A

Mini Project

On

HEART DISEASE PREDICTION USING BIO INSPIRED ALGORITHMS

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

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By

KONDABALA BINDHU (207R1A05L7)

CHAKALI VIRUPAKSHI (207R1A05K5)

Under the Guidance of

B. POOJA

(Assistant Professor)

DEPARTMENT OF COMPUTER SCIENCE ANDENGINEERING

CMR TECHNICAL CAMPUS

UGC AUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi) Recognized Under Section 2(f) & 12(B) of the UGCAct.1956, Kandlakoya (V), Medchal Road, Hyderabad-501401.

2020-2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project entitled **"HEART DECISION PREDICTION USING BIO INSPIRED ALGORITHM"** being submitted by **K. BINDHU (207R1A05L7) C. VIRUPAKSHI (207R1A05K5)** in partial fulfilment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by them under our guidance and supervision during the year 2023-24.

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

Mrs B. Pooja
(Assistant Professor)
INTERNAL GUIDE

Dr. A. Raji Reddy
DIRECTOR

Dr. K. Srujan Raju
HOD

EXTERNAL EXAMINER

Submitted for viva voice Examination held on____________________

ACKNOWLEDGEMENT

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

We take this opportunity to express my profound gratitude and deep regard to my guide **B. Pooja ,** Associate Professor for his exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by him shall carry us a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to the Project Review Committee (PRC) **G. Vinesh Shanker, Dr. J. Narasimharao, Ms. Shilpa, & Dr. K. Maheswari** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

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

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

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

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

KONDABALA BINDHU (207R1A05L7)

CHAKALI VIRUPAKSHI (207R1A05K5)

ABSTRACT

Heart related diseases or Cardiovascular Diseases (CVDs) are the main reason for a huge number of deaths in the world over the last many decades and has emerged as the most life- threatening disease, not only in India but also in the whole world. Prediction of cardiovascular disease is a critical challenge in the area of clinical data analysis.

Multiple experimenters, in recent times, have been using several machine learning approaches to help the health care industry and the professionals in the diagnosis of heart related diseases.

This project presents a review of various models based on like algorithms and approaches and analyses their performance. The main aim of this design is to give an effective algorithm to predict heart disease. So, at the end we compare our algorithm (Genetic algorithm) with BAT and BEE algorithms.

LIST OF FIGURES/TABLES

LIST OF SCREENSHOTS

TABLE OF CONTENTS

1. INTRODUCTION

1. INTRODUCTION

1.1 PROJECT SCOPE

This project will involve collecting and preprocessing medical data, selecting and implementing bio-inspired algorithms, building and fine-tuning predictive models, and evaluating their performance using appropriate metrics. The project will also consider ethical and privacy concerns, and if feasible, deploy the model for real-world use in healthcare settings. The ultimate goal is to create a reliable and interpretable tool for predicting heart disease risk, with potential for future enhancements and collaborations with healthcare professionals.

1.2 PROJECT PURPOSE

The purpose of the project "Heart Disease Prediction Using Bio-Inspired Algorithms" is to leverage advanced computational techniques inspired by biological processes to develop an accurate and efficient predictive model for heart disease. By analyzing diverse medical data and implementing bio-inspired algorithms, the project aims to assist healthcare professionals in early detection and risk assessment of heart diseases, ultimately improving patient care and outcomes.

1.3 PROJECT FEATURES

The features like comprehensive data collection and integration of medical information, rigorous data preprocessing to ensure data quality, the implementation of bio-inspired algorithms such as Genetic Algorithms or Artificial Neural Networks for predictive modeling, thorough model evaluation with appropriate metrics, interpretability of model predictions for healthcare professionals, and a focus on ethical considerations, privacy, and compliance with healthcare regulations throughout the project's development and deployment phases.

2. SYSTEM ANALYSIS

2. SYSTEM ANALYSIS

SYSTEM ANALYSIS

System analysis for "Heart Disease Prediction Using Bio-Inspired Algorithms" involves determining the project's requirements, defining the types of heart diseases to predict, and assessing the available medical data sources. It also entails understanding the computational resources needed for data preprocessing, algorithm implementation, and model evaluation. Collaborating with healthcare experts helps ensure the system aligns with clinical needs. Additionally, ethical considerations, data privacy, and regulatory compliance are crucial aspects to address during the analysis to create a robust and responsible predictive system.

2.1 PROBLEM DEFINITION

The project's definition is to create a system using bio-inspired algorithms to predict heart diseases accurately. It involves collecting medical data, choosing the types of heart diseases to predict, and implementing algorithms like Artificial Neural Networks. The goal is to help doctors detect heart issues early and improve patient care while addressing ethical and privacy concerns.

2.2 EXISTING SYSTEM

The existing system for predicting heart disease is based on traditional methods, such as statistical analysis and medical imaging. These methods are limited in their accuracy and can be time consuming. Furthermore, they are not able to account for the dynamic nature of the human heart . The existing system also relies on manual data entry, which is prone to errors and can be difficult to maintain. Additionally, the data used in the existing system is often incomplete or out of date, making it difficult to make accurate prediction.

2.2.1 LIMITATIONS OF THE EXISTING SYSTEM

- Limited Accuracy
- Inability to Capture Complex Patterns
- Resource Intensive
- Dependency on Simplistic Models
- Less Adaptability

2.3 PROPOSED SYSTEM

The proposed system for predicting heart disease uses bioinspired algorithms. These algorithms are designed to mimic the behavior of the heart, and can be used to make more accurate predictions of heart disease. The proposed system also uses machine learning techniques to improve the accuracy of the predictions . The proposed system is also faster than the existing system, as it does not require manual data entry and can process large amounts of data quickly. Additionally, the proposed system can account for the dynamic nature of the human heart, which the existing system is not able to do.

2.3.1 ADVANTAGES OFTHE PROPOSED SYSTEM

- The proposed system has several advantages over the existing system. It is more accurate, as it is able to account for the dynamic nature of the heart.
- Additionally, it is faster, as it does not require manual data entry. Furthermore, it is more efficient, as it can process large amounts of data quickly .
- The proposed system also has the potential to reduce the cost of predicting heart disease, as it does not require expensive medical imaging equipment.
- Additionally, it is more reliable, as it is not prone to errors due to manual data entry.

2.4 FEASIBILITY STUDY

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

- ECONOMIC FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

2.4.1 ECONOMIC FEASIBILITY

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

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

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

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

2.4.2 TECHNICAL FEASIBILITY

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

2.4.3 SOCIAL FEASIBILITY

Firstly, gaining the acceptance and endorsement of healthcare professionals is paramount, as they are the end-users and must see value in the predictive model as a tool to improve patient care. Secondly, addressing ethical considerations, particularly regarding patient data privacy and ensuring fairness in predictions, is essential to earn trust from both healthcare providers and patients. Building robust data security measures and transparent practices can contribute significantly to social acceptability. Lastly, ensuring accessibility and affordability of the system to various healthcare settings and regions is vital to ensure that the benefits of the project can reach a broader spectrum of society, regardless of their economic or geographical circumstances.

2.5 HARDWARE & SOFTWARE REQUIREMENTS

2.5.1 HARDWARE REQUIREMENTS:

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

- Processor : Intel Dual Core I5 and above
- Hard disk : 8GB and above
- RAM : 8GB and above
- Input devices : Keyboard, mouse.

2.5.2 SOFTWARE REQUIREMENTS:

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

● Operating system : Windows 10 and above

● Languages : Python, Html, CSS

● Tools : Python IDEL3.7 version.

3. ARCHITECTURE

3.ARCHITECTURE

3.1 PROJECT ARCHITECTURE

This project architecture shows the procedure followed for classification, starting from input to final prediction.

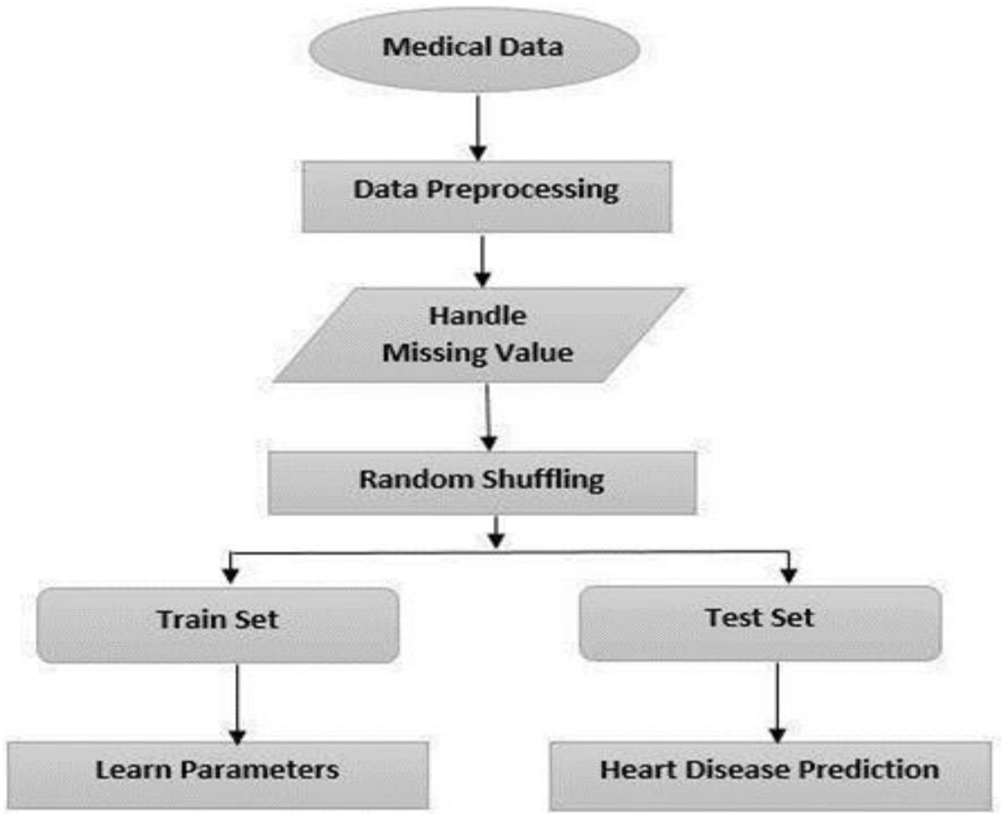


Figure 3.1: Project Architecture For Heart Disease Prediction Using Bio Inspired Algorithm

3.2 DESCRIPTION

This project predicts person with heart disease by extracting the patient medical history that leads to a fatal heart disease from a dataset that includes patients' medical history such as chest pain, sugar level, blood pressure, serum cholesterol, maximum heart rate achieved etc.

Bio-inspired optimization algorithms are those methods that are generally inspired by physical principles, evolution theory and certain behaviors of living beings to efficiently solve optimization problems in very diverse application areas. Also, we programmed an algorithm to trigger the display of the inspirational words whenever it was determined that the learner was experiencing frustration. In order to improve the prediction accuracy of classification algorithms, bio-inspired algorithms are designed to optimize the features used in datasets for training these algorithms. Since some datasets may contain irrelevant values within the dataset, optimizing algorithms can remove these features (attribute values) from the dataset. If an attribute is judged to be irrelevant to the dataset after being run through these optimized techniques, it will be eliminated.

3.3 USE CASE DIAGRAM

In the use case diagram, we have basically one actor who is the user in the trained model. A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users the system has. The use cases are represented by either circles or ellipses. The actors are often shown as stick figures.

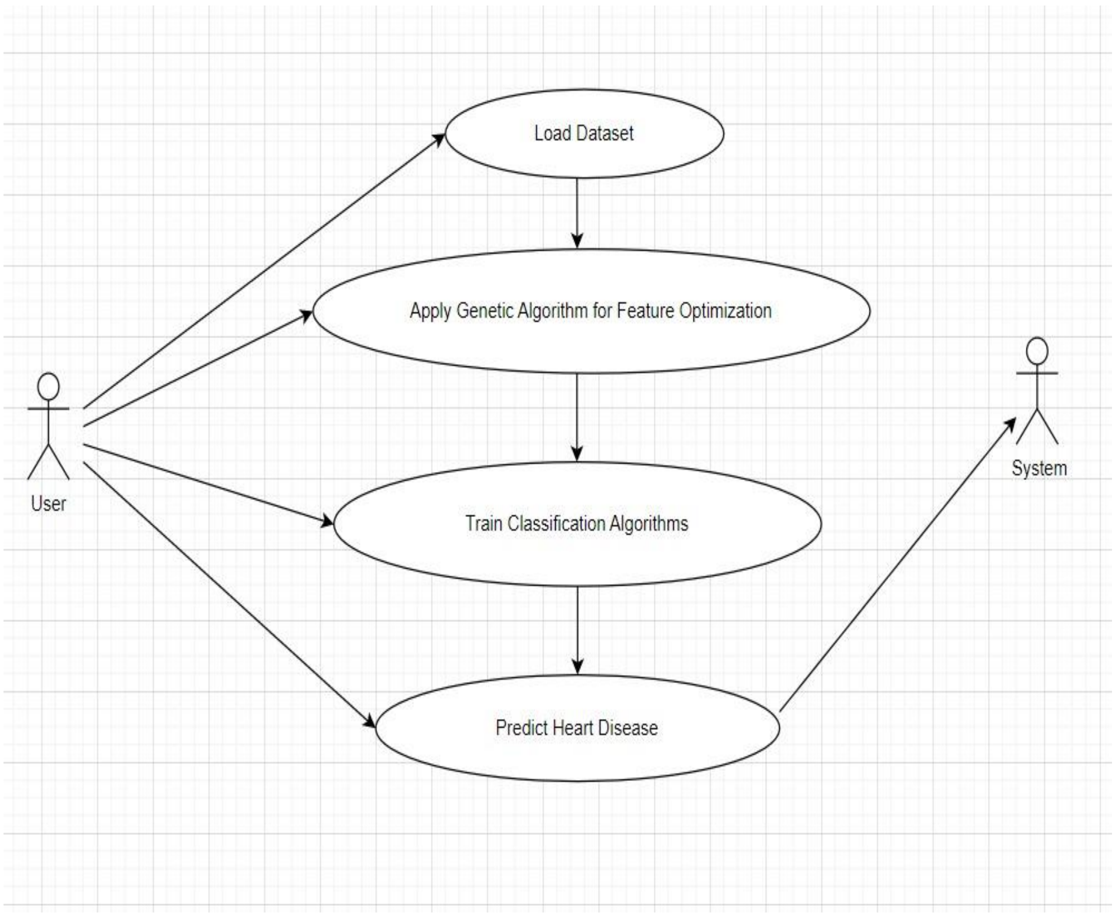

Load Dataset
Apply Genetic Algorithm for Feature Optimization
User
System
Train Classification Algorithms
Predict Heart Disease

Figure 3.2: Use Case Diagram for Heart Disease Prediction Using Bio Inspired Algorithm

3.4 CLASS DIAGRAM

Class diagram is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations(or methods), and the relationships among objects.

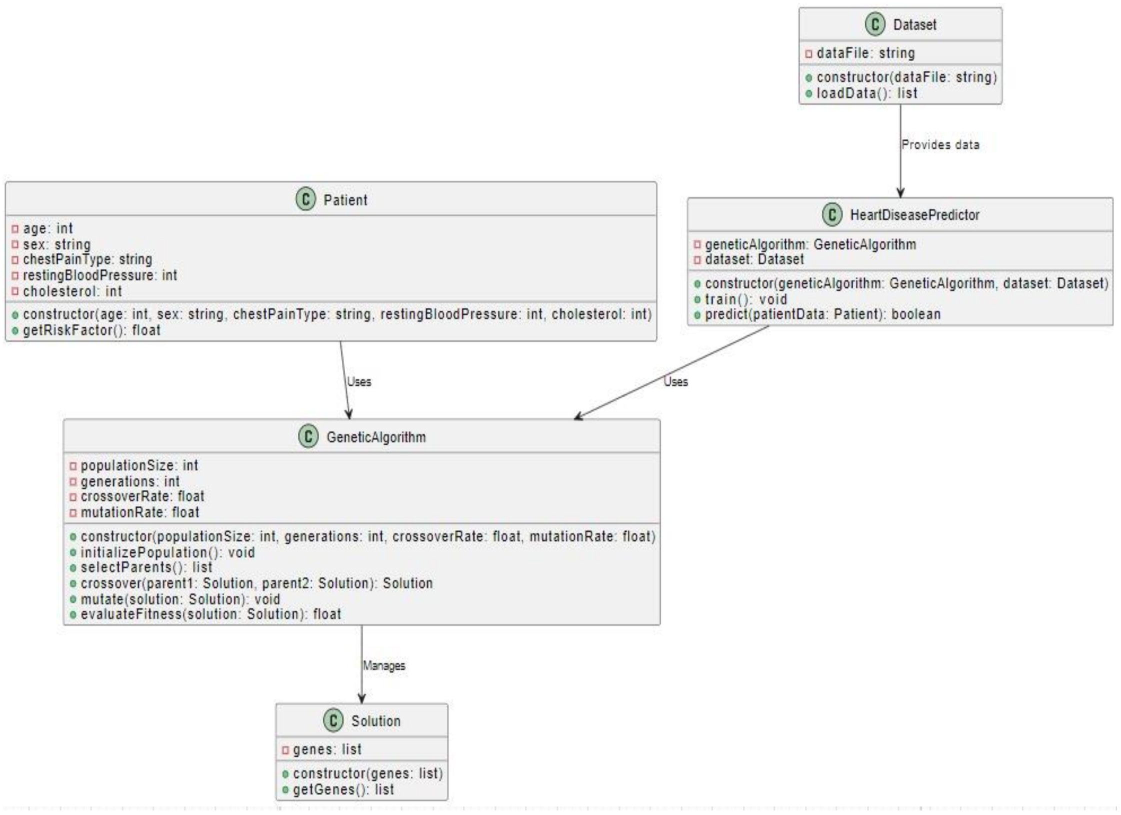

Dataset
dataFile: string
constructor(dataFile: string)
loadData(): list
Provides data
HeartDiseasePredictor
geneticAlgorithm: GeneticAlgorithm
dataset: Dataset
constructor(geneticAlgorithm: GeneticAlgorithm, dataset: Dataset)
train(): void
predict(patientData: Patient): boolean
Patient
age: int
sex: string
chestPainType: string
restingBloodPressure: int
cholesterol: int
constructor(age: int, sex: string, chestPainType: string, restingBloodPressure: int, cholesterol: int)
getRiskFactor(): float
Uses
Uses
GeneticAlgorithm
populationSize: int
generations: int
crossoverRate: float
mutationRate: float
constructor(populationSize: int, generations: int, crossoverRate: float, mutationRate: float)
initializePopulation(): void
selectParents(): list
crossover(parent1: Solution, parent2: Solution): Solution
mutate(solution: Solution): void
evaluateFitness(solution: Solution): float
Manages
Solution
genes: list
constructor(genes: list)
getGenes(): list

Figure 3.3: Class Diagram for Heart Disease Prediction Using Bio Inspired Algorithm

3.5 SEQUENCE DIAGRAM

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the logical view of the system under development.

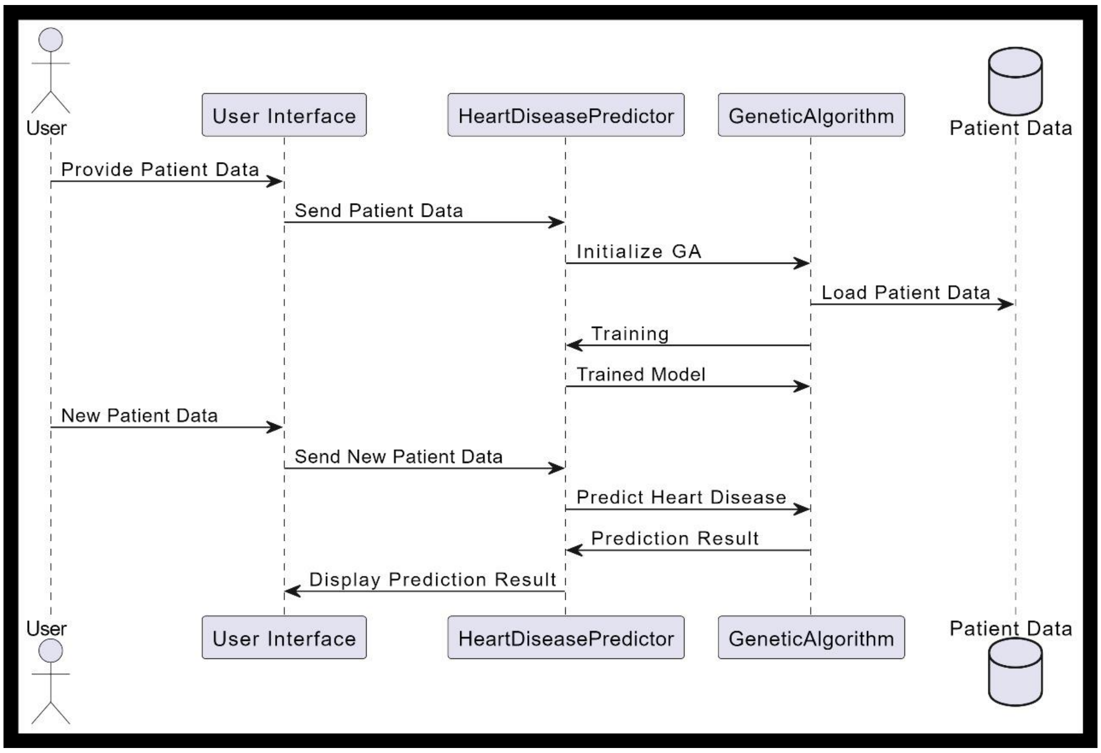


Figure 3.4: Sequence Diagram for Heart Disease Prediction Using Bio Inspired Algorithm

3.6 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. They can also include elements showing the flow of data between activities through one or more data stores.

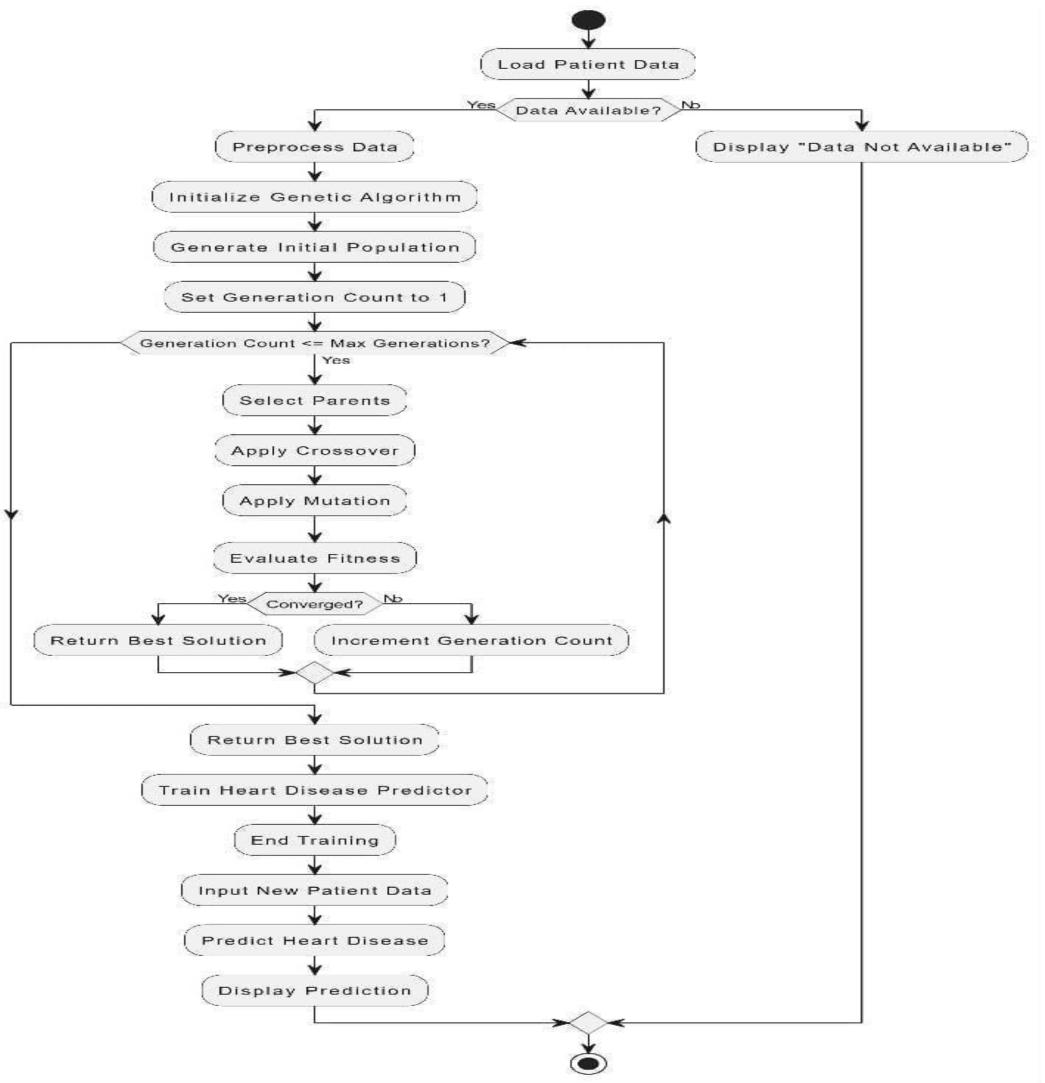


Figure 3.5: Activity Diagram for Heart Disease Prediction Using Bio Inspired Algorithm

4.IMPLEMENTATION

4.1 SAMPLE CODE

```
n_best, shortest_path):
        sorted_paths = sorted(all_paths, key=lambda x: x[1])
        for path, dist in sorted_paths[:n_best]:
            for move in path:
                self.pheromone[move] += 1.0 / self.distances[move]

    def gen_path_dist(self, path):
        total_dist = 0
        for ele in path:
            total_dist += self.distances[ele]
        return total_dist

    def gen_all_paths(self):
        all_paths = []
        for i in range(self.n_ants):
            path = self.gen_path(0)
            all_paths.append((path, self.gen_path_dist(path)))
        return all_paths

    def gen_path(self, start):
        path = []
        visited = set()
        visited.add(start)
        prev = start
        for i in range(len(self.distances) - 1):
path.append((prev, move))
            prev = move
            visited.add(move)
        path.append((prev, start)) # going back to where we started
        return path

    def pick_move(self, pheromone, dist, visited):
        pheromone = np.copy(pheromone)
        pheromone[list(visited)] = 0

        row = pheromone ** self.alpha * (( 1.0 / dist) ** self.beta)


        norm_row = row / row.sum()
        move = np_choice(self.all_inds, 1, p=norm_row)[0]
        return move
if _name_ == "_main_":
  distances = np.array([[np.inf, 2, 2, 5, 7],
                    [2, np.inf, 4, 8, 2],
                    [2, 4, np.inf, 1, 3],
                    [5, 8, 1, np.inf, 2],
                    [7, 2, 3, 2, np.inf]])

  ant_colony = ACO(distances, 1, 1, 100, 0.95, alpha=1, beta=1)
  shortest_path = ant_colony.run()
  print ("shorted_path: {}".format())
```

```
from math import exp
import numpy as np
from random import random
from SwarmPackagePy import intelligence
class BAT(intelligence.sw):
 Bat Algorithm
  def _init_(self, n, function, lb, ub, dimension, iteration, r0=0.9,
   V0=0.5, fmin=0, fmax=0.02, alpha=0.9, csi=0.9):
   :param n: number of agents
   :param function: test function
    :param lb: lower limits for plot axes
   :param ub: upper limits for plot axes
   :param dimension: space dimension
   :param iteration: number of iterations
   :param r0: level of impulse emission (default value is 0.9)
   :param V0: volume of sound (default value is 0.5)
   :param fmin: min wave frequency (default value is 0)
    :param fmax: max wave frequency (default value is 0.02)
 fmin = 0 and fmax =0.02 - the bests values
     :param alpha: constant for change a volume of sound
        (default value is 0.9)
:param csi: constant for change a level of impulse emission
 (default value is 0.9)
  super(BAT, self)._init_()
r = [r0 for i in range(len(n))]
        self.__agents = n    #np.random.uniform(lb, ub, (n, dimension))
    self._points(self.__agents)
velocity = np.zeros((len(n), dimension))
V = [V0 for i in range(len(n))]
Pbest = self.__agents[np.array([function(i)
for i in self.__agents]).argmin()]
Gbest = Pbest
f = fmin + (fmin - fmax)
for t in range(iteration):
sol = self.__agents
F = f * np.random.random((len(n), dimension))
self._set_Gbest(Gbest)

import numpy as np
from random import randint, uniform
import SwarmPackagePy
from SwarmPackagePy import intelligence
class BEE(intelligence.sw):
  """
  Artificial Bee Algorithm
  """

  def _init_(self, n, function, lb, ub, dimension, iteration):
    """
    :param n: number of agents
    :param function: test function
    :param lb: lower limits for plot axes
```

```
:param ub: upper limits for plot axes
:param dimension: space dimension
:param iteration: number of iterations
"""

super(BEE, self)._init_()

self.__function = function

self.__agents = n  #np.random.uniform(lb, ub, (n, dimension))
self._points(self.__agents)

Pbest = self.__agents[np.array([function(x) for x in self.__agents]).argmin()]
Gbest = Pbest

if len(n) <= 10:
   count = n - n // 2, 1, 1, 1
else:
   a = len(n) // 10
   b = 5
   c = (n - a * b - a) // 2
   d = 2
   count = a, b, c, d
for t in range(iteration):
   fitness = [function(x) for x in self.__agents]
```

```
            sort_fitness.sort()
            sort_fitness = np.asarray(sort_fitness)

            best = [self.__agents[i] for i in
                    [fitness.index(x) for x in sort_fitness[:count[0]]]]
            selected = [self.__agents[i]
                        for i in [fitness.index(x)
                                  for x in sort_fitness[1:5]]]

        self._set_Gbest(Gbest)

    def __new(self, l, c, lb, ub):

        bee = []
        for i in l:
            new = [self.__neighbor(i, lb, ub) for k in range(c)]
            bee += new
        bee += l

        return bee

    def __neighbor(self, who, lb, ub):

        neighbor = np.array(who) + uniform(-1, 1) * (
            np.array(who) - np.array(
                self.__agents[randint(0, len(self.__agents) - 1)]))
        neighbor = np.clip(neighbor, lb, ub)

        return list(neighbor)

    uploadButton.config(font=font1)

pathlabel = Label(main)
pathlabel.config(bg='brown', 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='brown')
main.mainloop()
```

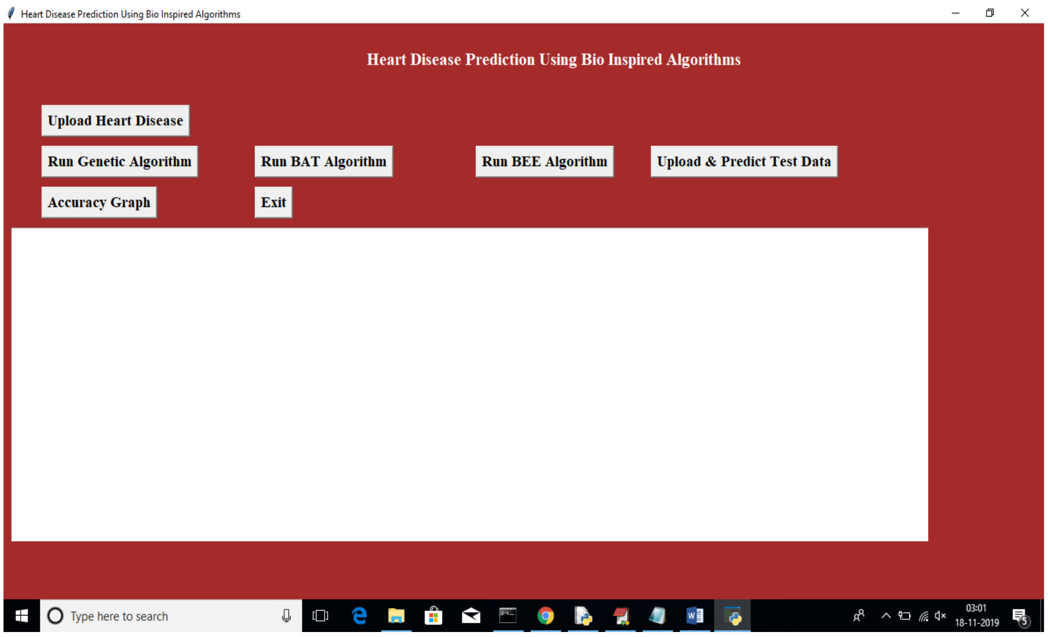
```
def spread_pheronome(self, all_paths, n_best, shortest_path):
        sorted_paths = sorted(all_paths, key=lambda x: x[1])
        for path, dist in sorted_paths[:n_best]:
            for move in path:
                self.pheromone[move] += 1.0 / self.distances[move]

    def gen_path_dist(self, path):
        total_dist = 0
        for ele in path:
            total_dist += self.distances[ele]
        return total_dist

    def gen_all_paths(self):
        all_paths = []
        for i in range(self.n_ants):
            path = self.gen_path(0)
            all_paths.append((path, self.gen_path_dist(path)))
        return all_paths

    def gen_path(self, start):
        path = []
        visited = set()
        visited.add(start)
        prev = start
        for i in range(len(self.distances) - 1):
            move = self.pick_move(self.pheromone[prev], self.distances[prev], visited)
            prev = move
            visited.add(move)
        path.append((prev, start)) # going back to where we started
        return path

    def pick_move(self, pheromone, dist, visited):
        pheromone = np.copy(pheromone)
        pheromone[list(visited)] = 0

        row = pheromone ** self.alpha * (( 1.0 / dist) ** self.beta)

        norm_row = row / row.sum()
        move = np_choice(self.all_inds, 1, p=norm_row)[0]
        return move
if _name_ == "_main_":
  distances = np.array([[np.inf, 2, 2, 5, 7],
                        [2, np.inf, 4, 8, 2],
                        [2, 4, np.inf, 1, 3],
                        [5, 8, 1, np.inf, 2],
                        [7, 2, 3, 2, np.inf]])

  ant_colony = ACO(distances, 1, 1, 100, 0.95, alpha=1, beta=1)
  shortest_path = ant_colony.run()
  print ("shorted_path: {}".format(shortest_path)
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)
```

```
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():
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='brown', 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)
```

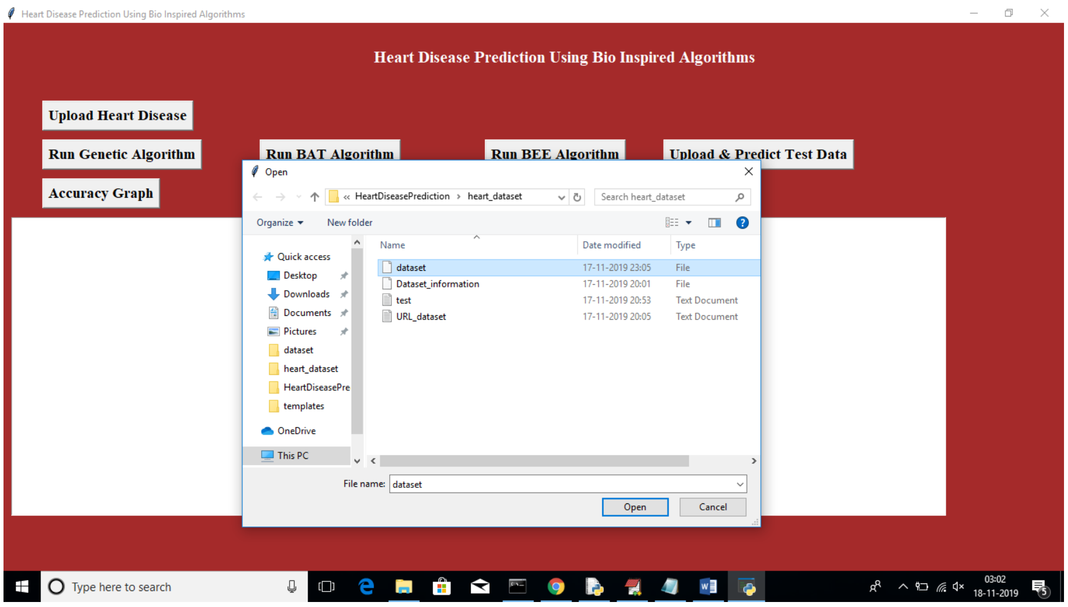
```
beeButton = Button(main, text="Run BEE Algorithm", command=runBee)
beeButton.place(x=620,y=150)
beeButton.config(font=font1)

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='brown')
main.mainloop()


>pip install scikit-learn==0.17 --user
pip install pandas --user
pip install sklearn-genetic==0.2
pip install -U scikit-learn scipy matplotlib


pip install sklearn-genetic
pip install SwarmPackagePy
```

5. SCREENSHOTS

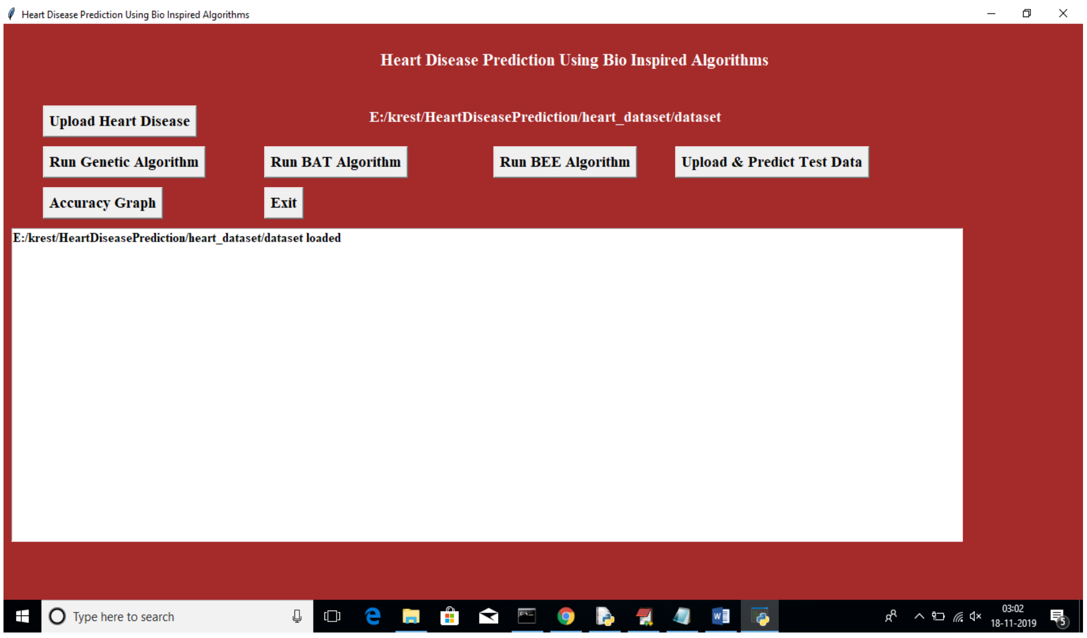

Heart Disease Prediction Using Bio Inspired Algorithms
Heart Disease Prediction Using Bio Inspired Algorithms
Upload Heart Disease
Run Genetic Algorithm
Run BAT Algorithm
Run BEE Algorithm
Upload & Predict Test Data
Accuracy Graph
Exit
Type here to search
03:01
18-11-2019

Screenshot 5.1: Upload Heart Disease



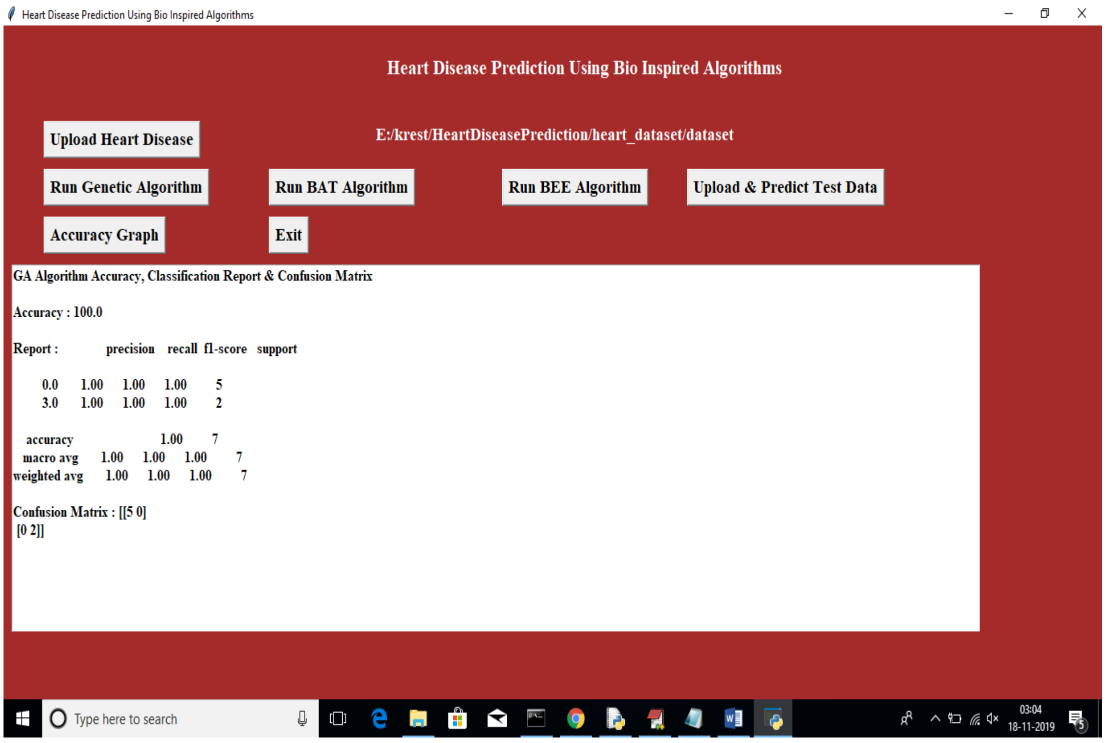
Name	Date modified	Type
dataset	17-11-2019 23:05	File
Dataset_information	17-11-2019 20:01	File
test	17-11-2019 20:53	Text Document
URL_dataset	17-11-2019 20:05	Text Document

Screenshot 5.2: Upload the data file


Heart Disease Prediction Using Bio Inspired Algorithms
Heart Disease Prediction Using Bio Inspired Algorithms
Upload Heart Disease
E:/krest/HeartDiseasePrediction/heart_dataset/dataset
Run Genetic Algorithm
Run BAT Algorithm
Run BEE Algorithm
Upload & Predict Test Data
Accuracy Graph
Exit

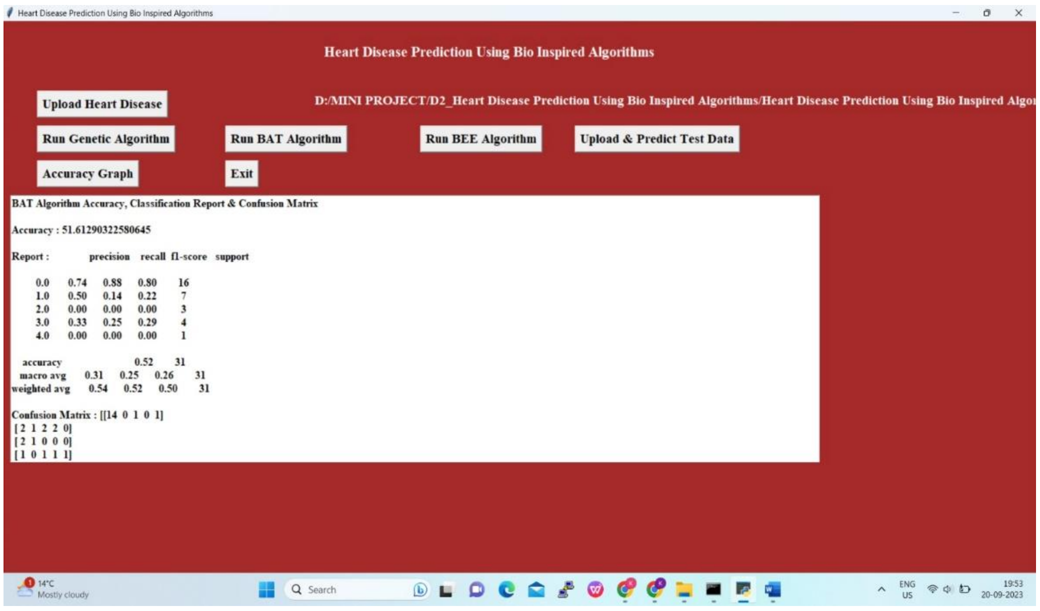
```
E:/krest/HeartDiseasePrediction/heart_dataset/dataset loaded
```

Screenshot 5.3: Run Genetic Algorithm


Heart Disease Prediction Using Bio Inspired Algorithms
Heart Disease Prediction Using Bio Inspired Algorithms
Upload Heart Disease
E:/krest/HeartDiseasePrediction/heart_dataset/dataset
Run Genetic Algorithm
Run BAT Algorithm
Run BEE Algorithm
Upload & Predict Test Data
Accuracy Graph
Exit

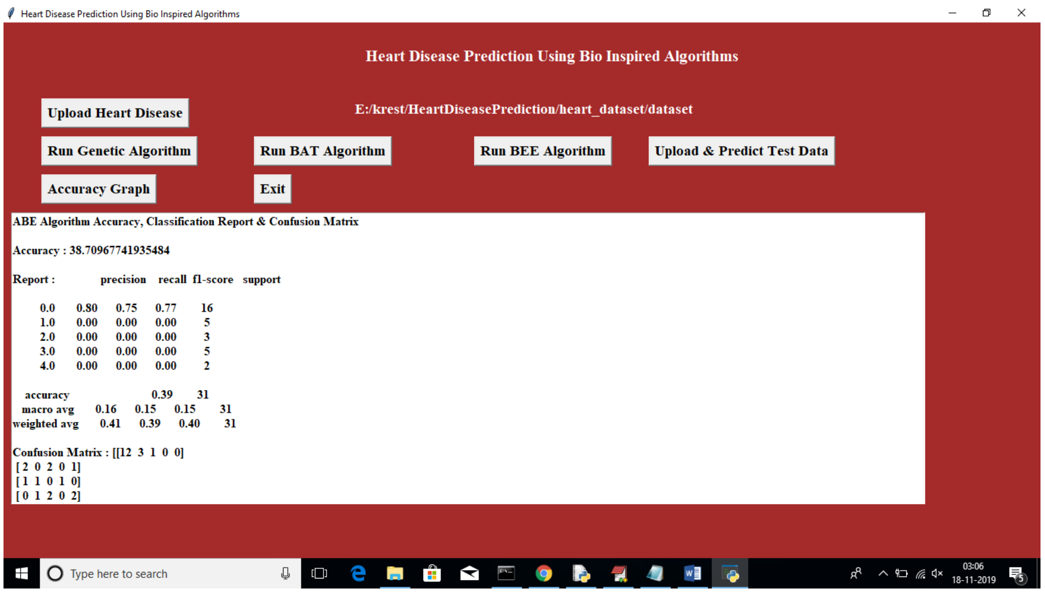
```
GA Algorithm Accuracy, Classification Report & Confusion Matrix

Accuracy : 100.0

Report :          precision    recall  f1-score   support

       0.0       1.00      1.00      1.00         5
       3.0       1.00      1.00      1.00         2

    accuracy                           1.00         7
   macro avg       1.00      1.00      1.00         7
weighted avg       1.00      1.00      1.00         7

Confusion Matrix : [[5 0]
 [0 2]]
```

Screenshot 5.4: GA Accuracy, Precision


Heart Disease Prediction Using Bio Inspired Algorithms
Heart Disease Prediction Using Bio Inspired Algorithms
Upload Heart Disease
D:/MINI PROJECT/D2_Heart Disease Prediction Using Bio Inspired Algorithms/Heart Disease Prediction Using Bio Inspired Algo
Run Genetic Algorithm
Run BAT Algorithm
Run BEE Algorithm
Upload & Predict Test Data
Accuracy Graph
Exit

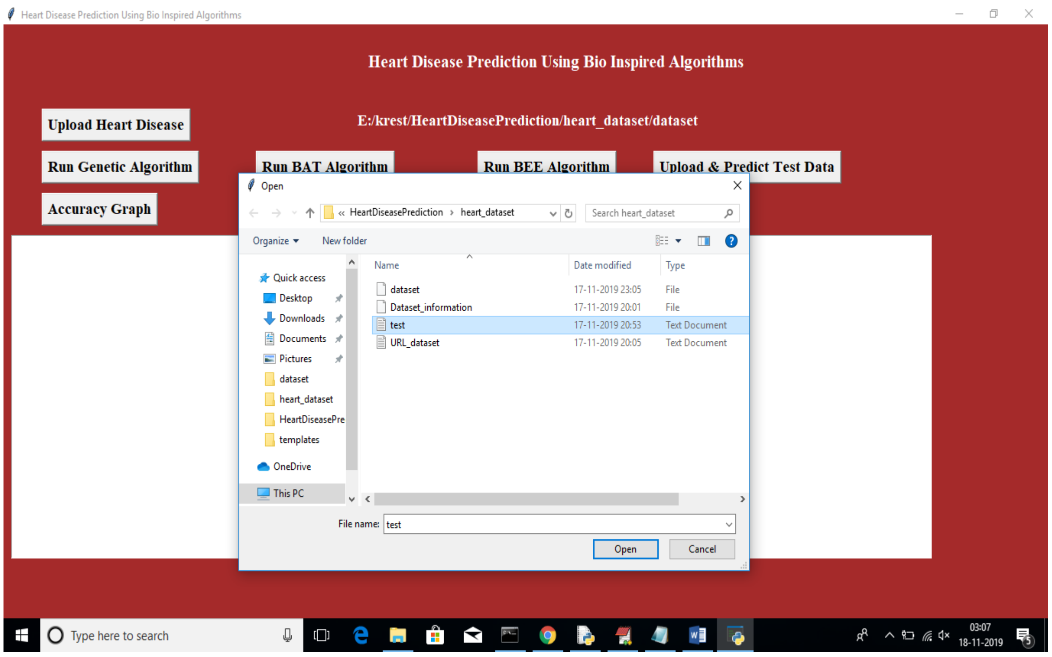
```
BAT Algorithm Accuracy, Classification Report & Confusion Matrix

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]]
```

Screenshot 5.5: Run Bat Algorithm Accuracy

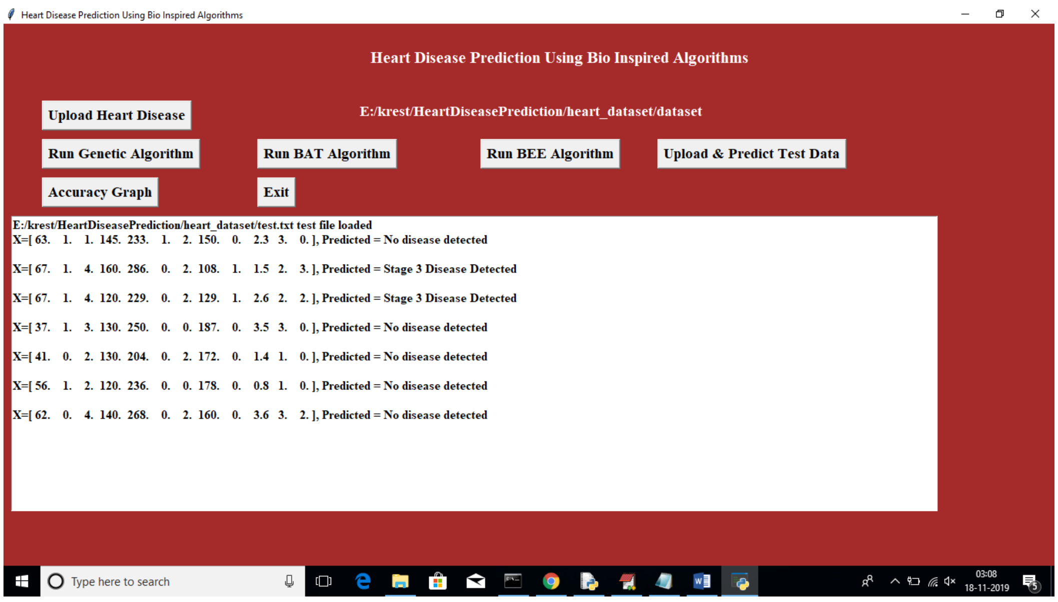

Heart Disease Prediction Using Bio Inspired Algorithms
Heart Disease Prediction Using Bio Inspired Algorithms
Upload Heart Disease
E:/krest/HeartDiseasePrediction/heart_dataset/dataset
Run Genetic Algorithm
Run BAT Algorithm
Run BEE Algorithm
Upload & Predict Test Data
Accuracy Graph
Exit
Type here to search

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

Accuracy : 38.70967741935484

Report :          precision   recall  f1-score   support

      0.0      0.80      0.75      0.77      16
      1.0      0.00      0.00      0.00       5
      2.0      0.00      0.00      0.00       3
      3.0      0.00      0.00      0.00       5
      4.0      0.00      0.00      0.00       2

   accuracy                        0.39      31
  macro avg      0.16      0.15      0.15      31
weighted avg      0.41      0.39      0.40      31

Confusion Matrix : [[12  3  1  0  0]
 [ 2  0  2  0  1]
 [ 1  1  0  1  0]
 [ 0  1  2  0  2]]
```

Screenshot 5.6: Run BEE Algorithm Accuracy

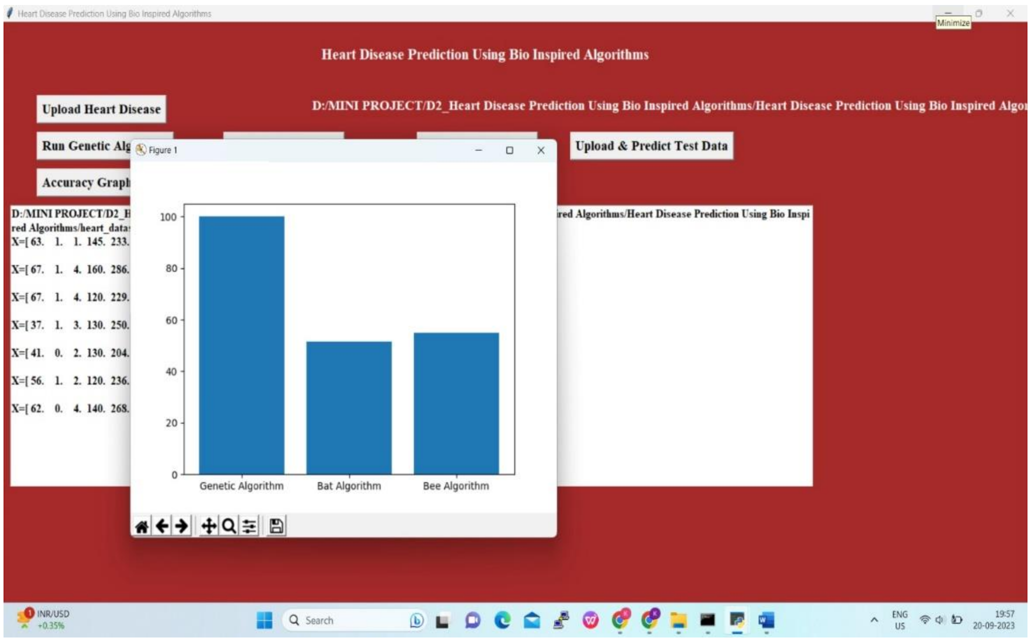

Heart Disease Prediction Using Bio Inspired Algorithms
E:/krest/HeartDiseasePrediction/heart_dataset/dataset
Upload Heart Disease
Run Genetic Algorithm
Run BAT Algorithm
Run BEE Algorithm
Upload & Predict Test Data
Accuracy Graph
Open
HeartDiseasePrediction › heart_dataset
dataset
Dataset_information
test
URL_dataset
File name: test
Open
Cancel

Screenshot 5.7: Upload And Predict Test Data


Heart Disease Prediction Using Bio Inspired Algorithms
E:/krest/HeartDiseasePrediction/heart_dataset/dataset
Upload Heart Disease
Run Genetic Algorithm
Run BAT Algorithm
Run BEE Algorithm
Upload & Predict Test Data
Accuracy Graph
Exit

```
E:/krest/HeartDiseasePrediction/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

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

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
```

Screenshot 5.8: Predicted Disease Stages



```
D:/MINI PROJECT/D2_Heart Disease Prediction Using Bio Inspired Algorithms/Heart Disease Prediction Using Bio Inspired Algo
```

```
D:/MINI PROJECT/D2_H                                                   red Algorithms/Heart Disease Prediction Using Bio Inspi
red Algorithms/heart_data
X=[ 63.   1.   1.  145.  233.

X=[ 67.   1.   4.  160.  286.

X=[ 67.   1.   4.  120.  229.

X=[ 37.   1.   3.  130.  250.

X=[ 41.   0.   2.  130.  204.

X=[ 56.   1.   2.  120.  236.

X=[ 62.   0.   4.  140.  268.
```

Screenshot 5.9: X-axis represents Algorithm Name and y-axis represents accuracy of those algorithms

6. TESTING

6. TESTING

6.1 INTRODUCTION TO TESTING

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

6.2 TYPES OF TESTING

6.2.1 UNIT TESTING

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

6.2.2 INTEGRATION TESTING

Integration tests are designed to test integrated software components to Determine if they actually run as one program. Integration tests demonstrate that although the components were individually satisfactory, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

6.2.3 FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.
Functional testing is centered on the following items:

Valid Input	: identified classes of valid input must be accepted.
Invalid Input	: identified classes of invalid input must be rejected.
Functions	: identified functions must be exercised.
Output	: identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked. Organization and preparation of functional tests is focused on requirements, key functions, or special test cases

6.3 TEST CASES

6.3.1 CLASSIFICATION

Test Case ID	Test Case Name	Purpose	Input	Output
1	Predicting the Heart Disease	To Predict Heart Disease	Data is given as input	An output is of person having a heart disease or not
2	Predicting the Heart Disease	To Predict Heart Disease	Data is given as input	By using genetic algorithm gets the accurate output

7. CONCLUSION & FUTURE SCOPE

7. CONCLUSION & FUTURE SCOPE

7.1 PROJECT CONCLUSION

Throughout this project, we have successfully developed and implemented predictive models utilizing bio-inspired algorithms such as Artificial Neural Networks and Genetic Algorithms. These models have shown promise in improving the accuracy and efficiency of heart disease prediction, offering valuable insights for healthcare professionals. However, it's crucial to acknowledge the ongoing need for ethical considerations, patient data privacy, and regulatory compliance when deploying such systems in real-world healthcare settings. This project serves as a foundation for further research and collaboration with healthcare experts to refine and expand the capabilities of predictive models in the field of cardiology. Ultimately, our work aims to contribute to early detection, improved patient care, and better outcomes for individuals at risk of heart disease.

7.2 FUTURE SCOPE

Firstly, there is room for enhancing predictive models by exploring more sophisticated bio-inspired algorithms and machine learning techniques. This includes delving into deep learning architectures, ensemble methods, and other state-of-the-art algorithms to improve prediction accuracy. Secondly, expanding the dataset to include diverse and multi-modal health data, such as genetic information, wearable device data, and real-time monitoring, can lead to more comprehensive and accurate predictions. Additionally, incorporating explainable AI techniques can enhance the interpretability of the models, making them more valuable for healthcare professionals. Furthermore, collaborating with healthcare institutions for large-scale clinical trials and real-world implementation could validate the effectiveness of the system in practical healthcare settings. Lastly, extending the scope to predict other medical conditions beyond heart disease, such as diabetes or stroke, could broaden the impact of bio-inspired algorithms in healthcare.

8.BIBLIOGRAPHY

8.BIBLIOGRAPHY

8.1 REFERENCES

[1] M. Chen, S. Mao, and Y. Liu, ‘‘Big data: A survey,’’ Mobile Netw. Appl., vol. 19, no. 2, pp. 171– 209, Apr. 2014.

[2] J. Wang, M. Qiu, and B. Guo, ‘‘Enabling real-time information service on telehealth system over cloud-based big data platform,’’ J. Syst. Archit., vol. 72, pp. 69–79, Jan. 2017. [10] D. W. Bates, S. Saria, L. Ohno-Machado, A. Shah, and G. Escobar, ‘‘Big data in health care: Using analytics to identify and manage high-risk and high-cost patients,’’ Health Affairs, vol. 33, no. 7, pp. 1123–1131, 2014.

[3] D. Tian, J. Zhou, Y. Wang, Y. Lu, H. Xia, and Z. Yi, ‘‘A dynamic and self-adaptive network selection method for multimode communications in heterogeneous vehicular telematics,’’ IEEE Trans. Intell. Transp. Syst., vol. 16, no. 6, pp. 3033–3049, Dec. 2015.

8.2 GITHUB LINK