HEART DISEASE PREDICTION SYSTEM

A MINI PROJECT REPORT

Submitted by

Abbas Hussain Muzammil

in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

AURORA'S TECHNOLOGICAL AND RESEARCH INSTITUTE

(Affiliated to JNTUH and approved by AICTE,New Delhi)

PARVATHAPUR,UPPAL,Hyderabad - 500039

JANUARY 2022

AURORA’S TECHNOLOGICAL AND RESEARCH INSTITUTE

(Affiliated to JNTU,Hyderabad)

PARVATHAPUR,UPPAL,Hyderabad - 500039

DECLARATION

We hereby declare that the work described in this project,entitled **HEART DISEASE PREDICTION SYSTEM** which is being submitted by me in partial fulfillment for the award of Bachelor Of Technology in Computer Science and Engineering to **Aurora’s Technological and Research Institute** is the result of investigation carried out by me under the guidance of **Ms. M Vineela,Assistant Professor, CSE**

The work is original and has not been submitted for my any degree of this or any other university.

Place: Hyderabad

Date:

Abbas Hussain Muzammil (18841A0562))

AURORA’S TECHNOLOGICAL AND RESEARCH INSTITUTE

(Affiliated to JNTU,Hyderabad)

PARVATHAPUR,UPPAL,Hyderabad - 500039

BONAFIDE CERTIFICATE

Certified that this project report **“HEART DISEASE PREDICTION SYSTEM**” is the bonafide work of **“Abbas Hussain Muzammil”** who carried out the project work under my supervision.

GUIDE	**MINI PROJECT COORDINATOR**
Ms. M Vineela	**Ms. V. Shilpa**
Assistant Professor	Assistant Professor
Dept.of CSE	Dept.of CSE

HEAD OF THE DEPARTMENT	**DIRECTOR**
Ms. A Durga Pavani	**Mr.Srikanth Jatla**
Dept.of CSE	

EXTERNAL EXAMINER

Acknowledgement

This work has been done during the project period and it was a very good opportunity to put theoretical knowledge into planned exercise with an aim to solve a real time problem and also to develop confidence to face various practical situations.

We would like to express our gratitude to Director **Mr.Srikanth Jatla**, Aurora's Technological and Research Institute for providing us with a congenial atmosphere and encouragement

We express our sincere thanks to Head of the Department **Ms. A Durga Pavani** for giving us the support and her kind attention and valuable guidance to us throughout the course.

We express our sincere thanks to Project Coordinator **Ms. V. Shilpa**for helping us to complete our project work by giving valuable suggestions.

We convey thanks to our project Guide **Ms. M Vineela** , Department Of Computer Science and Engineering, for providing encouragement,constant support and guidance which was of great help to complete this project successfully.

Abstract

The human heart is a muscular organ in the human body that pumps blood through the blood vessels of the circulatory system. The pumped blood carries oxygen and nutrients to the body and removes waste products from our bodies. Heart disease is regarded as the world's biggest killer in the world. It's very risky to human lives whenever heart disease or failure occurs. To avoid these risks, we predict the early disease symptoms. In the medicine and the health care domain, with the support of data mining techniques, we predict early disease detection and patient care. Different Algorithms are used for clinical decision support systems to get accurate results for effective prediction of heart disease. However, bio-inspired algorithms help in exploring new areas of application. This project focuses on different bio-inspired algorithms such as Genetic, Bee and Bat for heart disease prediction. It was observed that these Bio-Inspired algorithms are powerful algorithms for optimization and have a wide impact on computing. These algorithms assure that all the values are related to the dataset and remove irrelevant data

Contents

List of Figures

Chapter 1

INTRODUCTION

1.1 Overview

Worldwide, Mortality rate is highly effected due to the heart disease. It becomes life-threatening disease among all because, heart is the hardest working muscle in our body. It pumps blood, supplying oxygen and nutrients to our body. Due to the heart disease,all these activities done by heart are not properly functioned. So, it is better to identify the risk factors of heart diseases to minimize the deaths. There are many factors like cholesterol, smoking, blood pressure, diabetes, stress, etc., causes heart disease. Regular symptoms of heart disease include discomfort in left side of the chest area, pain in upper abdomen and back, heart burn or indigestion, numbness in legs, etc. These symptoms vary from person to person. So, it becomes thought-provoking topredict the heart disease

Since Heart Attacks are on the rise, the novel algorithms are not feasible to work with categorical clinical data.By making use of Bio-Inspired Algorithms we usually mimic the biological processes and are inspired by natural evolution. These bio-inspired algorithms are quite interesting and bridge a gap between computer science, biology, artificial intelligence as well as economics.

1.2 Scope

With a huge increase in data generation, it became extremely challenging to provide a highly efficient and optimal solution. The intelligent approaches like bio inspired algorithms solve these complex problems in time with high efficiency. To solve the traditional optimization problems the current trend uses bio inspired algorithms because their implementation is based on the biological feature extraction.
However,there is so much scope still left for bio inspired algorithms in computer Science as still very minor fraction being explored and there is still lot of room to explore these natural behaviours.
we have seen different classification methods for predicting heart disease by extracting the features of respective bio inspired algorithms. This study analyses that we can achieve accuracy of heart disease prediction depends on both feature extraction of bio inspired algorithm and type of classification methods used.

1.3 User Needs

A system which efficiently and accurately classifies patients sufferring from cardiovascular diseases and approprately take necessary action so as to save his/her life from later stages of Heart Disease.

1.4 Assumptions and Dependencies

1.4.1 Assumptions:

User need to have his/her clinical data

1.4.2 Dependencies:

Python programming language must be installed and ML libraries needs to be used

Chapter 2

REQUIREMENT SPECIFICATION

2.1 Software Requirements

Following are the Software's required followed by their detailed description

2.1.1 Python

- Python is a high-level, general-purpose and a very popular programming language.
- Python programming language (latest Python 3) is being used in web development, Machine Learning applications, along with all cutting edge technology in Software Industry.
- Python is currently the most widely used multi-purpose, high-level programming language.
- Python allows programming in Object-Oriented and Procedural paradigms.
- Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.
- Python is dynamically typed and has automatically garbage is collected.
- It supports multiple programming paradigms, including procedural, object-oriented, and functional programming

- The biggest strength of Python is huge collection of standard library which can be used for the following:
  - Machine Learning
  - GUI Applications (like Kivy, Tkinter, PyQt etc.)
  - Web frameworks like Django (used by YouTube, Instagram, Dropbox)
  - Image processing (like OpenCV, Pillow)
  - Web scraping (like Scrapy, BeautifulSoup, Selenium)
  - Test frameworks
  - Multimedia
  - Scientific computing
  - Text processing and many more...

2.1.2 Operating System

An operating system (OS) is system software that manages computer hardware, software resources, and provides common services for computer programs. The operating system should support Python 3.7.0 and all the libraries mentioned above.

2.2 Libraries Used

2.2.1 Pandas

Pandas is an open-source library that is used for data analysis and manipulation. It is a Python package that offers various data structures and operations for manipulating numerical data and time series. It is mainly popular for importing and analyzing data much easier. Pandas is fast and it has high-performance productivity for users. In this project we have used pandas version 0.25.3.

2.2.2 Numpy

Numpy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. In this project we have used Numpy version 1.81.1.

2.2.3 SwarmPackage

SwarmPackagepy is a library of swarm optimizaion algorithms. It include 14 optimizaion algorithms and each can be used for solving specific optimization problem such as BEE BAT

2.2.4 Sklearn

Sklearn is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python.

2.2.5 Tkinter

• Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit

2.3 Hardware Requirements

2.3.1 Processor

A central processing unit (CPU), also called a central processor or main processor, is the electronic circuitry within a computer that carries out the instructions of a computer program by performing the basic arithmetic, logic, controlling, and input/output (I/O) operations specified by the instructions. The computer industry has used the term ”central processing unit” at least since the early 1960s. Traditionally, the term ”CPU” refers to a processor, more specifically to its processing unit and control unit (CU), distinguishing these core elements of a computer from external components such Principal components of a CPU include the arithmetic logic unit (ALU) that performs arithmetic and logic operations, processor registers that supply operands to the ALU and store the results of ALU operations and a control unit that orchestrates the fetching (from memory) and execution of instructions by directing the coordinated operations of the ALU, registers and other components. Most modern CPUs are microprocessors, meaning they are contained on a single integrated circuit (IC) chip. An IC that contains a CPU may also contain memory, peripheral interfaces, and other components of a computer; such integrated devices are variously called microcontrollers or systems on a chip (SoC). Some computers employ a multi-core processor, which is a single chip containing two or more CPUs called ”cores”; in that context, one can speak of such single chips as ”sockets”. Array processors or vector processors have multiple processors that operate in parallel, with no unit considered central. There also exists the concept of virtual CPUs which are an abstraction of dynamical aggregated computational resources.

2.3.2 RAM

Random-access memory is a form of computer data storage that stores data and machine code currently being used. A random-access memory device allows data items to be read or written in almost the same amount of time irrespective of the physical location of data inside the memory. In contrast, with other direct-access data storage media such as hard disks, CD-RWs, DVD-RWs and the older magnetic tapes and drum memory, the time required to read and write data items varies significantly depending on their physical locations on the recording medium, due to mechanical limitations such as media rotation speeds and arm movement

2.3.3 Hard Disk

A hard disk drive (HDD), hard disk, hard drive, or fixed disk, is an electromechanical data storage device that uses magnetic storage to store and retrieve digital information using one or more rigid rapidly rotating disks (platters) coated with magnetic material.

The platters are paired with magnetic heads, usually arranged on a moving actuator arm, which read and write data to the platter surfaces. Data is accessed in a random-access manner, meaning that individual blocks of data can be stored or retrieved in any order and not only sequentially. HDDs are a type of non-volatile storage, retaining stored data even when powered off. Introduced by IBM in 1956. HDDs became the dominant secondary storage device for general-purpose computers by the early 1960s. Continuously improved, HDDs have maintained this position into the modern era of servers and personal computers. HDDs are a type of non-volatile storage, retaining stored data even when powered off. Introduced by IBM in 1956. HDDs became the dominant secondary storage device for general-purpose computers by the early 1960s. Continuously improved, HDDs have maintained this position into the modern era of servers and personal computers. More than 200 companies have produced HDDs historically, though after extensive industry consolidation most units are manufactured by Seagate, Toshiba, and Western Digital. HDDs dominate the volume of storage produced (Exabytes per year) for servers.

Chapter 3

LITERATURE SURVEY

3.1 Existing System

• Existing system uses novel algorithms that employ optimization techniques for any given complex problem. These novel algorithms require a lot of observations and are restricted by a large number of association rules. As the problem size increases, it requires huge amounts of computational effects for processing huge data.
• In the existing system, detection is not possible at an earlier stage and practical use of various collected data is time-consuming.

3.2 Problem Identification

- Novel algorithms are used for giving solutions of optimized and distributed control problems. But these novel algorithms are derived from the optimization algorithms, which comes from the observations. As the problem size increases, it requires huge amounts of computational efforts for traditional method.

- So, to avoid this we use bio inspired algorithms for efficient results in a deterministic approach.

- The explosion of data becomes more prominent in digital era. It is very difficult to find the solution from standard algorithms due to the complexity of computations.

Chapter 4

SOFTWARE REQUIREMENT ANALYSIS

4.1 Problem Definition

- Heart Disease Prediction is very important because it leads to human death. Different data mining techniques are implemented on medical data to predict heart disease.
- Heart disease is one of the expensive health problems to cure. Many people are died in the world due to this heart disease. Human heart has four chambers and the blood flows from one chamber to the other through valves.
- These valves open and close for the blood to be pumped across one chamber to the other. A situation where the valves are not able to open or close completely, leads to valvular diseases like regurgitation or stenosis.
- According to the estimates, one in four women or one in five men are prone to the heart ailments, which are varied through one's lifestyle habits, poor diet, obesity, lack of physical activity, diabetes, or excessive intake of alcohol.
- **Risk factors of heart disease are:**
  - Smoking: Smokershave high chance of risk than non-smokers.
  - Cholesterol: High Cholesterol leads to heart attacks in most of the cases.

- Blood pressure: High BP leads to heart attacks because high BP avoids theblood flow to the heart muscle.
- Diabetes: Diabetic patients are prone tomorerisk as high glucose levels cause damage to heart nerves and blood vessels
- Sedentary life style: Sedentary life style leads to many problems like high BP, diabetes, obesity, etc. Then there will be high chance to get heart diseases.
- Eating Habits: Eating habits plays crucial role because unhealthy eating habits leads to high risk.
- Stress: Having too much stress leads to high BP then there will be more chances to get heart diseases

4.2 Functional Requirements

These are the requirements that the end user specifically demands as basic facilities that the system should offer. All these functionalities need to be necessarily incorporated into the system as a part of the contract. These are represented or stated in the form of input to be given to the system, the operation performed and the output expected. They are basically the requirements stated by the user which one can see directly in the final product, unlike the non-functional requirements.

4.2.1 Upload Dataset

Using this module, we will upload clinical data recorded at diagnostic centers to our standalone application.

4.2.2 Data Preprocessing

Data Pre-processing is a data mining technique to turn the raw data gathered from diverse sources into cleaner information that's more suitable for work. In other words, it's a preliminary step that takes all of the available information to organize it, sort it, and merge it

4.2.3 Confusion Matrix

A confusion matrix, also known as an error matrix, is a specific table layout that allows visualization of the performance of an algorithm, typically a supervised learning one (in unsupervised learning it is usually called a matching matrix). Each row of the matrix represents the instances in an actual class while each column represents the instances in a predicted class, or vice versa.

4.2.4 Accuracy Graph

Shows the accuracy of various ML algorithms used and help us to rely on those algorithms

4.2.5 Prediction

After computing the training data, the test dataset is loaded and prediction is done.

4.2.6 Precision

Precision is the ratio of correctly predicted positive observations to the total predicted positive observations.
Precision is the ratio of correctly predicted positive observations to the total predicted positive observations

4.2.7 Recall

Recall refers to the percentage of total relevant results correctly classified by your algorithm.

4.2.8 f1-score

It is also called F score or the F Measure.
F1 score is the weighted average of the precision and the recall. The best value of F1 would be 1 and the worst would be 0.

4.2.9 Support

Support maybe defined as a number of samples of the true response that lies in each class of target values.

4.3 Non Functional Requirements

These are basically the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project to other.
They are also called non-behavioral requirements.

4.3.1 Usability

The system must be easy to operate and user friendly

4.3.2 Reliability

The results shown by the system must be reliable enough to take appropriate decisions.

4.3.3 Performance

The performance of the system is measured and compared by employing vaious related algorithms.

4.3.4 Scalability

The system must be ready to take large data sets to ensure scalability

4.3.5 Portability

The system must be able to run on various other System software and should maintain Integrated results.

Chapter 5

SOFTWARE DESIGN

5.1 Architectural Design

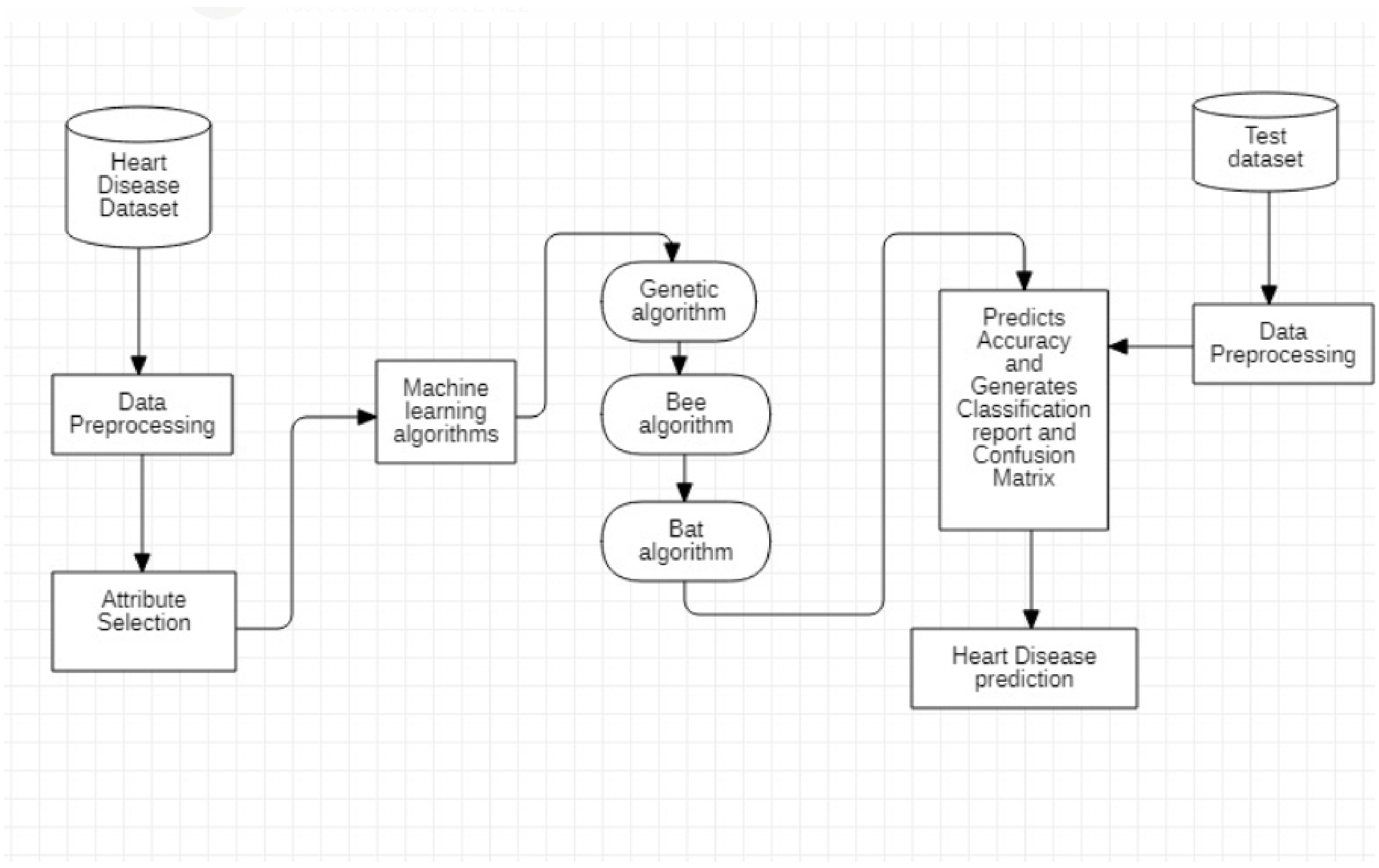


Figure 5.1: System Architecture

5.2 System Working

5.2.1 Data-Preprocessing

The heart disease data-set is loaded into this standalone application and then the process of Data preprocessing starts.
Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model.
Data preprocessing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model Data preprocessing involves following steps:

5.2.1.1 Getting the dataset

To create a machine learning model, the first thing we required is a dataset as a machine learning model completely works on data. The collected data for a particular problem in a proper format is known as the dataset.
Dataset may be of different formats for different purposes, such as, if we want to create a machine learning model for business purpose, then dataset will be different with the dataset required for a liver patient. So each dataset is different from another dataset. To use the dataset in our code, we usually put it into a CSV file. However, sometimes, we may also need to use an HTML or xlsx file.
What is a CSV File?
CSV stands for ”Comma-Separated Values” files; it is a file format which allows us to save the tabular data, such as spreadsheets. It is useful for huge datasets and can use these datasets in programs.
We can also create our dataset by gathering data using various API with Python and put that data into a .csv file.

5.2.1.2 Importing libraries

In order to perform data preprocessing using Python, we need to import some predefined Python libraries. These libraries are used to perform some specific jobs.

There are three specific libraries that we will use for data preprocessing, which are:

- The **Numpy** Python library is used for including any type of mathematical operation in the code. It is the fundamental package for scientific calculation in Python. It also supports to add large, multidimensional arrays and matrices. So, in Python, we can import it as:

  ```
  import numpy as np
  ```

- The **matplotlib**, which is a Python 2D plotting library, and with this library, we need to import a sub-library pyplot. we can import it as:

  ```
  import matplotlib.pyplot as mpt
  ```

- The **Pandas** library, which is one of the most famous Python libraries and used for importing and managing the datasets. It is an open-source data manipulation and analysis library We can import it as:

  ```
  import pandas as pd
  ```

5.2.1.3 Importing datasets

Now to import the dataset, we will use read_csv() function of pandas library, which is used to read a csv file and performs various operations on it. Using this function, we can read a csv file locally as well as through an URL.
We can use read_csv function as:

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

Then

- **Extracting dependent and independent variables**

5.2.1.4 Finding Missing Data

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

Ways to handle missing data:

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

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

- **By calculating the mean:** In this way, we will calculate the mean of that column or row which contains any missing value and will put it on the place of missing value. This strategy is useful for the features which have numeric data such as age, salary, year, etc.
  Here, we will use this approach.
  To handle missing values, we will use **Scikit-learn library** in our code, which contains various libraries for building machine learning models. Here we will use **Imputer** class of **sklearn.preprocessing** library.

5.2.1.5 Encoding Categorical Data

Categorical data is data which has some categories such as, in our dataset; there are three categorical variable, Stage 1,Stage 2,and Stage 3.
Since machine learning model completely works on mathematics and numbers, but if our dataset would have a categorical variable, then it may create trouble while building the model. So it is necessary to encode these categorical variables into numbers. We will make use of **LabelEncoder()** class from preprocessing library. such as:

```
from sklearn.preprocessing import LabelEncoder
```

5.2.1.6 Splitting dataset into training and test set

In machine learning data preprocessing, we divide our dataset into a training set and test set. This is one of the crucial steps of data preprocessing as by doing this, we can enhance the performance of our machine learning model.

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

- **Training Set:** A subset of dataset to train the machine learning model, and we already know the output.
- **Test set:** A subset of dataset to test the machine learning model, and by using the test set, model predicts the output.

For splitting the data-set we will use below line of code:

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test= train_test_split(x, y,
test_size= 0.2, random_state=0)
```

In the above code, the first line is used for splitting arrays of the dataset into random train and test subsets.
In the second line, we have used four variables for our output that are

- x_train: features for the training data
- x_test: features for testing data
- y_train: Dependent variables for training data
- y_test: Independent variable for testing data

  In **train_test_split()** function, we have passed four parameters in which first two are for arrays of data, and test_size is for specifying the size of the test set. The test_size maybe .5, .3, or .2, which tells the dividing ratio of training and testing sets. The last parameter random_state is used to set a seed for a random generator so that you always get the same result, and the most used value for this is 42.

5.2.1.7 Feature scaling

Feature scaling is the final step of data preprocessing in machine learning. It is a technique to standardize the independent variables of the dataset in a specific range. In feature scaling, we put our variables in the same range and in the same scale so that no any variable dominate the other variable.
For feature scaling, we will import **StandardScaler** class of **sklearn.preprocessing** library as:

```
from sklearn.preprocessing import StandardScaler
```

Now, we will create the object of **StandardScaler** class for independent variables or features. And then we will fit and transform the training dataset.

```
st_x= StandardScaler()
x_train= st_x.fit_transform(x_train)
```

For test dataset, we will directly apply transform() function instead of fit_transform() because it is already done in training set.

```
x_test= st_x.transform(x_test)
```

5.2.2 Machine Learning Algorithms

5.2.2.1 Genetic Algorithm

In 1975, John Holland introduced genetic algorithm for solving optimization problems. This evolutionary algorithm produces best solutions for search and optimization problems. Here we use three different biologically inspired operators known as selection, mutation and crossover. Genetic algorithm generates a pool of possible solutions for a given problem. This pool of solutions is called population and Each solution in a pool is known as chromosome.This population of solutions are generated repeatedly until the required fitness value is observed. This mechanism is adopted from Charles Darwin's theory. Genetic algorithm not only deals with discrete and continuous problems but also implemented in multi objective problems.

5.2.2.2 BEE Algorithm

Bee Algorithm was proposed by Dervis,Karaboga in 2005.Honey bees traverse in multiple directions in groups to locate food sources It has been observed that the density of honey bees over heavily pollen flower is higher when compared to less pollen flower. Honey bees are categorized into three types based on their searching of food. Namely, Employed bees, onlookers and scout bees.Once an employed bee finds food sources, it informs to other bees through a dance, is called waggle dance . Onlookers understand the location, quality and quantity of food sources through this waggle dance. Waggle dance comprises of various movements and each one has respective meaning.Based on these signs it will advertise the food location and encourage the remaining bees to follow. After the dance, some recruited members follow the scout bees to find the food source.Scout bees move randomly move from one place to another to discover new flower patches with rich food .

5.2.2.3 BAT Algorithm

Bat algorithm, one of the bio inspired algorithms is a recent one developed by Yang. It is heavily inspired by Bats way of communication using echo-based location.This algorithm can be employed in continuous solution domain to solve both single objective and multiple objective optimization problems. Echo based location is a technique used by bats to find food and navigate even in dark. Bats release a high sound pulse in a definite angle and listen the echo which comes back from the surrounding objects to detect the food, prey and other objects on the way.Bats change intensity of sound and frequency they make when food is found. Similar technique is taken to solve iterative model problems in the field of vector algebra.For global optimization,

5.3 UML Diagrams

The Unified Modelling Language is a general-purpose,development,modelling language in the field of software engineering that is intended to provide a standard way to visualize the design of a system.

The UML diagrams are categorized into structural diagrams, behavioral diagrams, and also interaction overview diagrams.

- **Behavioural Diagrams** portray a dynamic view of a system or the behavior of a system, which describes the functioning of the system. It includes use case diagrams, state diagrams, and activity diagrams. It defines the interaction within the system.
- **Structural Diagrams** depict a static view or structure of a system. It is widely used in the documentation of software architecture. It embraces class diagrams, composite structure diagrams, component diagrams, deployment diagrams, object diagrams, and package diagrams. It presents an outline for the system. It stresses the elements to be present that are to be modeled.
- **Interaction Diagrams** are a subclass of behavioral diagrams that give emphasis to object interactions and also depicts the flow between various use case elements of a system. In simple words, it shows how objects interact with each other and how the data flows within them. It consists of communication, interaction overview, sequence, and timing diagrams.

The diagrams are hierarchically classified in the following figure:

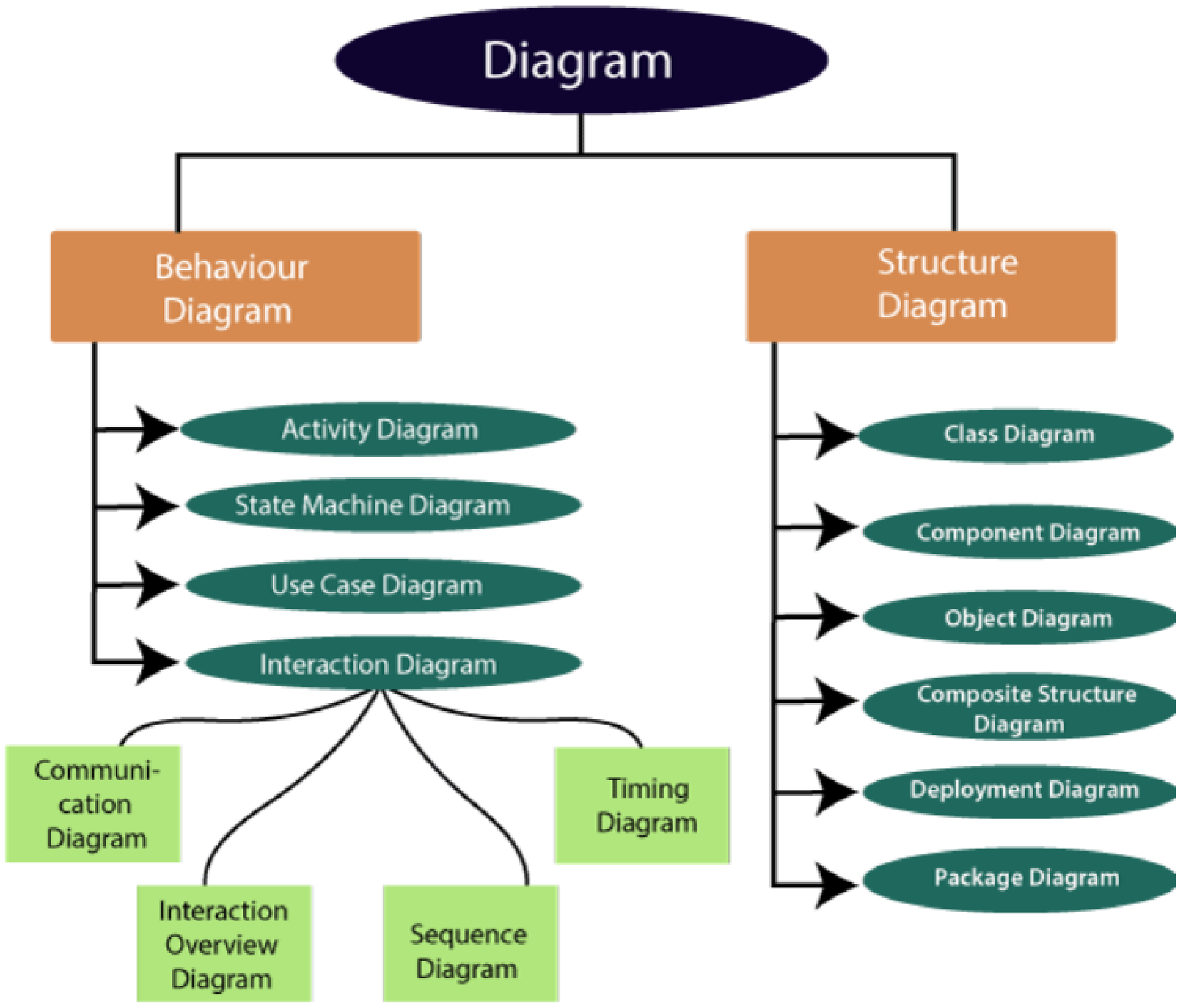


Figure 5.2: UML Diagram

5.3.1 Use Case Diagram

Use Case Diagram represents the functionality of a system by utilizing actors and use cases. It encapsulates the functional requirement of a system and its association with actors. It portrays the use case view of a system.

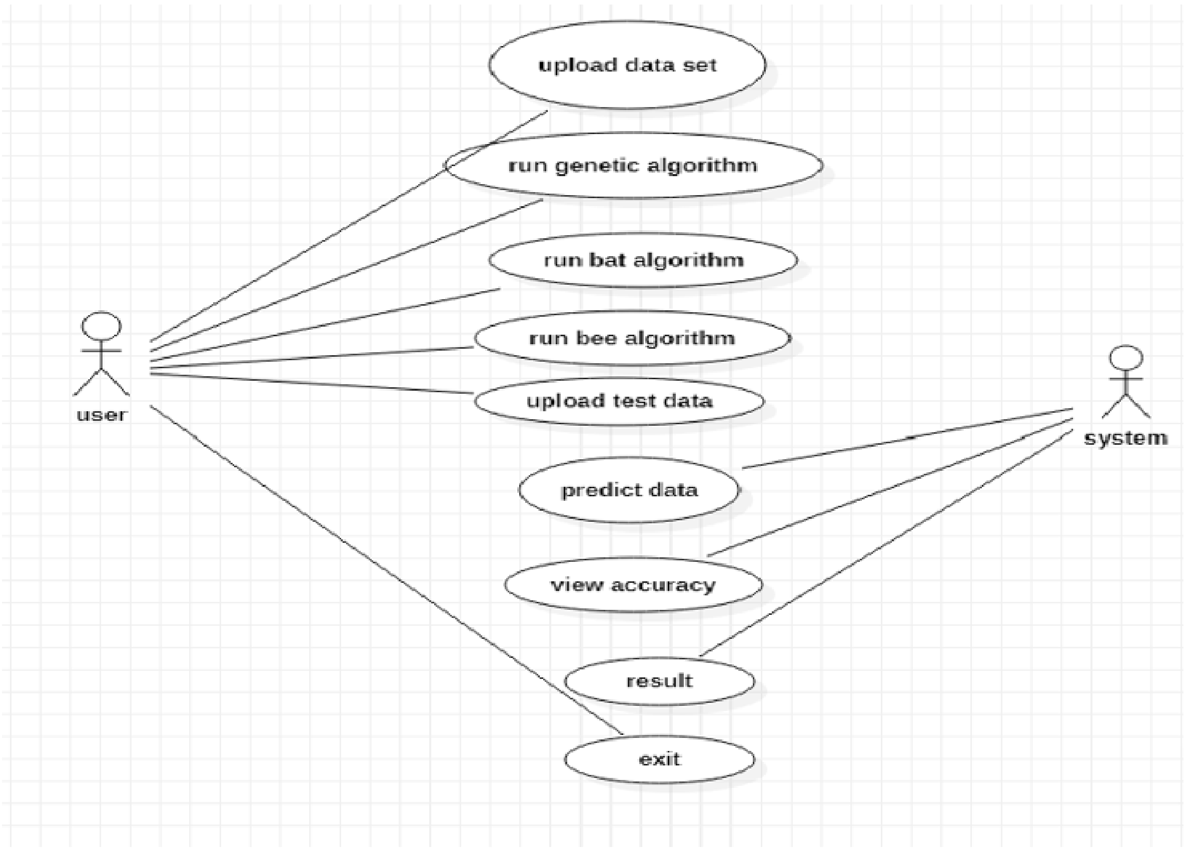


Figure 5.3: Use Case Diagram

5.3.2 Class Diagram

Class Diagrams are one of the most widely used diagrams. It is the backbone of all the object-oriented software systems. It depicts the static structure of the system. It displays the system's class, attributes, and methods. It is helpful in recognizing the relation between different objects as well as classes.

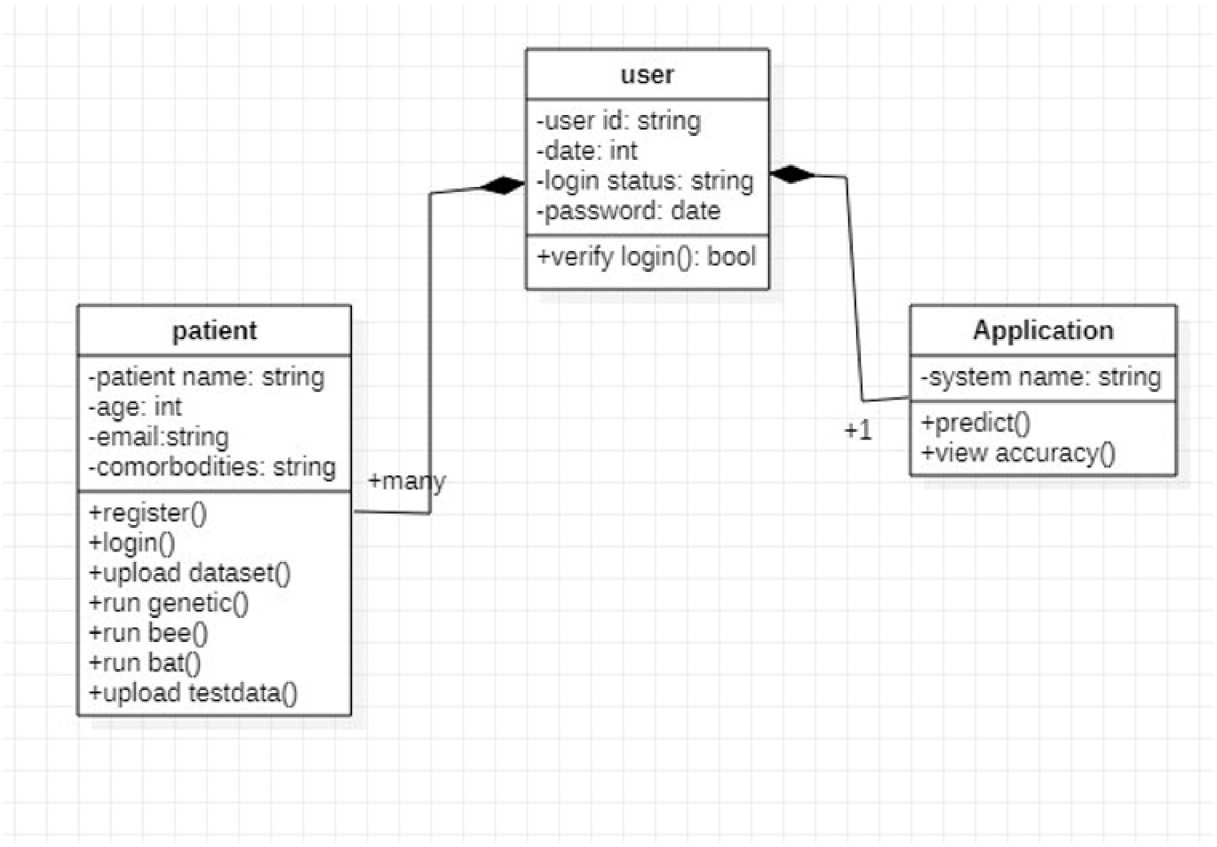


Figure 5.4: Class Diagram

5.3.3 Sequence Diagram

Sequence Diagram shows the interactions between the objects in terms of messages exchanged over time. It delineates in what order and how the object functions are in a system.

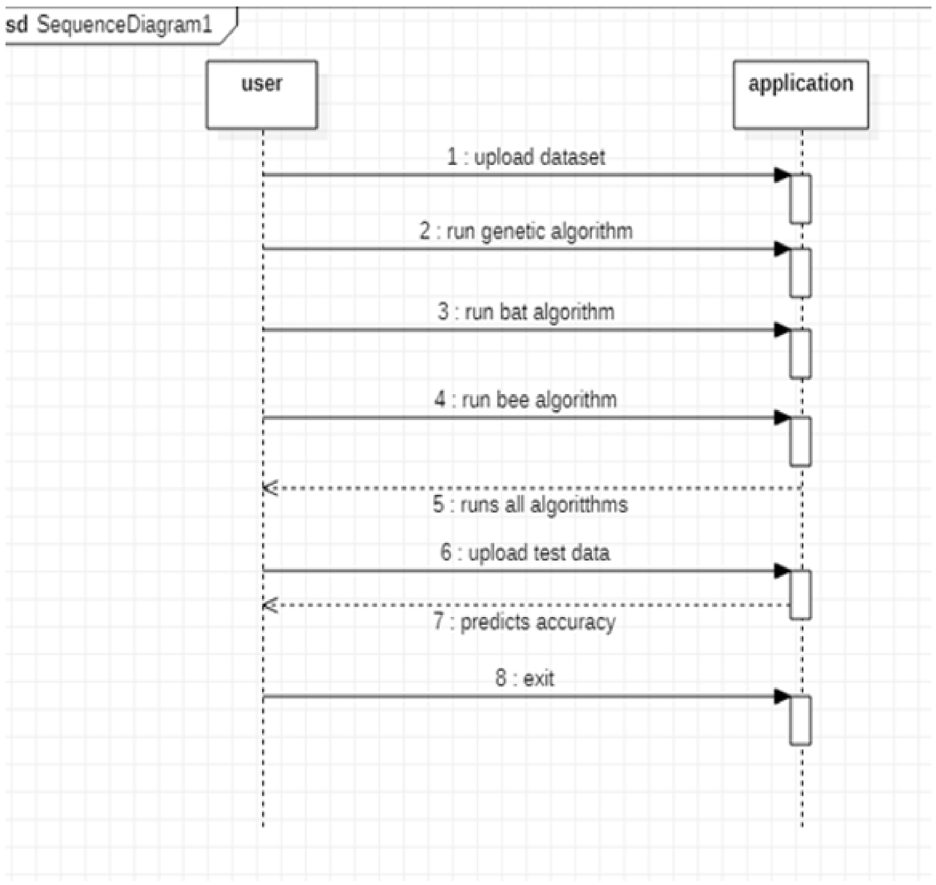


Figure 5.5: Sequence Diagram

5.3.4 Communication Diagram

Communication Diagram shows the interchange of sequence messages between the objects. It focuses on objects and their relations. It describes the static and dynamic behavior of a system.

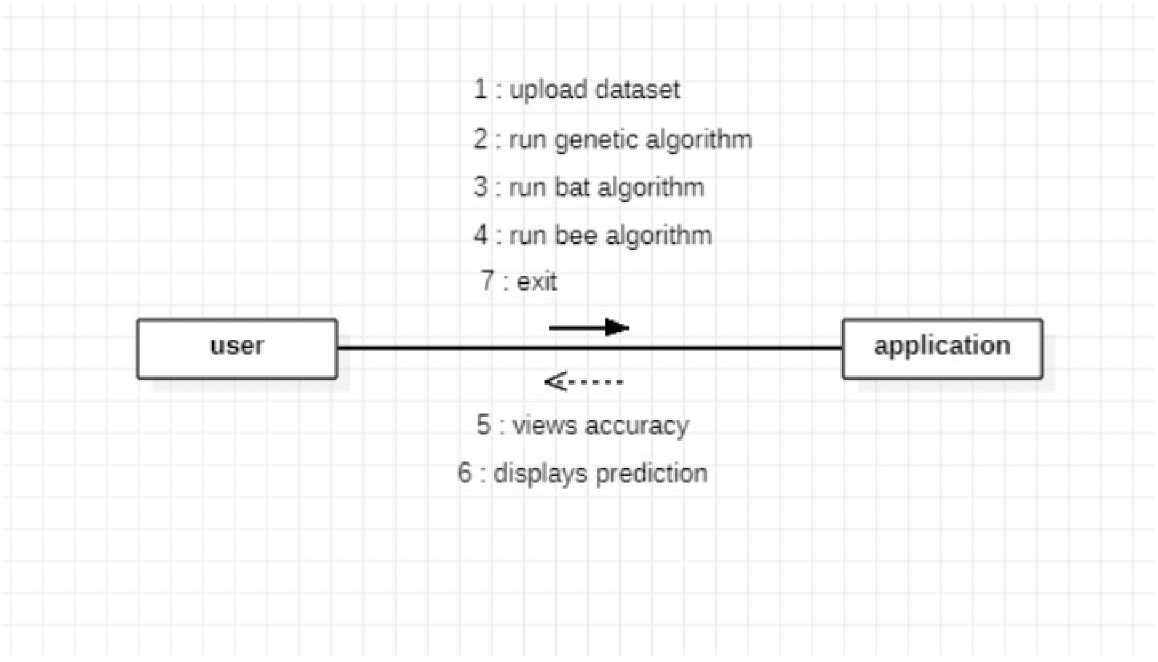


Figure 5.6: Communication Diagram

Chapter 6

SOURCE CODE

main.py

```
from __future__ import print_function
from tkinter import messagebox
from tkinter import *
from tkinter import simpledialog
import tkinter
from tkinter import filedialog
import matplotlib.pyplot as plt
from tkinter.filedialog import askopenfilename
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
import os
import re
from sklearn.metrics import accuracy_score
import numpy as np
```

```
from sklearn import datasets, linear_model
import pandas as pd
from genetic_selection import GeneticSelectionCV
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
import SwarmPackagePy
from sklearn import svm
from sklearn.ensemble import RandomForestClassifier
from BAT import BAT
from SwarmPackagePy import testFunctions as tf
from BEE import BEE

main = tkinter.Tk()
main.title(”Heart Disease Prediction System”)
main.geometry(”1300x1200”)

global filename
global train
global ga_acc, bat_acc, bee_acc
global classifier

def upload():
    global filename
    filename =
    filedialog.askopenfilename(
    initialdir="heart_dataset")
    pathlabel.config(text=filename)
    text.delete('1.0', END)
    text.insert(END,filename+" loaded\n");

def prediction(X_test, cls):  #prediction done here
    y_pred = cls.predict(X_test)
    for i in range(len(X_test)):
      print("X=%s, Predicted=%s" % (X_test[i], y_pred[i]))
    return y_pred

# Function to calculate accuracy
def cal_accuracy(y_test, y_pred, details):
    cm = confusion_matrix(y_test, y_pred)
    accuracy = accuracy_score(y_test,y_pred)*100
    text.insert(END,details+"\n\n")
    text.insert(END,"Accuracy : "+str(accuracy)+"\n\n")
    text.insert(END,"Report : "
    +str(classification_report(y_test, y_pred))+"\n")
    text.insert(END,"Confusion Matrix : "+str(cm)+"\n\n")
    return accuracy

def geneticAlgorithm():
    global classifier
    text.delete('1.0', END)
```

```
global ga_acc
train = pd.read_csv(filename)
test = pd.read_csv('heart_dataset/test.txt')
test_X = test.values[:, 0:12]
X = train.values[:, 0:12]
y = train.values[:, 13]

estimator = linear_model.LogisticRegression(
solver="liblinear", multi_class="ovr")

selector = GeneticSelectionCV(estimator,
                              cv=5,
                              verbose=1,
                              scoring="accuracy",
                              max_features=10,
                              n_population=50,
                              crossover_proba=0.5,
                              mutation_proba=0.2,
                              n_generations=200,
                              crossover_independent
                              _proba=0.5,
                              mutation_independent
                              _proba=0.05,
                              tournament_size=3,
                              n_gen_no_change=10,
                                    caching=True,
                                    n_jobs=-1)
    selector = selector.fit(X, y)
    y_pred = selector.predict(test_X)
    prediction_data = prediction(test_X, selector)
    ga_acc = cal_accuracy(prediction_data,
    prediction_data,'GA Algorithm Accuracy,
    Classification Report & Confusion Matrix')
    classifier = selector

def runBat():
    text.delete('1.0', END)
    global bat_acc
    train = pd.read_csv(filename)
    alh = BAT(train.values, tf.easom_function,
    -10, 10, 2, 20)
    data = alh.get_agents()
    X = []
    Y = []
    for i in range(len(data)):
        for j in range(len(data[i])):
            X.append(data[i][j][0:13])
            Y.append(data[i][j][13])

    X_train, X_test, y_train, y_test =
    train_test_split(X, Y,
    test_size = 0.1, random_state = 0)
    cls = RandomForestClassifier(n_estimators=50
    ,max_depth=2,
    random_state=0,class_weight='balanced')
    cls.fit(X_train, y_train)
    prediction_data = prediction(X_test, cls)
    bat_acc = cal_accuracy(y_test, prediction_data,
    'BAT Algorithm Accuracy,
    Classification Report & Confusion Matrix')

def runBee():
    text.delete('1.0', END)
    global bee_acc
    train = pd.read_csv(filename)
    alh = BEE(train.values, tf.easom_function,
    -10, 10, 2, 20)
    data = alh.get_agents()
    X = []
    Y = []
    for i in range(len(data)):
        for j in range(len(data[i])):
            X.append(data[i][j][0:13])
            Y.append(data[i][j][13])
    X_train , X_test , y_train , y_test =
    train_test_split(X, Y,
    test_size = 0.1 , random_state = 0)
    cls = RandomForestClassifier(n_estimators=30,
    max_depth=2,
    random_state=0,class_weight='balanced ')
    cls.fit(X_train , y_train)
    prediction_data = prediction(X_test , cls)
    bee_acc = cal_accuracy(y_test , prediction_data ,
    'ABE Algorithm Accuracy ,
    Classification Report & Confusion Matrix ')

def predict():
    text.delete('1.0', END)
    filename = filedialog.askopenfilename(
    initialdir="dataset")
    test = pd.read_csv(filename)
    test = test.values[:, 0:12]
    total = len(test)
    text.insert(END,filename+" test file loaded\n");
    y_pred = classifier.predict(test)
    for i in range(len(test)):
        print(str(y_pred[i]))
        if str(y_pred[i]) == '0.0':
            text.insert(END,"X=%s ,
                Predicted = %s" % (test[i],
                'No disease detected')+"\n\n")
            if str(y_pred[i]) == '1.0':
               text.insert(END,"X=%s,
               Predicted = %s" % (test[i],
               'Stage 1 Disease Detected and should Take
               necessary precautions')+"\n\n")
            if str(y_pred[i]) == '2.0':
               text.insert(END,"X=%s,
               Predicted = %s" % (test[i],
               'Stage 2 Disease Detected ,
               should  Take necessary precautions and
               maintain a healthy diet')+"\n\n")
            if str(y_pred[i]) == '3.0':
               text.insert(END,"X=%s,
               Predicted = %s" % (test[i],
               'Stage 3 Disease Detected ,
               should consult a Doctor ')+"\n\n")
            if str(y_pred[i]) == '4.0':
               text.insert(END,"X=%s,
               Predicted = %s" % (test[i],
               'Stage 4 Disease Detected ,
               should consult a Doctor immediately')+"\n\n")
def graph():
    height = [ga_acc,bat_acc,bee_acc]
    bars = ('Genetic Algorithm','Bat Algorithm',
    'Bee Algorithm')
    y_pos = np.arange(len(bars))
    plt.bar(y_pos, height)
    plt.xticks(y_pos, bars)
    plt.show()

def exit():
    main.destroy()

font = ('times', 16, 'bold')
title = Label(main, text='Heart Disease Prediction
System')
title.config(bg='purple', fg='white')
title.config(font=font)
title.config(height=3, width=120)
title.place(x=0,y=5)

font1 = ('times', 14, 'bold')
uploadButton = Button(main, text="Upload Heart Disease",
command=upload)
uploadButton.place(x=50,y=100)
uploadButton.config(font=font1)

pathlabel = Label(main)
```

```
pathlabel.config(bg='white', fg='white')
pathlabel.config(font=font1)
pathlabel.place(x=460,y=100)

geneticButton = Button(main,
text="Run Genetic Algorithm", command=geneticAlgorithm)
geneticButton.place(x=50,y=150)
geneticButton.config(font=font1)

batButton = Button(main, text="Run BAT Algorithm",
command=runBat)
batButton.place(x=330,y=150)
batButton.config(font=font1)

beeButton = Button(main, text="Run BEE Algorithm",
command=runBee)
beeButton.place(x=620,y=150)
beeButton.config(font=font1)

predictButton = Button(main,
text="Upload & Predict Test Data", command=predict)
predictButton.place(x=850,y=150)
predictButton.config(font=font1)

graphButton = Button(main, text="Accuracy Graph",
command=graph)
graphButton.place(x=50,y=200)
graphButton.config(font=font1)

exitButton = Button(main, text="Exit",
command=exit)
exitButton.place(x=330,y=200)
exitButton.config(font=font1)

font1 = ('times', 12, 'bold')
text=Text(main,height=20,width=150)
scroll=Scrollbar(text)
text.configure(yscrollcommand=scroll.set)
text.place(x=10,y=250)
text.config(font=font1)

#main.config(bg='pink')
main.mainloop()
```

Chapter 7

TESTING

Testing is the process of executing a program with the aim of finding errors. To make our software perform well it should be error-free. If testing is done successfully it will remove all the errors from the software.

7.1 Principles Of Testing

- All the test should meet the customer requirements
- To make our software testing should be performed by a third party
- Exhaustive testing is not possible. As we need the optimal amount of testing based on the risk assessment of the application.
- All the test to be conducted should be planned before implementing it.
- It follows the Pareto rule(80/20 rule) which states that 80% of errors come from 20% of program components.
- Start testing with small parts and extend it to large parts.

7.2 Software Testing Life Cycle

The procedure of software testing is also known as STLC (Software Testing Life Cycle) which includes phases of the testing process.The testing process is executed in a well-planned and systematic manner. All activities are done to improve the quality of the software product.

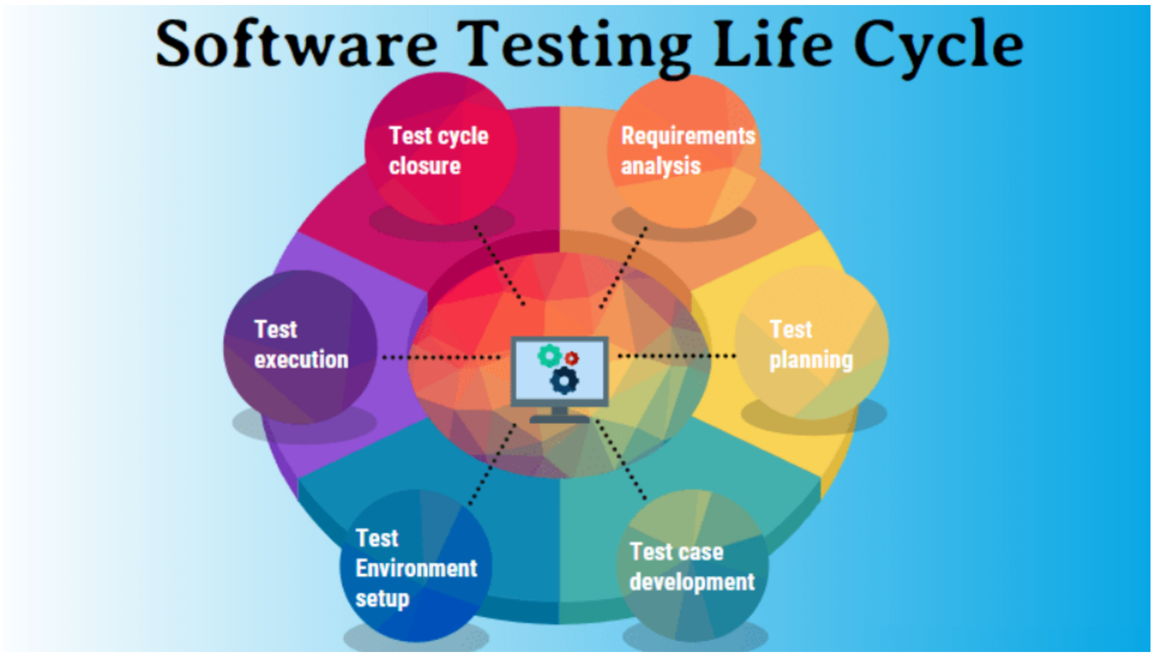


Figure 7.1: Software Testing Life Cycle

7.2.1 Requirement Analysis

The first step of the manual testing procedure is requirement analysis. In this phase, tester analyses requirement document of SDLC (Software Development Life Cycle) to examine requirements stated by the client. After examining the requirements, the tester makes a test plan to check whether the software is meeting the requirements or not.

7.2.2 Test Planning

Test plan creation is the crucial phase of STLC where all the testing strategies are defined. Tester determines the estimated effort and cost of the entire project. This phase takes place after the successful completion of the Requirement Analysis Phase. Testing strategy and effort estimation documents provided by this phase. Test case execution can be started after the successful completion of Test Plan Creation

7.2.3 Test Case Development

The Test Case Development Phase involves the creation, verification and rework of test cases test scripts after the test plan is ready. Initially, the Test data is identified then created and reviewed and then reworked based on the preconditions. Then the QA team starts the development process of test cases for individual units.

7.2.4 Test Environment Setup

Setup of the test environment is an independent activity and can be started along with Test Case Development. This is an essential part of the manual testing procedure as without environment testing is not possible. Environment setup requires a group of essential software and hardware to create a test environment. The testing team is not involved in setting up the testing environment, its senior developers who create it.

7.2.5 Test Execution

Test case Execution takes place after the successful completion of test planning. In this phase, the testing team starts case development and execution activity. The testing team writes down the detailed test cases, also prepares the test data if required. The prepared test cases are reviewed by peer members of the team or Quality Assurance leader.

7.2.6 Test Cycle Closure

The test cycle closure report includes all the documentation related to software design, development, testing results, and defect reports.
This phase evaluates the strategy of development, testing procedure, possible defects in order to use these practices in the future if there is a software with the same specification.

7.3 Types Of Testing

7.3.1 Unit Testing

It focuses on the smallest unit of software design. In this, we test an individual unit or group of interrelated units. It is often done by the programmer by using sample input and observing its corresponding outputs.
Example:
a) In a program we are checking if loop, method or function is working fine.
b) Misunderstood or incorrect, arithmetic precedence.

c) Incorrect initialization.

7.3.2 Integration Testing

The objective is to take unit tested components and build a program structure that has been dictated by design. Integration testing is testing in which a group of components is combined to produce output. **Integration testing is of four types:**

(i) Top-down

(ii) Bottom-up

(iii) Sandwich

(iv) Big-Bang

Example:

(a) Black Box testing:- It is used for validation. In this we ignore internal working mechanism and focuse on **What is the output?**.

(b) White Box testing:- It is used for verification. In this we focus on internal mechanism i.e. **How the output is achieved?**

7.3.3 Regression Testing

Every time a new module is added leads to changes in the program. This type of testing makes sure that the whole component works properly even after adding components to the complete program.

Example:

In school record suppose we have module staff, students and finance combining these modules and checking if on integration these module works fine is regression testing

7.3.4 Smoke Testing

This test is done to make sure that software under testing is ready or stable for further testing It is called a smoke test as the testing an initial pass is done to check if it did not catch the fire or smoke in the initial switch on.

Example:

If project has 2 modules so before going to module make sure that module 1 works properly

7.3.5 Alpha Testing

This is a type of validation testing. It is a type of acceptance testing which is done before the product is released to customers. It is typically done by QA people.

Example:

When software testing is performed internally within the organization

7.3.6 Beta Testing

The beta test is conducted at one or more customer sites by the end-user of the software. This version is released for a limited number of users for testing in a real-time environment.

Example:

When software testing is performed for the limited number of people

7.3.7 System Testing

This software is tested such that it works fine for the different operating systems. It is covered under the black box testing technique. In this, we just focus on the required input and output without focusing on internal working.
Example: This include functional as well as non functional testing

7.3.8 Stress Testing

In this, we give unfavorable conditions to the system and check how they perform in those conditions. **Example:** (a) Test cases that require maximum memory or other resources are executed (b) Test cases that may cause thrashing in a virtual operating system (c) Test cases that may cause excessive disk requirement

7.3.9 Performance Testing

It is designed to test the run-time performance of software within the context of an integrated system. It is used to test the speed and effectiveness of the program. It is also called load testing. In it we check, what is the performance of the system in the given load.
Example: Checking number of processor cycles.

7.3.10 Objec-Oriented Testing

This testing is a combination of various testing techniques that help to verify and validate object-oriented software. This testing is done in the following manner:

- Testing of Requirements,
- Design and Analysis of Testing,
- Testing of Code,
- Integration testing,
- System testing,
- User Testing.

We use this Object-Oriented Testing, for discussing test plans and for executing the projects.

7.4 Testing Levels

- Tests are grouped together based on where they are added in SDLC or the by the level of detailing they contain. In general, there are four levels of testing: unit testing, integration testing, system testing, and acceptance testing. The purpose of Levels of testing is to make software testing systematic and easily identify all possible test cases at a particular level.
- There are many different testing levels which help to check behavior and performance for software testing. These testing levels are designed to recognize missing areas and reconciliation between the development life-cycle states.
  In SDLC models there are characterized phases such as

- Requirement Gathering
- Analysis
- Design
- Coding Or Execution
- Testing Deployment.

All these phases go through the process of software testing levels.

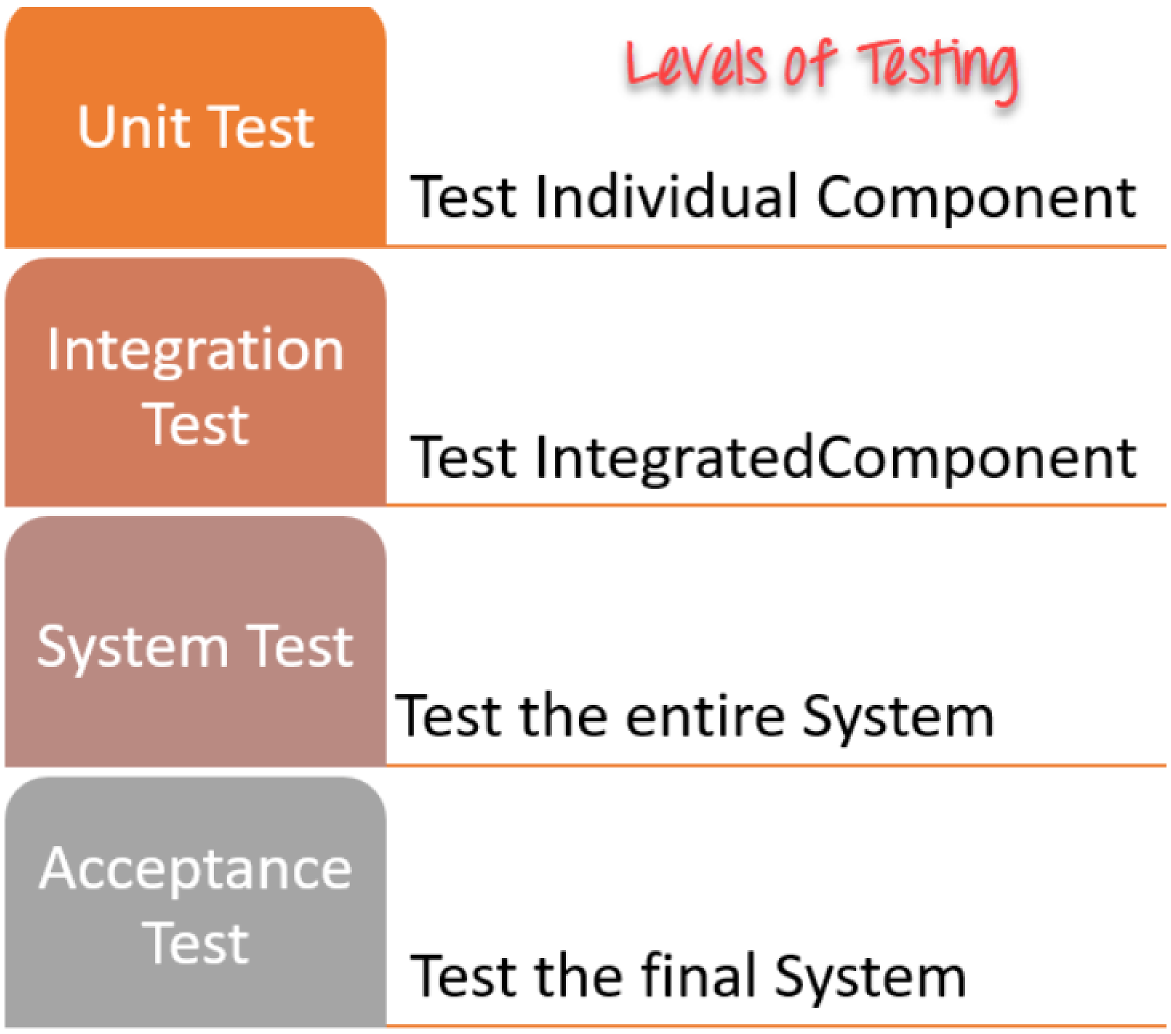


Figure 7.2: Levels Of Testing

Chapter 8

OUTPUT SCREENS

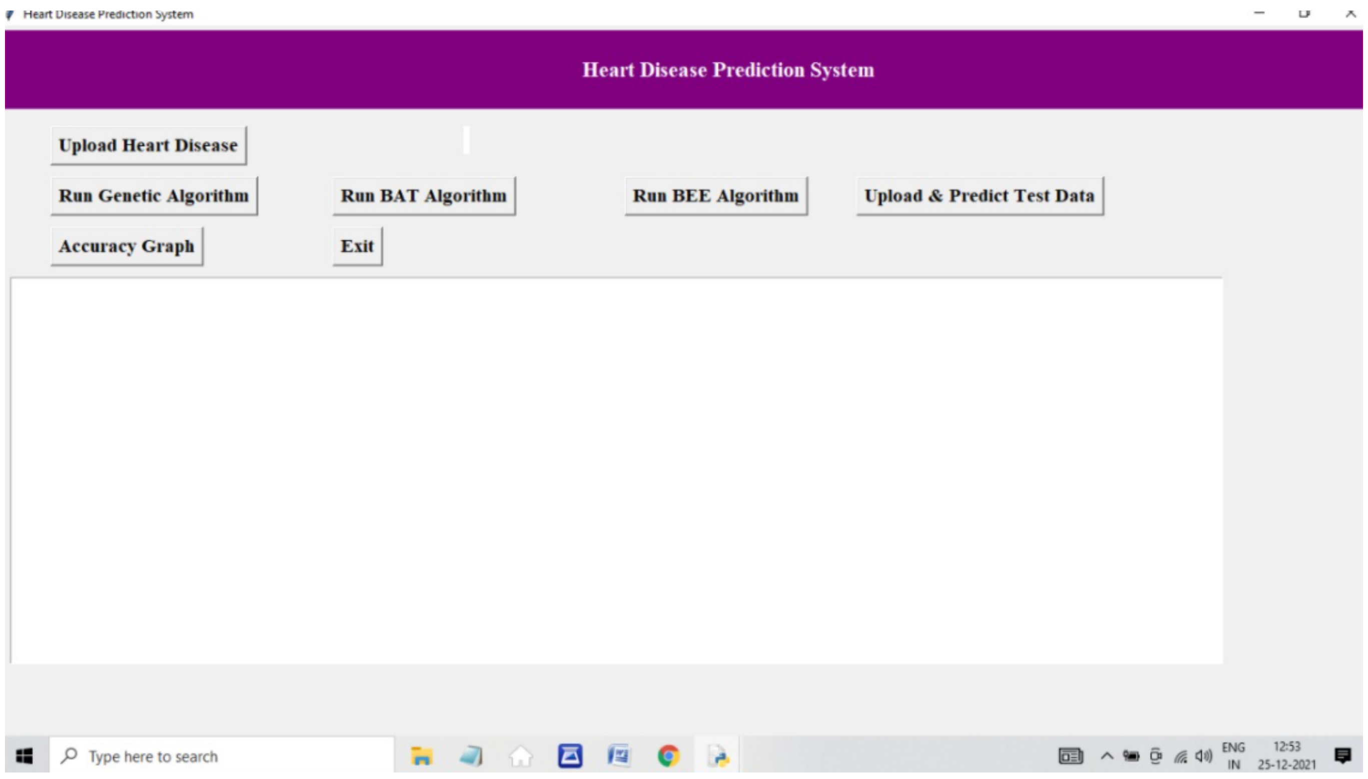
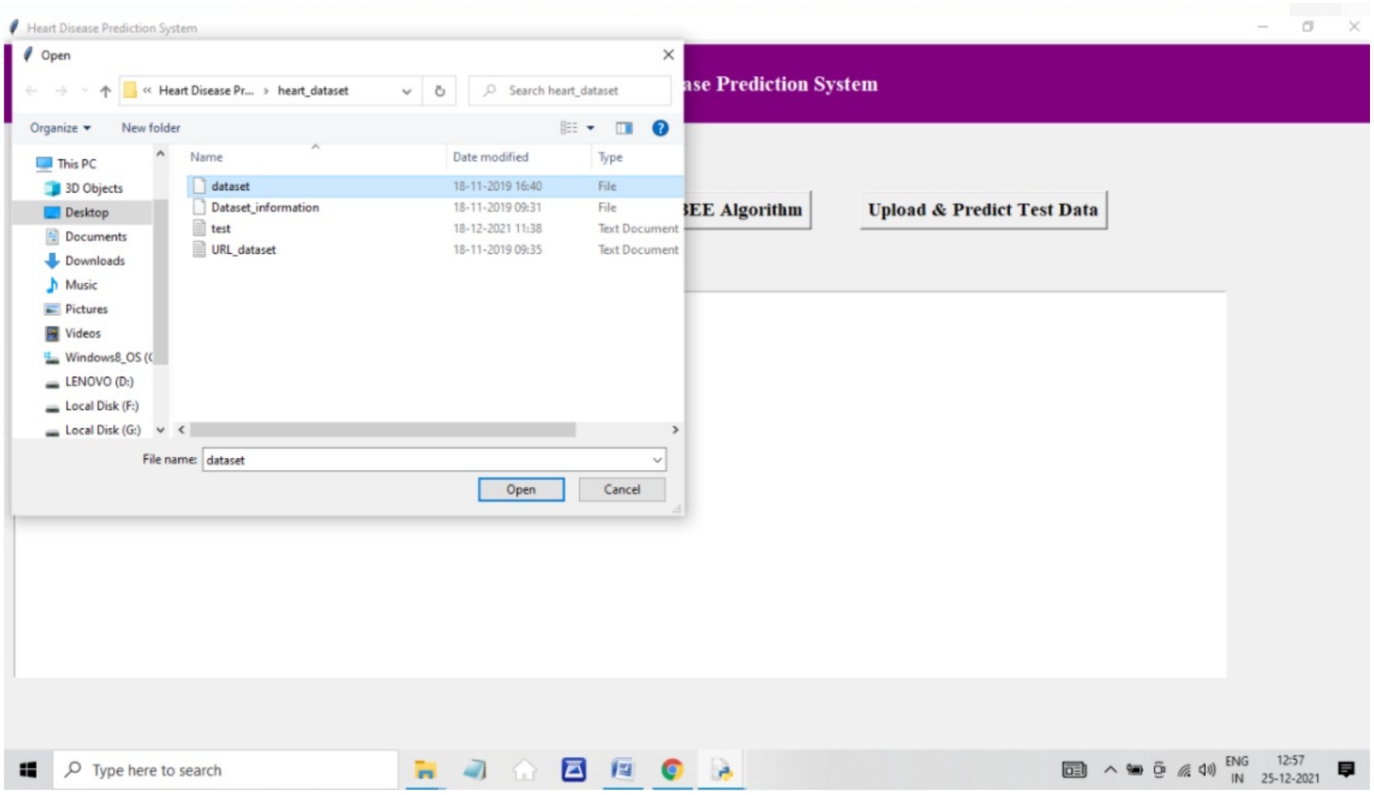


Figure 8.1: Main Screen

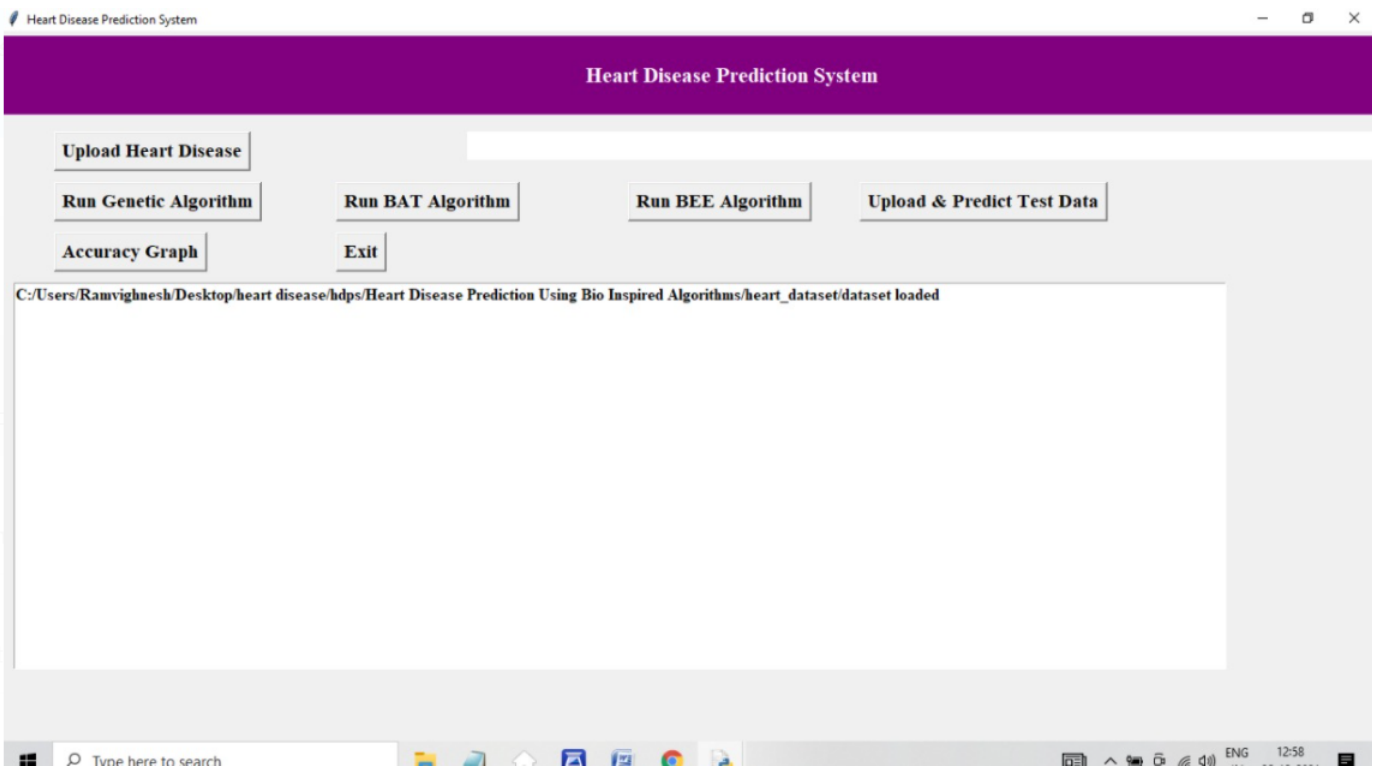
This Standalone application is opened and is ready to work.



Name	Date modified	Type
dataset	18-11-2019 16:40	File
Dataset_information	18-11-2019 09:31	File
test	18-12-2021 11:38	Text Document
URL_dataset	18-11-2019 09:35	Text Document

Figure 8.2: Uploading Training Data

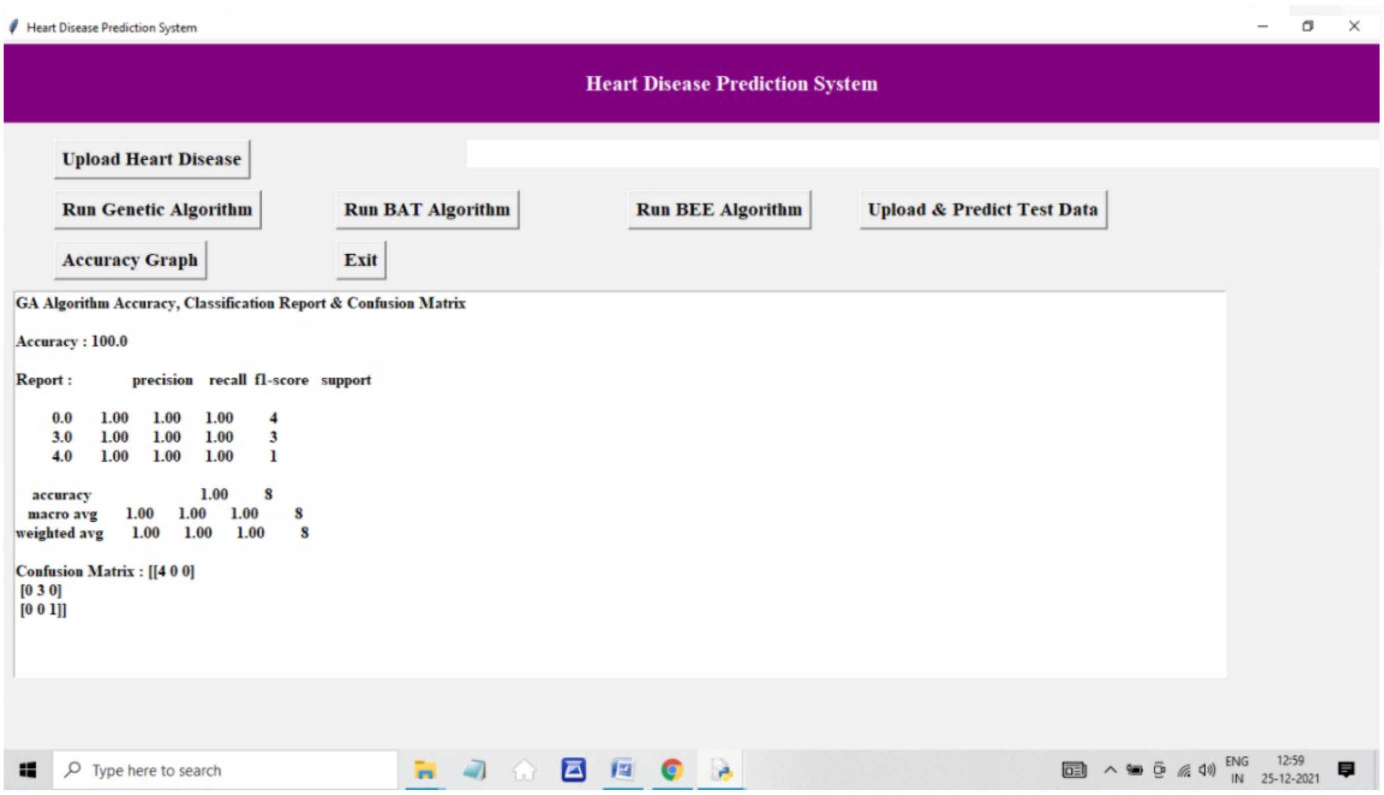
Start uploading Training Dataset



```
C:/Users/Ramvighnesh/Desktop/heart disease/hdps/Heart Disease Prediction Using Bio Inspired Algorithms/heart_dataset/dataset loaded
```

Figure 8.3: Training DataSet Loaded

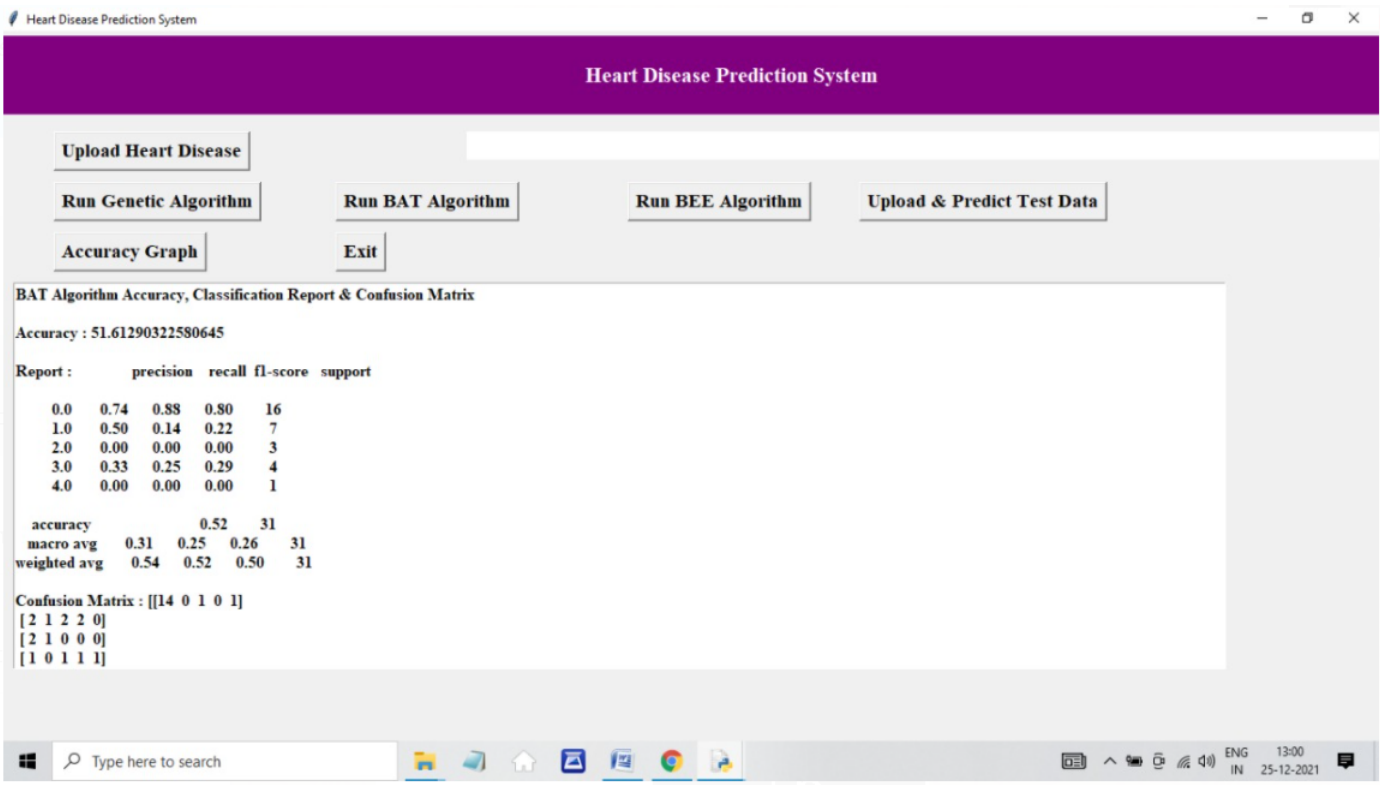
Make sure that Training DataSet is Loaded



```
GA Algorithm Accuracy, Classification Report & Confusion Matrix

Accuracy : 100.0

Report :          precision    recall  f1-score   support

         0.0       1.00      1.00      1.00         4
         3.0       1.00      1.00      1.00         3
         4.0       1.00      1.00      1.00         1

    accuracy                           1.00         8
   macro avg       1.00      1.00      1.00         8
weighted avg       1.00      1.00      1.00         8

Confusion Matrix : [[4 0 0]
 [0 3 0]
 [0 0 1]]
```

Figure 8.4: Run Genetic Algorithm

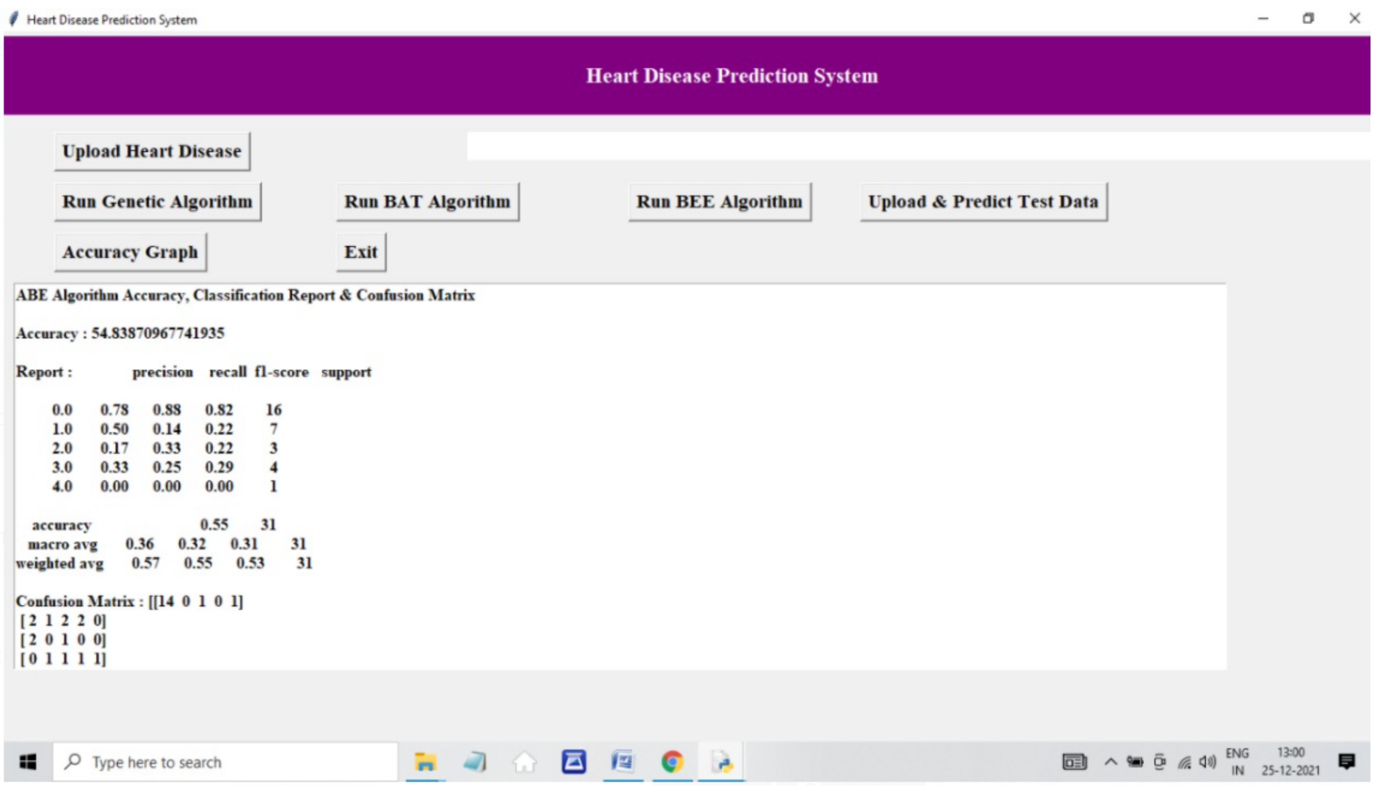
Run Genetic Algorithm and view the accuracy, classification report and confusion matrix.



```
BAT Algorithm Accuracy, Classification Report & Confusion Matrix

Accuracy : 51.61290322580645

Report :          precision   recall  f1-score   support

     0.0       0.74      0.88      0.80        16
     1.0       0.50      0.14      0.22         7
     2.0       0.00      0.00      0.00         3
     3.0       0.33      0.25      0.29         4
     4.0       0.00      0.00      0.00         1

   accuracy                        0.52        31
  macro avg    0.31      0.25      0.26        31
weighted avg   0.54      0.52      0.50        31

Confusion Matrix : [[14  0  1  0  1]
 [2  1  2  2  0]
 [2  1  0  0  0]
 [1  0  1  1  1]]
```

Figure 8.5: Run Bat Algorithm

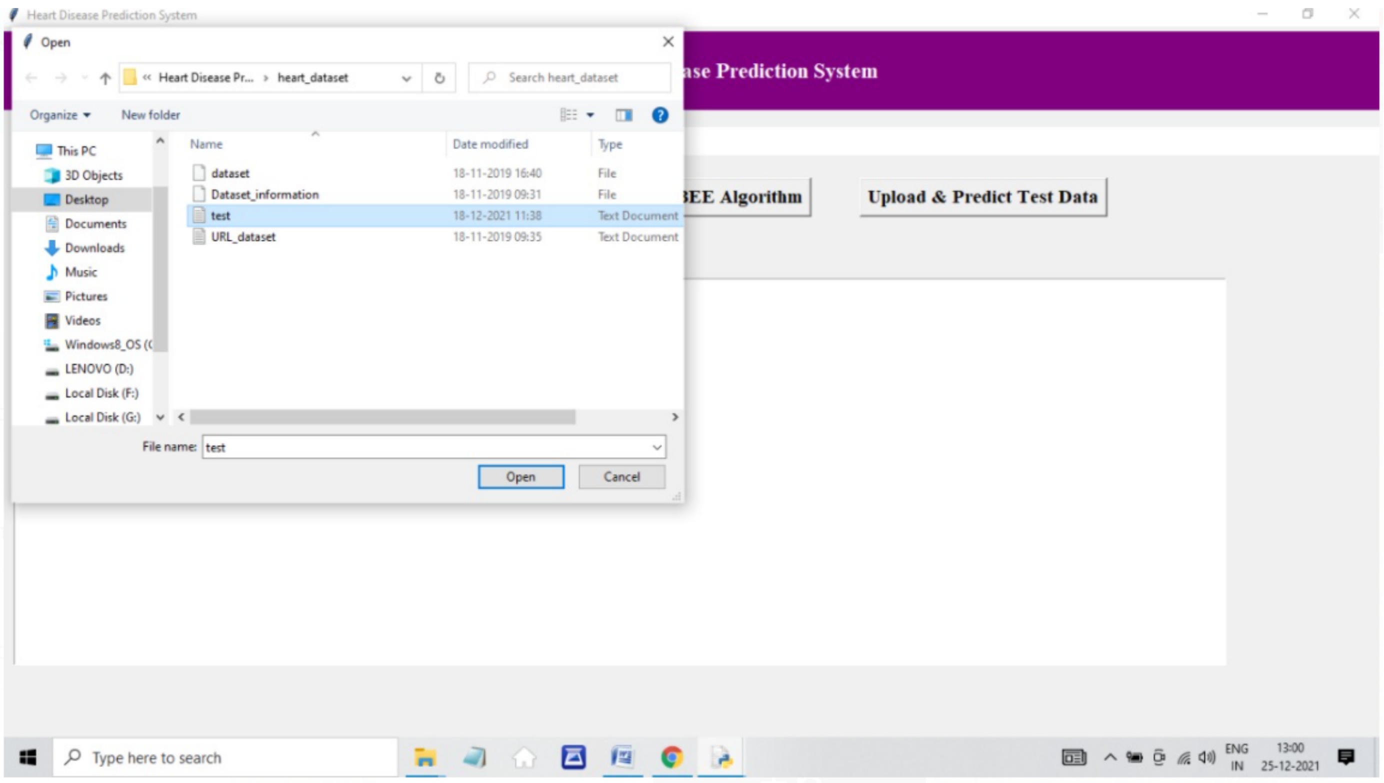
Run Bat Algorithm and view the accuracy, classification report and confusion matrix.



```
ABE Algorithm Accuracy, Classification Report & Confusion Matrix

Accuracy : 54.83870967741935

Report :          precision   recall  f1-score   support

     0.0       0.78      0.88      0.82        16
     1.0       0.50      0.14      0.22         7
     2.0       0.17      0.33      0.22         3
     3.0       0.33      0.25      0.29         4
     4.0       0.00      0.00      0.00         1

  accuracy                         0.55        31
 macro avg       0.36      0.32      0.31        31
weighted avg     0.57      0.55      0.53        31

Confusion Matrix : [[14  0  1  0  1]
 [ 2  1  2  2  0]
 [ 2  0  1  0  0]
 [ 0  1  1  1  1]]
```

Figure 8.6: Run BEE Algorithm

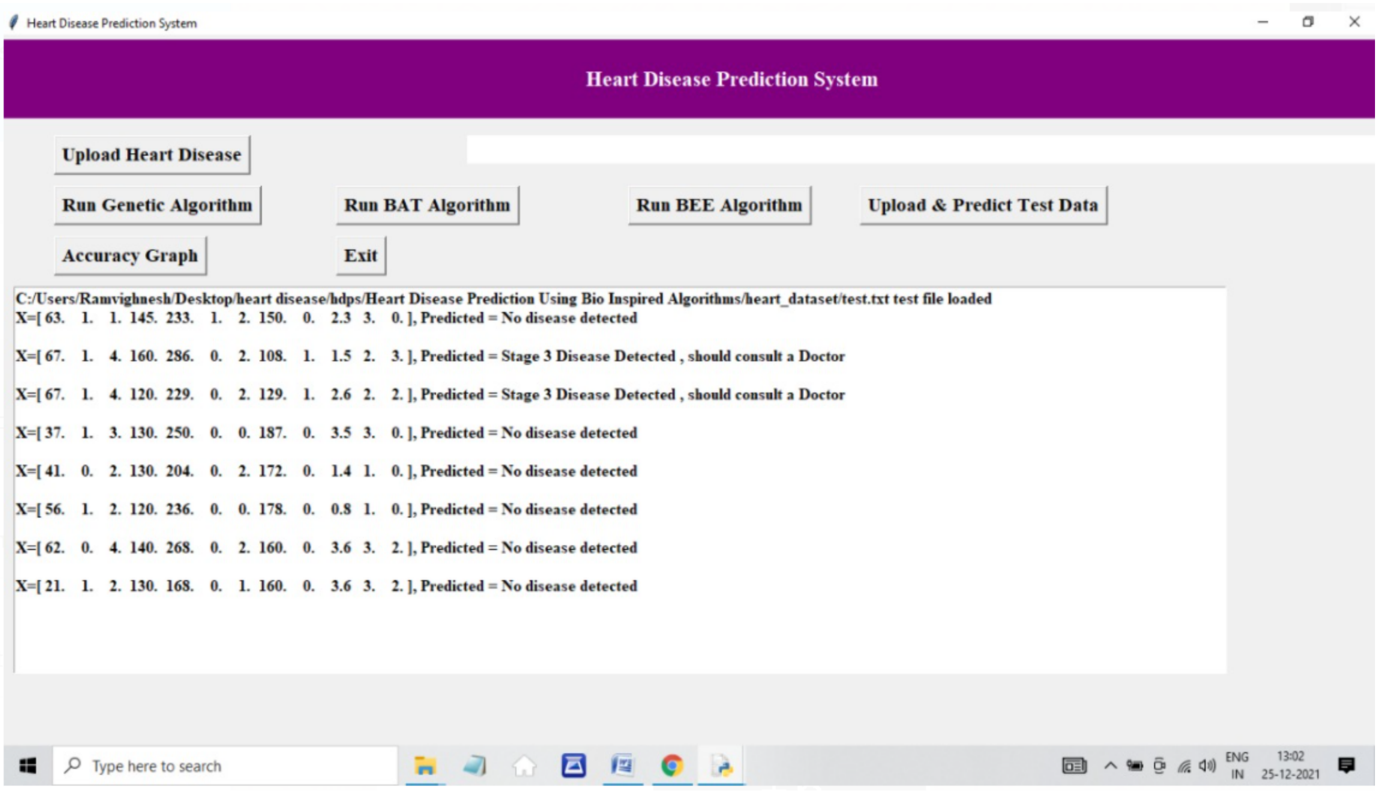
Run BEE Algorithm and view the accuracy, classification report and confusion matrix.



Name	Date modified	Type
dataset	18-11-2019 16:40	File
Dataset_information	18-11-2019 09:31	File
test	18-12-2021 11:38	Text Document
URL_dataset	18-11-2019 09:35	Text Document

Figure 8.7: Upload Test Data-Set

After running the bio-inspired algorithm, upload test dataset



```
C:/Users/Ramvighnesh/Desktop/heart disease/hdps/Heart Disease Prediction Using Bio Inspired Algorithms/heart_dataset/test.txt test file loaded
X=[ 63.   1.   1. 145. 233.   1.   2. 150.   0.   2.3  3.   0. ], Predicted = No disease detected

X=[ 67.   1.   4. 160. 286.   0.   2. 108.   1.   1.5  2.   3. ], Predicted = Stage 3 Disease Detected , should consult a Doctor

X=[ 67.   1.   4. 120. 229.   0.   2. 129.   1.   2.6  2.   2. ], Predicted = Stage 3 Disease Detected , should consult a Doctor

X=[ 37.   1.   3. 130. 250.   0.   0. 187.   0.   3.5  3.   0. ], Predicted = No disease detected

X=[ 41.   0.   2. 130. 204.   0.   2. 172.   0.   1.4  1.   0. ], Predicted = No disease detected

X=[ 56.   1.   2. 120. 236.   0.   0. 178.   0.   0.8  1.   0. ], Predicted = No disease detected

X=[ 62.   0.   4. 140. 268.   0.   2. 160.   0.   3.6  3.   2. ], Predicted = No disease detected

X=[ 21.   1.   2. 130. 168.   0.   1. 160.   0.   3.6  3.   2. ], Predicted = No disease detected
```

Figure 8.8: Disease Prediction Screen

Heart Disease is predicted along with stages of respective person from the test dataset

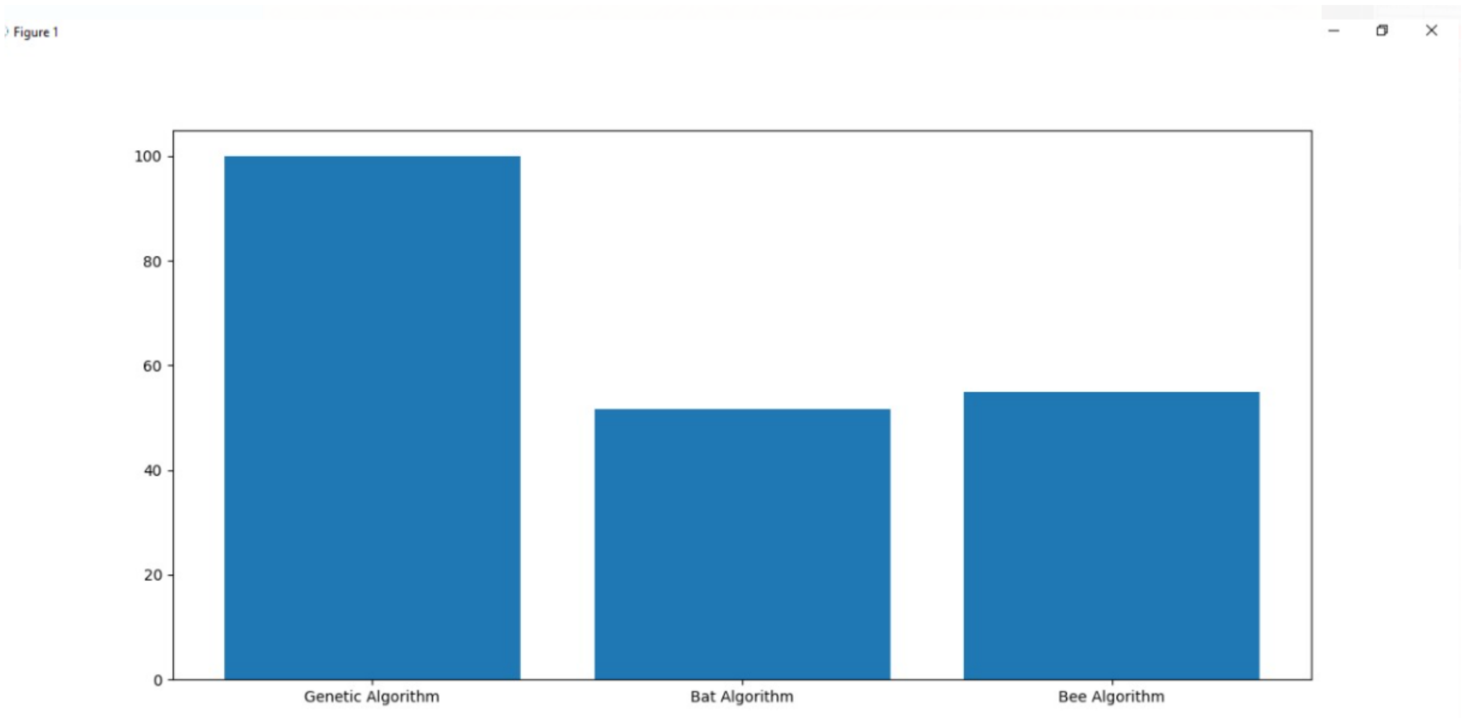


Figure 8.9: Accuracy Graph

The Accuracy graph shows that there is a large accuracy gap between Genetic, Bat and BEE bio-inspired ML Algorithms.

Chapter 9

CONCLUSION

Bio Inspired algorithms is a new paradigm that's being adopted in various educational fields. Computer Science, a very important field of education is heavily employing Bio Inspired algorithms to solve many real-world problems. Moreover, these bio inspired algorithms are quite interesting and bridge a gap between computer science, biology, artificial intelligence as well as economics. However,there is so much scope still left for bio inspired algorithms in computer Science as still very minor fraction being explored and there is still lot of room to explore these natural behaviours. We have seen different classification methods for predicting heart disease by extracting the features of respective bio inspired algorithms. This study analyses that we can achieve accuracy of heart disease prediction depends on both feature extraction of bio inspired algorithm and type of classification methods used

Chapter 10

REFERENCES

[1] *”Survey on Recent Bio-Inspired Optimization Algorithms ”*
IJCSN -International Journal of Computer Science and Network, Volume 7, Issue 6, December 2018 ISSN (Online): 2277-5420

[2] *”Artificial bee colony algorithm. ”*
cholarpedia, vol.5, no. 3, 6915, 2010

[3] *”Intelligent heart disease prediction system using CANFIS and genetic algorithm. ”*
International Journal of Biological, Biomedical andMedical Sciences, vol.3, no. 3, pp. 157-160, 2008

[4] *”Efficient Optimal Algorithm of Task Scheduling in Cloud Computing Environment International Journal of Computer Trends and Technology”*
(IJCTT)-Volume 9 Number 7 , by Dr. Amit Agarwal, Saloni Jain,March 2014

[5] ***"A Survey of Bio inspired Optimization Algorithms"***

International Journal ofSoft Computing and Engineering (IJSCE - : ISSN: 2231-2307, Volume-2, Issue-2

[6] ***"Implementation of Some Bio-Inspired Algorithms in Prediction of Heart Disease "***

doi: 10.1016/j.eswa.2016.04.0

[7] ***" Genetic Programming: on the programming of Computers by means of natural selection"***

by Koza, J.R. 1992 – MIT Press

[8] ***" Bio-inspired Algorithms for Engineering"***

by Alma Y. Alanis, Nancy Arana-Daniel and Carlos López-Franco

[9] ***"Machine Learning in Python: Essential Techniques for Predictive Analysis "***

by Michael Bowles