="e",padx=200)\
Label(text="SELECT YOUR CHOICE",bg="#8acddd", font=("Times",
17)).pack(pady=15,side=TOP,anchor="se",padx=20)
    Label(text="",bg="#8acddd").pack()
    Button(text="LOGIN", height="3", width="30", font=("Times",13),command =
login).pack(side=TOP,anchor="e",padx=15,pady=20)
    Label(text="",bg="#8acddd").pack()
    Button(text="REGISTER", height="3", width="30",
font=("Times",13),command=register).pack(side=TOP,anchor="e",padx=15,pady=20)

    main_screen.mainloop()
main_account_screen()


l1=['back_pain','constipation','abdominal_pain','diarrhoea','mild_fever','yellow_urine',
    'yellowing_of_eyes','acute_liver_failure','fluid_overload','swelling_of_stomach',
    'swelled_lymph_nodes','malaise','blurred_and_distorted_vision','phlegm','throat_irritation',
    'redness_of_eyes','sinus_pressure','runny_nose','congestion','chest_pain','weakness_in_limbs',
    'fast_heart_rate','pain_during_bowel_movements','pain_in_anal_region','bloody_stool',
    'irritation_in_anus','neck_pain','dizziness','cramps','bruising','obesity','swollen_legs',
    'swollen_blood_vessels','puffy_face_and_eyes','enlarged_thyroid','brittle_nails',
    'swollen_extremeties','excessive_hunger','extra_marital_contacts','drying_and_tingling_lips',
    'slurred_speech','knee_pain','hip_joint_pain','muscle_weakness','stiff_neck','swelling_joints',
    'movement_stiffness','spinning_movements','loss_of_balance','unsteadiness',
    'weakness_of_one_body_side','loss_of_smell','bladder_discomfort','foul_smell_of urine',
    'continuous_feel_of_urine','passage_of_gases','internal_itching','toxic_look_(typhos)',
    'depression','irritability','muscle_pain','altered_sensorium','red_spots_over_body','belly_pain',
    'abnormal_menstruation','dischromic
_patches','watering_from_eyes','increased_appetite','polyuria','family_history','mucoid_sputum',
    'rusty_sputum','lack_of_concentration','visual_disturbances','receiving_blood_transfusion',
    'receiving_unsterile_injections','coma','stomach_bleeding','distention_of_abdomen',
    'history_of_alcohol_consumption','fluid_overload','blood_in_sputum','prominent_veins_on_calf',
    'palpitations','painful_walking','pus_filled_pimples','blackheads','scurring','skin_peeling',
    'silver_like_dusting','small_dents_in_nails','inflammatory_nails','blister','red_sore_around_nose',
    'yellow_crust_ooze']


disease=['Fungal infection','Allergy','GERD','Chronic cholestasis','Drug Reaction',
    'Peptic ulcer diseae','AIDS','Diabetes','Gastroenteritis','Bronchial Asthma','Hypertension',
    ' Migraine','Cervical spondylosis',
    'Paralysis (brain hemorrhage)','Jaundice','Malaria','Chicken pox','Dengue','Typhoid','hepatitis A',
    'Hepatitis B','Hepatitis C','Hepatitis D','Hepatitis E','Alcoholic hepatitis','Tuberculosis',
    'Common Cold','Pneumonia','Dimorphic hemmorhoids(piles)',
    'Heartattack','Varicoseveins','Hypothyroidism','Hyperthyroidism','Hypoglycemia','Osteoarthristis',
    'Arthritis','(vertigo) Paroymsal  Positional Vertigo','Acne','Urinary tract infection','Psoriasis',
    'Impetigo']
  l2=[]
for i in range(0,len(l1)):
    l2.append(0)
print(l2)
```

```
df=pd.read_csv("training.csv")


df.replace({'prognosis':{'Fungal infection':0,'Allergy':1,'GERD':2,'Chronic cholestasis':3,'Drug Reaction':4,
    'Peptic ulcer diseae':5,'AIDS':6,'Diabetes ':7,'Gastroenteritis':8,'Bronchial Asthma':9,'Hypertension ':10,
    'Migraine':11,'Cervical spondylosis':12,
    'Paralysis (brain hemorrhage)':13,'Jaundice':14,'Malaria':15,'Chicken pox':16,'Dengue':17,'Typhoid':18,'hepatitis A':19,
    'Hepatitis B':20,'Hepatitis C':21,'Hepatitis D':22,'Hepatitis E':23,'Alcoholic hepatitis':24,'Tuberculosis':25,
    'Common Cold':26,'Pneumonia':27,'Dimorphic hemmorhoids(piles)':28,'Heart attack':29,'Varicose veins':30,'Hypothyroidism':31,
    'Hyperthyroidism':32,'Hypoglycemia':33,'Osteoarthristis':34,'Arthritis':35,
    '(vertigo) Paroymsal  Positional Vertigo':36,'Acne':37,'Urinary tract infection':38,'Psoriasis':39,
    'Impetigo':40}},inplace=True)


df.head()

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()
```

```
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)

   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)

tr=pd.read_csv("testing.csv")

tr.replace({'prognosis':{'Fungal infection':0,'Allergy':1,'GERD':2,'Chronic cholestasis':3,'Drug
Reaction':4,
    'Peptic ulcer diseae':5,'AIDS':6,'Diabetes ':7,'Gastroenteritis':8,'Bronchial Asthma':9,'Hypertension
':10,
    'Migraine':11,'Cervical spondylosis':12,
    'Paralysis (brain hemorrhage)':13,'Jaundice':14,'Malaria':15,'Chicken
pox':16,'Dengue':17,'Typhoid':18,'hepatitis A':19,
    'Hepatitis B':20,'Hepatitis C':21,'Hepatitis D':22,'Hepatitis E':23,'Alcoholic
hepatitis':24,'Tuberculosis':25,
    'Common Cold':26,'Pneumonia':27,'Dimorphic hemmorhoids(piles)':28,'Heart attack':29,'Varicose
veins':30,'Hypothyroidism':31,
    'Hyperthyroidism':32,'Hypoglycemia':33,'Osteoarthristis':34,'Arthritis':35,
    '(vertigo) Paroymsal  Positional Vertigo':36,'Acne':37,'Urinary tract infection':38,'Psoriasis':39,
    'Impetigo':40}},inplace=True)

tr.head()
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)
accuracy_list = []
cross_accuracy_list = []
model_list = []

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:
        dt= DecisionTreeClassifier()
        dt = dt.fit(X, y)
# prediction of labels for the test data
        scores_dt = cross_val_score(dt, X, y, cv=5)
# mean of cross val score (accuracy)
        score= round(Decimal(scores_dt.mean() * 100), 2)
        cross_accuracy_list.append(score)
        print(f"Cross Validation Accuracy (DT): {score+1}%")

        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 = dt.predict(inputtest)
        predicted=predict[0]
        h='no'
        for a in range(0,len(disease)):
            if(predicted == a):
                h='yes'
                break
        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('database1.db')
        c = conn.cursor()
        c.execute("CREATE TABLE IF NOT EXISTS DecisionTree(Name StringVar,Age
StringVar,Symtom1 StringVar,Symtom2 StringVar,Symtom3 StringVar,Symtom4 TEXT,Symtom5
TEXT,Disease StringVar)")
        c.execute("INSERT INTO
DecisionTree(Name,Age,Symtom1,Symtom2,Symtom3,Symtom4,Symtom5,Disease)
VALUES(?,?,?,?,?,?,?,?)",(NameEn.get(),AgeEn.get(),Symptom1.get(),Symptom2.get(),Symptom3.g
et(),Symptom4.get(),Symptom5.get(),pred1.get()))
        conn.commit()
        c.close()
        conn.close()


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:
        rf = RandomForestClassifier(n_estimators=10, criterion='entropy')
        rf = rf.fit(X, y)
        scores_rf = cross_val_score(rf, X, y, cv=5)
# mean of cross val score (accuracy)
        score= round(Decimal(scores_rf.mean() * 100), 2)
        cross_accuracy_list.append(score)
        t1=print(f"Cross Validation Accuracy (RF): {score}%")
        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 = rf.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('database1.db')
        c = conn.cursor()
        c.execute("CREATE TABLE IF NOT EXISTS RandomForest(Name StringVar,Age
StringVar,Symtom1 StringVar,Symtom2 StringVar,Symtom3 StringVar,Symtom4 TEXT,Symtom5
TEXT,Disease StringVar)")
        c.execute("INSERT INTO
RandomForest(Name,Age,Symtom1,Symtom2,Symtom3,Symtom4,Symtom5,Disease)
VALUES(?,?,?,?,?,?,?,?)",(NameEn.get(),AgeEn.get(),Symptom1.get(),Symptom2.get(),Symptom3.g
et(),Symptom4.get(),Symptom5.get(),pred2.get()))
        conn.commit()
        c.close()
        conn.close()

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:
    knn = KNeighborsClassifier(n_neighbors=7, weights='distance', n_jobs=4)
    knn = knn.fit(X, y)
    scores_knn = cross_val_score(knn, X, y, cv=5)

# prediction of labels for the test data
    score = round(Decimal(scores_knn.mean() * 100), 2)
    cross_accuracy_list.append(score)
    print(f"Cross Validation Accuracy (KNN): {score}%")
    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('database1.db')
    c = conn.cursor()
    c.execute("CREATE TABLE IF NOT EXISTS KNearestNeighbour(Name StringVar,Age
StringVar,Symtom1 StringVar,Symtom2 StringVar,Symtom3 StringVar,Symtom4 TEXT,Symtom5
TEXT,Disease StringVar)")
    c.execute("INSERT INTO
KNearestNeighbour(Name,Age,Symtom1,Symtom2,Symtom3,Symtom4,Symtom5,Disease)
VALUES(?,?,?,?,?,?,?,?)",(NameEn.get(),AgeEn.get(),Symptom1.get(),Symptom2.get(),Symptom3.g
et(),Symptom4.get(),Symptom5.get(),pred4.get()))
    conn.commit()
    c.close()
    conn.close()
```

```
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:
        mnb = MultinomialNB()
        mnb = mnb.fit(X, y)
# Cross Validation Accuracy MNB
# performing cross validation with 5 different splits
        scores_mnb = cross_val_score(mnb, X, y, cv=5)
# mean of cross val score (accuracy)
        score = round(Decimal(scores_mnb.mean() * 100), 2)
        cross_accuracy_list.append(score)
        print(f"Cross Validation Accuracy (MNB): {score}%")
        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 = mnb.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('database1.db')
        c = conn.cursor()
        c.execute("CREATE TABLE IF NOT EXISTS NaiveBayes(Name StringVar,Age
StringVar,Symtom1 StringVar,Symtom2 StringVar,Symtom3 StringVar,Symtom4 TEXT,Symtom5
TEXT,Disease StringVar)")
```

```
 c.execute("INSERT INTO
NaiveBayes(Name,Age,Symtom1,Symtom2,Symtom3,Symtom4,Symtom5,Disease)
VALUES(?,?,?,?,?,?,?,?)",(NameEn.get(),AgeEn.get(),Symptom1.get(),Symptom2.get(),Symptom3.get(),Symptom4.get(),Symptom5.get(),pred3.get()))
        conn.commit()
        c.close()
        conn.close()


root.configure(background='lightblue')
root.title('Human Disease Prediction System')

Symptom1 = StringVar()
Symptom1.set("Select Here")
Symptom2 = StringVar()
Symptom2.set("Select Here")
Symptom3 = StringVar()
Symptom3.set("Select Here")
Symptom4 = StringVar()
Symptom4.set("Select Here")
Symptom5 = StringVar()
Symptom5.set("Select Here")
Name = StringVar()
Age=StringVar()


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")

    pred1.set(" ")
    pred2.set(" ")
    pred3.set(" ")
    pred4.set(" ")
    try:
        prev_win.destroy()
        prev_win=None
    except AttributeError:
        pass
```

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

w2 = Label(root, justify=LEFT, text="HUMAN DISEASE PREDICTION", fg="blue",
bg="lightblue")
w2.config(font=("Times",30,"bold italic"))
w2.grid(row=1, column=0, columnspan=2, padx=100)

NameLb = Label(root, text="Name of the Patient", fg="black", bg="lightblue")
NameLb.config(font=("Times",15,"bold italic"))
NameLb.grid(row=3, column=0, pady=15, sticky=W)
AgeLb = Label(root, text="Age", fg="black", bg="lightblue")
AgeLb.config(font=("Times",15,"bold italic"))
AgeLb.grid(row=4, column=0, pady=15, sticky=W)

S1Lb = Label(root, text="Symptom 1", fg="Black", bg="lightblue")
S1Lb.config(font=("Times",15,"bold italic"))
S1Lb.grid(row=5, column=0, pady=10, sticky=W)

S2Lb = Label(root, text="Symptom 2", fg="Black", bg="lightblue")
S2Lb.config(font=("Times",15,"bold italic"))
S2Lb.grid(row=6, column=0, pady=10, sticky=W)

S3Lb = Label(root, text="Symptom 3", fg="Black",bg="lightblue")
S3Lb.config(font=("Times",15,"bold italic"))
S3Lb.grid(row=7, column=0, pady=10, sticky=W)

S4Lb = Label(root, text="Symptom 4", fg="Black", bg="lightblue")
S4Lb.config(font=("Times",15,"bold italic"))
S4Lb.grid(row=8, column=0, pady=10, sticky=W)

S5Lb = Label(root, text="Symptom 5", fg="Black", bg="lightblue")
S5Lb.config(font=("Times",15,"bold italic"))
S5Lb.grid(row=9, column=0, pady=10, sticky=W)


NameEn = Entry(root,textvariable=Name)
NameEn.grid(row=3, column=1)
AgeEn = Entry(root,textvariable=Age)
AgeEn.grid(row=4, column=1)


OPTIONS = sorted(l1)
S1 = OptionMenu(root, Symptom1,*OPTIONS)
S1.grid(row=5, column=1)
```

```
S2 = OptionMenu(root, Symptom2,*OPTIONS)
S2.grid(row=6, column=1)

S3 = OptionMenu(root, Symptom3,*OPTIONS)
S3.grid(row=7, column=1)

S4 = OptionMenu(root, Symptom4,*OPTIONS)
S4.grid(row=8, column=1)

S5 = OptionMenu(root, Symptom5,*OPTIONS)
S5.grid(row=9, column=1)

dst = Button(root,
text="Prediction",command=lambda:[DecisionTree(),randomforest(),NaiveBayes(),KNN()],
bg="blue",fg="lightblue")
dst.config(font=("Times",15,"bold italic"))
dst.grid(row=5, column=3,padx=10)

rs = Button(root,text="Reset Inputs", command=Reset,bg="blue",fg="lightblue",width=15)
rs.config(font=("Times",15,"bold italic"))
rs.grid(row=7,column=3,padx=10)

ex = Button(root,text="Exit System", command=Exit,bg="blue",fg="lightblue",width=15)
ex.config(font=("Times",15,"bold italic"))
ex.grid(row=9,column=3,padx=10)


t2=Label(root,font=("Times",15,"bold italic"),text="Result",height=1,bg="blue"
        ,width=40,fg="white",textvariable=pred2,relief="sunken").grid(row=13, column=1, padx=10)

 root.mainloop()
```

5. TESTING

5.TESTING

5.1 INTRODUCTION TO 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 and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

5.2 TYPES OF TESTING

5.2.1 UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual softwareunitsoftheapplication.itisdoneafterthecompletionofanindividualunitbeforeintegration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

5.2.2 INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with thebasicoutcomeofscreensorfields.Integrationtestsdemonstratethatalthoughthecomponentswereindividuallysatisfaction, as shown by successfully 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.

5.2.3 FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : Identified classes of valid input must be accepted.

Invalid Input : Identified classes of invalid input must be rejected.

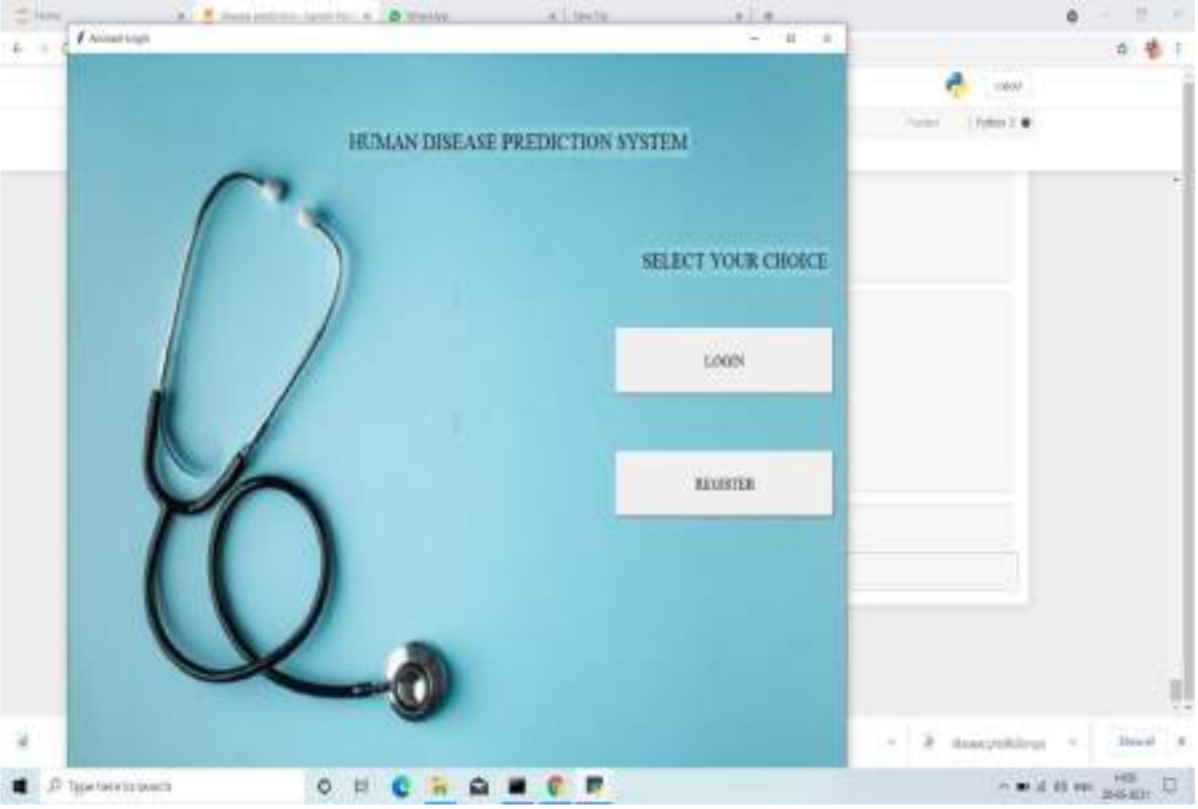
Functions : Identified functions must be exercised.

Output : Identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows, data fields, predefined processes.

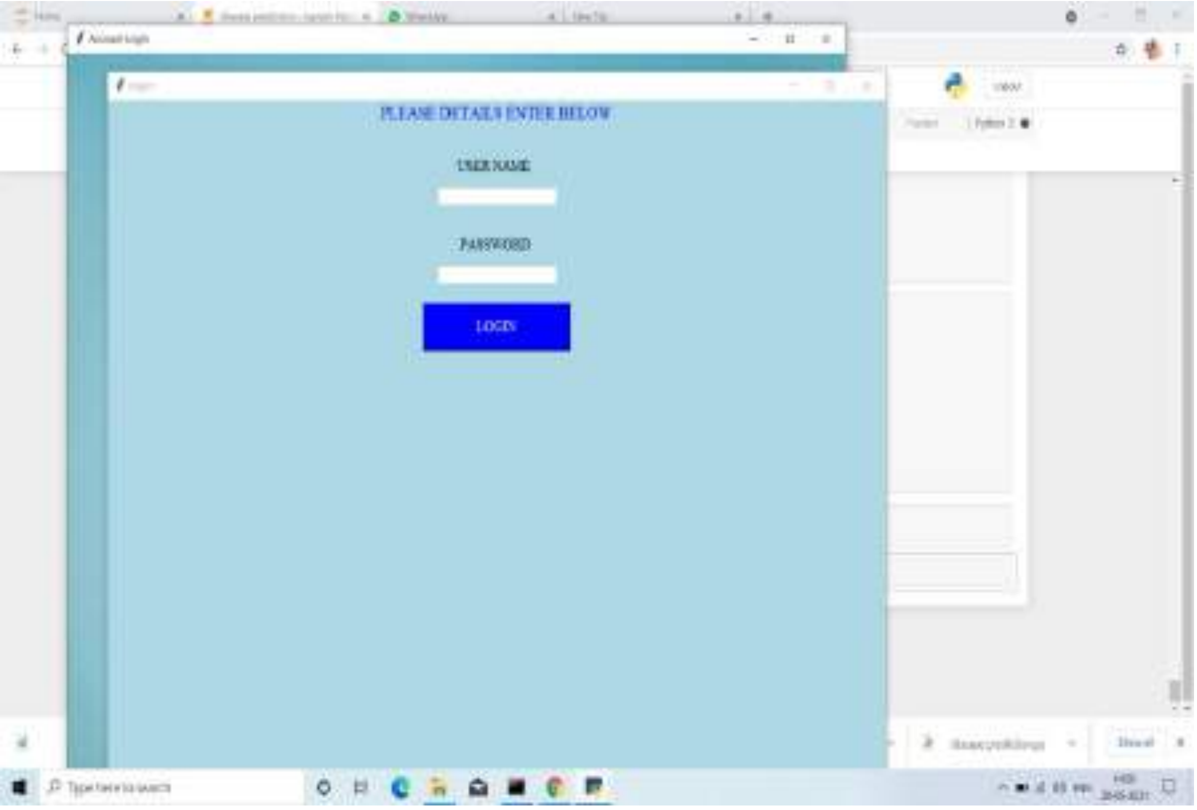
6. OUTPUT



HUMAN DISEASE PREDICTION SYSTEM

SELECT YOUR CHOICE

SCREEN SHOT 1: LOGIN AND REGISTRATION PAGE

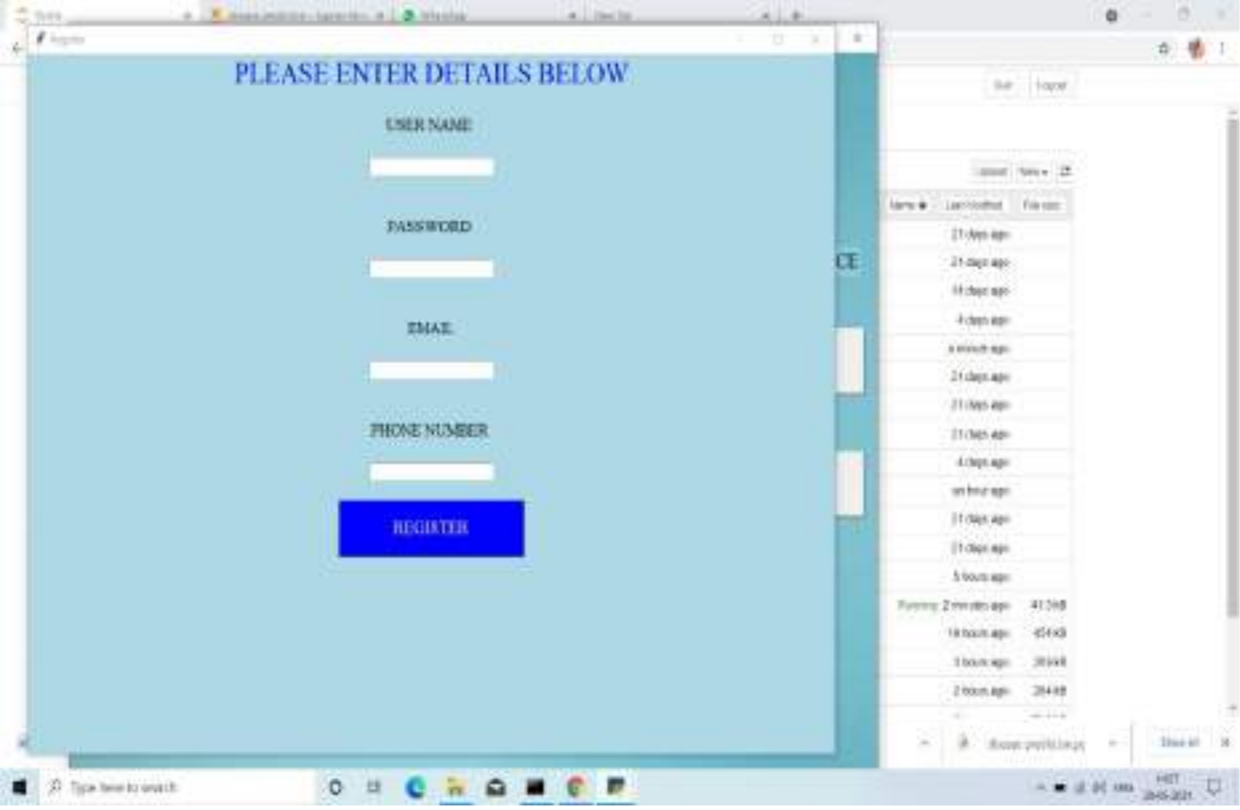


PLEASE DETAILS ENTER BELOW

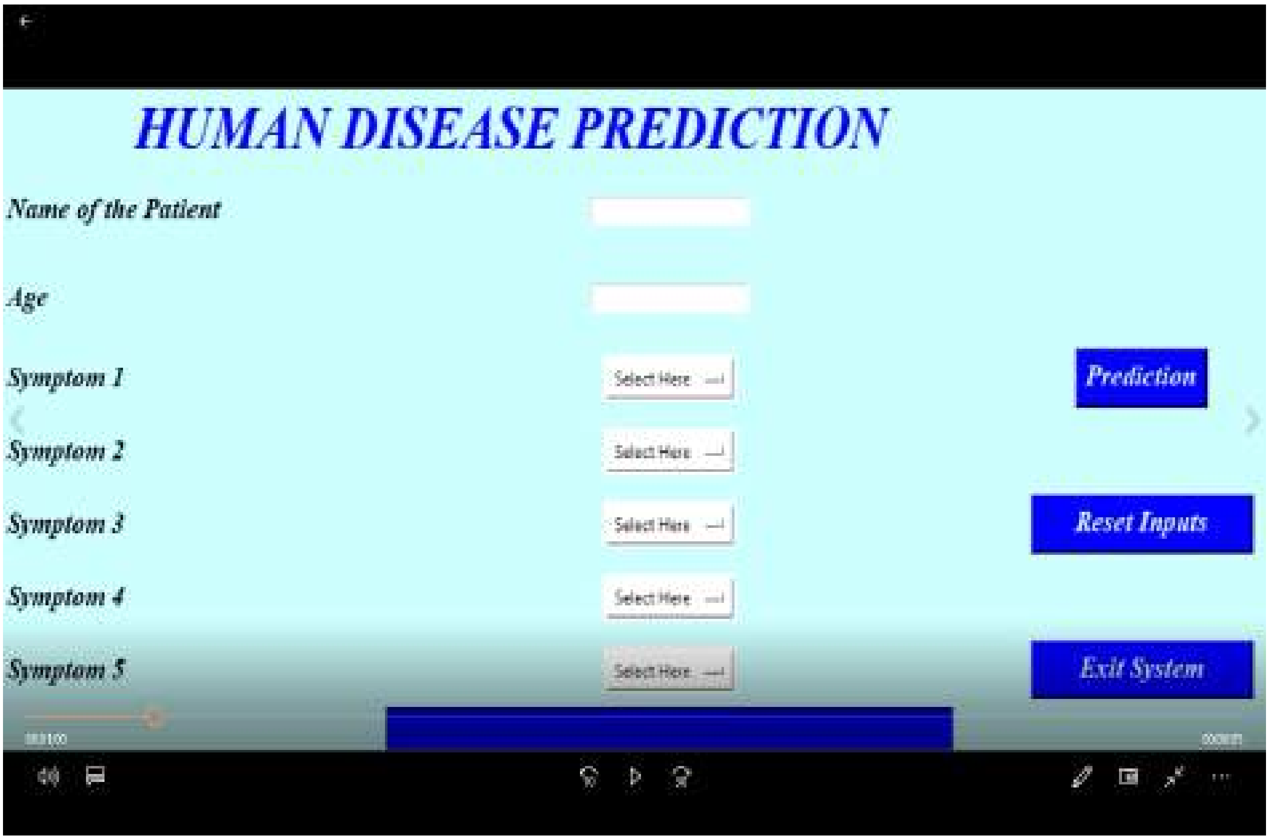
USER NAME

PASSWORD

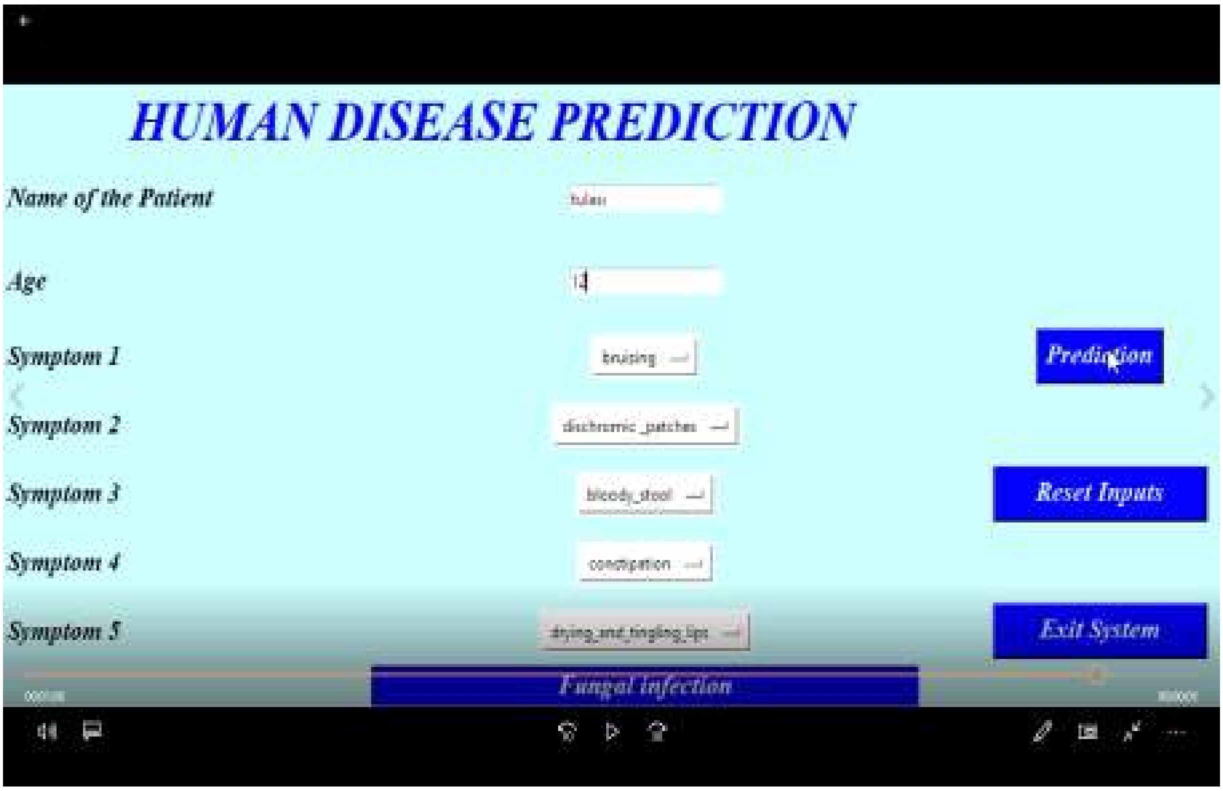
SCREEN SHOT 2: LOGIN PA



SCREEN SHOT 3: REGISTRATION PAGE



SCREEN SHOT 4: MAIN WINDOW OF DISEASE PREDICTIO



HUMAN DISEASE PREDICTION

Name of the Patient tulasi

Age 12

Symptom 1 bruising

Symptom 2 dischromic _patches

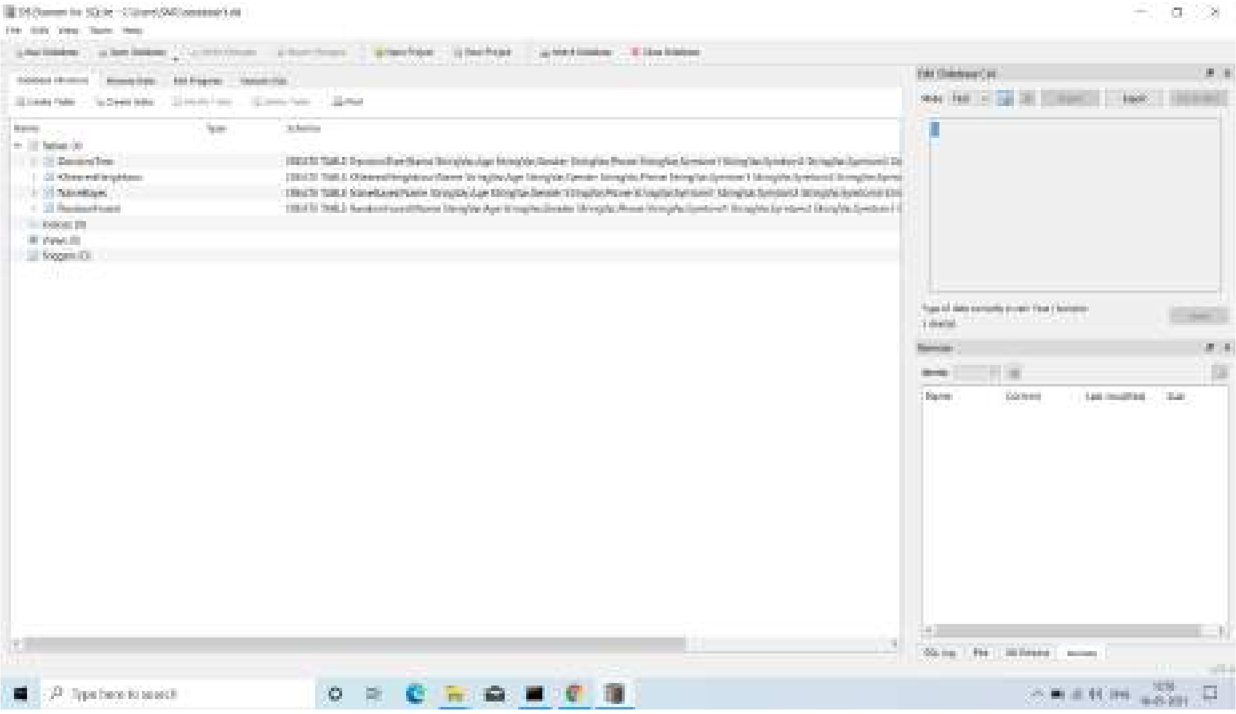
Symptom 3 bloody_stool

Symptom 4 constipation

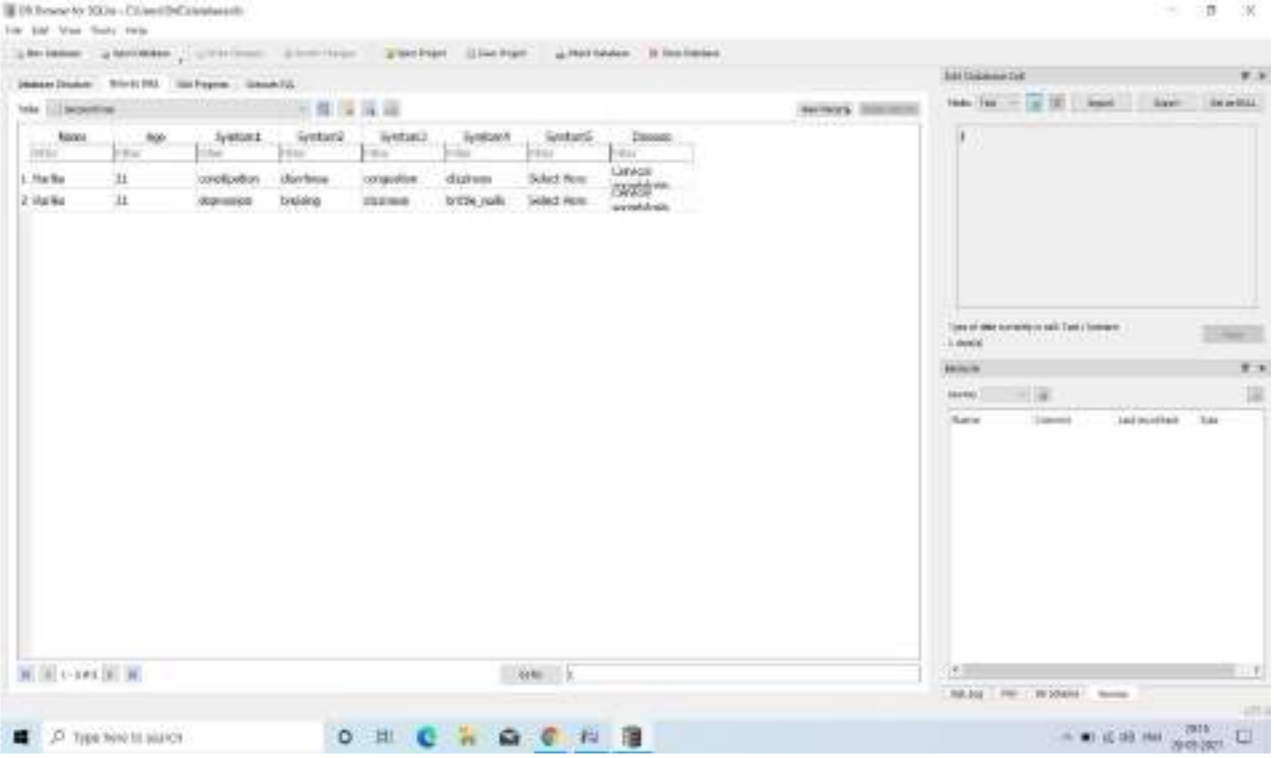
Symptom 5 drying_and_tingling_lips

Fungal infection

SCREEN SHOT 5: FINAL OUTPUT AFTER SYMPTOMS IS SLECTED



SCREEN SHOT 6: CREATION OF DATABAS



	Name	Age	Symptom1	Symptom2	Symptom3	Symptom4	Symptom5	Disease
1	Harika	11	constipation	diarrhoea	congestion	dizziness	Select Here	Cervical spondylosis
2	Harika	11	depression	bruising	dizziness	brittle_nails	Select Here	Cervical spondylosis

SCREEN SHOT 7: STORING SYMPTOMS AND THE PREDICTED DISEASE IN DATABASE

7. CONCLUSION

7. CONCLUSION & FUTURESCOPE

7.1 PROJECT CONCLUSION

Nowadays machine learning is playing important role in the medical prediction field of the health sector. Health is a very important aspect of everyone’s life.

So, after doing the research and comparison of all the algorithms and theorems of machine learning we have come to the conclusion that all those algorithms such as Decision Tree, KNN, Naive ayes, and Random Forest Algorithms all are used in building a disease prediction system which predicts the disease of the patients from which he/she is suffering from. But, according to accuracy percentage Decision tree is predicting the most accurate result.

7.2 FUTURE SCOPE

In future we can use other advanced algorithms by downloading the modules directly into the project files. The software can be developed further to include lot of modules because the proposed system is developed on the view of future. We can connect to other databases by including them.

8. GITHUB LINK:

https://github.com/Balemlaharika/Human-disease-prediction.git

9. BIBILOGRAPHY

9.1 REFERENCES

- Machine Learning by Tom M. Mitchell
- Python Machine Learning by Sebastian Raschka and Vahid Mirjalili
- Rules of Machine Learning: Best Practices for ML Engineering by Martin Zinkevich

9.2 WEBSITES

[1] https://link.springer.com/chapter/10.1007/978-981-15-50898_55#:~:text=Machine%20Learning%20is%20an%20emerging,dataset%20ana%20predict%20the%20disease.

[2] https://www.expert.ai/blog/machine-learning-definition/

[3] https://bmcmedinformdecismak.biomedcentral.com/articles/10.1186/s12911-019-1004-8

[4] https://www.youtube.com/watch?v=3YmAbta16yk&t=1s

[5] https://www.youtube.com/watch?v=NUw2cPNwDZs

[6] https://www.youtube.com/watch?v=I7NrVwm3apg

[7] https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/

[8] https://www.geeksforgeeks.org/decision-tree-introduction-example/#:~:text=Decision%20tree%20algorithm%20falls%20under%20the%20category%20of%20supervised%20learning.&text=Decision%20tree%20uses%20the%20tree,internal%20node%20of%20the%20tree

[9] https://www.tutorialspoint.com/machine_learning_with_python/machine_learning_with_python_classification_algorithms_random_forest.htm#:~:text=Random%20forest%20is%20a%20supervised,classification%20as%20well%20as%20regression.&text=Similarly%2C%20random%20forest%20algorithm%20creates,solution%20by%20means%20of%20voting

[10] https://www.tutorialspoint.com/machine_learning_with_python/machine_learning_with_python_knn_algorithm_finding_nearest_neighbors.htm#:~:text=With%20the%20help%20of%20KNN,Image%20Recognition%20and%20Video%20Recognition

HUMAN DISEASE PREDICTION

A.Mahendar[1], P.Krishna tulasi[2], B.Harika[3], N.Sindhu[4]

[1]Associate Professor, CSE, CMR Technical Campus, Hyderabad, India.

[2, 3, 4] Student, CSE, CMR Technical Campus, Hyderabad, India.

***ABSTRACT*— Disease prediction using Machine Learning is a system that predicts the disease based on the information or the symptoms provided by the user based on that accurate result shown. Currently, the health industry plays a major role in curing the illness of the patients, just in case he/she doesn't want to travel to the hospital, therefore just by giving the symptoms and all other validated information the user can get to know the illness they are suffering from. Machine Learning algorithms such as Naive Bayes, Decision Tree, KNN and Random forest are used on the provided dataset and predict the disease of the patient. The implementation is done through the Python Programming Language.**

***Keywords*— Machine Learning, Naive Bayes algorithm, KNN algorithm, Decision Tree algorithm, Random forest algorithm, Python.**

I. INTRODUCTION

Machine learning[1] is a type of artificial intelligence application that enables self-learning from data and then applies the learning without the need for human intervention and improves from experience without being explicitly programmed. In actuality, there are many different types of machine learning, as well as many strategies and algorithms so how to best employ them. It is totally different from traditional programming, here data and the output is given to the computer and in return, it gives us the program which provides a solution to various problems. Machine learning is the combination of algorithms, datasets and programs. Now-a day's. Machine learning is playing a major role in health industries, Stock market ,IT industry etc.,

Machine Learning [2] is complex in itself that is why it has been divided into two main areas, supervised learning [3] and unsupervised learning. Each one has a reason and action with Machine Learning, to yield particular results and utilizing various forms of data. Approximately 70 percent of Machine Learning is supervised learning such as Decision tree, Random forest, KNN etc. while unsupervised learning ranges from 10-20 percent such as K-means clustering, Apriority etc. Other method that is used less often is reinforcement learning.

II. LITERATURE REVIEW

Health care center's purpose is to improve the current health of the community. A health care institution such as hospitals or medical centers would essentially consist of many doctors who were qualified and specialized in treating various patients of their current disease that they are suffering from and trying to cure them and give proper health.

Currently, new technologies have developed and came into existence to improve people's daily life and routine especially in health care centers. The project disease prediction using machine learning is developed to overcome general disease or illness in early stages as we all know in the competitive environment of economic development the people has involved so much that he/she is not worried about health according to research there are 40% people who ignore the general disease or illness which leads to harmful disease later. The main reason for ignorance is laziness to consult a doctor or hospital and time concerns the people have involved themselves so much that they have no time to make an appointment and consult the doctor which later results in fatal dangerous disease. According to research there are 70% of people in India suffering from general disease and 25% of people are facing death due to early ignorance the main motive to develop this project Is that a user can sit at their place and have a check-up of their health the UI designed in such a simple way that everyone can easily operate it and can have a check-up.

III. ARCHITECTURE

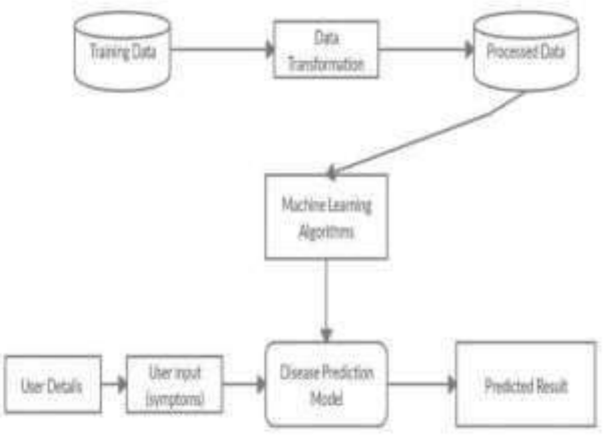


Fig 1: Architecture of Human Disease Prediction

From the above architectural diagram, first the training data from the dataset is transformed by removing the raw and the noisy data and then the data is processed further. Now the machine learning algorithms will be applied or predict the disease. When the user enters the basic details and the symptoms from which the patient is suffering. After entering the symptoms the Disease prediction model will predict the accurate disease as the output.

IV. EXISTING SYSTEM

Prediction using traditional methods [4] and models involves various risk factors and it contains various measures of algorithms such as datasets, programs and much more. In this system, there are many models which may not provide accurate results and use only one or two algorithms for predicting the resultant output. High-risk and low-risk patient classification is done based on the tests that are done in a group. But these models are only valuable in clinical situations and not in the big industry sector. So, to include the disease predictions in various health-related industries, we have used the concepts of machine learning and supervised learning methods to build the prediction system. This system can predict the disease but not the sub-type of the disease and it fails to predict the condition of the people, the predictions of the disease have been indefinite and non-specific. And the most difficult thing in the current system is it makes the patient undergo lengthy procedures and questionnaires which is time-consuming process.

Disadvantages:

- Only a few diseases can be detected.
- Less accuracy.
- Weak generalization.
- Time-consuming.

V. PROPOSED SYSTEM

The proposed system [5] of disease prediction using machine learning is that we have used many techniques and algorithms and all other various tools to build a system that 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 taking those symptoms we are comparing with the system's previously available dataset patient's disease.we will predict the accurate percentage of disease of the patient. In proposed system of disease prediction using machine learning is that we have used many different algorithms such as Decision tree, KNN, Naive Bayes, and Random Forest[6] which predicts the disease of the patient by entering the basic details of the patient like name and the symptoms from which the patient is suffering from. Now, by taking those symptoms we are comparing with the system dataset that is previously available which will predict accurate disease. Our main motive with this system is to be connecting the bridge between doctors and patients.

Advantages:

- Data can be stored and used for doctor reference.
- Analysis based on multiple algorithms.
- More accurate.

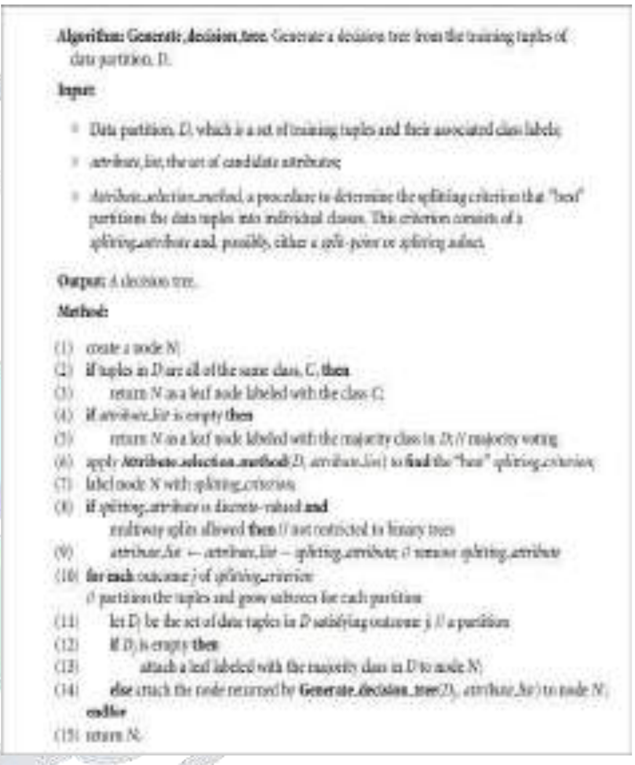
VI. ALGORITHMS

1. DECISION TREE ALGORITHM:

Decision Tree [8] algorithm belongs to supervised learning algorithms. Instead of other supervised learning algorithms, the decision tree algorithm can be used for solving regression and classification problems too.

The reason for using a Decision Tree is to create a training model that can use to predict the class or value of the variable by learning decision rules which are used and inferred from the training data.

In Decision Trees, for predicting a class label we start from the root point of the tree. Next, we will compare the values of the root point with the record's attribute values. At this point of Comparison, we follow the branch value corresponding to that value and shift to the next node.



Algorithm: Generate_decision_tree. Generate a decision tree from the training tuples of data partition, D.

Input:

- Data partition, D, which is a set of training tuples and their associated class labels;
- attribute_list, the set of candidate attributes;
- Attribute_selection_method, a procedure to determine the splitting criterion that "best" partitions the data tuples into individual classes. This criterion consists of a splitting_attribute and, possibly, either a split-point or splitting subset.

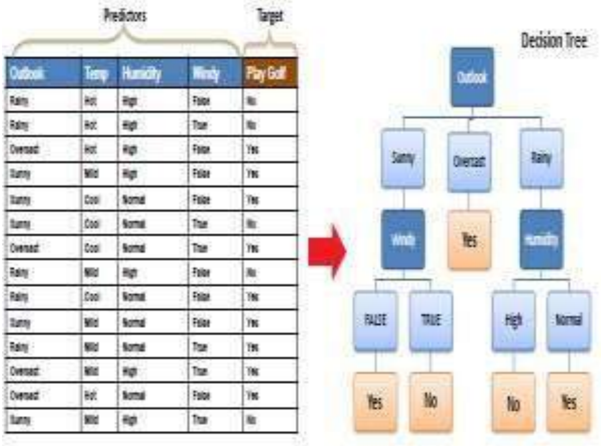
Output: A decision tree.

Method:

```
(1)  create a node N;
(2)  if tuples in D are all of the same class, C, then
(3)      return N as a leaf node labeled with the class C;
(4)  if attribute_list is empty then
(5)      return N as a leaf node labeled with the majority class in D; // majority voting
(6)  apply Attribute_selection_method(D, attribute_list) to find the "best" splitting_criterion;
(7)  label node N with splitting_criterion;
(8)  if splitting_attribute is discrete-valued and
         multiway splits allowed then // not restricted to binary trees
(9)      attribute_list ← attribute_list − splitting_attribute; // remove splitting_attribute
(10) for each outcome j of splitting_criterion
     // partition the tuples and grow subtrees for each partition
(11)     let Dj be the set of data tuples in D satisfying outcome j; // a partition
(12)     if Dj is empty then
(13)         attach a leaf labeled with the majority class in D to node N;
(14)     else attach the node returned by Generate_decision_tree(Dj, attribute_list) to node N;
     endfor
(15) return N;
```

Fig 2: Decision tree Algorithm

EXAMPLE:



Outlook	Temp	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

Fig 3: Decision tree Example

2. RANDOM FOREST ALGORITHM:

Random forest [9] is an algorithm. The "forest" it builds, is an ensemble of decision trees, typically trained with the “bagging” technique. The main idea of the bagging technique is that a mixture of learning models wills increases the final result.

In simple words, random forest builds multiple decision trees and combines them to get a more accurate and stable prediction and prevents the drawback of overfitting.

One massive advantage of random forest is that it may be used for both classification and regression issues, which form the bulk of current machine learning systems.

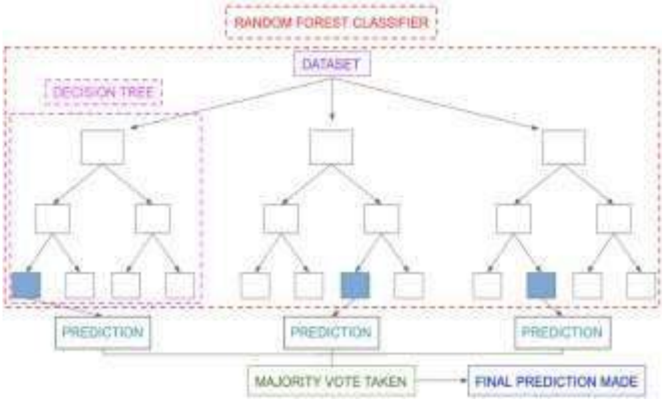
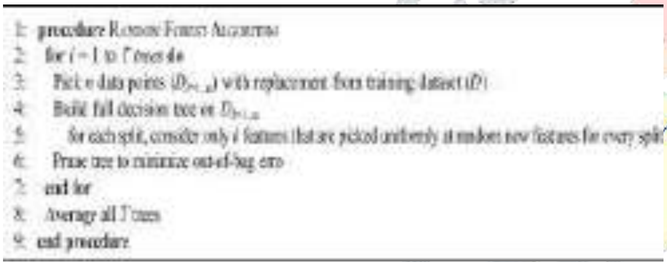


Fig 3: Working of Random Forest

The following are the steps for the Random Forest algorithm:



```
procedure Random Forest Algorithm
  for i = 1 to T trees do
    Pick n data points (D_{i=1..n}) with replacement from training dataset (D)
    Build full decision tree on D_{i=1..n}
      for each split, consider only k features that are picked uniformly at random new features for every split
    Prune tree to minimize out-of-bag error
  end for
  Average all T trees
end procedure
```

Fig 4: Random Forest Algorithm

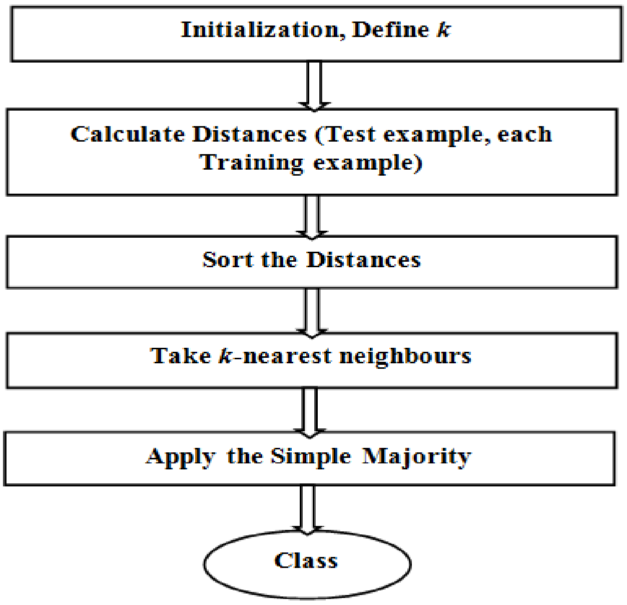
3. K-NEAREST NEIGHBOR ALGORITHM:

K-Nearest Neighbour [10] is one of the easiest Machine Learning algorithms based on the Supervised Learning technique. KNN algorithm detects the similarity between the new data and available data cases and puts the new case data into the file that is most similar to the available one.

KNN algorithm stores all the data which is available and classifies a new case data point based on the similarity. This means when new data appears then it can be easily classified into a well-suited category by using the KNN algorithm technique. It can be used for Regression issues and also for Classification but mostly it is used for Classification issues.KNN is a non-parametric algorithm technique, which means it can not make any assumption or idea on underlying data.

KNN is also called a **lazy** learner algorithm because it does not learn the issues generated from the training set immediately instead it stores the issues in the dataset and at the time of classification, it performs on the dataset.

KNN algorithm at the training phase just stores the dataset and when it gets new case data, and then it classifies that case data into a category based that is much familiar to the new data. The following are the steps for KNN algorithm:



Initialization, Define *k*

Calculate Distances (Test example, each Training example)

Sort the Distances

Take *k*-nearest neighbours

Apply the Simple Majority

Class

Fig 5: KNN Algorithm

EXAMPLE:

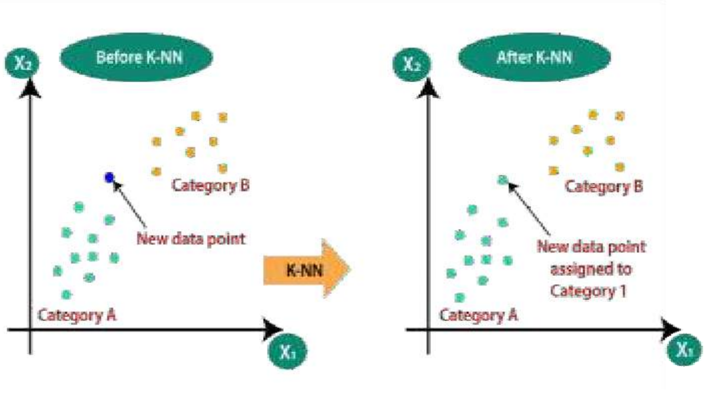


Fig 6: KNN Example

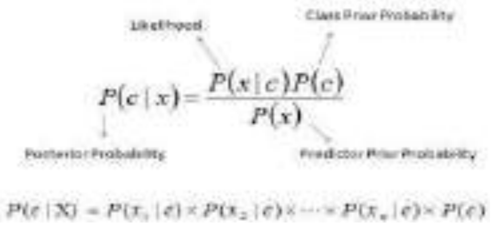
4. NAIVE BAYES ALGORITHM:

It is a supervised learning technique based on Bayes’ Theorem [7]. In simple words, a Naive Bayes classifier thinks that the presence of the main feature in a class is not related to any of the other class features. For example, a fruit may be said to be an apple if it is red in color, round in shape, and about 4 inches in diameter.

Even if these common features depend upon each other, all of these property methods separately say that this fruit is an apple and that is why it is called ‘Naive’. The Navie Bayes model is very easy to build and mainly useful for very large data sets problems/issues. Along with simplicity, Naive Bayes is called to perform even highly sophisticated classification techniques.

The following are the steps for the Naive Bayes algorithm:

Bayes theorem gives a way for calculating posterior probability P(c|x) from P(c), P(x) and P(x|c). See the equation given below:



$$P(c\mid x)=\frac{P(x\mid c)P(c)}{P(x)}$$

$$P(c\mid X)=P(x_1\mid c)\times P(x_2\mid c)\times\cdots\times P(x_n\mid c)\times P(c)$$

- $P(c|x)$ is a posterior probability of the *class* (c, *target*) and given *predictor* (x, *attributes*).
- $P(c)$ is the prior probability of *class*.
- $P(x|c)$ is the likelihood which is the probability of the *predictor* given *class*.
- $P(x)$ is the prior probability of *predictor*.

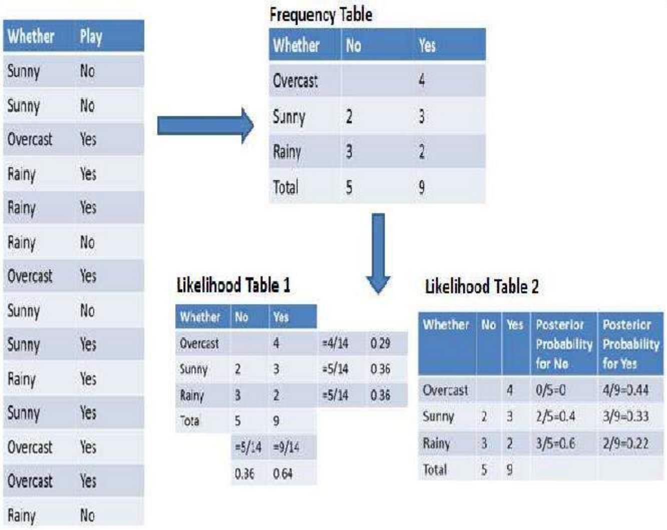
Algorithm Steps:

Step 1: Convert the data set into a frequency table model.

Step 2: Create Likelihood tables by finding the probabilities of each dataset.

Step 3: Now, use Naive Bayesian equation to calculate the posterior probability for each of the class separately. The data classes with the high posterior probability is the result of the prediction method.

EXAMPLE:



Whether	Play
Sunny	No
Sunny	No
Overcast	Yes
Rainy	Yes
Rainy	Yes
Rainy	No
Overcast	Yes
Sunny	No
Sunny	Yes
Rainy	Yes
Sunny	Yes
Overcast	Yes
Overcast	Yes
Rainy	No

Frequency Table

Whether	No	Yes
Overcast		4
Sunny	2	3
Rainy	3	2
Total	5	9

Likelihood Table 1

Whether	No	Yes		
Overcast		4	=4/14	0.29
Sunny	2	3	=5/14	0.36
Rainy	3	2	=5/14	0.36
Tota	5	9		
	=5/14	=9/14		
	0.36	0.64		

Likelihood Table 2

Whether	No	Yes	Posterior Probability for No	Posterior Probability for Yes
Overcast		4	0/5=0	4/9=0.44
Sunny	2	3	2/5=0.4	3/9=0.33
Rainy	3	2	3/5=0.6	2/9=0.22
Total	5	9		

Fig 7: Naive Bayes Example

VII. RESULT

- ➢ First, the user needs to register into the system with the username and the password. If the user is already registered then he/she need to log in into the system with their credentials.
- ➢ Now the user needs to select minimum of two symptoms as compulsory from the given dropdown menu, for getting more accurate results.
- ➢ After entering all the symptoms which they are suffering from, the user needs to press the buttons of the respective algorithm to predict the resultant disease the disease will be displayed that they are suffering from and here we are going to use four different algorithms to provide a clearer picture and accurate result to the user.

Based on our statistics,the Decision tree algorithm is giving more accurate results than the other algorithms.

This is the resultant output of the particular patient which will be stored in our database system.

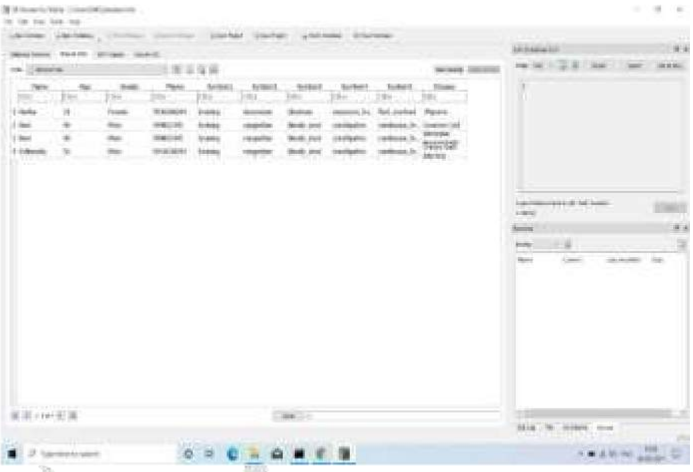


Fig 8: Resultant output of database stored by Human disease Prediction project

VIII. CONCLUSION

Nowadays machine learning is playing important role in the medical prediction field of the health sector. Health is a very important aspect of everyone's life.

So, after doing the research and comparison of all the algorithms and theorems of machine learning we have come to the conclusion that all those algorithms such as Decision Tree, KNN, Naive ayes, and Random Forest Algorithms all are used in building a disease prediction system which predicts the disease of the patients from which he/she is suffering from. But, according to accuracy percentage Decision tree is predicting the most accurate result.

IX. REFERENCES

[1] https://link.springer.com/chapter/10.1007/978-981-15-50898_55#:~:text=Machine%20Learning%20is%20an%20emerging,dataset%20ana%20predict%20the%20disease.

[2] https://www.expert.ai/blog/machine-learning-definition/

[3] https://bmcmedinformdecismak.biomedcentral.com/articles/10.1186/s12911-019-1004-8

[4] https://www.youtube.com/watch?v=3YmAbta16yk&t=1s

[5] https://www.youtube.com/watch?v=NUw2cPNwDZs

[6] https://www.youtube.com/watch?v=I7NrVwm3apg

[7] https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/

[8] https://www.geeksforgeeks.org/decision-tree-introduction-example/#:~:text=Decision%20tree%20algorithm%20falls%20under%20the%20category%20of%20supervised%20learning.&text=Decision%20tree%20uses%20the%20tree,internal%20node%20of%20the%20tree.

[9] https://www.tutorialspoint.com/machine_learning_with_python/machine_learning_with_python_classification_algorithms_random_forest.htm#:~:text=Random%20forest%20is%20a%20supervised,classification%20as%20well%20as%20regression.&text=Similarly%2C%20random%20forest%20algorithm%20creates,solution%20by%20means%20of%20voting.

[10] https://www.tutorialspoint.com/machine_learning_with_python/machine_learning_with_python_knn_algorithm_finding_nearest_neighbors.htm#:~:text=With%20the%20help%20of%20KNN,Image%20Recognition%20and%20Video%20Recognition.

JETIR

Journal of Emerging Technologies and Innovative Research

An International Open Access Journal Peer-reviewed, Refereed Journal

www.jetir.org | editor@jetir.org An International Scholarly Indexed Journal

Certificate of Publication

The Board of

Journal of Emerging Technologies and Innovative Research (ISSN : 2349-5162)

Is hereby awarding this certificate to

Pallakila Krishna tulasi

In recognition of the publication of the paper entitled

HUMAN DISEASE PREDICTION

Published In JETIR (www.jetir.org) ISSN UGC Approved (Journal No: 63975) & 7.95 Impact Factor

Published in Volume 8 Issue 6 , June-2021 | Date of Publication: 2021-06-11

Parin P

EDITOR

EDITOR IN CHIEF

ISSN 2349-5162

JETIR2106195 **Research Paper Weblink http://www.jetir.org/view?paper=JETIR2106195** **Registration ID : 310470**

Journal of Emerging Technologies and Innovative Research
An International Open Access Journal Peer-reviewed, Refereed Journal
www.jetir.org | editor@jetir.org An International Scholarly Indexed Journal

Certificate of Publication

The Board of

Journal of Emerging Technologies and Innovative Research (ISSN : 2349-5162)

Is hereby awarding this certificate to

Balemla Harika

In recognition of the publication of the paper entitled

HUMAN DISEASE PREDICTION

Published In JETIR (www.jetir.org) ISSN UGC Approved (Journal No: 63975) & 7.95 Impact Factor

Published in Volume 8 Issue 6 , June-2021 | Date of Publication: 2021-06-11

EDITOR

EDITOR IN CHIEF

ISSN
2349-5162

JETIR2106195 **Research Paper Weblink http://www.jetir.org/view?paper=JETIR2106195** **Registration ID : 310470**

Journal of Emerging Technologies and Innovative Research

An International Open Access Journal Peer-reviewed, Refereed Journal

www.jetir.org | editor@jetir.org An International Scholarly Indexed Journal

Certificate of Publication

The Board of

Journal of Emerging Technologies and Innovative Research (ISSN : 2349-5162)

Is hereby awarding this certificate to

Nadilla Sindhu

In recognition of the publication of the paper entitled

HUMAN DISEASE PREDICTION

Published In JETIR (www.jetir.org) ISSN UGC Approved (Journal No: 63975) & 7.95 Impact Factor

Published in Volume 8 Issue 6 , June-2021 | Date of Publication: 2021-06-11

Pariu P

EDITOR

EDITOR IN CHIEF

ISSN
2349-5162

JETIR2106195

Research Paper Weblink http://www.jetir.org/view?paper=JETIR2106195

Registration ID : 310470

JETIR

Journal of Emerging Technologies and Innovative Research

An International Open Access Journal Peer-reviewed, Refereed Journal

www.jetir.org | editor@jetir.org An International Scholarly Indexed Journal

Certificate of Publication

The Board of

Journal of Emerging Technologies and Innovative Research (ISSN : 2349-5162)

Is hereby awarding this certificate to

A Mahendar

In recognition of the publication of the paper entitled

HUMAN DISEASE PREDICTION

Published In JETIR (www.jetir.org) ISSN UGC Approved (Journal No: 63975) & 7.95 Impact Factor

Published in Volume 8 Issue 6 , June-2021 | Date of Publication: 2021-06-11

Parin P

EDITOR

EDITOR IN CHIEF

ISSN
2349-5162

JETIR2106195 **Research Paper Weblink http://www.jetir.org/view?paper=JETIR2106195** **Registration ID : 310470**