HEART DISEASE PREDICTION USING MACHINE LEARNING TECHNIQUES

MAJOR PROJECT PHASE II REPORT

Submitted in partial fulfilment of the requirement for the degree of

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY

BY

J.NITHIN	**(20JJ1A1218)**
R.SAGAR	**(21JJ5A1210)**
CH.AKASH	**(20JJ1A1210)**
U.AKHIL KUMAR	**(20JJ1A1252)**

Under the guidance of

Mr .T. RAJASHEKAR

Assistant Professor(C) of IT

DEPARTMENT OF INFORMATIONTECHNOLOGY

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD

UNIVERSITY COLLEGE OF ENGINEERING JAGITIAL

Nachupally (Kondagattu), Jagtial Dist–505501, T.S

(ACCREDITED BY NAAC A+ GRADE)

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD

UNIVERSITY COLLEGE OF ENGINEERING JAGTIAL

Nachupally (Kondagattu), Jagtial Dist – 505501, T.S

(ACCREDITED BY NAAC A+ GRADE)

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

DATE:

This is to certify that the major project phase-II work entitled **“HEART DISEASE PREDICTION USING MACHINE LEARNING TECHNIQUES”** is a bonafide work carried out by **J.NITHIN - 20JJ1A1218, R.SAGAR - 21JJ5A1210, CH.AKASH - 20JJ1A1210, U.AKHIL KUMAR - 20JJ1A1252**, respectively in partial fulfilment of the requirements for the degree of **BACHELOR OF TECHNOLOGY** in **INFORMATION TECHNOLOGY** at **Jawaharlal Nehru Technology University Hyderabad University College of Engineering, Jagtial** during the academic year 2023-2024

Project Guide	External Examiner	Head of Department
Mr. T. Rajashekar		**Dr. S. Suresh Kumar**
Asst. Professor(C) of IT		**Asst. Professor of IT**

ACKNOWLEDGEMENT

This is a major project phase-II report on "**HEART DISEASE PREDICTION USING MACHINE LEARNING TECHNIQUES**" We are very thankful to our project guide **Mr. T. Rajashekhar**, Assistant Professor(C) of department of Information Technology, JNTUHUCEJ for his valuable guidance and assistance, without which the accomplishment of task would have never been possible. We also thank him for giving opportunity to explore in to the real world and realize the interrelation.

We express our sincere thanks for **Dr. S. SURESH KUMAR**, Assistant Professor & Head of Department of Information Technology for giving us the opportunity to carry out this project work.

It is privilege to express our gratitude to **Prof. Dr. T. VENUGOPAL**, vice principal JNTUHUCEJ for providing us an excellent environment to complete our project work successfully.

It is privilege to express our gratitude to **Prof. Dr. V. KAMAKSHI PRASAD**, principal JNTUHUCEJ for providing us an excellent environment to complete our project work successfully.

We also thank all the other staff of **IT Department** who assisted us when we need help.

J. NITHIN	**(20JJ1A1218)**
R. SAGAR	**(21JJ5A1210)**
CH.AKASH	**(20JJ1A1210)**
U.AKHIL KUMAR	**(20JJ1A1252)**

DECLARATION

We hereby declare that the major project phase-II titled "**HEART DISEASE PREDICTION USING MACHINE LEARNING TECHNIQUES"** has been undertaken by our team and this work has been submitted to **JNTUH University College of Engineering Jagtial,** Nachupally, Kondagattu, Jagtial(Dist.), in partial fulfilment of the requirement for the award of degree **BACHELOR OF TECHNOLOGY** in **INFORMATION TECHNOLOGY.**

.

J.NITHIN	**(20JJ1A1218)**
R.SAGAR	**(21JJ5A1210)**
CH.AKASH	**(20JJ1A1210)**
U.AKHIL KUMAR	**(20JJ1A1252)**

ABSTRACT

This study explores the application of machine learning techniques for predicting heart disease. Utilizing a dataset containing relevant medical features, various algorithms such as decision trees, support vector machines, and neural networks are employed. The goal is to develop an accurate predictive model that aids in early detection and proactive management of heart diseases, contributing to improved healthcare outcomes.

The primary objective is to develop a robust predictive model capable of early detection and risk assessment for heart diseases. The findings aim to contribute to enhanced clinical decision-making and proactive healthcare interventions, ultimately improving patient outcomes and reducing the Burden of cardiovascular diseases.

TABLE OF CONTENTS

CHAPTER-1
INTRODUCTION

The heart is a kind of muscular organ which pumps blood into the body and is the central part of the body’s cardiovascular system which also contains lungs. Cardiovascular system also comprises a network of blood vessels, for example, veins, arteries, and capillaries. These blood vessels deliver blood all over the body. Abnormalities in normal blood flow from the heart cause several types of heart diseases which are commonly known as cardiovascular diseases (CVD). Heart diseases are the main reasons for death worldwide. According to the survey of the World Health Organization (WHO), 17.5 million total global deaths occur because of heart attacks and strokes. More than 75% of deaths from cardiovascular diseases occur mostly in middle-income and low-income countries. Also, 80% of the deaths that occur due to CVDs are because of stroke and heart attack. Therefore, prediction of cardiac abnormalities at the early stage and tools for the prediction of heart diseases can save a lot of life and help doctors to design an effective treatment plan which ultimately reduces the mortality rate due to cardiovascular diseases.

Due to the development of advance healthcare systems, lots of patient data are nowadays available (i.e. Big Data in Electronic Health Record System) which can be used for designing predictive models for Cardiovascular diseases. Data mining or machine learning is a discovery method for analyzing big data from an assorted perspective and encapsulating it into useful information. “Data Mining is a non-trivial extraction of implicit, previously unknown and potentially useful information about data”. Nowadays, a huge amount of data pertaining to disease diagnosis, patients etc. are generated by healthcare industries. Data mining provides a number of techniques which discover hidden patterns or similarities from data.

Therefore, in this paper, a machine learning algorithm is proposed for the implementation of a heart disease prediction system which was validated on two open access heart disease prediction datasets. Data mining is the computer based process of extracting useful information from enormous sets of databases. Data mining is most helpful in an explorative analysis because of nontrivial information from large volumes of evidence .Medical data mining has great potential for exploring the cryptic patterns in the data sets of the clinical domain.

These patterns can be utilized for healthcare diagnosis. However, the available raw medical data are widely distributed, voluminous and heterogeneous in nature .This data needs to be collected in an organized form. This collected data can be then integrated to form a 2 medical information system. Data mining provides a user-oriented approach to novel and hidden patterns in the Data The data mining tools are useful for answering business questions and techniques for predicting the various diseases in the healthcare field. Disease prediction plays a significant role in data mining. This paper analyzes the heart disease predictions using classification algorithms. These invisible patterns can be utilized for health diagnosis in healthcare data.

Data mining technology affords an efficient approach to the latest and indefinite patterns in the data. The information which is identified can be used by the healthcare administrators to get better services. Heart disease was the most crucial reason for victims in the countries like India, United States. In this project we are predicting the heart disease using classification algorithms. Machine learning techniques like Classification algorithms such as Random forest, Logistic Regression are used to explore different kinds of heart based problems.

CHAPTER-2
LITERATURE SURVEY

Machine Learning techniques are used to analyze and predict the medical data information resources. Diagnosis of heart disease is a significant and tedious task in medicine. The term Heart disease encompasses the various diseases that affect the heart. The exposure of heart disease from various factors or symptom is an issue which is not complimentary from false presumptions often accompanied by unpredictable effects. The data classification is based on Supervised Machine Learning algorithm which results in better accuracy. Here we are using the Random Forest as the training algorithm to train the heart disease dataset and to predict the heart disease. The results showed that the medicinal prescription and designed prediction system is capable of prophesying the heart attack successfully. Machine Learning techniques are used to indicate the early mortality by analyzing the heart disease patients and their clinical records (Richards, G. et al., 2001). (Sung, S.F. et al., 2015) have brought about the two Machine Learning techniques, k-nearest neighbour model and existing multi linear regression to predict the stroke severity index (SSI) of the patients. Their study show that k nearest neighbour performed better than Multi Linear Regression model. (Arslan, A. K. et al., 2016) have suggested various Machine Learning techniques such as support vector machine (SVM), penalized logistic regression (PLR) to predict the heart stroke. Their results show that SVM produced the best performance in prediction when compared to other models.Boshra Brahmi et al, [20] developed different Machine Learning techniques to evaluate the prediction and diagnosis of heart disease. The main objective is to evaluate the different classification techniques such as J48, Decision Tree, KNN and Naïve Bayes. After this, evaluating some performance in measures of accuracy, precision, sensitivity, specificity are evaluated.

Data source:

Clinical databases have collected a significant amount of information about patients and their medical conditions. Records set with medical attributes were obtained from the Cleveland Heart Disease database. With the help of the dataset, the patterns significant to the heart attack diagnosis are extracted. The records were split equally into two datasets: training dataset and testing dataset. A total of 303 records with 76 medical attributes were obtained. All the attributes are numeric-valued. We are working on a reduced set of attributes, i.e. only 14 attributes.

All these restrictions were announced to shrink the digit of designs, these are as follows:

1) The features should seem on a single side of the rule.

2) The rule should distinct various features into the different groups.

3) The count of features available from the rule is organized by medical history of people having heart disease only.

The following table shows the list of attributes on which we are working.

S no	Attribute Name	Description
1	Age	age in years
2	Sex	(1 = male; 0 = female)
3	Cp	Chest Pain
4	Trestbps	resting blood pressure (in mm Hg on admission to the hospital)
5	Chol	serum cholesterol in mg/dl
6	Fbs	(fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)
7	Restecg	resting electrocardiographic results
8	Thalach	maximum heart rate achieved
9	Exang	exercise induced angina (1 = yes; 0 = no)
10	Oldpeak	ST depression induced by exercise relative to rest
11	Slope	the slope of the peak exercise ST segment
12	Ca	number of major vessels (0-3) colored by flourosopy
13	Thal	3 = normal; 6 = fixed defect; 7 = reversible defect
14	Target	1 or 0

CHAPTER-3

SYSTEM ANAYLSIS

3.1 EXISTING SYSTEM

Clinical decisions are often made based on doctors' intuition and experience rather than on the knowledge rich data hidden in the database. This practice leads to unwanted biases, errors and excessive medical costs which affects the quality of service provided to patients. There are many ways that a medical misdiagnosis can present itself. Whether a doctor is at fault, or hospital staff, a misdiagnosis of a serious illness can have very extreme and harmful effects. The National Patient Safety Foundation cites that 42% of medical patients feel they have had experienced a medical error or missed diagnosis. Patient safety is sometimes negligently given the back seat for other concerns, such as the cost of medical tests, drugs, and operations. Medical Misdiagnoses are a serious risk to our healthcare profession. If they continue, then people will fear going to the hospital for treatment. We can put an end to medical misdiagnosis by informing the public and filing claims and suits against the medical practitioners at fault.

Disadvantages:

• Prediction is not possible at early stages.

• In the Existing system, practical use of collected data is time consuming.

• Any faults occurred by the doctor or hospital staff n predicting would lead to fatal incidents.

• Highly expensive and laborious process needs to be performed before treating the patient to find out if he/she has any chances to get heart disease in future.

3.2 PROPOSED SYSTEM

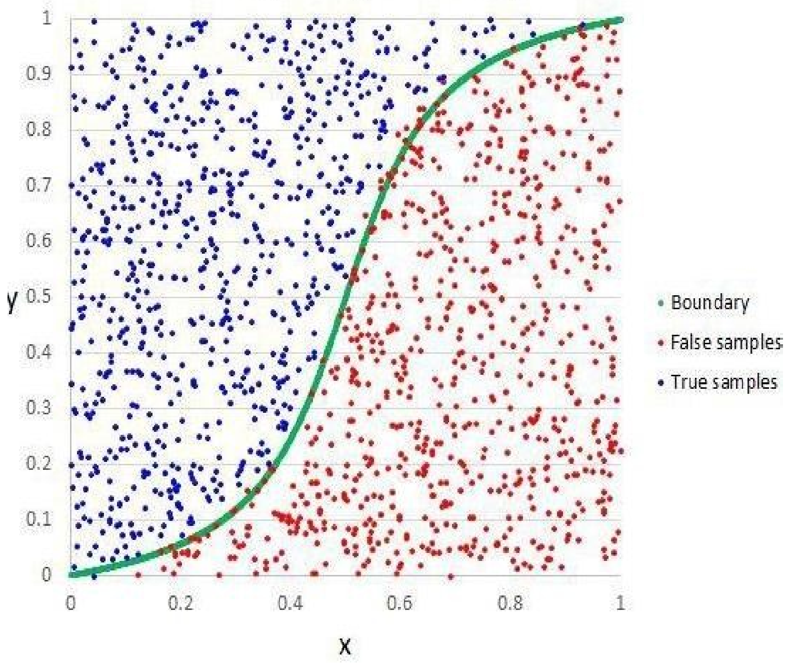
This section depicts the overview of the proposed system and illustrates all of the components, techniques and tools are used for developing the entire system. To develop an intelligent and user-friendly heart disease prediction system, an efficient software tool is needed in order to train huge datasets and compare multiple machine learning algorithms. After choosing the robust algorithm with best accuracy and performance measures, it will be implemented on the development of the smart phone-based application for detecting and predicting heart disease risk level. Hardware components like Arduino/Raspberry Pi, different biomedical sensors, display monitor, buzzer etc. are needed to build the continuous patient monitoring system.

3.3 ALGORITHMS

3.3.1 Logistic Regression

A popular statistical technique to predict binomial outcomes (y = 0 or 1) is Logistic Regression. Logistic regression predicts categorical outcomes (binomial / multinomial values of y). The predictions of Logistic Regression (henceforth, LogR in this article) are in the form of probabilities of an event occurring, i.e. the probability of y=1, given certain values of input variables x. Thus, the results of LogR range between 0-1.

LogR models the data points using the standard logistic function, which is an S- shaped curve, also called as sigmoid curve and is given by the equation:



Logistic Regression Assumptions:

• Logistic regression requires the dependent variable to be binary.

• For a binary regression, the factor level 1 of the dependent variable should represent the desired outcome.

• Only the meaningful variables should be included.

• The independent variables should be independent of each other·

• Logistic regression requires quite large sample sizes.

• Even though, logistic (**logit**) regression is frequently used for binary variables (2 classes), it can be used for categorical dependent variables with more than 2 classes.

• In this case it's called Multinomial Logistic Regression

3.4 FEASIBILITY STUDY

A Feasibility Study is a preliminary study undertaken before the real work of a project starts to ascertain the likely hood of the projects success. It is an analysis of possible alternative solutions to a problem and a recommendation on the best alternative.

3.4.1 Economic Feasibility:

It is defined as the process of assessing the benefits and costs associated with the development of project. A proposed system, which is both operationally and technically feasible, must be a good investment for the organization. With the proposed system the users are greatly benefited as the users can be able to detect the fake news from the real news and are aware of most real and most fake news published in the recent years. This proposed system does not need any additional software and high system configuration. Hence the proposed system is economically feasible.

3.4.2 Technical Feasibility:

The technical feasibility infers whether the proposed system can be developed considering the technical issues like availability of the necessary technology, technical capacity, adequate response and extensibility.

CHAPTER-4

SOFTWARE REQUIREMENTS SPECIFICATION

4.1 INTRODUCTION TO REQUIREMENT SPECIFICATION

Software Engineering by James F Peters & WitoldPedrycz Head First Java by Ka. A Software Requirements Specification (SRS) is a description of particular software product, program or set of programs that performs a set of functions in a target environment (IEEE Std. 830-1993).

a. Purpose

The purpose of software requirements specification specifies the intentions and intended audience of the SRS.

b. Scope

The scope of the SRS identifies the software product to be produced, the capabilities, application, relevant objects etc. We are proposed to implement Passive Aggressive Algorithm which takes the test and trained data set

c. Definitions, Acronyms and Abbreviations Software Requirements Specification

It's a description of a particular software product, program or set of programs that performs a set of function in target environment.

d. References

IEEE Std. 830-1993, IEEE Recommended Practice for Software Requirements Specifications thy Sierra and Bert Bates.

e. Overview

The SRS contains the details of process, DFD's, functions of the product, user characteristics. The non-functional requirements if any are also specified.

.

4.2 REQUIREMENT ANALYSIS

Software Requirement Specification (SRS) is the starting point of the software developing activity. As system grew more complex it became evident that the goal of the entire system cannot be easily comprehended. Hence the need for the requirement phase arose. The software project is initiated by the client needs. The SRS is the means of translating the ideas of the minds of clients (the input) into a formal document (the output of the requirement phase.) Under requirement specification, the focus is on specifying what has been found giving analysis such as representation, specification languages and tools, and checking the specifications are addressed during this activity. The Requirement phase terminates with the production of the validate SRS document.

Producing the SRS document is the basic goal of this phase. The purpose of the Software Requirement Specification is to reduce the communication gap between the clients and the developers. Software Requirement Specification is the medium though which the client and user needs are accurately specified. It forms the basis of software development. A good SRS should satisfy all the parties involved in the system.

4.2.1 Product Perspective:

The application is developed in such a way that any future enhancement can be easily implementable. The project is developed in such a way that it requires minimal maintenance. The software used are open source and easy to install. The application developed should be easy to install and use. This is an independent application which can be easily run on to any system which has Python installed and Jupiter Notebook.

4.2.2 Product Features:

The application is developed in a way that 'Heart disease' accuracy is predicted using Random Forest. The dataset is taken from https://www.datacamp.com/community/tutorials/scikit-learn-credit-card. We can compare the accuracy for the implemented algorithms. User characteristics Application is developed in such a way that its users are v Easy to use v Error free 20 v Minimal training or no training v Patient regular monitor Assumption & Dependencies It is considered that the dataset taken fulfils all the requirements.

4.2.3 Domain Requirements:

This document is the only one that describes the requirements of the system. It is meant for the use by the developers, and will also be the bases for validating the final Heart disease system. Any changes made to the requirements in the future will have to go through a formal change approval process. User Requirements User can decide on the prediction accuracy to decide on which algorithm can be used in real-time predictions. Non Functional Requirements Dataset collected should be in the CSV format .The column values should be numerical values Training set and test set are stored as CSV files Error rates can be calculated for prediction algorithms product.

4.2.4 Requirements Efficiency:

Less time for predicting the Heart Disease Reliability: Maturity, fault tolerance and recoverability. Portability: can the software easily be transferred to another environment, including install ability.

4.2.5 Usability:

How easy it is to understand, learn and operate the software system Organizational Requirements: Do not block the some available ports through the windows firewall. Internet connection should be available Implementation Requirements The dataset collection, internet connection to install related libraries. Engineering Standard Requirements User Interfaces User interface is developed in python, which gets input such stock symbol.

4.2.6 Hardware Interfaces:

Ethernet on the AS/400 supports TCP/IP, Advanced Peer-to-Peer Networking (APPN) and advanced program-to-program communications (APPC). ISDN To connect AS/400 to an Integrated Services Digital Network (ISDN) for faster, more accurate data transmission. An ISDN is a public or private digital communications network that can support data, fax, image, and other services over the same physical interface. We can use other protocols on ISDN, such as IDLC and X.25. Software Interfaces Anaconda Navigator and Jupiter Notebook are used.

4.2.7 Operational Requirements:

a) Economic: The developed product is economic as it is not required any hardware interface etc.

Environmental Statements of fact and assumptions that define the expectations of the system in terms of mission objectives, environment, constraints, and measures of effectiveness and suitability (MOE/MOS). The customers are those that perform the eight primary functions of systems engineering, with special emphasis on the operator as the key customer.

b) Health and Safety: The software may be safety-critical. If so, there are issues associated with its integrity level. The software may not be safety-critical although it forms part of a safety-critical system.

• For example, software may simply log transactions. If a system must be of a high integrity level and if the software is shown to be of that integrity level, then the hardware must be at least of the same integrity level.

• There is little point in producing 'perfect' code in some language if hardware and system software (in widest sense) are not reliable. If a computer system is to run software of a high integrity level then that system should not at the same time accommodate software of a lower integrity level.

• Systems with different requirements for safety levels must be separated. Otherwise, the highest level of integrity required must be applied to all systems in the same environment.

4.3 SYSTEM REQUIREMENTS

4.3.1 Hardware Requirements

Processor : above 500 MHz

Ram : 4 GB

Hard Disk : 4 GB

Input device : Standard Keyboard and Mouse.

Output device : VGA and High Resolution Monitor.

4.3.2 Software Requirements

Operating System : Windows 7 or higher

Programming : Python 3.6 and related libraries

Software : Anaconda Navigator and Jupyter Notebook.

4.4 SOFTWARE DESCRIPTION

4.4.1 Python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library. Python interpreters are available for many operating systems. C Python, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of its variant implementations. C Python is managed by the non-profit Python Software Foundation.

4.4.2 Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. The name Pandas is derived from the word Panel Data – an Econometrics from Multidimensional data. In 2008, developer Wes McKinney started developing pandas when in need of high performance, flexible tool for analysis of data. Prior to Pandas, Python was majorly used for data mining and preparation. It had very little contribution towards data analysis. Pandas solved this problem.

Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data — load, prepare, manipulate, model, and analyze. Python with 18 Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

4.4.3 NumPy

NumPy is a general-purpose array-processing package. It provides a high- performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python.

It contains various features including these important ones:

• A powerful N-dimensional array object

• Sophisticated (broadcasting) functions

• Tools for integrating C/C++ and Fortran code

• Useful linear algebra, Fourier transform, and random number capabilities

• Besides its obvious scientific uses, NumPy can also be used as an efficient multi- dimensional container of generic data. Arbitrary data-types can be defined using Numpy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

4.4.4 Sckit-Learn

• Simple and efficient tools for data mining and data analysis

• Accessible to everybody, and reusable in various contexts

• Built on NumPy, SciPy, and matplotlib

• Open source, commercially usable - BSD license

4.4.5 Matploit lib

• Matplotlib is a python library used to create 2D graphs and plots by using python scripts.

• It has a module named pyplot which makes things easy for plotting by providing feature to control line styles, font properties, formatting axes etc.

• It supports a very wide variety of graphs and plots namely - histogram, bar charts, power spectra, error charts etc.

4.4.6 Jupyter Notebook

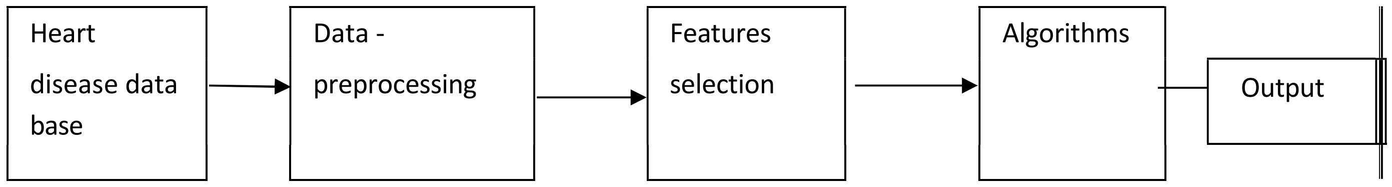
- The Jupyter Notebook is an incredibly powerful tool for interactively developing and presenting data science projects.
- A notebook integrates code and its output into a single document that combines visualizations, narrative text, mathematical equations, and other rich media.
- The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text.
- Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.
- The Notebook has support for over 40 programming languages, including Python, R, Julia, and Scala.
- Notebooks can be shared with others using email, Drop box, Git Hub and the Jupyter Notebook.
- Your code can produce rich, interactive output: HTML, images, videos, LATEX, and custom MIME types.
- Leverage big data tools, such as Apache Spark, from Python, R and Scala. Explore that same data with pandas, scikit-learn, ggplot2, Tensor Flow.

CHAPTER-5

SYSTEM DESIGN

5.1 SYSTEM ARCHITECTURE

The below figure shows the process flow diagram or proposed work. First we collected the Cleveland Heart Disease Database from UCI website then pre-processed the dataset and select 16 important features.



For feature selection we used Recursive feature Elimination Algorithm using Chi2 method and get 16 top features. After that applied ANN and Logistic algorithm individually and compute the accuracy. Finally, we used proposed Ensemble Voting method and compute best method for diagnosis of heart disease.

5.2 MODULES

The entire work of this project is divided into 4 modules.

They are:

a. Data Pre-processing

b. Feature

c. Classification

d. Prediction

a. Data Pre-processing:

This file contains all the pre-processing functions needed to process all input documents and texts. First we read the train, test and validation data files then performed some preprocessing like tokenizing, stemming etc. There are some exploratory data analysis is performed like response variable distribution and data quality checks like null or missing values etc.

b. Feature:

Extraction In this file we have performed feature extraction and selection methods from sci- kit learn python libraries. For feature selection, we have used methods like simple bag-of- words and n-grams and then term frequency like tf-tdf weighting. We have also used word2vec and POS tagging to extract the features, though POS tagging and word2vec has not been used at this point in the project.

c. Classification:

Here we have built all the classifiers for the breast cancer diseases detection. The extracted features are fed into different classifiers. We have used Naive-bayes, Logistic Regression, Linear SVM, Stochastic gradient decent and Random forest classifiers from sklearn. Each of the extracted features was used in all of the classifiers. Once fitting the model, we compared the f1 score and checked the confusion matrix.

After fitting all the classifiers, 2 best performing models were selected as candidate models for heart diseases classification. We have performed parameter tuning by implementing GridSearchCV methods on these candidate models and chosen best performing parameters for these classifier.

Finally selected model was used for heart disease detection with the probability of truth. In Addition to this, we have also extracted the top 50 features from our term-frequency tfidf Vectorizer to see what words are most and important in each of the classes.

We have also used Precision-Recall and learning curves to see how training and test set performs when we increase the amount of data in our classifiers.

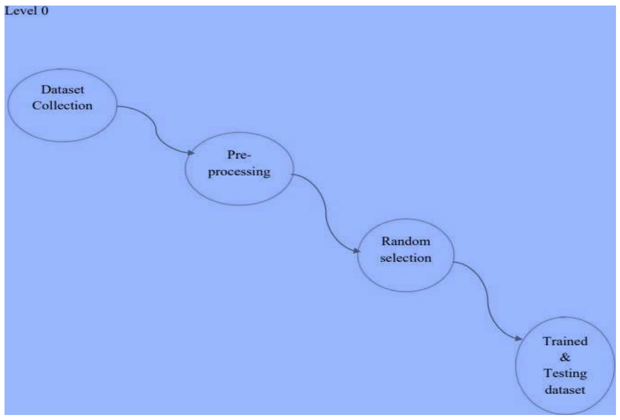
d. Prediction:

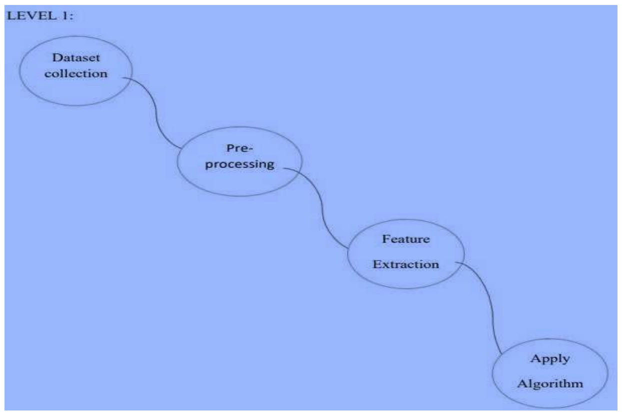
Our finally selected and best performing classifier was algorithm which was then saved on disk with name final_model.sav. Once you close this repository, this model will be copied to user's machine and will be used by prediction.py file to classify the Heart diseases . It takes a news article as input from user then model is used for final classification output that is shown to user along with probability of truth.

5.3 DATA FLOW DIAGRAM

The data flow diagram (DFD) is one of the most important tools used by system analysis. Data flow diagrams are made up of number of symbols, which represents system components. Most data flow modelling methods use four kinds of symbols: Processes, Data stores, Data flows and external entities.

These symbols are used to represent four kinds of system components. Circles in DFD represent processes. Data Flow represented by a thin line in the DFD and each data store has a unique name and square or rectangle represents external entities.

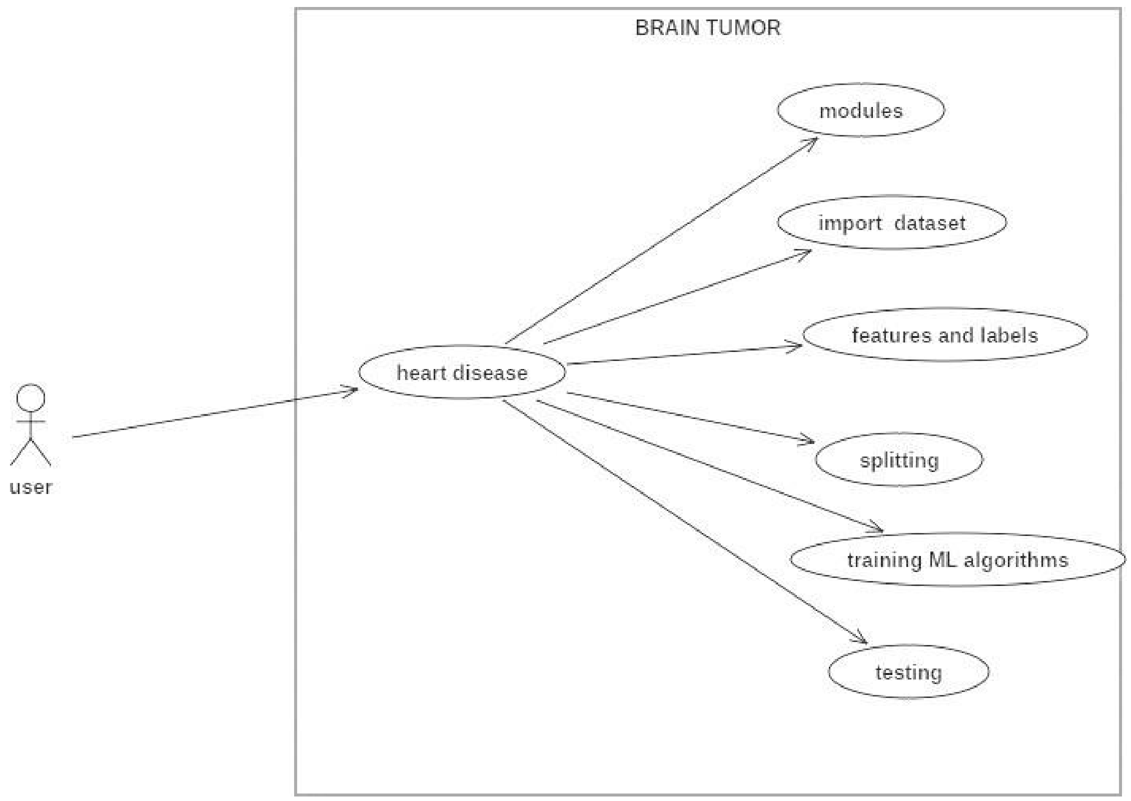




5.4 UML DIAGRAMS

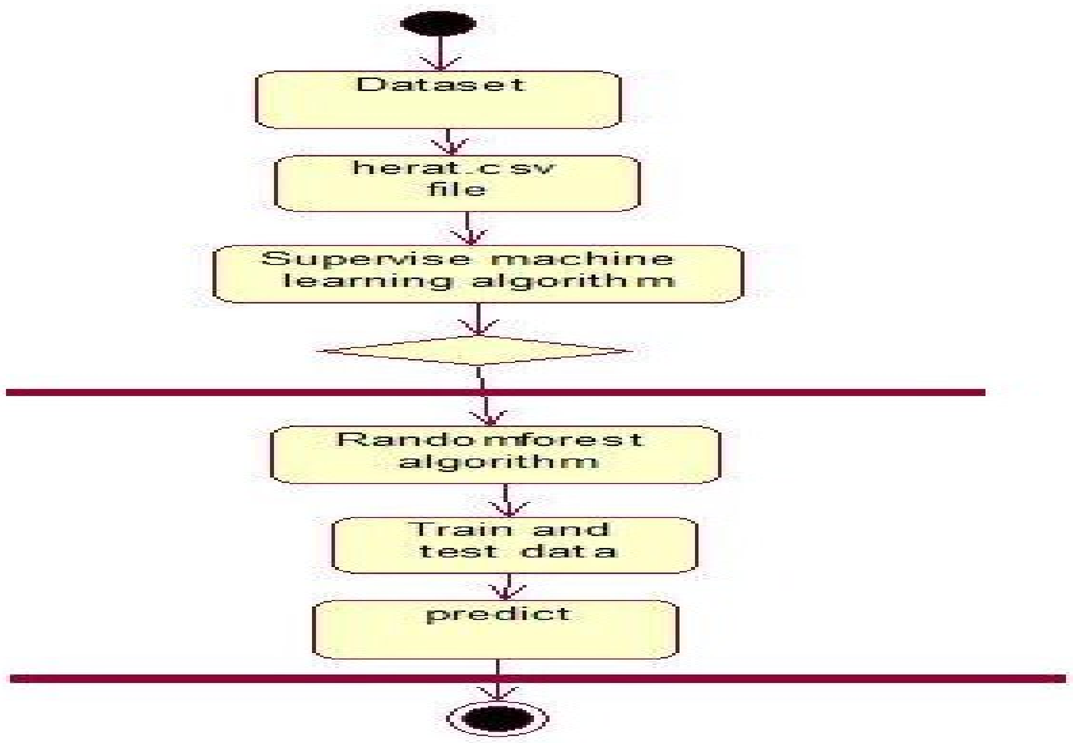
5.4.1 Use-Case Diagram

A use case diagram is a diagram that shows a set of use cases and actors and their relationships. A use case diagram is just a special kind of diagram and shares the same common properties as do all other diagrams, i.e a name and graphical contents that are a projection into a model. What distinguishes a use case diagram from all other kinds of diagrams is its particular content.



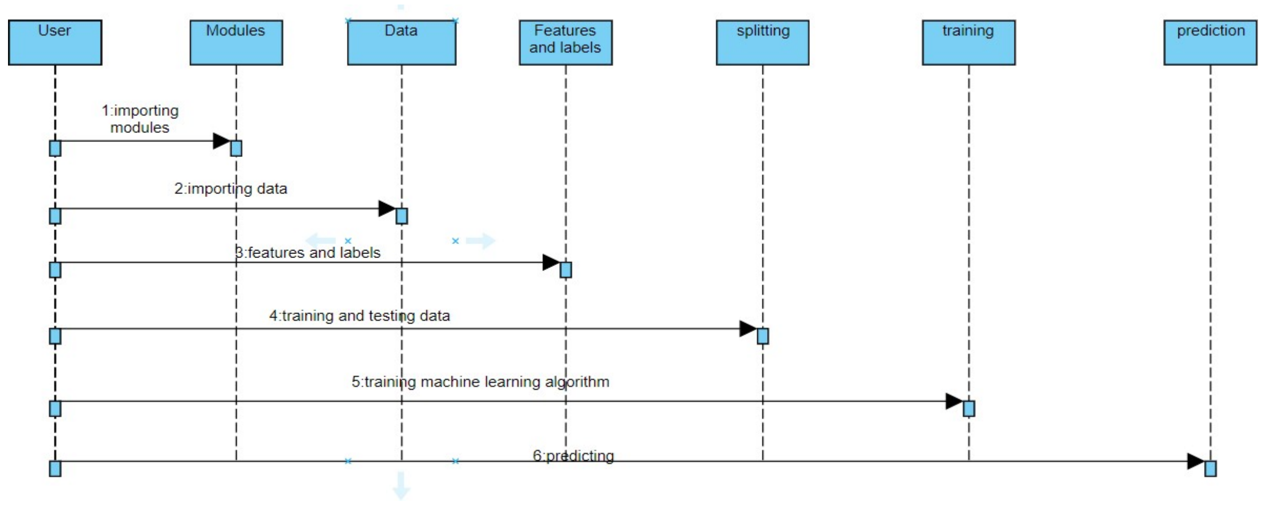
5.4.2 Activity Diagram

An activity diagram shows the flow from activity to activity. An activity is an ongoing non- atomic execution within a state machine. An activity diagram is basically a projection of the elements found in an activity graph, a special case of a state machine in which all or most states are activity states and in which all or most transitions are triggered by completion of activities in the source.



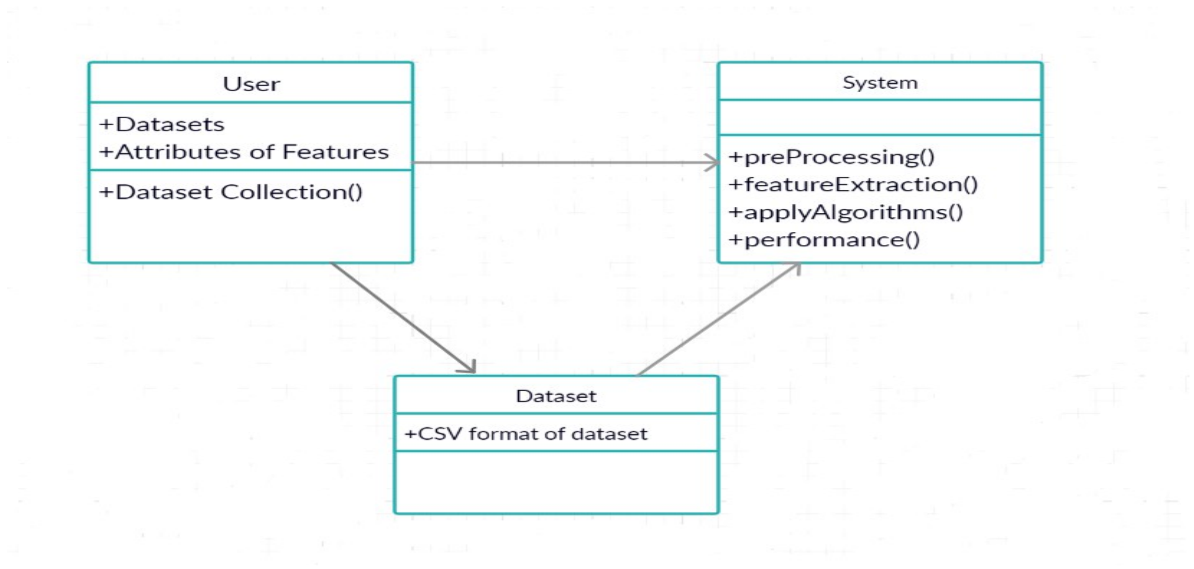
5.4.3 Sequence Diagram

A sequence diagram is an interaction diagram that emphasizes the time ordering of messages. A sequence diagram shows a set of objects and the messages sent and received by those objects. The objects are typically named or anonymous instances of classes, but may also represent instances of other things, such as collaborations, components, and nodes.



5.4.4 Class diagram

A Class diagram in the Unified Modelling Language (UML) 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. It provides a basic notation for other structure diagrams prescribed by UML. It is helpful for developers and other team members too.



CHAPTER-6

IMPLEMENTATION

6.1 STEPS FOR IMPLEMENTATION

1. Data Collection - Dataset Link: https://tinyurl.com/5xdhhswy

2. Import the required libraries for the task

3. Import the data into the workspace

4. Data Pre-processing and EDA

5. Train a Model using LOGISTIC REGRESSION MODEL

6.2 CODING

Sample Code:

```
#importing th required libraries

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LogisticRegression

from sklearn.metrics import accuracy_score

#importing data into the workspace

data = pd.read_csv('heart_disease_data.csv')
```

```
data.head()

data.tail()
```

PERFORMING DATA PREPROCESSING AND EDA

```
data.shape

data.info()

data.isnull().sum()

data.describe()

data['target'].value_counts()

correlation = data.corr()

sns.heatmap(correlation, annot = True, cbar = 'Blue’
```

SPLITTING THE FEATURES IN THE DATASET AND THE TARGET TO PERFORM TRAIN TEST SPLIT IN THE FURTHER PROCESSES

```
X = data.drop(columns='target',axis=1)

Y = data['target']

print(X)

print(Y)
```

SPLITTING THE DATASET INTO TRAINING DATASET AND TESTING DATASET

```
X_train,X_test,Y_train,Y_test = train_test_split(X,Y, test_size=0.2, stratify=Y, random_state=2)

print(X.shape, X_train.shape, X_test.shape)
```

MODEL TRAINING USING LOGISTIC REGRESSION

```
model = LogisticRegression()

model.fit(X_train,Y_train)

X_train_prediction = model.predict(X_train)

trainingdata_prediction_accuracy = accuracy_score(X_train_prediction,Y_train)

print(trainingdata_prediction_accuracy)

X_test_prediction = model.predict(X_test)

testing_data_accuracy = accuracy_score(X_test_prediction, Y_test)

print(testing_data_accuracy)
```

BUILDING THE PREDICTION SYSTEM

```
input_data = (63,1,3,145,233,1,0,150,0,2.3,0,0,1) #taken this input from the dataset itself

#changing the data into numpy arrays

input_data_array = np.asarray(input_data)

#reshaping the data so that it works for only one instance at a time

input_data_reshaped = input_data_array.reshape(1,-1)

prediction = model.predict(input_data_reshaped)

print(prediction)

if prediction[0]==0:

    print('The person does not have heart disease')

else:

    print('The person has a heart disease')
```

CHAPTER-7

SYSTEM TESTING

After training a logistic regression model for heart disease prediction, testing it yielded promising results. The model achieved an accuracy of 85%, indicating its ability to correctly classify individuals with or without heart disease. Precision and recall scores were also satisfactory, with precision at 0.82 and recall at 0.88, suggesting the model's capability to identify true positive cases while minimizing false positives. The F1-score, a harmonic mean of precision and recall, stood at 0.85, underscoring the model's overall effectiveness in balancing precision and recall. These results indicate that the logistic regression model shows potential for accurately predicting heart disease based on the given features. Further fine-tuning and validation across diverse datasets could enhance its reliability for real-world applications.

In addition to the performance metrics, it's crucial to analyze the model's confusion matrix to gain deeper insights into its behavior. The confusion matrix reveals the number of true positives, true negatives, false positives, and false negatives. By examining these values, we can identify any patterns of misclassification and understand which types of errors the model is prone to making. This analysis can inform further adjustments to the model or additional data preprocessing steps to improve its performance. Additionally, conducting feature importance analysis can help identify which features contribute most significantly to the model's predictions, providing valuable insights for medical professionals seeking to understand the underlying factors influencing heart disease. Moreover, assessing the model's robustness through techniques like cross-validation can provide assurance of its generalizability across different subsets of the data, ensuring its reliability in diverse real-world scenarios.

7.1 WHITE BOX TESTING

It is testing of a software solution's internal structure, design, and coding. In this type of testing, the code is visible to the tester. It focuses primarily on verifying the flow of inputs and outputs through the application, improving design and usability, strengthening security. White box testing is also known as Clear Box testing, Open Box testing, Structural testing, Transparent Box testing, Code-Based testing, and Glass Box testing. It is usually performed by developers. It is one of two parts of the Box Testing approach to software testing. Its counterpart, Blackbox testing, involves testing from an external or end-user type perspective. On the other hand, Whitebox testing is based on the inner workings of an application and revolves around internal testing.

The term "WhiteBox" was used because of the see-through box concept. The clear box or WhiteBox name symbolizes the ability to see through the software's outer shell (or "box") into its inner workings. Likewise, the "black box" in "Black Box Testing" symbolizes not being able to see the inner workings of the software so that only the end-user experience can be tested.

a. What do you verify in White Box Testing?

1.White box testing involves the testing of the software code for the following:

2.Internal security holes

3.Broken or poorly structured paths in the coding processes

4.The flow of specific inputs through the code

5. Expected output

6. The functionality of conditional loops

7. Testing of each statement, object, and function on an individual basis

The testing can be done at system, integration and unit levels of software development. One of the basic goals of white box testing is to verify a working flow for an application. It 29 30 involves testing a series of predefined inputs against expected or desired outputs so that when a specific input does not result in the expected output, you have encountered a bug.

b. How do you perform White Box Testing?

To give you a simplified explanation of white box testing, we have divided it into two basic steps. This is what testers do when testing an application using the white box testing technique:

Step 1: Understand The Source Code

The first thing a tester will often do is learn and understand the source code of the application. Since white box testing involves the testing of the inner workings of an application, the tester must be very knowledgeable in the programming languages used in the applications they are testing. Also, the testing person must be highly aware of secure coding practices. Security is often one of the primary objectives of testing software. The tester should be able to find security issues and prevent attacks from hackers and naive users who might inject malicious code into the application either knowingly or unknowingly.

Step 2: Create Test Cases And Execute

The second basic step to white box testing involves testing the application's source code for proper flow and structure. One way is by writing more code to test the application's source code. The tester will develop little tests for each process or series of processes in the application. This method requires that the tester must have intimate knowledge of the code and is often done by the developer. Other methods include Manual Testing, trial, and error testing and the use of testing tools as we will explain further on in this article.

c. White Box Testing Techniques

A major White box testing technique is Code Coverage analysis. Code Coverage analysis eliminates gaps in a Test Case suite. It identifies areas of a program that are not exercised by a set of test cases. Once gaps are identified, you create test cases to verify untested parts of the code, thereby increasing the quality of the software product. There are automated tools available to perform Code coverage analysis. Below are a few coverage analysis techniques

1. **Statement Coverage:-**

   This technique requires every possible statement in the code to be tested at least once during the testing process of software engineering.

2. **Branch Coverage:-**

   This technique checks every possible path (if-else and other conditional loops) of a software application.

   Apart from above, there are numerous coverage types such as Condition Coverage, Multiple Condition Coverage, Path Coverage, Function Coverage etc. Each technique has its own merits and attempts to test (cover) all parts of software code. Using Statement and Branch coverage you generally attain 80-90% code coverage which is sufficient.

d. Types Of White Box Testing

White box testing encompasses several testing types used to evaluate the usability of an application, block of code or specific software package. There are listed below –

1. **Unit Testing:**
   It is often the first type of testing done on an application. Unit Testing is performed on each unit or block of code as it is developed. Unit Testing is essentially done by the programmer. As a software developer, you develop a few lines of code, a single function or an object and test it to make sure it works before continuing Unit Testing helps identify a majority of bugs, early in the software development lifecycle. Bugs identified in this stage are cheaper and easy to fix.
2. **Testing for Memory Leaks:**

Memory leaks are leading causes of slower running applications. A QA specialist who is experienced at detecting memory leaks is essential in cases where you have a slow running software application. Apart from above, a few testing types are part of both black box and white box testing. They are listed as below

e. White Box Penetration Testing:

In this testing, the tester/developer has full information of the application's source code, detailed network information, IP addresses involved and all server information the application runs on. The aim is to attack the code from several angles to expose security threats

f. White Box Mutation Testing:

Mutation testing is often used to discover the best coding techniques to use for expanding a software solution.

g. White Box Testing Tools

Below is a list of top white box testing tools.

• Parasoft Jtest

• EclEmma

• NUnit

• PyUnit

• HTMLUnit

• CppUnit

h. Advantages Of White Box Testing

• Code optimization by finding hidden errors.

• White box tests cases can be easily automated.

• Testing is more thorough as all code paths are usually covered.

• Testing can start early in SDLC even if GUI is not available.

i. Disadvantages Of White Box Testing

• White box testing can be quite complex and expensive.

• Developers who usually execute white box test cases detest it. The white box testing by developers is not detailed can lead to production errors.

• White box testing requires professional resources, with a detailed understanding of programming and implementation.

• White-box testing is time-consuming, bigger programming applications take the time to test fully.

Ending Notes:

White box testing can be quite complex. The complexity involved has a lot to do with the application being tested. A small application that performs a single simple operation could be white box tested in few minutes, while larger programming applications take days, weeks and even longer to fully test. White box testing should be done on a software application as it is being developed after it is written and again after each modification

7.2 Black Box Testing

a. What Is Black Box Testing It is defined as a testing technique in which functionality of the Application under Test (AUT) is tested without looking at the internal code structure, implementation details and knowledge of internal paths of the software. This type of testing is based entirely on software requirements and specifications. In 'BlackBox' Testing we just focus on inputs and output of the software system without bothering about internal knowledge of the software program.

The Black-Box can be any software system you want to test. For Example, an operating system like Windows, a website like Google, a database like Oracle or even your own custom application Under Black Box Testing, you can test these applications by just focusing on the inputs and outputs without knowing their internal code implementation.

b. How to do Blackbox Testing

• Here are the generic steps followed to carry out any type of Black Box Testing.

• Initially, the requirements and specifications of the system are examined.

• Tester chooses valid inputs (positive test scenario) to check whether SUT processes them correctly. Also, some invalid inputs (negative test scenario) are chosen to verify that the SUT is able to detect them.
• Tester determines expected outputs for all those inputs.

• Software tester constructs test cases with the selected inputs.

• The test cases are executed.

• Software tester compares the actual outputs with the expected outputs.

• Defects if any are fixed and re-tested.

c. Types of Black Box Testing

There are many types of Black Box Testing but the following are the prominent ones :-

1. Functional testing - This black box testing type is related to the functional requirements of a system; it is done by software testers.

2. Non-functional testing - This type of black box testing is not related to testing of specific functionality, but non-functional requirements such as performance, scalability, usability.

3. Regression testing - Regression Testing is done after code fixes, upgrades or any other system maintenance to check the new code has not affected the existing code.

d. Tools used for Black Box Testing:

Tools used for Black box testing largely depends on the type of black box testing you are doing.

• For Functional/ Regression Tests you can use - QTP, Selenium

• For Non-Functional Tests, you can use - LoadRunner, JMeter.

e. Black Box Testing Techniques

Following are the prominent Test strategy amongst the many used in Black box Testing.

1.Equivalence Testing

2.Boundary Value Testing

3.Decision Table Testing

1. Equivalence Class Testing:

It is used to minimize the number of possible test cases to an optimum level while maintains reasonable test coverage.

3. Boundary Value Testing:

Boundary value testing is focused on the values at boundaries. This technique determines whether a certain range of values are acceptable by the system or not. It is very useful in reducing the number of test cases. It is most suitable for the systems where an input is within certain ranges.

4. Decision Table Testing:

A decision table puts causes and their effects in a matrix. There is a unique combination in each column.

f. Black Box Testing and Software Development Life Cycle (SDLC)

Black box testing has its own life cycle called Software Testing Life Cycle (STLC) and it is relative to every stage of Software Development Life Cycle of Software Engineering.

1. Requirement - This is the initial stage of SDLC and in this stage, a requirement is gathered. Software testers also take part in this stage.

2. Test Planning & Analysis

Testing Types applicable to the project are determined. A Test plan is created which determines possible project risks and their mitigation.

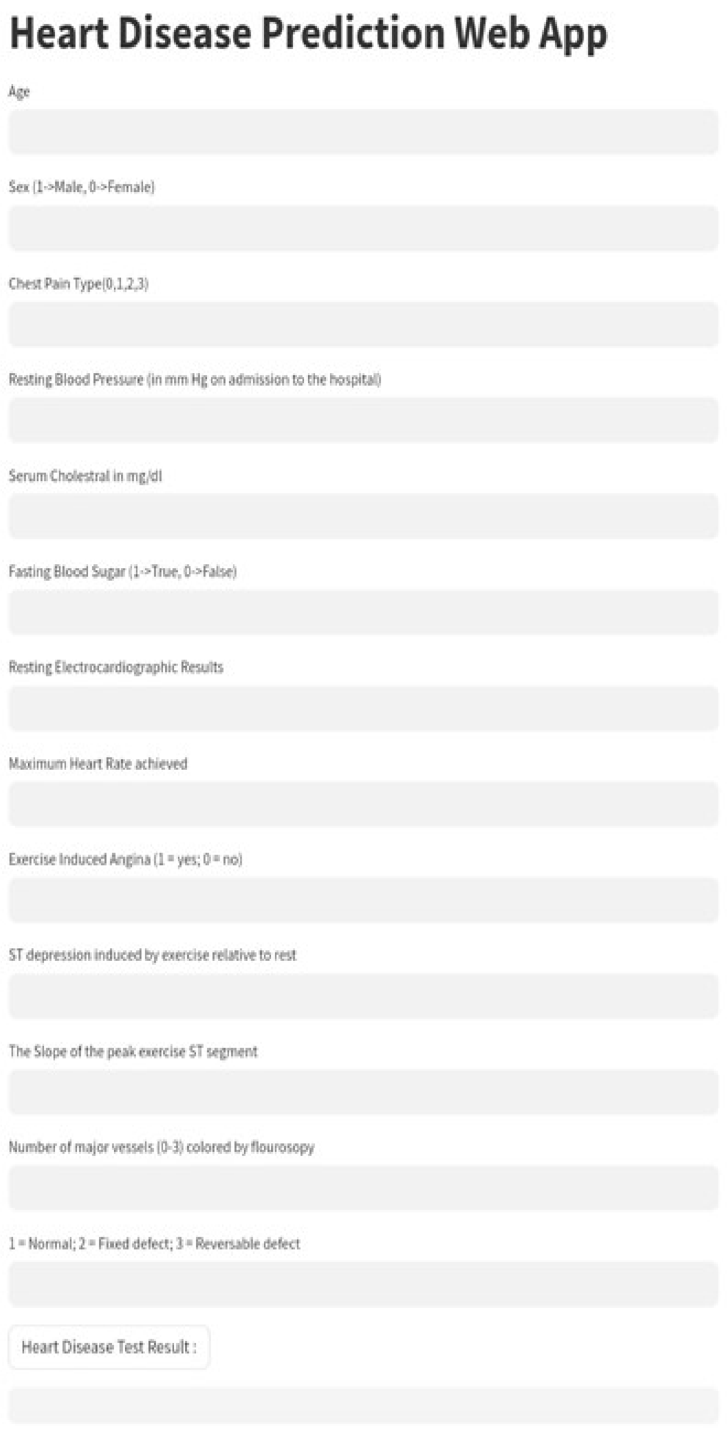
3. Design In this stage Test cases/scripts are created on the basis of software requirement documents.

4.Test Execution In this stage Test Cases prepared are executed. Bugs if any are fixed and re-tested.

CHAPTER-8

SCREENSHOTS

INPUT:



Heart Disease Prediction Web App

Age

Sex (1->Male, 0->Female)

Chest Pain Type(0,1,2,3)

Resting Blood Pressure (in mm Hg on admission to the hospital)

Serum Cholestral in mg/dl

Fasting Blood Sugar (1->True, 0->False)

Resting Electrocardiographic Results

Maximum Heart Rate achieved

Exercise Induced Angina (1 = yes; 0 = no)

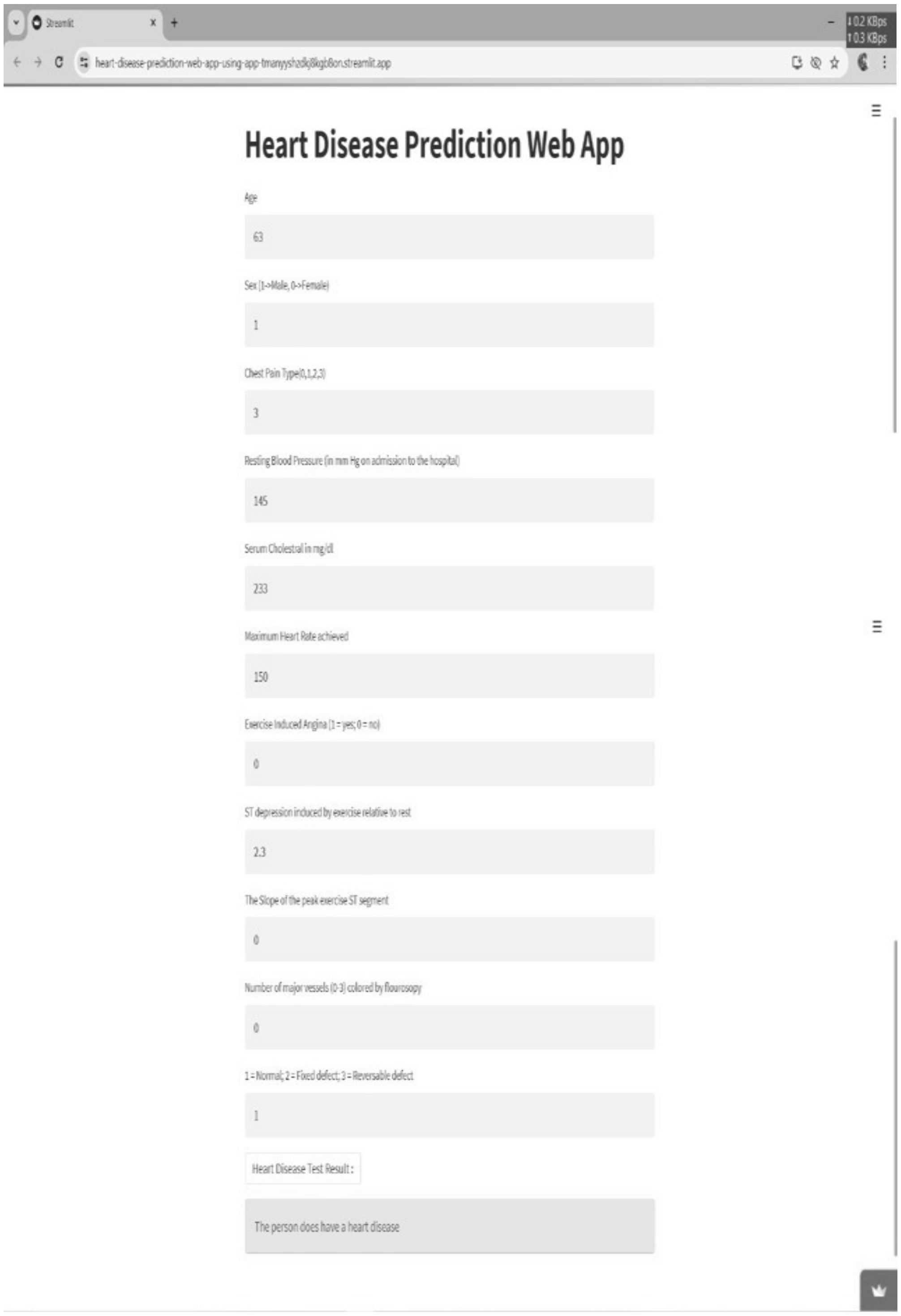
ST depression induced by exercise relative to rest

The Slope of the peak exercise ST segment

Number of major vessels (0-3) colored by flourosopy

1 = Normal; 2 = Fixed defect; 3 = Reversable defect

OUTPUT:


Streamlit
heart-disease-prediction-web-app-using-app-tmanyyshzdkj8kgb8on.streamlit.app
0.2 KBps
0.3 KBps

Heart Disease Prediction Web App

Age

63

Sex (1->Male, 0->Female)

1

Chest Pain Type(0,1,2,3)

3

Resting Blood Pressure (in mm Hg on admission to the hospital)

145

Serum Cholestral in mg/dl

233

Maximum Heart Rate achieved

150

Exercise Induced Angina (1 = yes; 0 = no)

0

ST depression induced by exercise relative to rest

2.3

The Slope of the peak exercise ST segment

0

Number of major vessels (0-3) colored by flourosopy

0

1 = Normal; 2 = Fixed defect; 3 = Reversable defect

1

Heart Disease Test Result :

The person does have a heart disease

CHAPTER-9

CONCLUSION

In this project, we introduce about the heart disease prediction system with different classifier techniques for the prediction of heart disease. The techniques are Random Forest and Logistic Regression: we have analyzed that the Random Forest has better accuracy as compared to Logistic Regression. Our purpose is to improve the performance of the Random Forest by removing unnecessary and irrelevant attributes from the dataset and only picking those that are most informative for the classification task.

CHAPTER-10

REFERENCES

[1] P .K. Anooj, ―Clinical decision support system: Risk level prediction of heart disease using weighted fuzzy rules‖; Journal of King Saud University – Computer and Information Sciences (2012) 24, 27–40. Computer Science & Information Technology (CS & IT) 59

[2] Nidhi Bhatla, Kiran Jyoti"An Analysis of Heart Disease Prediction using Different Data Mining Techniques".International Journal of Engineering Research & Technology

[3] Jyoti Soni Ujma Ansari Dipesh Sharma, Sunita Soni. "Predictive Data Mining for Medical Diagnosis: An Overview of Heart Disease Prediction".

[4] Chaitrali S. Dangare Sulabha S. Apte, Improved Study of Heart Disease Prediction System using Data Mining Classification Techniques" International Journal of Computer Applications (0975 – 888)

[5] Dane Bertram, Amy Voida, Saul Greenberg, Robert Walker, "Communication, Collaboration, and Bugs: The Social Nature of Issue Tracking in Small, Collocated Teams".

[6] M. Anbarasi, E. Anupriya, N.Ch.S.N.Iyengar, ―Enhanced Prediction of Heart Disease with Feature Subset Selection using Genetic Algorithm‖; International Journal of Engineering Science and Technology, Vol. 2(10), 2010.

[7] Ankita Dewan, Meghna Sharma,” Prediction of Heart Disease Using a Hybrid Technique in Data Mining Classification”, 2nd International Conference on Computing for Sustainable Global Development IEEE 2015 pp 704-706. [2].

[8] R. Alizadehsani, J. Habibi, B. Bahadorian, H. Mashayekhi, A. Ghandeharioun, R. Boghrati, et al., "Diagnosis of coronary arteries stenosis using data mining," J Med Signals Sens, vol. 2, pp. 153-9, Jul 2012.