

HEART DISEASE PREDICTION USING DATA MINING TECHNIQUES

Submitted in partial fulfilment of the requirements for the award of

MASTER OF BUSINESS ADMINISTRATION

By

ANTO RATHESH A R

REGISTER NO:42410526



**DEPARTMENT OF BUSINESS ADMINISTRATION
SCHOOL OF MANAGEMENT STUDIES**

SATHYABAMA

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

**Accredited with Grade "A" by NAAC I 12B Status by UGC I Approved by AICTE
JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119**

DECLARATION

I'M ANTO RATHESH A R (42410526) hereby declare that the Project Report entitled **“HEART DISEASE PREDICTION USING DATA MINING TECHNIQUES”** done by me under the guidance of **DR.R.THAMILSELVAN** is submitted in partial fulfilment of the requirements for the award of Master of Business Administration degree.

DATE:

PLACE:

ANTO RATHESH. A .R

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 sincere thanks to **Dr. G. Bhuvaneswari, MBA., Ph.D., and Dean - School of Management Studies** and **Dr. A. Palani, M.Com., M.Phil., M.B.A., Ph.D., - Head - School of Management Studies** 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.THAMILSELVAN** for her 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 **School of Management Studies** who were helpful in many ways for the completion of the Training.

ANTO RATHESH A R

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	
CHAPTER-1	INTRODUCTION	
	AIM	
	OBJECTIVE	
	SCOPE	
CHAPTER-2	LITERATURE SURVEY	
	PROJECT DESCRIPTION	
CHAPTER-3	FUNCTIONAL REQUIREMENTS	
	NON-FUNCTIONAL REQUIREMENTS	
CHAPTER-4	PROPOSED WORK	
	SYSTEM ARCHITECTURE	
	UML DIAGRAM	
	FLOW DIAGRAM	
CHAPTER-5	SYSTEM REQUIREMENTS	
	BUILD TEST PLAN	
	TEST CASES	
CHAPTER-6	CODE ANALYSIS	
	PREDICTION SCREENSHOTS	
	RESULT AND DISCUSSION	
	CONCLUSION	
	REFERENCES	

ABSTRACT

Cardiovascular diseases are the most common cause of death worldwide over the last few decades in the developed as well as underdeveloped and developing countries. Early detection of cardiac diseases and continuous supervision of clinicians can reduce the mortality rate. However, accurate detection of heart diseases in all cases and consultation of a patient by a doctor is not available since it requires more sapience, time and expertise. In this study, a tentative design of a cloud-b heart disease prediction system had been proposed to detect impending heart disease using Machine learning techniques. For the accurate detection of the heart disease, an efficient machine learning technique should be used which had been derived from a distinctive analysis among several machine learning algorithms in a python Based Open Access Data Mining Platform. The proposed algorithm was validated using two widely used open-access database, where 10-fold cross-validation is applied in order to analyze the performance of heart disease detection.

CHAPTER-1

1.1 INTRODUCTION

The heart is a kind of muscular organ which pumps blood into the body and is the central part of the body's cardiovascular system which also contains lungs. Cardiovascular system also comprises a network of blood vessels, for example, veins, arteries, and capillaries. These blood vessels deliver blood all over the body. Abnormalities in normal blood flow from the heart cause several types of heart diseases which are commonly known as cardiovascular diseases (CVD). Heart diseases are the main reasons for death worldwide. Therefore, detection of cardiac abnormalities at the early stage and tools for the prediction of heart diseases can save a lot of life and help doctors to design an effective treatment plan which ultimately reduces the mortality rate due to cardiovascular diseases.

1.2 AIM:

- We aim to build support vector machine naïve bayes and logical regression models to predict the heart disease by using machine learning and data mining techniques to these three volumes separately.

1.3 OBJECTIVE:

The performance of each classification algorithm was increased while 10 attributes are used. From the above experimental results, it can also be said that using all the 14 attributes for heart disease prediction are not so useful. It decreases the performance of the classifier as it contains irrelevant attributes. Using the attribute selection method for removing the irrelevant attributes, it increases the classifiers' performance in prediction of heart disease patients.

1.4 SCOPE:

Data mining techniques have been widely used to mine knowledgeable information from medical data bases. In data mining classification is a supervised learning that can be used to design models describing important data classes, where class attribute is involved in the construction of the classifier.

CHAPTER-2

2.1 LITERATURE SUREY

- **LITERATURE SURVEY-1**

TITLE: Classification and development of tool for heart diseases (MRI images) using machine learning

AUTHOR: Lalaantika Sharma; Gaurav Gupta; Varun Jaiswal

DESCRIPTION:

Heart diseases are one of the major killers worldwide. Early detection of heart disease such as Global Hypokinesia can reduce this global burden. Computational method has potential to predict disease in early stages automatically and especially helpful in resources limited countries. Computational method to predict global hypokinesia based on confirms cases of global hypokinesia through MRI was developed. Almost all feature extraction method was used on MRI images and model was generated on merged and different images separately. High accuracy of model independent test set justified our approaches and reliability of model. The newly developed was implemented in python and available for open use.

- **LITERATURE SURVEY - 2**

TITLE: Efficient machine learning based detection of heart disease

AUTHOR: Ricardo Buettner; Marc Schunter

DESCRIPTION:

This paper describes a method to detect possible heart disease using the Random Forests algorithm. Cardiovascular diseases are the number 1 cause of death globally - an estimated 17.9 million people died from it in 2016. This machine learning work contributes to healthcare and can detect heart disease on the basis of clinical data and test data from different patients. The result and contribution of this paper is to identify whether a patient has heart disease or not, based on the information of clinical data and test results and so support doctors in making decisions about patient treatments.

- **LITERATURE SURVEY- 3**

TITLE: Supervised Machine Learning Algorithm for Detection of Cardiac Disorders

AUTHOR: Sneha Borkar; M. N. Annadate

DESCRIPTION:

The echocardiogram is the technique which is used in the diagnosis of most of the heart related diseases. Echocardiogram contains less information hence the diagnosis of the disease from the Echocardiogram videos is time consuming task. It required more human efforts to make a decision. Hence the automatic approach to detect the cardiovascular diseases with minimum computing time and high accuracy is necessary. In this approach, the automatic dilated cardiomyopathy (DCM) and Atrial Septal Defect (ASD) disease detection using machine learning approach has been proposed. The database contains the ultrasound videos of ASD, DCM and normal cases. The features extracted from the image and the classified the extracted features using supervised support vector machine algorithm. The proposed system achieved the accuracy of 98.30%.

- **LITERATURE SURVEY- 4**

TITLE: A Patient-Adapting Heartbeat Classifier Using ECG Morphology and Heartbeat Interval Features

AUTHOR: P. de Chazal and R.B.Reilly.

DESCRIPTION:

An adaptive system for the automatic processing of the electrocardiogram (ECG) for the classification of heartbeats into one of the five beat classes recommended by ANSI/AAMI EC57:1998 standard is presented. The heartbeat classification system processes an incoming recording with a global-classifier to produce the first set of beat annotations. An expert then validates and if necessary corrects a fraction of the beats of the recording. The system then adapts by first training a local-classifier using the newly annotated beats and combines this with the global-classifier to produce an adapted classification system. The adapted system is then used to update beat annotations. The results of this study show that the performance of a patient

adaptable classifier increases with the amount of training of the system on the local record. Crucially, the performance of the system can be significantly boosted with a small amount of adaptation even when all beats used for adaptation are from a single class. This study illustrates the ability to provide highly beneficial automatic arrhythmia monitoring and is an improvement on previously reported results for automated heartbeat classification systems LESS.

- **LITERATURE SURVEY- 5**

TITLE: Geometrical aspects of the interindividual variability of multilead ECG recordings.

AUTHOR: R. Hoekema et al.

DESCRIPTION:

The electrocardiogram (ECG) as measured from healthy subjects shows a considerable interindividual variability. This variability is caused by geometrical as well as by physiological factors. In this study, the relative contribution of the geometrical factors is estimated. In addition a method aimed at correcting for these factors is described. First, a measure (RV) for quantifying the overall variability is presented, and for healthy individuals its value is estimated as 0.52. Next, based on a simulation study using the individual (heart-lung-torso) geometry of 25 subjects, the variability caused by geometrical factors is estimated as 0.40, indicating that in healthy subjects the RV for healthy individuals resulting from electrophysiology is of the order of 0.33. In an evaluation of the correction procedure, applied to realistic, simulated body surface potentials, it is shown that RV caused by geometrical factors can be reduced from 0.40 to 0.06. When applying the correction procedure to measured ECG data no reduction of the RV value could be demonstrated. These results indicate that the involved procedure of the inverse computation of a cardiac equivalent source, at the present time, is of insufficient quality to cash in on the substantial reduction of RV values from 0.52 down to 0.33 that might be obtainable.

2.2 PROJECT DESCRIPTION

MODULES AND DESCRIPTION

PERSONAL DETAILS

The user or patient should provide their personal details such as name, age and sex and these details will be saved within the database and the web framework will access the details during each session and the health details of a particular user and its corresponding predicted result by the machine learning algorithm will be saved for each person based on the input provided by the user.

HEALTH CONDITION

Healthcare databases have collected a significant amount of patient's records. The term heart disease circulates on various conditions which are harmful to the human heart. Heart disease is one of the deadliest diseases in nature. The term "Heart/Cardiovascular Disease" deals with the situation by which the heart and the blood veins are affected and the result by which the blood pumping and circulation of blood takes place in the body.

PREDICTION RESULT

The prediction is based on the pre-trained machine learning model where previous medical datasets are stored and certain criteria of previously heart disease affected patient and the reasons for disease. The stored dataset will be fit into the model then it has trained and tested based on the dataset the model will learn to predict the result based on the health condition of a particular user/patient.

CHAPTER-3

3.1 FUNCTIONAL REQUIREMENTS

In Software engineering and systems engineering, a functional requirement defines a function of a system or its component. A function is described as a set of inputs, the behaviour and outputs. ... This should be contrasted with non-functional requirements which specify overall characteristics such as cost and reliability

PRODUCT PERSPECTIVE:

The product is supposed to be an open source, under the GNU general Public License. It is a web-based system implementing client-server model. The portal System provides simple mechanism for users to share and acquire knowledge.

USER CHARACTERISTICS:

It is considered that the user do have the basic knowledge of operating the internet and to have access to it. The administrator is expected to be familiar with the interface of the tech support system.

ASSUMPTION AND DEPENDENCIES:

This software highly depends on type and version of browser being installed in the system i.e. browser version should be used which have HTML5 support

.DOMAIN REQUIREMENT:

Domain requirement is the Requirement that comes from the application domain of the system that reflects the characteristics of that domain. Therefore, as our System is an inventory System, the domain requirement of this system should concern

about the requirements that reflect characteristic of Inventory System.

3.2 NON-FUNCTIONAL REQUIREMENTS

In systems engineering and **requirements** engineering, a **non-functional requirement** (NFR) is a **requirement** that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. They are contrasted with **functional requirements** that **define** specific behavior or functions.

EFFICIENCY

SPACE EFFICIENCY:

Storage **efficiency** is the ability to store and manage data that consumes the least amount of **space** with little to no impact on performance; resulting in a lower total operational cost. **Efficiency** addresses the real-world demands of managing costs, reducing complexity and limiting risk.

TIME EFFICIENCY:

The state or quality of being **efficient**, or able to accomplish something with the least waste of **time** and effort is Time efficiency; competency in performance. And accomplishment of or ability to accomplish a job with a minimum expenditure of **time** and effort.

PORTABILITY:

Portability is a characteristic attributed to a computer program if it can be used in an operating system other than the one in which it was created without requiring major rework. Porting is the task of doing any work necessary to make the computer program run in the new environment.

USABILITY:

Usability is the ease of use and learnability of a human-made object such as a tool or device. In **software engineering**, **usability** is the degree to which a **software**

can be used by specified consumers to achieve quantified objectives with effectiveness, efficiency, and satisfaction in a quantified context of use.

CHAPTER-4

4.1 PROPOSED WORK:

Proposed system Approach Combines SVM, Naive Bayes, Logistic Regression Classification. To Access the severity of heart disease risk using the data mining models.

ADVANTAGES

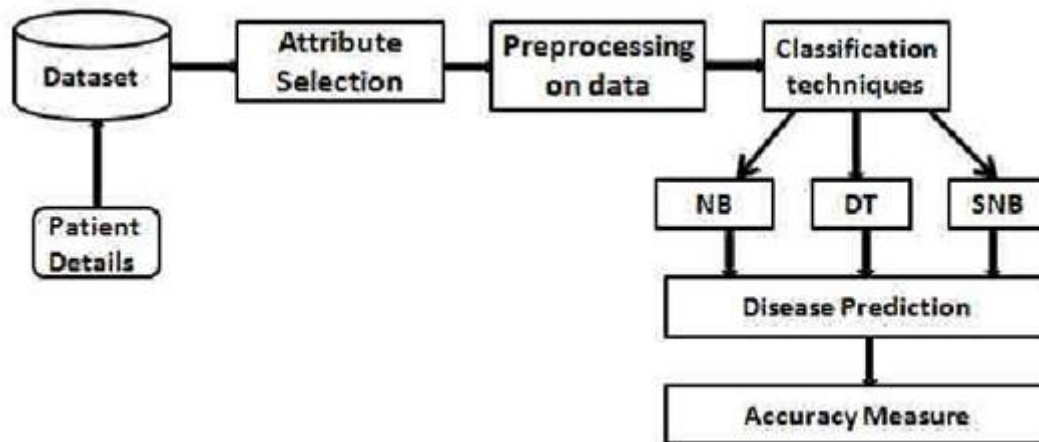
The performance of each classification algorithm was increased while 10 attributes are used. From the above experimental results, it can also be said that using all the 14 attributes for heart disease prediction are not so useful. It decreases the performance of the classifier as it contains irrelevant attributes. Using the attribute selection method for removing the irrelevant attributes, it increases the classifiers' performance, and logistic regression showed better performance in prediction of heart disease patients.

EXISTING SYSTEM DISADVANTAGES:

Most of the existing classification methods tend to perform poorly on dataset which is extremely imbalanced. However there are mainly two issues namely performance issues and data source issues. Hence, an alternative method of modelling the objects is required .Thus in this paper an efficient algorithm is proposed to overcome the above issues. The healthcare industry contains large information, which is tedious to process by manual methods. Medical datasets are often not balanced in their class labels. Therefore in this paper we propose an algorithm for healthcare system to accurately predict the result from the large

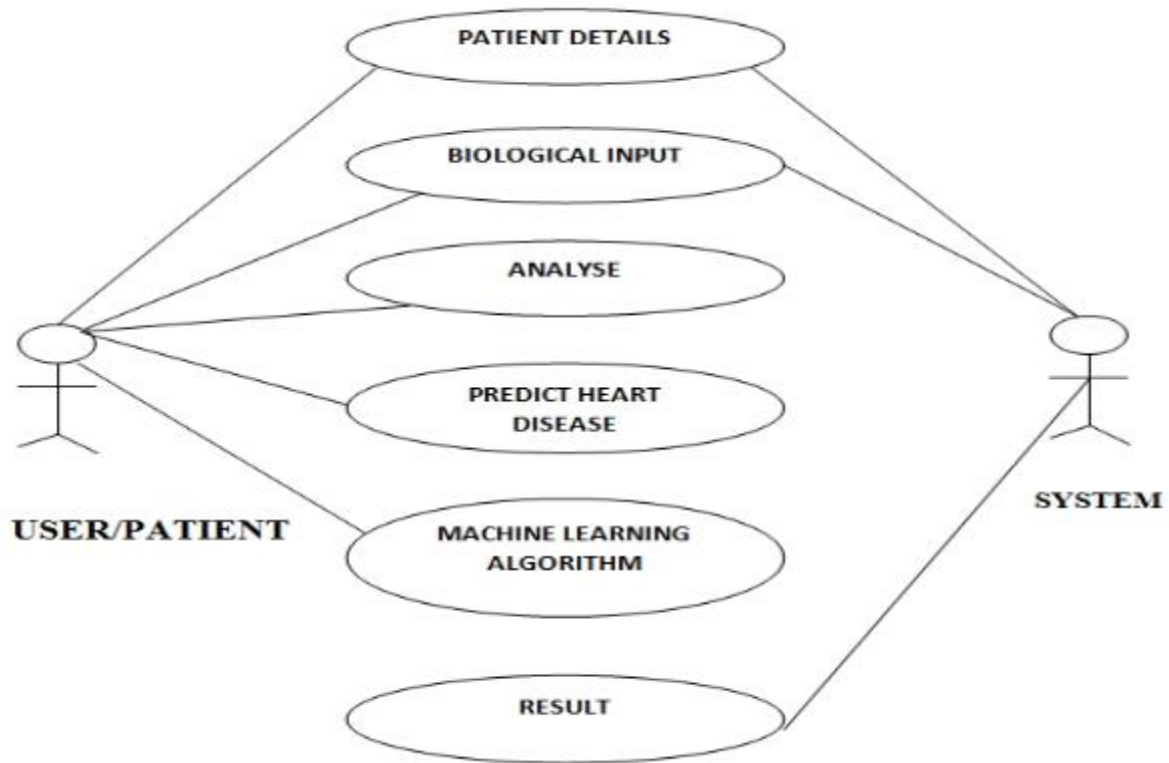
amount of data.

4.2 SYSTEM ARCHITECTURE:



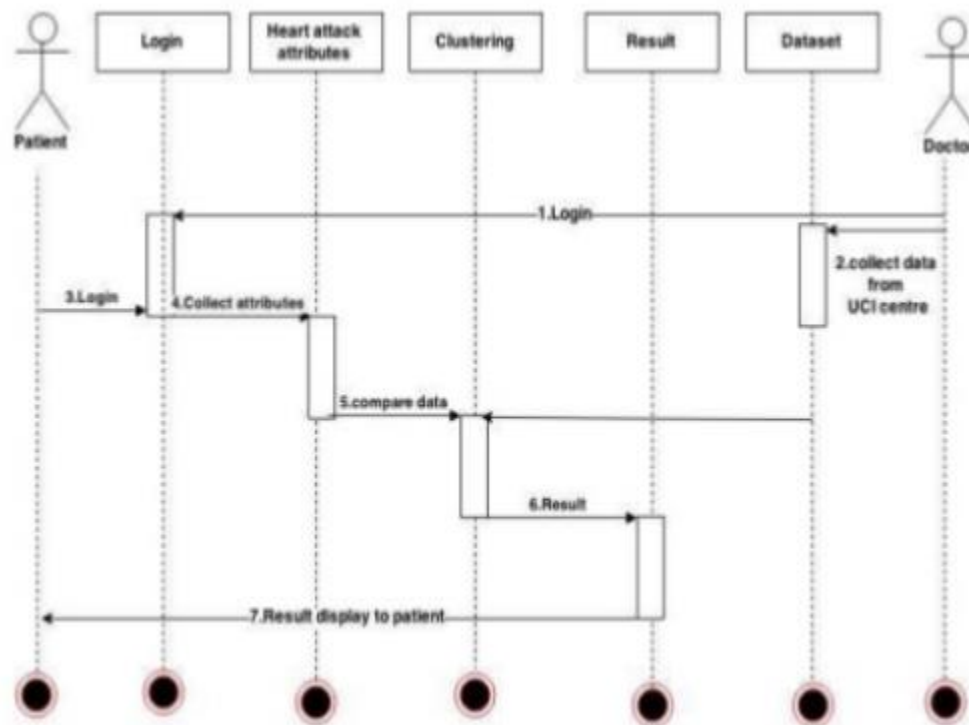
4.3 UML DIAGRAMS:

4.3.1 USE CASE

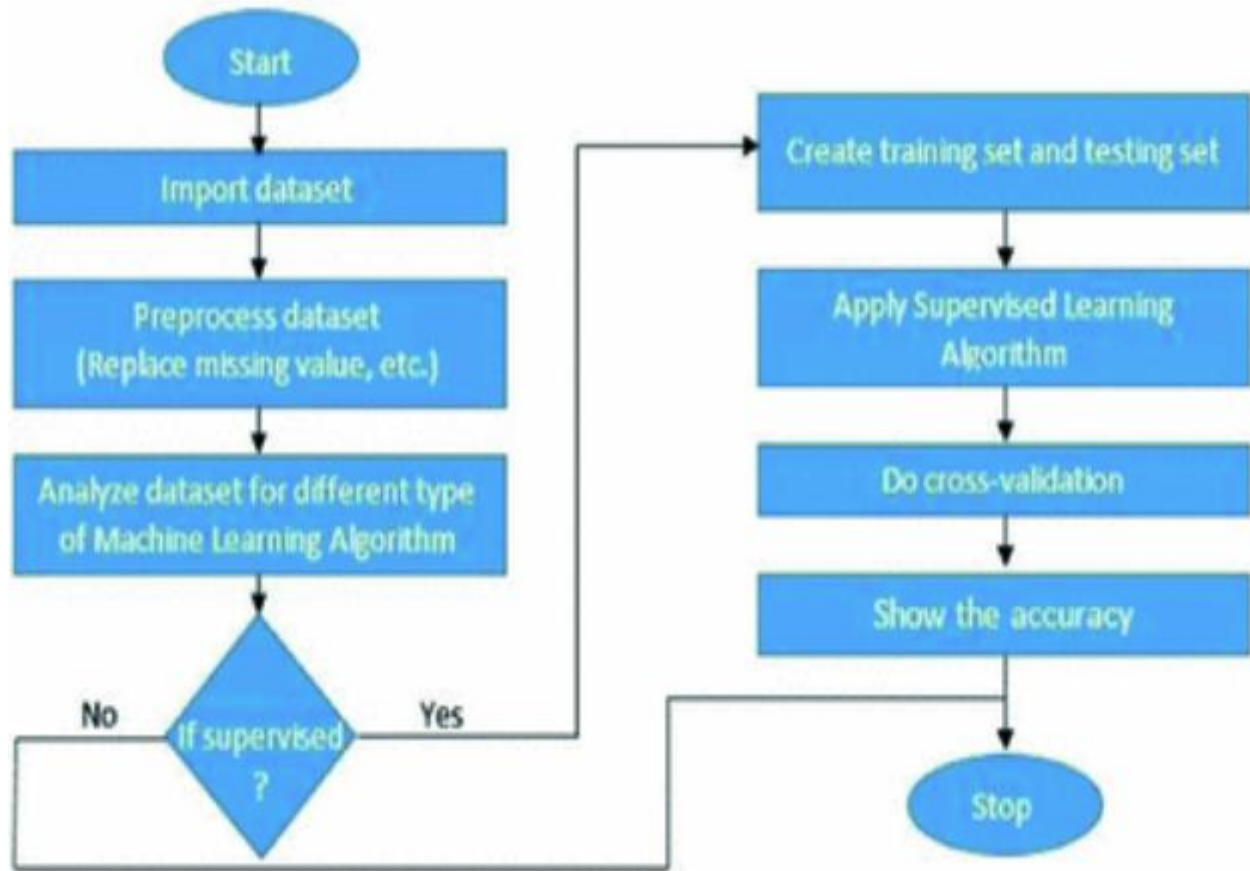


- Use case diagrams consists of actors, use cases and their relationships. The diagram is used to model the system/subsystem of an application. A single use case diagram captures a particular functionality of a system. The identified patient is user. User has to perform the following process to predict heart disease, giving personal information, the medical history, and reports

4.3.2 SEQUENCE DIAGRAM



4.4 FLOW DIAGRAM:



- A flowchart is simply a graphical representation of steps. It shows steps in sequential order and is widely used in presenting the flow of algorithms, workflow or processes. Typically, a flowchart shows the steps as boxes of various kinds, and their order by connecting them with arrows. The flow chart explain the process of our prediction from top to bottom manner.

CHAPTER-5

5.1 SYSTEM REQUIREMENTS

SOFTWARE REQUIREMENTS:

- Operating system - Windows/ Ubuntu
- Coding Language - Python
- Front-End - Jupyter Notebook.
- Back-End - Anaconda 3.6.

HARDWARE REQUIREMENTS:

- Processor - Intel Pentium Dual Core
- Speed - 2.2GHz
- RAM - 4 GB
- Hard Disk - 200 G
- Operating System - Windows XP/07/10

5.2 BUILT TEST PLAN

Any project can be divided into units that can be further performed for detailed processing. Then a testing strategy for each of this unit is carried out. Unit testing helps to identify the possible bugs in the individual component, so the component that has bugs can be identified and can be rectified from errors.

5.3 TEST CASES

TEST CASE	TEST CASE	TEST CASE DESCRIPTION	TEST CASE	EXPECTED RESULT	ACTUAL RESULT	STATUS
-----------	-----------	-----------------------	-----------	-----------------	---------------	--------

ID	TITLE		STEPS			
01	Testing the functionality of python development overview	Testing the functionality of anaconda.	Run the python code.	The software starts to compile and execute the program.	The software starts to compile and execute the program.	Pass
02			The software takes the data sets of cascade style sheet	The python software takes the input data sets	The python software take the input data sets and shows	Pass
03			The main program displays the restored.	The execution stops and displays a restored data sets.	The execution stops and displays a restored data sets.	Pass
04			Now run the validation module code.	The software starts to compile and execute the process.	The software starts to compile and execute the process.	Pass

5.3.1 UNIT TESTING Unit

testing verification efforts on the smallest unit of software design, module. This is known as “Module Testing”. The modules are tested separately. This testing is carried out during programming stage itself. In these testing steps, each module is found to be working satisfactorily as regard to the expected output from the module.

TEST CASE ID	AGE	SEX	CP	BP	CH OL	FBS	ECG	MAXHR	EXANG	EXST	SLOPE	VESS	THA L	S T A T U S
01	63	1	3	145	233	1	0	150	0	2.3	0	0	1	G o o d
02	37	1	2	130	250	0	1	187	0	3.5	0	0	2	M i l d
03	41	0	1	130	204	0	0	172	0	1.4	2	0	2	G o o d
04	56	1	1	120	236	0	1	178	0	0.8	2	0	2	G o o d
05	57	0	3	150	354	1	2	183	1	3.6	3	2	6	S e v e r e

CHAPTER-6

6.1 CODE ANALYSIS

➤ PYTHON CODING:

```
import tkinter #importing tkinter library for GUI creation
from tkinter import *
import pandas as pnd # importing pandas data analysis toolkit
import numpy as np # importing numpy library for array operations
from time import time # importing time library for time calculations
from sklearn.model_selection import train_test_split # importing module model_classification
from sklearn library

header_row = ['age', 'sex', 'pain', 'BP', 'chol', 'fbs', 'ecg', 'maxhr', 'eiang', 'eist', 'slope', 'vessels',
'thal',
'diagnosis'] # Declaring the header row for getting data from the dataset files

# filter to only those diagnosed with heart disease

master = Tk() # Defining the Tkinter widget
master.wm_title("Heart Disease Prediction")
import sklearn # Importing scikit-learn functions

Lab=Label(master,text=" Automatic Heart Disease Detection ") # Adding Label to the Tkinter
widget
Lab.grid(row=1,column=5,columnspan=2) # Packing the label data to the tkinter
widget in user defined rows and columns
Lab.configure(height=2) # Changing dimensions of the Label

Lab=Label(master,text="")
Lab.grid(row=2,column=5,columnspan=2)

Lab1=Label(master,text="Classification Report")
Lab1.grid(row=11,column=2,columnspan=3)
Lab2=Label(master,text="Confusion Matrix")
Lab2.grid(row=11,column=7,columnspan=3)
T = Text(master, height=6, width=30) # Declaring Text Widget for Result
Displaying
T.grid(row=13,column=2,rowspan=4, columnspan=3, sticky= 'nsew')
```

```
T1 = Text(master, height=6, width=30)
T1.grid(row=13,column=7,rowspan=4,columnspan=3, sticky= 'nsew')
```

```
var = StringVar(master)
var.set("Select Dataset") # initial value
```

```
option = OptionMenu(master, var, "Cleveland", "Hungarian", "VA", "all") # Declaring the
OptionMenu (Drop-Down list) widget
option.grid(row=2,column=5,sticky='nsew') # N-North (Top), S-South
(Bottom), E-East (Right), W-West (Left)
option.configure(width=6)
```

```
field1="Age" # Defining the field names which user has to
input for heart disease detection
field2="Sex"
field3="Pain"
field4="BP"
field5="Chol"
field6="FBS"
field7="ECG"
field8="Maxhr"
field9="Eiang"
field10="Eist"
field11="Slope"
field12="Vessels"
field13="Thal"
```

```
L1=Label(master,text=field1)
L1.grid(row = 4, column = 0, sticky='nsew')
L1.configure(width=14)
L2=Label(master,text=field2)
L2.grid(row = 4, column = 1, sticky='nsew')
L2.configure(width=14)
L3=Label(master,text=field3)
L3.grid(row = 4, column = 2, sticky='nsew')
L3.configure(width=14)
L4=Label(master,text=field4)
L4.grid(row = 4, column = 3, sticky='nsew')
L4.configure(width=14)
L5=Label(master,text=field5)
L5.grid(row = 4, column = 4, sticky='nsew')
L5.configure(width=14)
L6=Label(master,text=field6, )
L6.grid(row = 4, column = 5, sticky='nsew')
L6.configure(width=14)
```

```
L7=Label(master,text=field7)
L7.grid(row = 4, column = 6, sticky='nsew')
L7.configure(width=14)
L8=Label(master,text=field8)
L8.grid(row = 4, column = 7, sticky='nsew')
L8.configure(width=14)
L9=Label(master,text=field9)
L9.grid(row = 4, column = 8, sticky='nsew')
L9.configure(width=14)
L10=Label(master,text=field10)
L10.grid(row = 4, column = 9, sticky='nsew')
L10.configure(width=14)
L11=Label(master,text=field11)
L11.grid(row = 4, column = 10, sticky='nsew')
L11.configure(width=14)
L12=Label(master,text=field12)
L12.grid(row = 4, column = 11, sticky='nsew')
L12.configure(width=14)
L13=Label(master,text=field13)
L13.grid(row = 4, column = 12, sticky='nsew')
L13.configure(width=14)
```

```
E1=Entry(master)
user input values
E1.grid(row = 5, column = 0, sticky='nsew')
E1.configure(width=14)
E2=Entry(master)
E2.grid(row = 5, column = 1, sticky='nsew')
E2.configure(width=14)
E3=Entry(master)
E3.grid(row = 5, column = 2, sticky='nsew')
E3.configure(width=14)
E4=Entry(master)
E4.grid(row = 5, column = 3, sticky='nsew')
E4.configure(width=14)
E5=Entry(master)
E5.grid(row = 5, column = 4, sticky='nsew')
E5.configure(width=14)
E6=Entry(master)
E6.grid(row = 5, column = 5, sticky='nsew')
E6.configure(width=14)
E7=Entry(master)
E7.grid(row = 5, column = 6, sticky='nsew')
E7.configure(width=14)
E8=Entry(master)
E8.grid(row = 5, column = 7, sticky='nsew')
```

Declaring the Entry widget for taking

```

E8.configure(width=14)
E9=Entry(master)
E9.grid(row = 5, column = 8, sticky='nsew')
E9.configure(width=14)
E10=Entry(master)
E10.grid(row = 5, column = 9, sticky='nsew')
E10.configure(width=14)
E11=Entry(master)
E11.grid(row = 5, column = 10, sticky='nsew')
E11.configure(width=14)
E12=Entry(master)
E12.grid(row = 5, column = 11, sticky='nsew')
E12.configure(width=14)
E13=Entry(master)
E13.grid(row = 5, column = 12, sticky='nsew')
E13.configure(width=14)

```

```

Labx=Label(master,text="")
Labx.grid(row=21,column=4,columnspan=4)
Labx.visible=False

```

```

#T3 = Text(master, height=2, width=30)                                # Declaring Text
Widget for Displaying Prediction
#T3.grid(row=23,column=4, columnspan=4, sticky= 'nsew')

```

```

def train_classifier(x_train,x_test,y_train,y_test,string):           # Declaring the function
for training classifiers and classification analysis

```

```

    global clf                                                         # Declaring clf as a Global Variable for
using throughtot the code

```

```

    global outclass
    from sklearn.metrics import classification_report
    from sklearn.metrics import confusion_matrix
    T.delete(1.0,END)                                                  # Deleting the text in the Text

```

Widget

```

    T1.delete(1.0,END)
    if string=="SVM":
        from sklearn import svm
        from sklearn.svm import SVC
        from sklearn.model_selection import GridSearchCV

```

```

    t1= time()
    param_grid = {'C': [1, 5, 10, 50, 100],
        'gamma': [0.0001, 0.0005, 0.001, 0.005, 0.01, 0.1], }
    clf = GridSearchCV(SVC(kernel='rbf', class_weight='balanced'), param_grid) #
Initialization of GridSearch Optimization for SVM with RBF kernel

```

```

        clf.fit(x_train, y_train)                                # Fitting the classifier to the training
and testing the SVM Classifier
        y_pred = clf.predict(x_test)                            # Predict Results for Test Data
        title = "Learning Curves (SVM)"
        geterror(x_train,y_train,clf,title);
        t= time()-t1
        print("Training Complete")

elif string=="Naive Bayes":
    from sklearn.naive_bayes import GaussianNB
    t2=time()
    clf = GaussianNB()                                          # Initializing the Naive Bayes
Classifier
    clf.partial_fit(x_train, y_train, np.unique(y_train))      # Fitting the classifier to
the training and testing the Naive Bayes Classifier
    y_pred = clf.predict(x_test)
    t = time() - t2
    title = "Learning Curves (Naive Bayes)"
    geterror(x_train,y_train,clf,title);

    print("Training Complete")

elif string=="Logistic Regression":
    from sklearn.linear_model import LogisticRegression
    t3=time()
    clf=LogisticRegression(penalty='l2', dual=False, tol=0.0001, C=1.0, fit_intercept=True,
intercept_scaling=1, solver='liblinear', max_iter=100, verbose=0, warm_start=False, n_jobs=1)
# Initializing the Logistic Regression Classifier
    clf.fit(x_train,y_train)
    y_pred = clf.predict(x_test)
    t = time() - t3
    title = "Learning Curves (Logistic Regression)"
    geterror(x_train,y_train,clf,title);
    print("Training Complete")

t=str(t)

classre=classification_report(y_test,y_pred)                  # Generating Classification Report

T.insert(END,classre[1:5]+classre[1:32]+classre[1:13]+classre[60:90]+classre[1:11]+classre[1:2
]+classre[115:140]+classre[1:7]+classre[161:195]) # Printing Precision and Recall Results
    print(classre)
    confmat=confusion_matrix(y_test, y_pred)                  # Calculating the
Confusion Matrix for the classification
    T1.insert(END, confmat)
    T.insert(END, classre[1:9])

```

```

T.insert(END, "Accuracy ")
T.insert(END, classre[1:5])
T.insert(END, int(float((y_test==y_pred).sum())/len(y_test.T)*100))
T.insert(END, "%")
T.insert(END, classre[1:10]+classre[1:10])
T.insert(END, "Class. Time")
T.insert(END, classre[1:8])
T.insert(END, t[0:4]+" sec")
import matplotlib.pyplot as plt
def process_dataset(string):

    if string=="Cleveland":

        heart = pd.read_csv('processed.cleveland.data', names=header_row)      # Reading the
dataset file in .data format using Pandas library function read_csv()
        print("Unprocessed Cleveland Dataset")

    print("*****")
    print(heart.loc[:, 'age': 'diagnosis'])

    print("*****")

    import numpy as np
    has_hd_check = heart['diagnosis'] > 0                                     # Getting the
indices of individuals having heart disease
    has_hd_patients = heart[has_hd_check]
    heart['vessels'] = heart['vessels'].apply(lambda vessels: 0.0 if vessels == "?" else vessels) #
Replacing the unknown values in the dataset with float
    heart['vessels'] = heart['vessels'].astype(float)
    heart['thal'] = heart['thal'].apply(lambda thal: 0.0 if thal == "?" else thal)
    heart['thal'] = heart['thal'].astype(float)
    heart['diag_int'] = has_hd_check.astype(int)

    ind1 = np.where((heart['diagnosis'] == 1)|(heart['diagnosis'] == 2));
    ind2 = np.where((heart['diagnosis'] == 3)|(heart['diagnosis'] == 4));

    temp = heart['diagnosis'];
    temp.ix[ ind1 ] = 1;
    temp.ix[ ind2 ] = 2;
    heart['diagnosis'] = temp;

    global x_train

```

```

    global y_train
    global x_test
    global y_test
    x_train, x_test, y_train, y_test = train_test_split(heart.loc[:, 'age':'thal'], heart.loc[:,
'diagnosis'], # Splitting the processed data into training data and testing data
                                                    test_size=0.30, random_state=42) # test_size
= percent of data used for testing,
                                                    # random_state = for
initializing the random number generator

    print("Processed Cleveland Dataset")

print("*****")
    print(heart.loc[:, 'age':'diagnosis'])

print("*****")

elif string=="VA":

    import numpy as np

    heart_va = pnd.read_csv('processed.va.data', names=header_row)
    print("Unprocessed VA Dataset")

print("*****")
    print(heart_va.loc[:, 'age':'diagnosis'])

print("*****")

    has_hd_check = heart_va['diagnosis'] > 0

    heart_va['diag_int'] = has_hd_check.astype(int)
    heart_va = heart_va.replace(to_replace='?', value=0.0)
    heart_va['diag_int'] = has_hd_check.astype(int)

    ind1 = np.where((heart_va['diagnosis'] == 1)|(heart_va['diagnosis'] ==2));
    ind2 = np.where((heart_va['diagnosis'] == 3)|(heart_va['diagnosis'] ==4));

```

```

temp = heart_va['diagnosis'];
temp.ix[ ind1 ] = 1;
temp.ix[ ind2 ] = 2;

heart_va['diagnosis'] = temp;

print("Processed VA Dataset")

print("*****")
print(heart_va.loc[:, 'age':'diagnosis'])

print("*****")

x_train, x_test, y_train, y_test = train_test_split(heart_va.loc[:, 'age':'thal'], heart_va.loc[:,
'diagnosis'],
                                                    test_size=0.30, random_state=42)

elif string=="Hungarian":
    import numpy as np
    heart_hu = pnd.read_csv('processed.hungarian.data', names=header_row)
    print("Unprocessed Hungarian Dataset")

print("*****")
print(heart_hu.loc[:, 'age':'diagnosis'])

print("*****")

has_hd_check = heart_hu['diagnosis'] > 0
heart_hu['diag_int'] = has_hd_check.astype(int)
heart_hu = heart_hu.replace(to_replace='?', value=0.0)

ind1 = np.where((heart_hu['diagnosis'] == 1)|(heart_hu['diagnosis'] ==2));
ind2 = np.where((heart_hu['diagnosis'] == 3)|(heart_hu['diagnosis'] ==4));

temp = heart_hu['diagnosis'];
temp.ix[ ind1 ] = 1;
temp.ix[ ind2 ] = 2;
heart_hu['diagnosis'] = temp;

print("Processed Hungarian Dataset")

```

```

print("*****")
    print(heart_hu.loc[:, 'age':'diagnosis'])

print("*****")
    heart_hu['diag_int'] = has_hd_check.astype(int)

    x_train, x_test, y_train, y_test = train_test_split(heart_hu.loc[:, 'age':'thal'], heart_hu.loc[:,
'diagnosis'],
                                                    test_size=0.30, random_state=42)

    elif string=="all":
        import numpy as np
        heart_cl = pnd.read_csv('processed.cleveland.data', names=header_row)
        print("Unprocessed Cleveland Dataset")

print("*****")
    print(heart_cl.loc[:, 'age':'diagnosis'])

print("*****")
    has_hd_check = heart_cl['diagnosis'] > 0
    has_hd_patients = heart_cl[has_hd_check]
    heart_cl['diag_int'] = has_hd_check.astype(int)
    heart_cl['vessels'] = heart_cl['vessels'].apply(lambda vessels: 0.0 if vessels == "?" else
vessels)
    heart_cl['vessels'] = heart_cl['vessels'].astype(float)
    heart_cl['thal'] = heart_cl['thal'].apply(lambda thal: 0.0 if thal == "?" else thal)
    heart_cl['thal'] = heart_cl['thal'].astype(float)

    ind1 = np.where((heart_cl['diagnosis'] == 1)|(heart_cl['diagnosis'] ==2));
    ind2 = np.where((heart_cl['diagnosis'] == 3)|(heart_cl['diagnosis'] ==4));

    temp = heart_cl['diagnosis'];
    temp.ix[ ind1 ] = 1;
    temp.ix[ ind2 ] = 2;
    heart_cl['diagnosis'] = temp;

    heart_va = pnd.read_csv('processed.va.data', names=header_row)
    print("Unprocessed VA Dataset")

print("*****")

```

```

    print(heart_va.loc[:, 'age':'diagnosis'])

print("*****")

    has_hd_check = heart_va['diagnosis'] > 0
    heart_va['diag_int'] = has_hd_check.astype(int)
    heart_va = heart_va.replace(to_replace='?', value=0.0)

    ind1 = np.where((heart_va['diagnosis'] == 1)|(heart_va['diagnosis'] == 2));
    ind2 = np.where((heart_va['diagnosis'] == 3)|(heart_va['diagnosis'] == 4));

    temp = heart_va['diagnosis'];
    temp.ix[ ind1 ] = 1;
    temp.ix[ ind2 ] = 2;
    heart_va['diagnosis'] = temp;

    print("Processed VA Dataset")

print("*****")
    print(heart_va.loc[:, 'age':'diagnosis'])

print("*****")

    heart_hu = pnd.read_csv('processed.hungarian.data', names=header_row)
    print("Unprocessed Hungarian Dataset")

print("*****")
    print(heart_hu.loc[:, 'age':'diagnosis'])

print("*****")

    has_hd_check = heart_hu['diagnosis'] > 0
    heart_hu['diag_int'] = has_hd_check.astype(int)
    heart_hu = heart_hu.replace(to_replace='?', value=0.0)

    ind1 = np.where((heart_hu['diagnosis'] == 1)|(heart_hu['diagnosis'] == 2));
    ind2 = np.where((heart_hu['diagnosis'] == 3)|(heart_hu['diagnosis'] == 4));

    temp = heart_hu['diagnosis'];
    temp.ix[ ind1 ] = 1;
    temp.ix[ ind2 ] = 2;

```

```

heart_hu['diagnosis'] = temp;

print("Processed Hungarian Dataset")

print("*****")
print(heart_hu.loc[:, 'age':'diagnosis'])

print("*****")

x_train1, x_test1, y_train1, y_test1 = train_test_split(heart_cl.loc[:, 'age':'thal'],
heart_cl.loc[:, 'diagnosis'],
                                test_size=0.30, random_state=42)
x_train2, x_test2, y_train2, y_test2 = train_test_split(heart_va.loc[:, 'age':'thal'],
heart_va.loc[:, 'diagnosis'],
                                test_size=0.30, random_state=42)
x_train3, x_test3, y_train3, y_test3 = train_test_split(heart_hu.loc[:, 'age':'thal'],
heart_hu.loc[:, 'diagnosis'],
                                test_size=0.30, random_state=42)

# Combining the dataset for Cleveland, VA and Hungarian Dataset
x_train4= x_train1.append(x_train2);
x_train = x_train4.append(x_train3);

y_train4 = y_train1.append(y_train2);
y_train = y_train4.append(y_train3);

x_test4 = x_test1.append(x_test2);
x_test = x_test4.append(x_test3)

y_test4 = y_test1.append(y_test2);
y_test = y_test4.append(y_test3);

button = Button(master, text="Process Dataset", command=lambda: process_dataset(var.get()))
#Defining the button in the Tkinter Widget
button.grid(row=2, column=6, sticky='nsew')
button.configure(width=14)
var1 = StringVar(master)
var1.set("Select Classifier") # initial value

option1 = OptionMenu(master, var1, "SVM", "Naive Bayes", "Logistic Regression")
option1.grid(row=3,column=5, sticky='nsew')
option1.configure(width=14)
#option.place ( relx=0.5, rely=0.1)

```

```
button1 = Button(master, text=" Train Classifier", command=lambda:
train_classifier(x_train,x_test,y_train,y_test,var1.get()))
button1.grid(row=3,column=6, sticky='nsew')
button1.configure(width=14)
```

```
#e1.bind('<Button-1>',e1.delete(0,END))
```

```
def predres(clf):
result from the user input data
```

```
# Defining function to predict the
```

```
    E14=E10.get()
according to sign and decimal point
    if len(E14)==3 or len(E14)==4:
        E15=float(E14)
    else:
        E16=E14+'.0'
        E15=float(E16)
```

```
# Converting the Eist data
```

```
test=[float(E1.get()+'.0'),float(E2.get()+'.0'),float(E3.get()+'.0'),float(E4.get()+'.0'),float(E5.get()+
'.0'),float(E6.get()+'.0'),float(E7.get()+'.0'),float(E8.get()+'.0'),float(E9.get()+'.0'),E15,float(E11.g
et()+'.0'),float(E12.get()+'.0'),float(E13.get()+'.0')]
test=np.reshape(test,(1,-1))
if clf.predict(test) == 1:
```

```
    Labx1=Label(master,text="The Person has Mild Heart Disease", bg='orange')
    Labx1.visible=False
    Labx1.grid(row=23,column=4,columnspan=4, sticky = 'nsew')
    Labx1.visible=True
    #T3.insert(END,"The Person has Heart Disease")
```

```
elif clf.predict(test) == 2:
```

```
    Labx1=Label(master,text="The Person has Severe Heart Disease", bg='red')
    Labx1.visible=False
    Labx1.grid(row=23,column=4,columnspan=4, sticky = 'nsew')
    Labx1.visible=True
```

```
else:
```

```
    Labx1=Label(master,text="The Person does not have Heart Disease", bg='green')
    Labx1.visible=False
    Labx1.grid(row=23,column=4,columnspan=4, sticky = 'nsew')
    Labx1.visible=True
```

```
#T3.insert(END, "The Person does not have Heart Disease")
```

```
def geterror(x_train,y_train,clf,title):
```

```
    #global outclass
```

```
    import matplotlib.pyplot as plt
```

```
    from sklearn.model_selection import learning_curve
```

```
    from sklearn.model_selection import ShuffleSplit
```

```
    clas=[];
```

```
def plot_learning_curve(estimator, title, X, y, ylim=None, cv=None,  
                        n_jobs=1, train_sizes=np.linspace(.1, 1.0, 5)):
```

```
    #plt.figure()
```

```
    plt.title(title)
```

```
    if ylim is not None:
```

```
        plt.ylim(*ylim)
```

```
    plt.xlabel("Training examples")
```

```
    plt.ylabel("Score")
```

```
    train_sizes, train_scores, test_scores = learning_curve(  
        estimator, X, y, cv=cv, n_jobs=n_jobs, train_sizes=train_sizes)
```

```
    train_scores_mean = np.mean(train_scores, axis=1)
```

```
    train_scores_std = np.std(train_scores, axis=1)
```

```
    test_scores_mean = np.mean(test_scores, axis=1)
```

```
    test_scores_std = np.std(test_scores, axis=1)
```

```
    plt.grid()
```

```
    plt.fill_between(train_sizes, train_scores_mean - train_scores_std,  
                    train_scores_mean + train_scores_std, alpha=0.1,  
                    color="r")
```

```
    plt.fill_between(train_sizes, test_scores_mean - test_scores_std,  
                    test_scores_mean + test_scores_std, alpha=0.1, color="g")
```

```
    plt.plot(train_sizes, train_scores_mean, 'o-', color="r",  
            label="Training score")
```

```
    plt.plot(train_sizes, test_scores_mean, 'o-', color="g",  
            label="Cross-validation score")
```

```
    plt.legend(loc="best")
```

```
    plt.axis([ 0,len(y),0,1.1])
```

```
    return plt
```

```
for index in range(len(y_train)):
```

```
    x_train1=np.reshape(x_train.iloc[index,:],(1,-1));
```

```
    outclass = clf.predict(x_train1);
```

```
    clas.append(outclass[0]);
```

```
ind3 = np.where((clas == y_train));  
ind4 = np.where((clas != y_train));  
l=0  
m=0
```

```
cv = ShuffleSplit(n_splits=4, test_size=0.2, random_state=0)
```

```
plot_learning_curve(clf, title, x_train, y_train, (0.7, 1.01), cv=cv, n_jobs=1)
```

```
plt.show()
```

```
button2 = Button(master, text=" Predict Heart Disease ", command=lambda:predres(clf))  
button2.grid(row=8,column=5,columnspan=2,sticky='we')  
button2.configure(width=14)  
#button1.place(relx=0.1,rely=0.2)
```

```
master.mainloop()  
Tkinter widget until manually closed
```

```
# To continuously show the
```

6.2 SCREENSHOTS OF PREDICTION TAB

1) ANACONDA PROMPT:



2) COLLECTION OF CLINICAL DATA SETS:

processed.no - Notepad	processed.hospital - Notepad	processed.cleveland - Notepad
61,1,0,140,200,0,1,112,1,1,2,7,7,2 64,1,0,130,200,0,1,127,0,0,7,7,7,0 60,1,0,132,210,0,1,100,1,1,5,3,7,7,2 55,1,0,142,220,0,1,109,1,2,5,1,7,7,1 66,1,1,110,213,1,2,00,1,1,3,2,7,7,0 60,1,1,120,0,0,1,130,0,0,5,7,7,7,0 65,1,0,150,230,1,1,105,1,0,7,7,7,3 60,1,1,100,0,0,1,140,1,1,5,2,7,7,0 60,1,1,120,0,0,1,141,1,1,2,7,7,1 60,1,1,100,107,1,1,157,0,0,5,2,7,7,1 56,1,1,120,100,0,1,100,0,0,7,7,7,0 59,1,0,140,0,0,1,117,1,1,2,7,7,1 62,1,0,110,0,0,1,120,1,1,5,2,7,7,1 61,1,1,1,7,0,0,2,7,7,7,7,7,1 57,1,0,120,0,1,1,140,1,1,2,7,7,1 62,1,0,120,220,0,1,100,0,0,7,7,7,0 63,1,0,170,177,0,0,0,1,2,5,1,7,7,4 40,1,0,110,230,0,0,125,1,2,2,7,7,1 61,1,0,120,0,0,1,120,0,1,5,5,7,7,0 60,1,0,152,0,0,1,110,1,0,7,7,7,0 50,1,0,110,0,0,0,124,0,1,1,7,7,1 64,1,0,120,0,1,1,100,0,2,2,7,7,1 63,1,1,130,0,0,1,111,1,0,7,7,7,1 74,1,1,1,7,0,0,0,7,7,7,7,7,0 52,1,1,120,0,0,1,100,0,1,1,7,7,2 67,1,0,130,0,1,1,120,0,1,2,7,7,2 53,1,0,1,7,0,1,1,7,7,7,7,7,0 60,1,0,130,100,1,1,100,0,1,5,2,7,7,1 56,1,0,120,100,0,0,120,1,1,5,2,0,7,1 55,1,1,1,220,0,1,7,7,7,7,7,1 54,1,0,1,7,0,0,1,7,7,7,7,7,2 77,1,0,124,173,0,1,110,1,1,1,7,7,1 61,1,0,100,230,1,0,105,1,1,2,7,7,1 55,1,1,1,0,0,0,135,0,1,1,2,7,7,1 52,1,1,122,0,0,0,110,1,1,2,7,7,2 64,1,0,144,0,0,1,122,1,1,2,7,7,2 60,1,0,1,240,0,0,1,7,7,7,7,7,2 60,1,0,120,0,0,0,135,1,2,1,7,7,0 58,1,1,1,280,1,0,7,7,7,7,7,1 59,1,0,154,0,0,1,111,1,1,5,7,7,0 61,1,1,120,0,0,0,80,1,0,2,7,7,1 40,1,0,125,0,1,0,105,0,0,1,7,7,1 61,1,0,1,1,0,0,1,1,5,2,7,7,1 41,1,0,104,0,0,1,111,0,0,7,7,7,0	20,1,2,130,232,0,2,185,0,0,7,7,7,0 29,1,2,120,243,0,0,140,0,0,7,7,7,0 29,1,2,100,7,0,0,170,0,0,7,7,7,0 30,1,1,170,237,0,1,170,0,0,7,7,6,0 31,0,2,100,235,0,1,150,0,0,7,7,7,0 32,0,2,105,180,0,0,165,0,0,7,7,7,0 32,1,2,110,225,0,0,184,0,0,7,7,7,0 32,1,2,125,254,0,0,155,0,0,7,7,7,0 33,1,1,120,250,0,0,165,0,0,7,7,7,0 34,0,2,130,161,0,0,190,0,0,7,7,7,0 36,1,2,150,214,0,1,160,0,0,7,7,7,0 34,1,2,100,220,0,0,130,0,0,7,7,7,0 25,0,1,120,160,0,1,185,0,0,7,7,7,0 35,0,4,160,167,0,0,150,0,0,7,7,7,0 35,1,2,120,300,0,2,200,0,0,7,7,7,0 35,1,2,150,204,0,0,160,0,0,7,7,7,0 30,1,2,120,210,0,0,200,0,0,7,7,7,0 30,1,1,112,140,0,0,184,0,1,2,7,7,0 36,1,1,130,200,0,0,170,0,0,7,7,7,0 36,1,1,150,160,0,0,172,0,0,7,7,7,0 37,0,1,120,200,0,0,130,0,0,7,7,7,0 37,0,1,130,211,0,0,142,0,0,7,7,7,0 37,0,4,130,173,0,1,184,0,0,7,7,7,0 37,1,2,130,283,0,1,90,0,0,7,7,7,0 37,1,1,130,294,0,0,150,0,0,7,7,7,0 37,1,4,120,223,0,0,160,0,0,7,7,7,0 37,1,4,130,175,0,0,170,0,0,7,7,7,0 38,0,2,120,275,7,0,120,0,0,7,7,7,0 38,1,2,140,287,0,0,150,0,0,7,7,7,0 38,1,1,185,292,0,0,160,0,0,7,7,7,0 39,0,1,110,182,0,1,180,0,0,7,7,7,0 39,1,2,120,7,0,1,140,0,2,1,7,7,0 39,1,2,120,200,0,0,160,1,1,2,7,7,0 39,1,2,120,204,0,0,145,0,0,7,7,7,0 39,1,2,130,7,0,0,120,0,7,7,7,0 39,1,2,100,141,0,0,100,0,0,7,7,7,0 39,1,1,120,130,0,0,130,0,0,7,7,7,0 39,1,1,160,147,1,0,160,0,0,7,7,7,0 39,1,4,110,247,0,0,132,0,0,7,7,7,0 39,1,4,130,307,0,0,140,0,0,7,7,7,0 40,1,2,130,275,0,0,150,0,0,7,7,7,0 40,1,2,140,280,0,0,172,0,0,7,7,7,0 40,1,1,130,235,0,0,130,0,0,7,7,7,0 40,1,1,104,181,0,0,167,0,0,7,7,7,0	61,0,1,0,1,0,145,0,233,0,1,0,2,0,150,0,0,0,2,5,1,0,0,0,0,0,0 67,0,1,0,0,0,160,0,200,0,0,0,2,0,100,0,1,0,1,5,2,0,3,0,3,0,2 67,0,1,0,0,0,100,0,220,0,0,0,2,0,130,0,1,0,2,0,2,0,2,7,0,1 37,0,1,0,1,0,130,0,250,0,0,0,0,180,0,0,0,2,5,1,0,0,0,3,0,0 61,0,0,0,2,0,130,0,200,0,0,0,2,0,172,0,0,0,1,4,1,0,0,0,3,0,0 56,0,1,0,2,0,120,0,230,0,0,0,0,170,0,0,0,0,1,0,0,0,3,0,0 62,0,0,0,4,0,140,0,260,0,0,0,2,0,160,0,0,0,3,0,1,0,2,0,3,0,3 57,0,0,0,4,0,120,0,254,0,0,0,0,165,0,1,0,0,0,3,0,0,0,3,0,0 63,0,1,0,4,0,130,0,254,0,0,0,2,0,147,0,0,0,1,4,2,0,1,0,7,0,2 53,0,1,0,4,0,140,0,203,0,1,0,2,0,155,0,1,0,3,1,1,0,0,0,7,0,1 57,0,1,0,4,0,160,0,192,0,0,0,0,0,140,0,0,0,0,2,2,0,0,0,6,0,0 56,0,0,0,2,0,160,0,204,0,0,0,2,0,153,0,0,0,1,1,2,0,0,0,3,0,0 56,0,1,0,1,0,130,0,256,0,1,0,2,0,142,0,1,0,0,0,2,0,1,0,6,0,2 44,0,1,0,2,0,120,0,263,0,0,0,0,0,173,0,0,0,0,0,1,0,0,0,7,0,0 52,0,1,0,1,0,172,0,109,0,1,0,0,0,162,0,0,0,0,5,1,0,0,0,7,0,0 57,0,1,0,1,0,150,0,160,0,0,0,0,0,174,0,0,0,0,1,0,1,0,0,0,3,0,0 44,0,1,0,2,0,110,0,220,0,0,0,0,0,160,0,0,0,1,0,1,0,0,0,7,0,1 54,0,1,0,4,0,140,0,239,0,0,0,0,0,160,0,0,0,1,2,1,0,0,0,3,0,0 48,0,0,0,3,0,130,0,275,0,0,0,0,0,139,0,0,0,0,2,1,0,0,0,1,0,0 49,0,1,0,2,0,130,0,266,0,0,0,0,0,171,0,0,0,0,0,1,0,0,0,3,0,0 61,0,1,0,2,0,110,0,211,0,0,0,2,0,144,0,1,0,1,0,2,0,0,0,7,0,0 58,0,0,0,1,0,150,0,203,0,1,0,2,0,162,0,0,0,1,0,1,0,0,0,3,0,0 58,0,1,0,2,0,120,0,204,0,0,0,2,0,160,0,0,0,1,0,2,0,0,0,3,0,1 58,1,0,1,0,3,0,132,0,224,0,0,0,2,0,173,0,0,0,0,3,2,1,0,2,0,7,0,3 60,0,1,0,4,0,130,0,200,0,0,0,2,0,132,0,1,0,2,4,2,0,2,0,6,0,4 58,0,0,0,1,0,120,0,219,0,0,0,0,0,150,0,0,0,1,0,2,0,0,3,1,0,0 54,0,0,0,1,0,120,0,240,0,0,0,0,0,172,0,0,0,0,0,1,0,0,0,3,0,0 66,0,0,0,1,0,150,0,220,0,0,0,0,0,214,0,0,0,2,0,1,0,0,0,3,0,0 43,0,1,0,4,0,150,0,247,0,0,0,0,0,171,0,0,0,1,5,1,0,0,0,3,0,0 60,0,1,0,4,0,110,0,167,0,0,0,2,0,116,0,1,0,2,0,2,0,0,0,7,0,3 69,0,0,0,1,0,140,0,230,0,0,0,0,0,151,0,0,0,1,0,1,0,2,0,3,0,0 60,0,1,0,4,0,117,0,230,0,1,0,0,0,160,0,0,0,1,0,1,0,2,0,7,0,2 64,0,1,0,3,0,140,0,335,0,0,0,0,0,150,0,0,0,0,3,1,0,0,0,3,0,1 59,0,1,0,4,0,135,0,234,0,0,0,0,0,161,0,0,0,0,0,3,2,0,0,0,7,0,2 44,0,1,0,1,0,130,0,233,0,0,0,0,0,179,0,1,0,0,4,1,0,0,0,3,0,0 42,0,1,0,4,0,140,0,220,0,0,0,0,0,170,0,0,0,0,0,1,0,0,0,3,0,0 41,0,1,0,4,0,120,0,177,0,0,0,2,0,120,0,1,0,2,1,0,0,0,7,0,0,3 57,0,1,0,4,0,150,0,270,0,0,0,2,0,112,0,1,0,0,6,2,0,1,0,6,0,1 55,0,1,0,4,0,132,0,153,0,0,0,0,0,132,0,1,0,1,2,2,0,1,0,7,0,2 61,0,1,0,3,0,150,0,243,0,1,0,0,0,137,0,1,0,1,0,2,0,0,0,3,0,0 65,0,0,0,4,0,150,0,225,0,0,0,2,0,114,0,0,0,1,0,2,0,3,0,7,0,4 40,0,1,0,1,0,140,0,199,0,0,0,0,0,170,0,1,0,1,4,1,0,0,0,7,0,0 71,0,0,0,2,0,100,0,302,0,0,0,0,0,162,0,0,0,0,4,1,0,2,0,3,0,0 44,0,1,0,1,0,150,0,177,0,1,0,0,0,157,0,0,0,1,0,1,0,0,0,1,0,0
Ln 1, Col 1	100%	Ln 12, Col 34

1) CLEVELAND DATASET AND SVM:

Heart Disease Prediction

Automatic Heart Disease Detection

Cleveland → Process Dataset
SVM → Train Classifier

Age: 63 Sex: 1 Pain: 3 BP: 145 Chol: 233 FBS: 1 ECG: 0 Maxr: 150 Gang: 0 Est: 2.3 Slope: 0 Vessels: 0 Thal: 1

Predict Heart Disease

Classification Report

Confusion Matrix

This Patient has a Low Heart Disease

2) CLEVELAND DATASET AND NAÏVE BAYES:

Heart Disease Prediction

Automatic Heart Disease Detection

Cleveland → Process Dataset
Naive Bayes → Train Classifier

Age: 57 Sex: 0 Pain: 3 BP: 150 Chol: 354 FBS: 1 ECG: 2 Maxr: 183 Gang: 1 Est: 3.6 Slope: 3 Vessels: 2 Thal: 6

Predict Heart Disease

Classification Report

Confusion Matrix

This Patient has a Low Heart Disease

3) CLEVELAND DATASET AND LOGISTIC REGRESSION:

Heart Disease Prediction

Automatic Heart Disease Detection

Cleveland → Process Dataset
Logistic Regression → Train Classifier

Age: 70 Sex: 1 Pain: 4 BP: 140 Chol: 300 FBS: 1 ECG: 3 Maxhr: 180 Restng: 1 Restb: 3.0 Slope: 2 Vessels: 2 Thal: 3

Predict Heart Disease

Classification Report

Confusion Matrix

The Person has Mild Heart Disease

4) HUNGARIAN DATASET AND SVM:

Heart Disease Prediction

Automatic Heart Disease Detection

Hungarian → Process Dataset
SVM → Train Classifier

Age: 50 Sex: 0 Pain: 3 BP: 125 Chol: 220 FBS: 0 ECG: 0 Maxhr: 140 Restng: 0 Restb: 0.9 Slope: 2 Vessels: 0 Thal: 3

Predict Heart Disease

Classification Report

Confusion Matrix

The Person has Mild Heart Disease

5) HUNGARIAN DATASET AND NAÏVE BAYES:

Heart Disease Prediction

Automatic Heart Disease Detection

Hungarian → Process Dataset
Naive Bayes → Train Classifier

Age: 37 Sex: 1 Pain: 3 BP: 130 Chol: 250 FBS: 0 ECG: 1 Maxr: 187 Gang: 0 Est: 3.5 Slope: 0 Vessels: 0 Thal: 3

Predict Heart Disease

Classification Report

Confusion Matrix

The Person has Mild Heart Disease

6) HUNGARIAN DATASET AND LOGISTIC REGRESSION:

Heart Disease Prediction

Automatic Heart Disease Detection

Hungarian → Process Dataset
Logistic Regression → Train Classifier

Age: 80 Sex: 0 Pain: 4 BP: 150 Chol: 400 FBS: 0 ECG: 3 Maxr: 190 Gang: 0 Est: 4.0 Slope: 3 Vessels: 3 Thal: 7

Predict Heart Disease

Classification Report

Confusion Matrix

The Person has Mild Heart Disease

7) VA DATASET AND SVM:

Heart Disease Prediction

Automatic Heart Disease Detection

VA → Process Dataset
SVM → Train Classifier

Age: 50 Sex: 1 Pain: 4 BP: 130 Chol: 320 FBS: 1 ECG: 1 Maxhr: 170 Tang: 1 Est: 2.5 Slope: 1 Vessels: 3 Thal: 3

Predict Heart Disease

Classification Report

Confusion Matrix

The Person has Mild Heart Disease

8) VA DATASET AND NAÏVE BAYES:

Heart Disease Prediction

Automatic Heart Disease Detection

VA → Process Dataset
Naive Bayes → Train Classifier

Age: 77 Sex: 0 Pain: 3 BP: 120 Chol: 250 FBS: 0 ECG: 3 Maxhr: 160 Tang: 0 Est: 1.8 Slope: 0 Vessels: 0 Thal: 3

Predict Heart Disease

Classification Report

Confusion Matrix

The Person has Very Mild Heart Disease

9) VA DATASET AND LOGISTIC REGRESSION:

Heart Disease Prediction

Automatic Heart Disease Detection

VA → Process Dataset
Logistic Regression → Train Classifier

Age: 41 Sex: 0 Pain: 1 BP: 130 Chol: 204 FBS: 0 ECG: 0 Maxhr: 172 Bang: 0 Est: 1.4 Slope: 2 Vessels: 0 Thal: 1

Predict Heart Disease

Classification Report

Confusion Matrix

No Patient Found With Given Heart Disease

10) ALL DATASET AND SVM:

Heart Disease Prediction

Automatic Heart Disease Detection

all → Process Dataset
SVM → Train Classifier

Age: 56 Sex: 1 Pain: 1 BP: 120 Chol: 236 FBS: 0 ECG: 1 Maxhr: 176 Bang: 0 Est: 0.8 Slope: 2 Vessels: 0 Thal: 2

Predict Heart Disease

Classification Report

Confusion Matrix

No Patient Found With Given Heart Disease

11) ALL DATASET AND NAÏVE BAYES:

Heart Disease Prediction

Automatic Heart Disease Detection

all → Process Dataset
Naive Bayes → Train Classifier

FBS ECG

Age Sex Pain BP Cholesterol Mchc Hgb Est Slope Vessels Thal

50 1 4 130 320 1 1 170 1 2.5 1 3 3

Predict Heart Disease

Classification Report

Confusion Matrix

The Patient has **Not** Heart Disease

12) ALL DATASET AND LOGISTIC REGRESSION:

Heart Disease Prediction

Automatic Heart Disease Detection

all → Process Dataset
Logistic Regression → Train Classifier

FBS ECG

Age Sex Pain BP Cholesterol Mchc Hgb Est Slope Vessels Thal

70 0 3 140 290 1 2 160 1 3.0 3 0 1

Predict Heart Disease

Classification Report

Confusion Matrix

The Patient has **Yes** Heart Disease

13)ALL DATASET AND NAÏVE BAYES:

Heart Disease Prediction

Automatic Heart Disease Detection

all → Process Dataset
Naive Bayes → Train Classifier

Age: 50 Sex: 1 Pain: 4 BP: 130 Chol: 320 FBS: 1 ECG: 1 Maxhr: 170 Tang: 1 Est: 2.5 Slope: 1 Vessels: 3 Thal: 3

Predict Heart Disease

Classification Report

Confusion Matrix

This Patient has **Not** Heart Disease

14)ALL DATASET AND LOGISTIC REGRESSION:

Heart Disease Prediction

Automatic Heart Disease Detection

all → Process Dataset
Logistic Regression → Train Classifier

Age: 70 Sex: 0 Pain: 3 BP: 140 Chol: 290 FBS: 1 ECG: 3 Maxhr: 168 Tang: 1 Est: 3.0 Slope: 3 Vessels: 0 Thal: 3

Predict Heart Disease

Classification Report

Confusion Matrix

This Patient has **Not** Heart Disease

6.3 RESULTS AND DISCUSSION

The accuracy arises from the attributes collected from the datasets. These accuracies can also be increased with the help of better datasets and better computational systems. Based on the accuracy produced by the algorithms the best algorithm which gives accuracy is chosen for finding the prediction. The user interface is designed in such a way to use effectively by the user. The outputs and the accuracies generated are reviewed and the results are displayed. In this mostly the algorithm which has the highest accuracy gives the more accurate result.

CONCLUSION:

The correlative application and analysis of distinct machine learning algorithms in the software which gives an immediate mechanism for the user to use the machine learning algorithms in python and using flask framework to create a web application for forecasting the cardiovascular diseases. This is non-ethical study aims to use available machine learning techniques. Future work includes different ensemble methods of these classifiers which can advance to better performance with more parameter settings for these classifiers.

REFERENCE:

- [1] C. S. Dangare and S. S. Apte, "Improved study of heart disease prediction system using data mining classification techniques," *International Journal of Computer Applications*, vol. 47, no. 10, pp. 44–48, 2012.
- [2] S. Palaniappan and R. Awang, "Intelligent heart disease prediction system using data mining techniques," pp. 108–115, 2008.
- [3] Y. E. Shao, C.-D. Hou and C.-C. Chiu, "Hybrid intelligent modeling schemes for heart disease classification," *Applied Soft Computing*, vol. 14, pp. 47–52, 2014.
- [4] M. Shouman, T. Turner, and R. Stocker, "Using data mining techniques in heart disease diagnosis and treatment," pp. 173–177, 2012.;3
- [5] P. V. Ankur Makwana, "Identify the patients at high risk of re-admission in hospital in the next year," *International Journal of Science and Research*, vol. 4, pp. 2431–2434, 2015.
- [6] J. Nahar, T. Imam, K. S. Tickle, and Y.-P. P. Chen, "Computational intelligence for heart disease diagnosis: A medical Knowledge driven approach," *Expert Systems with Applications*, vol. 40, no. 1, pp. 96–104, 2013.
- [7] Y. Xing, J. Wang, Z. Zhao, and Y. Gao, "Combination data mining methods with new medical data to predicting outcome of coronary heart disease," pp. 868–872, 2007.
- [8] Combination data mining methods with new medical data to predicting outcome of coronary heart disease," in *Convergence Information Technology*, 2007. *International Conference on. IEEE*, 2007, pp. 868–872.
- [9] Y. E. Shao, C.-D. Hou, and C.-C. Chiu, "Hybrid intelligent modeling, schemes for heart disease classification," *Applied Soft Computing*, vol. 14, pp. 47–52, 2014.