Chapter1

Introduction

The heart is an important organ of the human body. It pumps blood to every part of our anatomy. If it fails to function correctly, then the brain and various other organs will stop working, and within a few minutes, the person will die. Changes in lifestyle, work-related stress, and bad food habits contribute to the increase in the rate of several heart-related diseases. Heart diseases have emerged as one of the most prominent causes of death all around the world. According to the World Health Organization, heart-related diseases are responsible for taking 17.7 million lives every year, 31% of all global deaths. In India too, heart-related diseases have become the leading cause of mortality. Heart diseases have killed 1.7 million Indians in 2016, according to the 2016 Global Burden of Disease Report, released on September 15, 2017. Heart-related diseases increase the spending on health care and reduce the productivity of an individual. Estimates made by the World Health Organization (WHO), suggest that India has lost up to $237 billion, from 2005- 2015, due to heart-related or cardiovascular diseases. Thus, feasible and accurate prediction of heart-related diseases is very important.

Medical organizations, all around the world, collect data on various health-related issues. This data can be exploited using various machine-learning techniques to gain useful insights. But the data collected is very massive and, many times, this data can be very noisy. These datasets, which are too overwhelming for human minds to comprehend, can be easily explored using various machine-learning techniques. Thus, these algorithms have become very useful, in recent times, to predict the presence or absence of heart-related diseases accurately the usage of information technology in the healthcare industry is increasing day by day to aid doctors in decision-making activities. It helps doctors and physicians in disease management, medications, and the discovery of patterns and relationships among diagnosis data. Current approaches to predicting cardiovascular risk fail to identify many people who would benefit from preventive treatment. Machine learning offers an opportunity to improve accuracy by exploiting complex interactions between risk factors.

1.1 Project Scope

The major challenge in heart disease is its detection. There are instruments available that can predict heart disease but either they are expensive or is not efficient to calculate the chance of heart disease in humans. Early detection of cardiac diseases can decrease the mortality rate and overall complications. However, it is not possible to monitor patients every day in all cases accurately and consultation of a patient for 24 hours by a doctor is not available since it requires more sapience, time, and expertise. Since we have a good amount of data in today's world, we can use various machine-learning algorithms to analyze the data for hidden patterns. The hidden patterns can be used for health diagnosis in medicinal data.

Chapter 2

Literature Survey

2.1 Introduction to Literature Survey

A few approaches for the prediction and diagnosis of heart disease.

Prediction and analysis of the occurrence of heart disease using data mining techniques

ChalaBeyene et al [1], recommended Prediction and Analysis of the occurrence of Heart Disease Using Data Mining Techniques. The main objective is to predict the occurrence of heart disease for early automatic diagnosis of the disease within result in a short time. The proposed methodology is also critical in a healthcare organization with experts that have no more knowledge and skill. It uses different medical attributes such as blood sugar and heart rate, age, and sex are some of the attributes are included to identify if the person has heart disease or not. Analyses of the dataset are computed using WEKA software.

Hybrid random forest linear method (HRFLM) machine learning for heart disease prediction

Senthilkumar Mohan et al [2], implemented hybrid machine learning for heart disease prediction. The data set used is the Cleveland data set. They have proposed their own Hybrid Random Forest Linear Method (HRFLM) which is the combination of Random Forest (RF) and Linear method (LM). In the HRFLM algorithm, the authors have used four algorithms. The first algorithm deals with partitioning the input dataset. It is based on a decision tree which is executed for each sample of the dataset. After identifying the feature space, the dataset is split into leaf nodes. The output of the first algorithm is the Partition of the data set. After that in the second algorithm, they apply rules to the data set and the output here is the classification of data with those rules. In the third algorithm, features are extracted using Less Error Classifier. This algorithm deals with finding the minimum and maximum error rates from the classifier. The output of this algorithm is the features with classified attributes. In the fourth algorithm, they apply a Classifier which is a hybrid method based on the error rate of the Extracted Features. Finally, they compared the results obtained after applying HRFLM with other classification algorithms such as a decision tree and support vector machine. As a result, as RF and LM are giving better results than others, both the algorithms are put together and a new unique algorithm HRFLM is created.

System containing two models based on linear support vector machine for prediction of heart disease

Ali, Liaqat, et al [3], propose a system containing two models based on a linear Support Vector Machine (SVM). The first one is called L1 regularized and the second one is called L2

regularized. The first model is used for removing unnecessary features by making the coefficient of those features zero. The second model is used for prediction. Prediction of disease is done in this part. To optimize both models they proposed a hybrid grid search algorithm. This algorithm optimizes two models based on metrics: accuracy, sensitivity, septicity, the Matthews correlation coefficient, ROC chart, and area under the curve. They used the Cleveland data set. Data split into 70% training and 30% testing using holdout validation. There are two experiments carried out and each experiment is carried out for various values of C1, C2, and k where C1 is the hyperparameter of the L1 regularized model, C2 is the hyperparameter of the L2 regularized model and k is the size of the selected subset of features. The first experiment is the L1-linear SVM model stacked with the L2-linear SVM model which gives maximum testing accuracy of 91.11% and training accuracy of 84.05%. The second experiment is L1- the linear SVM model cascaded with the L2-linear SVM model with RBF kernel. This gives maximum testing accuracy of 92.22% and a training accuracy of 85.02. They have obtained an improvement in accuracy over conventional SVM models by 3.3%.Various supervised machine learning algorithms for the prediction of heart disease.

Various supervised machine learning algorithms for the prediction of heart disease

Singh, Yeshvendra K. et al [4], deal with various supervised machine learning algorithms such as Random Forest, Support Vector Machine, Logistic Regression, Linear Regression, and Decision Tree with 3-fold, 5-fold, and 10-fold cross-validation techniques. They have used the Cleveland data set. In the pre-processing of data, they just removed the null values and duplicates and then they divided the data as training 70% and testing 30%. They applied all the algorithms and got the highest accuracy for Logistic regression. Here are the accuracies that have been obtained for every algorithm: Random Forest - 82.16%, Linear Regression - 82%, Logistic regression - 83.83%, SVM - 83.17%, and Decision tree - 79.54% with 5-fold.

Feature selection for medical diagnosis: Evaluation for cardiovascular diseases

C.-L. Chang and C.-H. Chen, et al [5], came up with an objective to predict more accurately the presence of cardiovascular disease with a reduced number of attributes. They proposed a hybrid forward selection technique for cardiovascular disease diagnosis. This experiment demonstrates that this approach finds smaller subsets and increases the accuracy of diagnosis compared to forward inclusion and back-elimination techniques.

They investigate intelligent systems to generate feature subsets with improvements in diagnostic performance. Features ranked with distance measures are searched through forward inclusion, forward selection, and backward elimination search techniques to find a subset that improves classification results.

An efficient approach for classifying and predicting heart disease using machine learning techniques

S. Shilaskar and A.Ghatol et al [6], applied different classification methods such as KNN, Random Forest & Naïve Bayes to original data sets as well as on datasets with feature selection methods. They used Heart Disease Data Set which databases bases namely Cleveland, Hungary, Switzerland and the VA Long Beach. All these processes are applied on first three different Heart Disease Datasets to analyses the performance of effect of preprocessing in terms of accuracy rate. They have used two methods the first one is "Filtering method" – it applies the data preprocessing on the databases, the second method "Recursive Feature Elimination" - gives the subset of features which gives accurate result. A random forest algorithm is used on each iteration to evaluate the model. The algorithm is configured to explore all possible subsets of the attributes. The accuracies of all the three algorithms during the first method on the first three databases in the heart disease dataset are as follows, KNN – 93%, 32%, 65%, Random forest 54%, 27%, 72%, Naive Bayes 99%, 84%, 97% for Cleveland, Hungary, Switzerland respectively and during the second method : KNN –96%, 90%, 94%, Random forest 57%, 44%, 82%, Naive Bayes 100%, 92%, 100% for Cleveland, Hungary, Switzerland respectively.

Diagnosis prediction via Recurrent Neural Networks

Yangzi Mu, Mengxing Huang et al [7], used the Electronic health records(EHR) data. It consists of patient health data, including demographics, diagnoses, procedures, and medications. EHR data are temporally sequenced by patient medical visits that are represented by a set of high dimensional clinical variables. To model the sequential EHR data, recurrent neural networks (RNNs) are used in the literature to obtain accurate and robust representations of patient visits in diagnostic predictive tasks.

They performed several experiments & the best parameters are further adjusted by experience and individually optimized for each parameter. They choose Adam [24] as the optimization. method. In terms of deep network architecture, the number of layers is set to 2 in our experiments with 128 hidden units in each layer. The dropout rate is set to 0.4 and the regularization parameter λ , 0.01. Other parameters are uniformly initialized between [-0.03, 0.03].

Risk prediction of cardiovascular disease using machine learning classifiers

Madhumita Pal, Smita Parija et al [8], In this study, they used two reliable machine learning techniques, multi-layer perceptron (MLP), and *K*-nearest neighbor (K-NN) have been employed for cardiovascular disease (CVD) detection using publicly available University of California Irvine repository data. The performance of the models are optimally increased by removing outliers and attributes having null values. Experimental-based results demonstrate

that a higher accuracy in detection of 82.47% and an area-under-the-curve value of 86.41% are obtained using the MLP model, unlike the K-NN model. Therefore, the proposed MLP model was recommended for automatic CVD detection. The proposed methodology can also be employed in detecting other diseases. In addition, the performance of the proposed model can be accessed via other standard data sets.

2.2 Existing System

Heart disease is even being highlighted as a silent killer which leads to the death of a person without obvious symptoms. The nature of the disease is the cause of growing anxiety about the disease and its consequences. Hence continued efforts are being made to predict the possibility of this deadly disease in the future. So various tools and techniques are regularly being experimented with to suit present-day health needs.

Machine Learning techniques can be a boon in this regard. Even though heart disease can occur in different forms, there is a common set of core risk factors that influence whether someone will ultimately be at risk for heart disease or not. By collecting the data from various sources, classifying them under suitable headings & finally analyzing them to extract the desired data we can conclude. This technique can be very well adapted to the do the prediction of heart disease. As the well-known quote says, “Prevention is better than cure”, early prediction & its control can be helpful to prevent & decrease the death rates due to heart disease.

Chapter 3

Requirement Specification

3.1 Software Requirements:

Operating System : Windows 7 and above

Languages : Python 3.9 & above

IDE : Anaconda Navigator

3.2 Hardware Requirements:

Processor : 32 or 64-bit, i3 and above dual-core processer

Ram : 4GB and Above

Disk Space : Minimum 10 GB disk space for Anaconda

3.3 Libraries Requirement

NumPy:

NumPy can be used to perform a wide variety of mathematical operations on arrays. It adds powerful data structures to Python that guarantee efficient calculations with arrays and matrices, and it supplies an enormous library of high-level mathematical functions that operate on these arrays and matrices.

Pandas:

Pandas is an open-source Python package that is most widely used for data science/data analysis and machine learning tasks. Pandas allow us to analyze big data and make conclusions based on statistical theories. Pandas can clean messy data sets and make them readable and relevant. Relevant data is very important in data science.

Matplotlib:

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible. Export to many file formats.

Hvplot:

It is a powerful graphical plotting program for scientists and engineers. It is used to visualize, analyze, and compare data from a variety of sources, including measurements, simulation results, and even images. HvPlot offers a wide range of features and options that make it a powerful tool for quickly and easily creating high-quality plots.

PyTorch:

It is an open-source machine learning library based on the Torch library, used for applications such as deep learning and computer vision. It provides an easy-to-use API to work with a variety of deep learning models. PyTorch is easy to use, efficient, and can be used for a wide range of applications**.**

IPyWidget

IPyWidgets is a Python library of HTML interactive widgets for Jupyter notebook. Each UI element in the library can respond to events and invokes specified event handler functions. They enhance the interactive feature of Jupyter notebook application.

Scikit-Learn

It is the. most useful library for machine learning in Python. The Sklearn library contains a lot of efficient tools for machine learning and statistical modeling including classification, regression, clustering, and dimensionality reduction.

Chapter 4

Requirement Analysis

4.1 Machine Learning

In machine learning, classification refers to a predictive modeling problem where a class label is predicted for a given example of input data.

4.1.1 Supervised Learning

Supervised Learning is the type of machine learning in which machines are trained using well "labeled" training data, and based on that data, machines predict the output. The labeled data means some input data is already tagged with the correct output. In supervised learning, the training data provided to the machines work as the supervisor that teaches the machines to predict the output correctly. It applies the same concept as a student learning under the supervision of the teacher. Supervised learning is a process of providing input data as well as correct output data to the machine learning model. The aim of a supervised learning algorithm is to find a mapping function to map the input variable(x) with the output variable(y).

4.1.2 Unsupervised Learning

Unsupervised learning cannot be directly applied to a regression or classification problem because, unlike supervised learning, we have the input data but no corresponding output data. The goal of unsupervised learning is to find the underlying structure of the dataset, group that data according to similarities, and represent that dataset in a compressed format.

• Unsupervised learning is helpful for finding useful insights from the data.

• Unsupervised learning is much like how a human learns to think by their own experiences, which makes it closer to real AI.

• Unsupervised learning works on unlabeled and uncategorized data which makes unsupervised learning more important.

• In the real world, we do not always have input data with the corresponding output so to solve such cases, we need unsupervised learning.

4.1.3 Reinforcement Learning

Reinforcement learning is an area of Machine Learning. It is about taking suitable action to maximize reward in a particular situation. It is employed by various software and machines to find the best possible behavior or path it should take in a specific situation. Reinforcement learning differs from supervised learning in a way that in supervised learning the training data has the answer key with it, so the model is trained with the correct answer itself whereas, in reinforcement learning, there is no answer, but the reinforcement agent decides what to do to perform the given task.

Chapter 5
System Design
5.1 UML Diagrams

UML stands for Unified Modeling Language. UML is a standardized general purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying. Visualization, Constructing and documenting the artifacts of software systems, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. UML is a very important part of developing object-oriented software and the software. development process. UML uses mostly graphical notations to express the design of software projects.

Goals:

The Primary goals in the design of the UML are as follows:

1) Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2) Provide extendibility and specialization mechanisms to extend the core concepts.
3) Be independent of programming languages and development process.
4) Provide a formal basis for understanding the modeling language.
5) Encourage the growth of OO tools market.
6) Support higher level development concepts such as collaborations, frameworks, patterns and components and integrate best practices.

5.1.1 Use Case Diagram

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

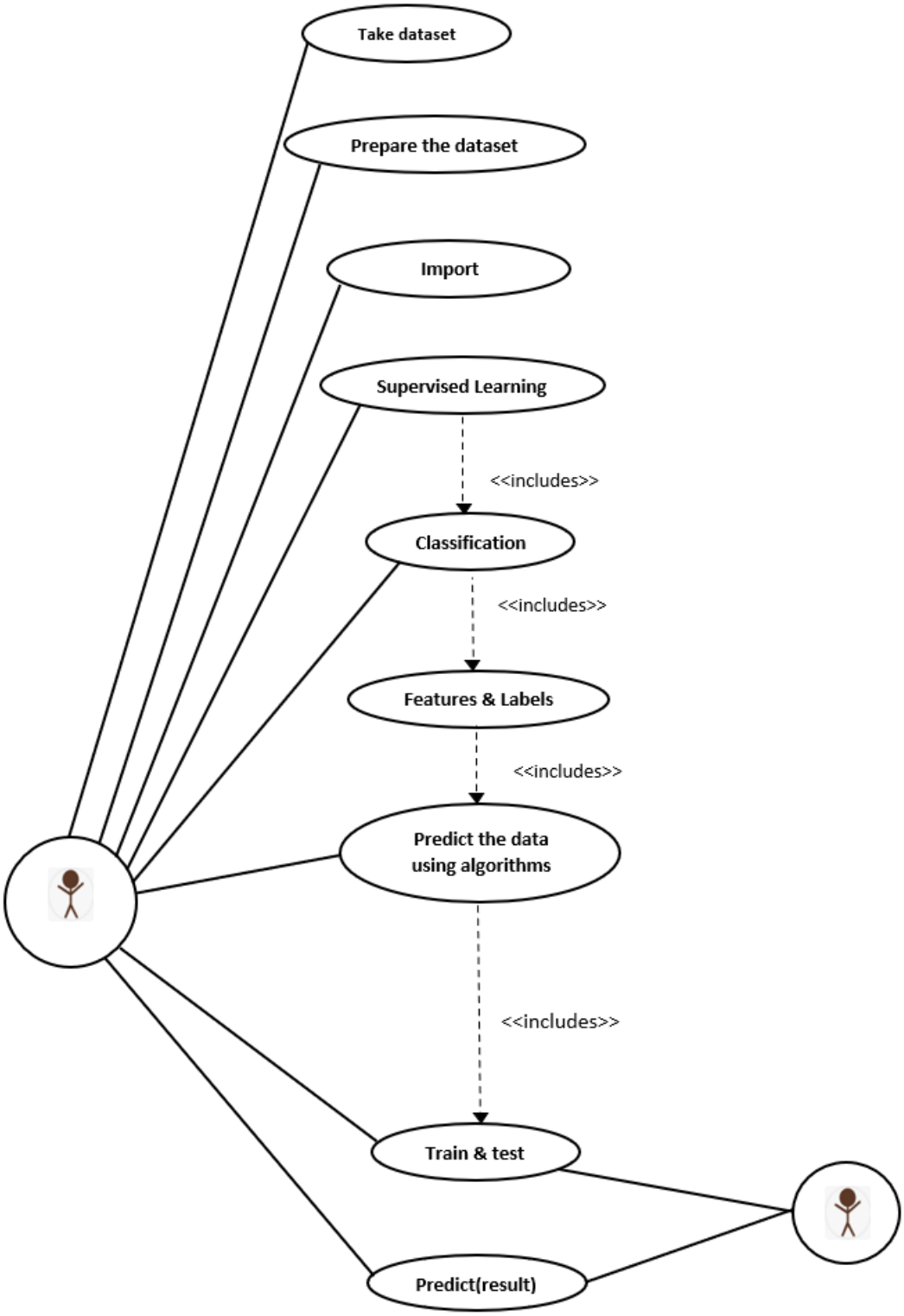


Figure 5.1: Use case Diagram

5.1.2 Activity Diagram

Activity diagram is another important behavioral diagram in UML diagram to describe dynamic aspects of the system. Activity diagram is essentially an advanced version of flow chart that modeling the flow from one activity to another activity.

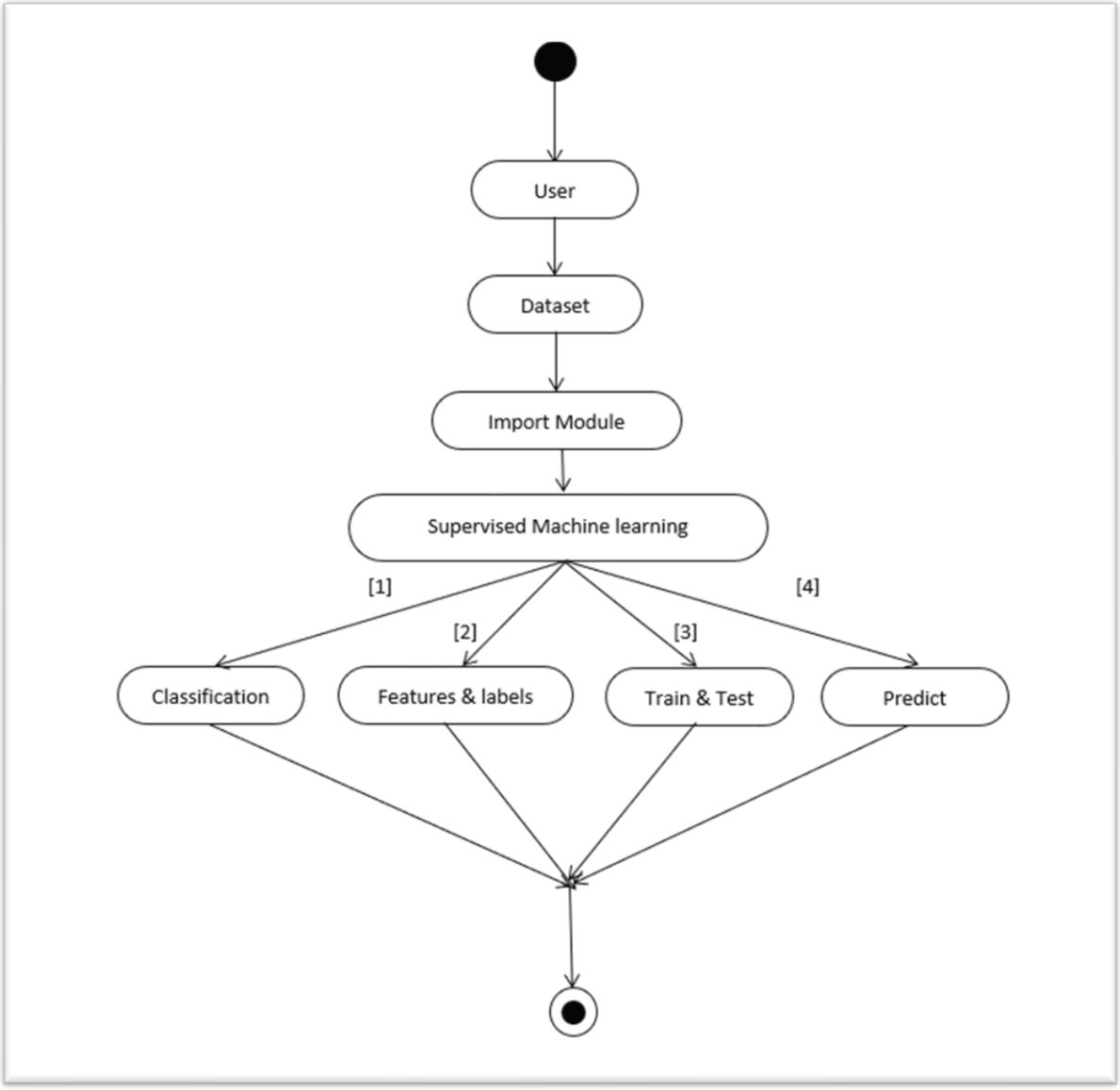


Figure 5.2: Activity Diagram

Chapter 6

System 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, sub-assemblies, 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.

6.1 Types of Tests

6.1.1 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 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. Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

6.1.2 Integration Testing

Integration tests are designed to test integrated software components to determine if they run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, 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. Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g., components in a software system or – one step up – software applications at the company level – interact without error.

6.1.3 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. Functionals 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 experienced.

Systems/Procedures: Interfacing systems or procedures must be invoked. Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows, data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

6.1.4 System Testing

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is configuration-oriented system integration test. System testing is based on process description and flows, emphasizing pre-driven process links and integration points.

6.1.5 Module Testing

Module testing is defined as a software testing type, which checks individual subprograms, subroutines, classes, or procedures in a program. Instead of testing the whole software program at once, module testing recommends testing the smaller building blocks of the program.

6.1.6 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Chapter 7

Project Planning

7.1 Project Overview

Heart-related diseases or cardiovascular diseases (CVDs) are the main reason for a huge number of deaths in the world over the last few decades and have emerged as the most life-threatening disease, not only in India but in the whole world. So, there is a need for a reliable, accurate, and feasible system to diagnose such diseases in time for proper treatment. Machine Learning algorithms and techniques have been applied to various medical datasets to automate the analysis of large and complex data. Many researchers, in recent times, have been using several machine learning techniques to help the healthcare industry and professionals in the diagnosis of heart-related diseases. The heart is the next major organ compared to the brain which has more priority in the Human body. It pumps the blood and supplies it to all organs of the whole body. Prediction of occurrences of heart diseases in the medical field is significant work.

Data analytics is useful for predicting from more information and it helps the medical center to predict various diseases. A huge amount of patient-related data is maintained monthly. The stored data can be useful for the source of predicting the occurrence of future diseases. Some of the data mining and machine learning techniques are used such as Random Forest, Support Vector Machine (SVM) and TabNet model. Prediction and diagnosing heart disease become a challenging factor faced by doctors and hospitals in India and abroad. To reduce the large scale of deaths from heart diseases, a quick and efficient detection technique is to be discovered. Data mining techniques and machine learning algorithms are very important in this area. The researchers accelerate their research works to develop software with the help of machine learning algorithms which can help doctors to decide both prediction and diagnosis of heart disease. The main objective of this research project is to predict a patient's heart disease using machine learning algorithms.

7.2 Proposed System

The working of the system starts with the collection of data and selecting the important attributes. Then the required data is preprocessed into the required format. The data is then divided into two parts training and testing data. The algorithms are applied, and the model is trained using the training data. The accuracy of the system is obtained by testing the system using the testing data. This system is implemented using the following modules.

1.Collection of Dataset

2.Selection of attributes

3.Data Pre-Processing

4.Balancing of Data

5.Disease Prediction

Chapter 8

Implementation

8.1 Problem Statement

Detecting heart disease with proper accuracy is a challenging task. Our goal is to go through different algorithms and find out which one gives better accuracy.

Given a set of patient medical data, the task is to predict whether a patient is at risk of having or developing heart disease. Detecting a heart patient with one model is not that quite difficult but training multiple models according to our needs and get high accuracy is quite challenging. Training our model with database is comparatively not that challenging but when we finish building our model and after that check to see new patients with our input data to find out whether that person has a heart disease or not that evaluation is difficult.

8.2 System Architecture

The system architecture gives an overview of the working of the system. The working of this system is described as follows: Dataset collection is collecting data that contains patient details. The attributes selection process selects useful attributes for the prediction of heart disease. After identifying the available data resources, they are further selected, cleaned, and made into the desired form. Different classification techniques as stated will be applied to preprocessed data to predict the accuracy of heart disease. The accuracy measure compares the accuracy of different classifiers.

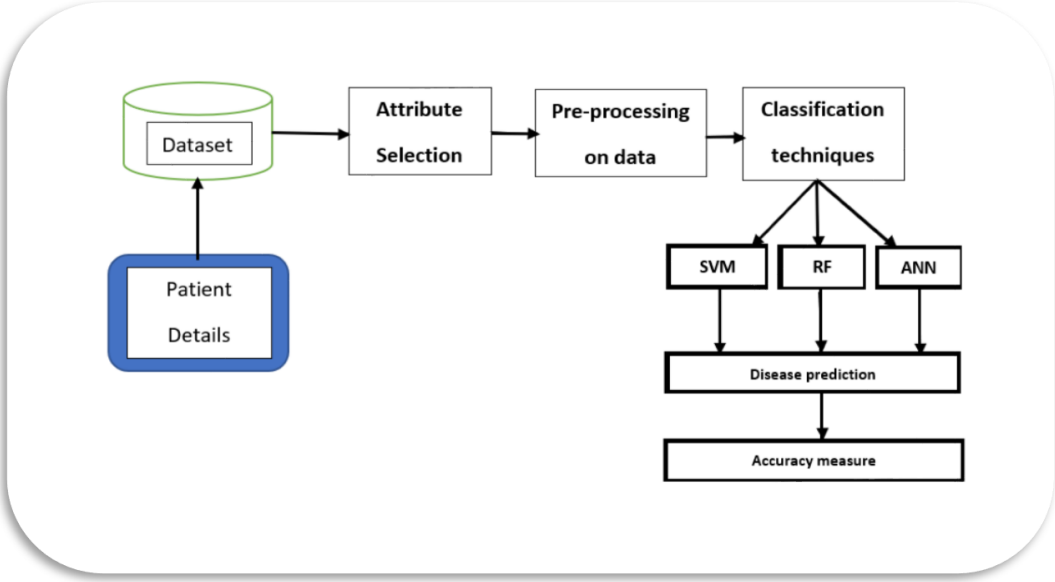


Figure 8.1: System Architecture

8.3 System Flow

8.3.1 Collection Of dataset

Initially, we collected a dataset for our heart disease prediction system. After the collection of the dataset, we split the dataset into training data and testing data. The training dataset is used for prediction model learning and testing data is used for evaluating the prediction model. For this project, 80% of training data is used and 20% of the data is used for testing. The dataset used for this project is Heart Disease UCI. The dataset consists of 14 attributes, which were used in this project.

8.3.2 Selection of attributes

Attribute or Feature selection includes the selection of appropriate attributes for the prediction system. This is used to increase the efficiency of the system. Various attributes of the patient like gender, chest pain type, fasting blood sugar, serum cholesterol, exang, electro cardio results, etc are selected for the prediction.

8.3.3 Preprocessing of data

Data pre-processing is an important step for the creation of a machine learning model. Initially, data may not be clean or in the required format for the model which can cause misleading outcomes. In pre-processing of data, we transform data into our required format. It is used to deal with noises, duplicates, and missing values of the dataset. Data pre-processing has activities like importing datasets, splitting datasets, attribute scaling, etc. Preprocessing of data is required for improving the accuracy of the model.

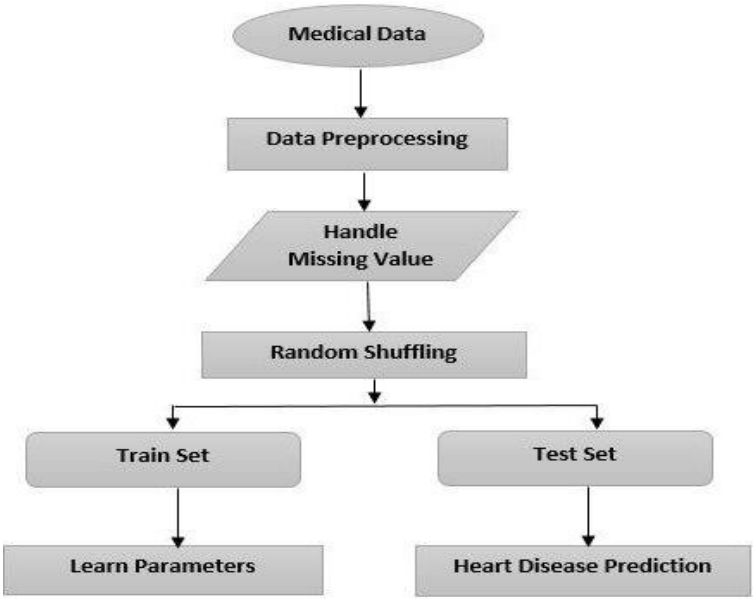


Figure 8.2: Preprocessing of data

8.3.4 Prediction of Disease

Various machine learning algorithms like SVM, Random Tree, TabNet are used for classification. Comparative analysis is performed among algorithms and the algorithm that gives the highest accuracy is used for heart disease prediction.

8.4 Dataset Details

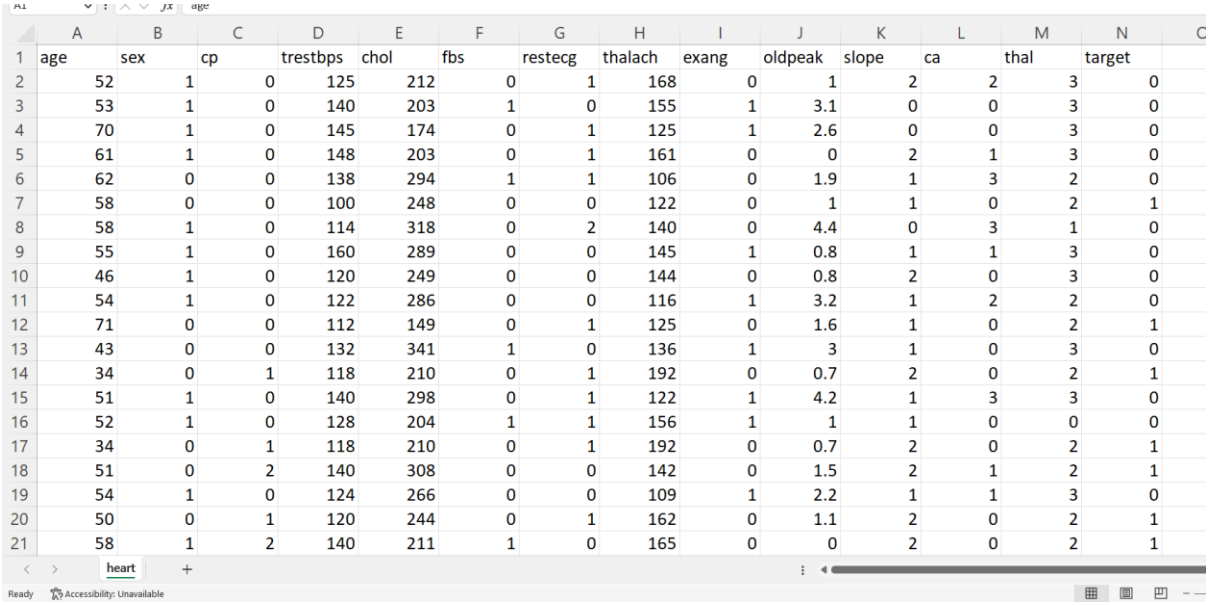
The Cleveland heart dataset from the UCI Machine learning repository has been used for the experiments [6]. The dataset consists of 14 attributes and 1025 records. There are 8 categorical attributes and 6 numeric attributes. The description of the dataset is shown in the table.

Feature information of the cleveland dataset.

S.No	Attribute Name	Description	Range of Values
1	Age	Age of the person in years	29 to 79
2	Sex	Gender of the person [1: Male, 0: Female]	0, 1
3	Cp	Chest pain type [1-Typical Type 1 Angina 2- Atypical Type Angina 3-Non-angina pain 4-Asymptomatic)	1, 2, 3, 4
4	Trestbps	Resting Blood Pressure in mm Hg	94 to 200
5	Chol	Serum cholesterol in mg/dl	126 to 564
6	Fbs	Fasting Blood Sugar in mg/dl	0, 1
7	Restecg	Resting Electrocardiographic Results	0, 1, 2
8	Thalach	Maximum Heart Rate Achieved	71 to 202
9	Exang	Exercise Induced Angina	0, 1
10	OldPeak	ST depression induced by exercise relative to rest	1 to 3
11	Slope	Slope of the Peak Exercise ST segment	1, 2, 3
12	Ca	Number of major vessels colored by fluoroscopy	0 to 3
13	Thal	3 – Normal, 6 – Fixed Defect, 7 – Reversible Defect	3, 6, 7
14	Num	Class Attribute	0 or 1

Figure 8.3: Dataset

Dataset link: https://www.kaggle.com/datasets/johnsmith88/heart-disease-dataset



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
2	52	1	0	125	212	0	1	168	0	1	2	2	3	0
3	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
4	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
5	61	1	0	148	203	0	1	161	0	0	2	1	3	0
6	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0
7	58	0	0	100	248	0	0	122	0	1	1	0	2	1
8	58	1	0	114	318	0	2	140	0	4.4	0	3	1	0
9	55	1	0	160	289	0	0	145	1	0.8	1	1	3	0
10	46	1	0	120	249	0	0	144	0	0.8	2	0	3	0
11	54	1	0	122	286	0	0	116	1	3.2	1	2	2	0
12	71	0	0	112	149	0	1	125	0	1.6	1	0	2	1
13	43	0	0	132	341	1	0	136	1	3	1	0	3	0
14	34	0	1	118	210	0	1	192	0	0.7	2	0	2	1
15	51	1	0	140	298	0	1	122	1	4.2	1	3	3	0
16	52	1	0	128	204	1	1	156	1	1	1	0	0	0
17	34	0	1	118	210	0	1	192	0	0.7	2	0	2	1
18	51	0	2	140	308	0	0	142	0	1.5	2	1	2	1
19	54	1	0	124	266	0	0	109	1	2.2	1	1	3	0
20	50	0	1	120	244	0	1	162	0	1.1	2	0	2	1
21	58	1	2	140	211	1	0	165	0	0	2	0	2	1

Figure 8.4: Samples of the dataset.

8.5 Algorithms

8.5.1 Support Vector Machine

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning. The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane. SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called support vectors, and hence the algorithm is termed as Support Vector Machine. Support vector machines (SVMs) are powerful yet flexible supervised machine learning algorithms that are used both for classification and regression. But generally, they are used in classification problems. In the 1960s, SVMs were first introduced but later they got refined in 1990. SVMs have their unique way of implementation as compared to other machine learning algorithms. Lately, they are extremely popular because of their ability to handle multiple continuous and categorical variables. The following are important concepts in SVM - Support Vectors - Data Points that are closest to the hyperplane are called support vectors. Separating lines will be defined with the help of these data points. Hyperplane - As we can see in the above diagram, it is a decision plane or space which is divided between a set of objects having different classes. Margin - It may be defined as the gap between two lines on the closest data points of different classes. It can be calculated as the perpendicular distance from the line to the support vectors. A large

margin is considered as a good margin and a small margin is considered as a bad margin.

Types of SVM: There are two types.

Linear SVM:

Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data and the classifier is used called Linear SVM classifier. For linearly non-separable data the input is mapped to high-dimensional feature space where they can be separated by a hyperplane. This projection into high-dimensional feature space is efficiently performed by using kernels. More precisely, given a set of training samples and the corresponding decision values -1, 1 the SVM aims to find the best-separating hyperplane.

Non-linear SVM:

Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as nonlinear data, and the classifier used is called a Non-linear SVM classifier. The objective of the support vector machine algorithm is to find a hyperplane in N-dimensional space (N - the number of features) that distinctly classifies the data points.

The advantages of support vector machines are:

- Effective in high-dimensional spaces.
- Still effective in cases where the number of dimensions is greater than the number of samples.
- Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
- Versatile: different kernel functions can be specified for the decision function. Common kernels are provided, but it is also possible to specify custom kernels.

The disadvantages of support vector machines include:

If the number of features is much greater than the number of samples, avoiding overfitting in choosing Kernel functions and the regularization term is crucial. SVMs do not directly provide probability estimates, these are calculated using an expensive five-fold cross-validation.

8.5.2 Random Forest:

Random Forest is a supervised learning algorithm. It is an extension of machine learning classifiers which include the bagging to improve the performance of the Decision Tree. It combines tree predictors, and trees are dependent on a random vector that is independently sampled. The distribution of all trees is the same. Random Forests splits nodes using the best among of a predictor subset that are randomly chosen from the node itself, instead of splitting nodes based on the variables. It can be used both for classification and regression. It is also the most flexible and easy-to-use algorithm. A forest consists of trees. It

is said that the more trees it has, the more robust a forest is [2]. Random Forests create Decision Trees on randomly selected data samples, get predictions from each tree and select the best solution by means of voting. It also provides a pretty good indicator of the feature's importance. Random Forests have a variety of applications, such as recommendation engines, image classification, and feature selection. It can be used to classify loyal loan applicants, identify fraudulent activity, and predict diseases. It lies at the base of the Boruta algorithm, which selects important features in a dataset.

It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and improve the performance of the model. As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset [4]." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions and predicts the final output. The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

Assumptions:

Since the random forest combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not. But together, all the trees predict the correct output. Therefore, below are two assumptions for a better Random Forest classifier.

There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result. And the predictions from each tree must have very low correlations.

Algorithm Steps:
Step 1: Select random samples from a given data or training set.
Step 2: This algorithm will construct a decision tree for every training data.
Step 3: Voting will take place by averaging the decision tree.
Step 4: Finally, select the most voted prediction result as the final prediction result.

Advantages:

- Random Forest can perform both Classification and Regression tasks.
- It is capable of handling large datasets with high dimensionality.
- It enhances the accuracy of the model and prevents the overfitting issue.

Disadvantages:
Although Random Forest can be used for both classification and regression tasks, it is not more suitable for Regression tasks.

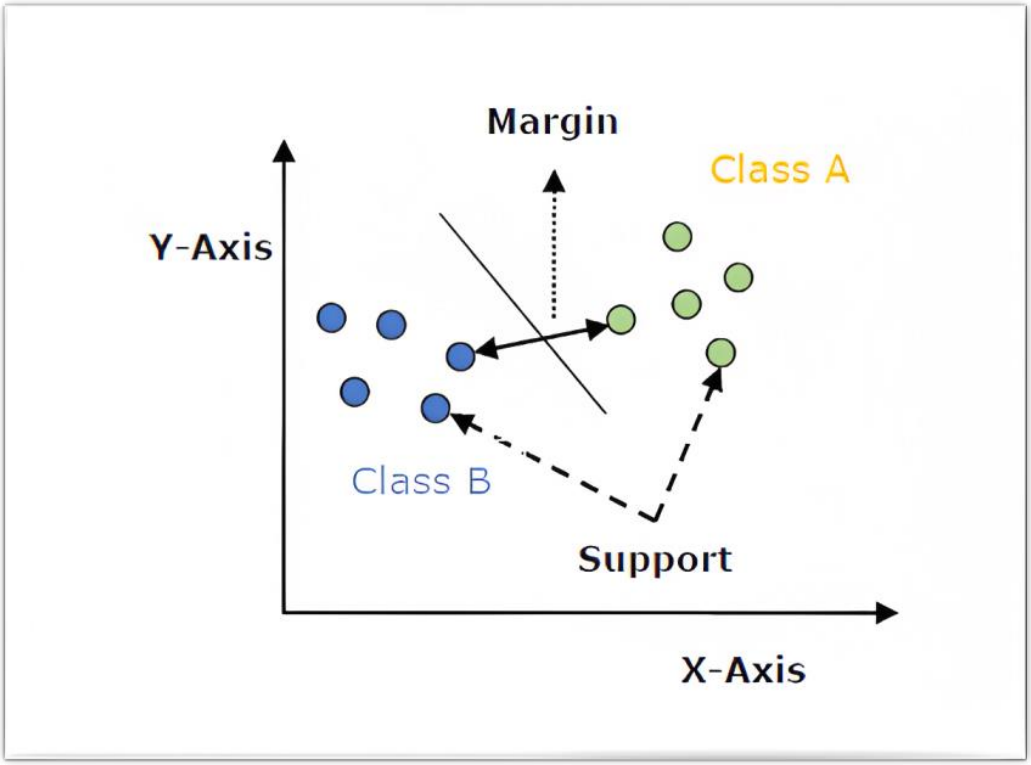


Figure 8.5: Support Vector Machine

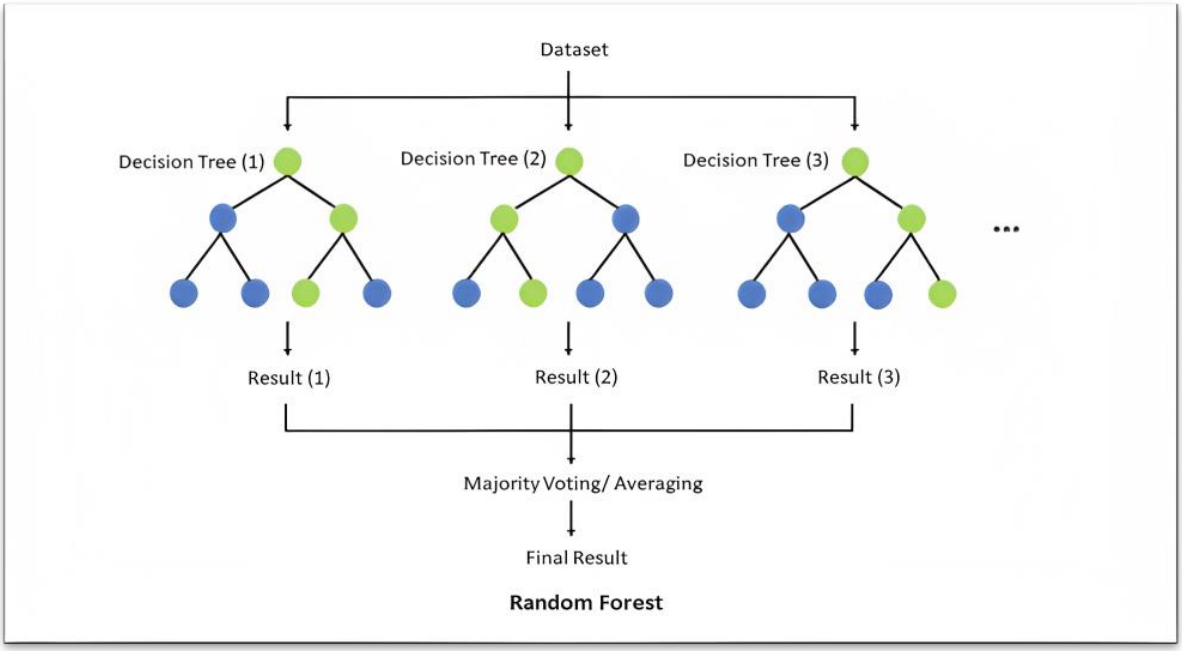


Figure 8.6: Random Forest

8.5.3 TabNet Model

TabNet mimics the behavior of decision trees using the idea of Sequential Attention. Simplistically speaking, you can think of it as a multi-step neural network that applies two key operations at each step:

1. An Attentive Transformer selects the most important features to process at the next step.
2. A Feature Transformer processes the features into a more useful representation.

The output of the Feature Transformer is later used in the prediction. Using both Attentive and Feature Transformers, TabNet can simulate the decision-making process of tree-based models.

TabNet encoder, composed of a feature transformer, an attentive transformer and feature masking. A split block divides the processed representation to be used by the attentive transformer of the subsequent step as well as for the overall output. For each step, the feature selection mask provides interpretable information about the model's functionality, and the masks can be aggregated to obtain global feature important attribution.

Feature Blocks

Feature Blocks consist of sequentially applied Fully Connected (FC) (or Dense) layers and Batch Normalization (BN).
In addition, for Feature Transformers the output gets passed through the GLU activation layer.

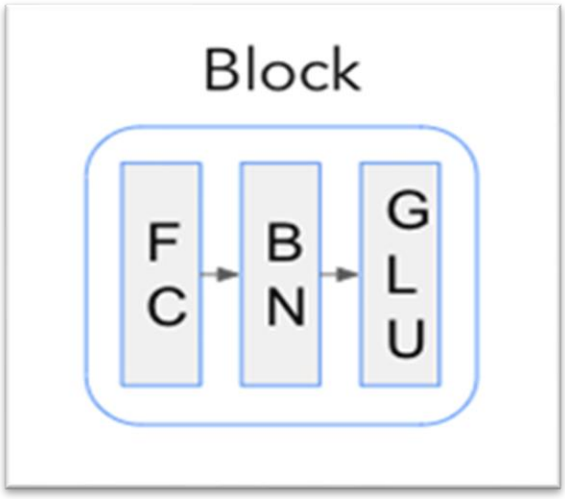


Figure 8.7: Feature Block

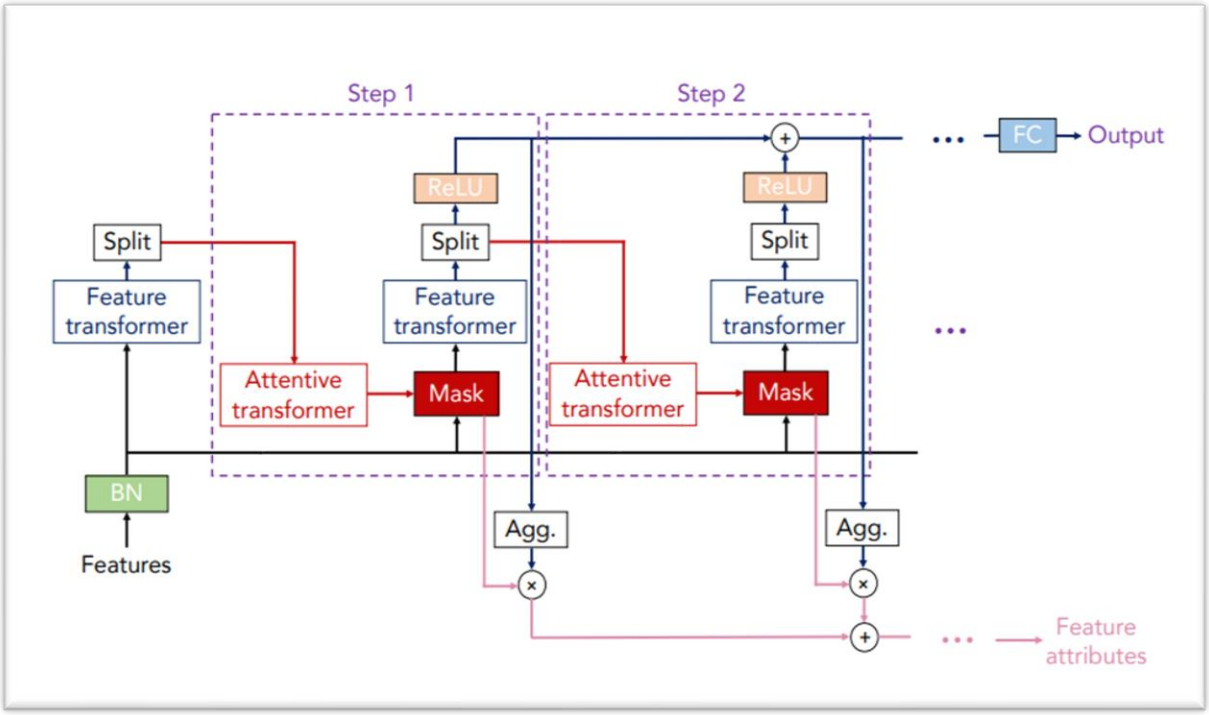


Figure 8.8: Structure of TabNet algorithm

A feature transformer block example – 4-layer network is shown, where 2 are shared across all decision steps and 2 are decision step-dependent. Each layer is composed of a fully connected (FC) layer, BN and GLU nonlinearity.

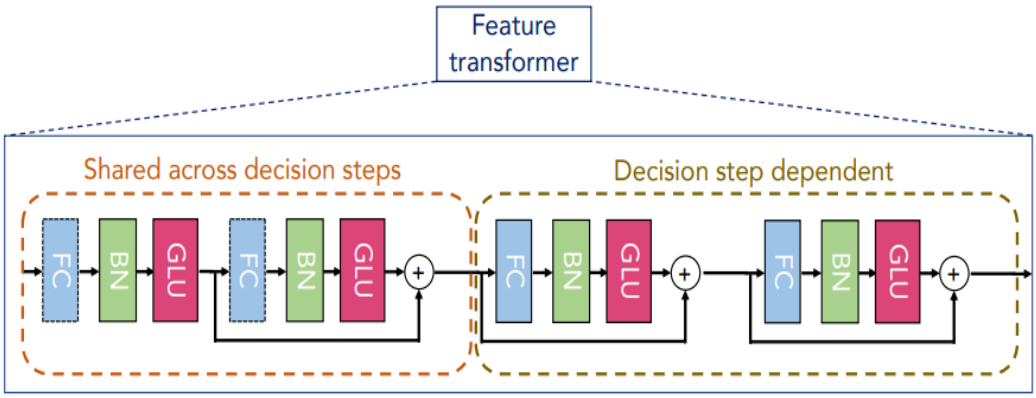


Figure 8.9: Feature transformer layer

Batch Normalization

Batch normalization is a technique used to normalize the input layer of a neural network. It helps to reduce the internal covariate shift, which is the change in the distribution of the input layer of a neural network due to the different layers of the network. This helps to prevent the overfitting of the model. In the TabNet model, batch normalization is used to normalize the input layers and to reduce the need for regularization. This helps to improve the overall performance of the model.

Feature Transformer

The output from a TabNet model after using the split method will depend on the specific task it is being used for. Generally, this method can be used to identify important features within the data, allowing for better prediction accuracy. It can also be used to help identify correlations between different variables and can help to reduce the complexity of the model.

Attentive Transformer

The Attentive Transformer in TabNet is a component of the model which allows it to learn feature importance and automatically select the most relevant features during training. This allows the model to not only learn the most relevant features, but also to use them in the most effective way to improve the model's performance. The attention mechanism of the transformer also helps the model to better learn the correlations between the features, which makes it easier to generalize the model across different datasets.

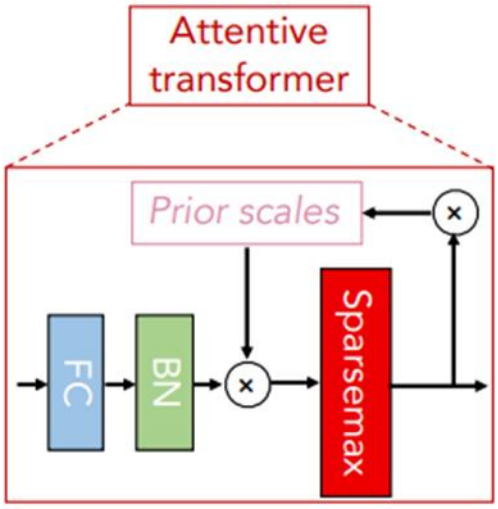


Figure 8.10: Attentive Transformer layer

Mask

The mask in TabNet is used to represent the importance of each input feature. It is used to identify which features should be used for the model's prediction. The algorithm assigns each feature a score, which is used to determine the importance of that feature for the prediction. The mask assigns a higher score to those features that are more important for the prediction, and a lower score to those that are less important. This helps the model focus on the most important features, which improves its accuracy.

Aggregation

Aggregation in TabNet is to group similar data points together. This allows the algorithm to identify patterns and create more accurate predictions. Aggregation helps TabNet to capture complex relationships quickly and accurately, improving the accuracy of the model**.**

Split

The split in a TabNet model is used to create multiple layers of feature importance and capture interactions between different features. By splitting the data into multiple layers, TabNet can identify and capture non-linear relationships between different features. This allows TabNet to generate more accurate predictions than traditional models.

8.6 Implementation and Results

In this project, various machine learning algorithms like SVM, Random Forest, TabNet are used to predict heart disease. Heart Disease UCI dataset (Cleveland) has a total of 76 attributes, out of those only 14 attributes are considered for the prediction of heart disease. Various attributes of the patient like gender, chest pain type, fasting blood pressure, serum cholesterol, exang, etc. are considered for heart disease prediction.

The isnull() method is used to replace all the null values in the dataframe with a Boolean value True for NULL values, and otherwise False. The dropna() method is used to remove the null values. The drop_duplicates() is used to remove duplicate values. We are using StandardScaler() for converting numerical columns and one-hot encoding for categorical values. The SVM, Random Forest, TabNet are used for the classification and the SVM has an accuracy of 84.91% , the Random Forest has an accuracy of 92.45% and the TabNet has an accuracy of 90.15% as maximum and 86.79% as minimum accuracy. For evaluating the experiment, various evaluation metrics like accuracy, confusion matrix, precision, recall, and f1-score are considered.

Accuracy- Accuracy is the ratio of the number of correct predictions to the total.
number of inputs in the dataset.
It is expressed as: Accuracy = (TP + TN) /(TP+FP+FN+TN)
Confusion Matrix- It is a table that is commonly used in machine learning and statistics to evaluate the performance of a classification model. The matrix summarizes the number of

correct and incorrect classifications made by the model on a set of test data, organized into four categories:

- True Positives (TP): Instances that are actually positive and are correctly classified as positive by the model.
- False Positives (FP): Instances that are actually negative but are incorrectly classified as positive by the model.
- True Negatives (TN): Instances that are actually negative and are correctly classified as negative by the model.
- False Negatives (FN): Instances that are actually positive but are incorrectly classified as negative by the model.

The confusion matrix is a 2x2 table that displays these categories in a matrix format. The rows correspond to the actual class labels, while the columns correspond to the predicted class labels.

The Positive class is represented by TP FN

The Negative class is represented by FP TN

The confusion matrix allows us to calculate various performance metrics for the classification model, such as accuracy, precision, recall, and F1-score, which can be useful for understanding the strengths and weaknesses of the model and for making improvements.

8.7 Source Code

```
# importing necessary libraries

import pandas as pd # for data analysis

import numpy as np #for numerical operations

import matplotlib.pyplot as plt

import hvplot.pandas

data=pd.read_csv(r"C:\Users\naga_\3. Coding\Heart Disease Code Implementation\HeartAttack.csv",na_values='?')

data.head()

data.tail()

data.info()

data.describe()

dictionary = {

            "age     ": "Age of person",
```

```
        "sex     ": "1=male, 0=female",
        "cp      ": "Chest pain type, 1: typical angina, 2: atypical angina, 3: non-anginal
pain, 4: asymptomatic",
        "trestbps": "Resting blood pressure",
        "chol    ": "Serum cholestoral in mg/dl",
        "fbs     ": "Fasting blood sugar > 120 mg/dl",
        "thalach ": "Maximum heart rate achieved",
        "restecg ": "Resting electrocardiographic results (values 0,1,2)",
        "exang   ": "Exercise induced angina",
        "oldpeak ": "Oldpeak = ST depression induced by exercise relative to rest",
        "slope   ": "The slope of the peak exercise ST segment",
        "ca      ": "Number of major vessels (0-3) colored by flourosopy",
        "thal    ": "Thalassemia  3 = normal; 6 = fixed defect; 7 = reversable defect"
}
for i in dictionary:
   print(i, ": ",dictionary[i])
print("Count of each column's null values\n",data.isnull().sum())
#dropping the remaining null values
data=data.dropna()
print(data.isnull().sum())
print("Index  Existence of duplicate value\n")
print(data.duplicated())
print("Count of Duplicate rows : ",data.duplicated().sum())
print("Count of Non-Duplicate rows : ",(~data.duplicated()).sum())
data.loc[data.duplicated(),:]
#one-hot encoding
data=pd.get_dummies(data,columns = ["cp","restecg"])
numerical_cols=["age","trestbps","chol","thalach","oldpeak"]
cat_cols = list(set(data.columns)-set(numerical_cols)-{"target"})
# formula { y = (x – mean) / standard_deviation }
```

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()

from sklearn.svm import SVC
from sklearn.metrics import mean_squared_error,accuracy_score
#one is for graph
svm_clf = SVC(kernel='rbf',C=2)
svm_clf.fit(X_train,y_train)
#one is for prediction
svm_clas = SVC(kernel='rbf',C=2)
svm_clas.fit(X_train,y_train)
y_predict = svm_clf.predict(X_test)
y_predic = svm_clas.predict(X_test)
t = mean_squared_error(y_test,y_predic)
print('Mean squred error is : ',t)
svm_accuracy = accuracy_score(y_test,y_predic)
print("Accuracy of the SVM model is : {0:.2f}".format(svm_accuracy*100),'%')
#random forest
from sklearn.ensemble import RandomForestClassifier
rf_max_accuracy = 0
for x in range(200):
    for j in range(1,20):
        rf = RandomForestClassifier(random_state=x,n_estimators= j,criterion="entropy")
        rf.fit(X_train,y_train)
        Y_pred_rf = rf.predict(X_test)
        current_accuracy = round(accuracy_score(Y_pred_rf,y_test)*100,2)
        if(current_accuracy>rf_max_accuracy):
            rf_max_accuracy = current_accuracy
            best_x = x
            best_y = j
```

```
rf_clf = RandomForestClassifier(random_state=best_x,n_estimators = best_y)
rf_clf.fit(X_train,y_train)
Y_pred_rf = rf.predict(X_test)
print("Accuracy of Random Forest: ",rf_max_accuracy,"%")
print("Used Random state ",best_x)
print("Number of Decision Trees used :",best_y)
from pytorch_tabnet.tab_model import TabNetClassifier
import torch

# define the tabnet model
tn_clf= TabNetClassifier(optimizer_fn=torch.optim.Adam,
                         scheduler_params={"step_size":20,"gamma":1.2},
                         scheduler_fn=torch.optim.lr_scheduler.StepLR)

# fit the model
tn_clf.fit(
    X_train,y_train,
    eval_set=[(X_train, y_train), (X_test, y_test)],
    eval_name=['train', 'test'],
    eval_metric=['auc','balanced_accuracy'],
    max_epochs=53,
    patience=40,
    batch_size=203,
    virtual_batch_size=203,
    num_workers=0,
    weights=1,
    drop_last=False
)
from sklearn.svm import SVC
from sklearn.metrics import mean_squared_error,accuracy_score
```

```
svm_clf = SVC(kernel='rbf',C=2)
svm_clf.fit(X_train,y_train)
svm_clas = SVC(kernel='rbf',C=2)
svm_clas.fit(X_train,y_train)
y_predict = svm_clf.predict(X_test)
y_predic = svm_clas.predict(X_test)
t = mean_squared_error(y_test,y_predic)
print('Mean squred error is : ',t)
svm_accuracy = accuracy_score(y_test,y_predic)
print("Accuracy of the SVM model is : {0:.2f}".format(svm_accuracy*100),'%')
import matplotlib.colors as colors
from sklearn.decomposition import PCA
pca = PCA()
X_train_pca = pca.fit_transform(X_train)
X_train_pca[0],X_train_pca[1]
pc1 = X_train_pca[:, 0]
pc2 = X_train_pca[:, 1]
svm_clf.fit(np.column_stack((pc1, pc2)), y_train)
x_min = pc1.min() - 1
x_max = pc1.max() + 1
y_min = pc2.min() - 1
y_max = pc2.max() + 1
a, b = np.meshgrid(np.arange(start=x_min, stop=x_max, step=0.1),np.arange(start=y_min,
stop=y_max, step=0.1))
Z = svm_clf.predict(np.column_stack((a.ravel(), b.ravel()))) ## Array of zeros and ones
Z = Z.reshape(a.shape)
fig, ax = plt.subplots(figsize=(10,10))
ax.contourf(a, b, Z, alpha=0.5)
cmap = colors.ListedColormap(['#4daf4a','#e41a1c'])
scatter = ax.scatter(pc1, pc2, c=y_train,
```

```
            s=100,
            cmap=cmap,
            edgecolors='black',
            alpha=0.7)
legend = ax.legend(scatter.legend_elements()[0],
                scatter.legend_elements()[1],
                 loc="upper right")
legend.get_texts()[0].set_text("Don't have heart disease")
legend.get_texts()[1].set_text("Have heart disease")
ax.set_xlabel('PC1')
ax.set_ylabel('PC2')
ax.set_title('Decison surface using the PCA transformed/projected features')
plt.show()
from sklearn.ensemble import RandomForestClassifier
rf_max_accuracy = 0
for x in range(200):
    for j in range(1,20):
        rf = RandomForestClassifier(random_state=x,n_estimators= j,criterion="entropy")
        rf.fit(X_train,y_train)
        Y_pred_rf = rf.predict(X_test)
        current_accuracy = round(accuracy_score(Y_pred_rf,y_test)*100,2)
        if(current_accuracy>rf_max_accuracy):
            rf_max_accuracy = current_accuracy
            best_x = x
            best_y = j
rf_clf = RandomForestClassifier(random_state=best_x,n_estimators = best_y)
rf_clf.fit(X_train,y_train)
Y_pred_rf = rf.predict(X_test)
print("Accuracy of Random Forest: ",rf_max_accuracy,"%")
print("Used Random state ",best_x)
```

```
print("Number of Decision Trees used :",best_y)
from pytorch_tabnet.tab_model import TabNetClassifier
import torch
# define the model
tn_clf= TabNetClassifier(optimizer_fn=torch.optim.Adam,
                         scheduler_params={"step_size":20,"gamma":1.2},
                         scheduler_fn=torch.optim.lr_scheduler.StepLR)
# fit the model
tn_clf.fit(
    X_train,y_train,
    eval_set=[(X_train, y_train), (X_test, y_test)],
    eval_name=['train', 'test'],
    eval_metric=['auc','balanced_accuracy'],
    max_epochs=53,
    patience=40,
    batch_size=203,
    virtual_batch_size=203,
    num_workers=0,
    weights=1,
    drop_last=False
)
tn_predicted=tn_clf.predict(X_test)
error=0
error = mean_squared_error(y_test,tn_predicted)
print('Mean squred error is : ',error)
tn_accuracy = accuracy_score(y_test,tn_predicted)
print("Accuracy of the TabNet model is : {0:.2f}".format(tn_accuracy*100),'%')
print("Accuracy of the SVM model is : {0:.2f}".format(svm_accuracy*100),'%')
print("Accuracy of Random Forest: ",rf_max_accuracy,"%")
```

```
import ipywidgets as widgets
from IPython.display import display, Image
print("The test values for predicting Heart Disease were derived from various medical tests(the names of tests are given below) \nand examinations.Here are the tests and measurements that the person has to obtain inorder to know whether they \nhad heart disease or not: \n\n")
print("[1].Age: The age of a person who is going to take the test")
print("[2].Sex: The gender of the person\n\n")
print("Note: Every time you click the button pplease scrol down to see the information of it.\n")
print("[3].Chest pain type (cp): Required tests: ECG,Computerized tomography (CT)-scan,Chest X-ray,Blood test")
button = widgets.Button(description="Click Me!")
def print_text(b):
    print("[3].Chest pain type (cp): Determined from the person's medical history and symptoms\nThere are 4 types of chest pains are there:\n1 = typical angina    -- occurs when the person is working hard\n\n2 = atypical angina   -- This Usually feels like a stabbing or burning pain in your chest and may \n\t\t\tsometimes have characteristics similar to indigestion. If the pain also \n\t\t\tspreads to your arms, back, or neck, or if you are feeling sick or have breathing \n\t\t\tdifficulties, you should immediately look for medical attention.\n\n3 = non-anginal pain  -- Happens due to small blood vessels bringing oxygenated blood to heart muscles\n\t\t\t(cause damage to muscle due to lack of blood flow)\n\n4 = asymptomatic      -- A silent heart attack is a heart attack, a silent heart attack might \n\t\t\tnot cause chest pain or shortness of breath, which are typically associated \n\t\t\twith a heart attack.\n\n\n\n")
    img_path = r'C:\Users\naga_\3. Coding\Heart Disease Code Implementation\Imgs\cp.png'
    display(Image(filename=img_path, width=400, height=400))
button.on_click(print_text)
display(button)
print("[4].Serum cholesterol (chol): Requires lipid profile test\n")
button = widgets.Button(description="Click Me!")
def print_text(b):
    print("[4].Cholesterol: serum cholesterol levels in mg/dl (milligrams per deciliter). Serum cholesterol levels refer to the amount \n\tof cholesterol present in the blood, and high levels
```

```
of cholesterol are a known risk factor for heart disease.\n\nA total cholesterol level of less than 200 mg/dL is normal.\nA total cholesterol level of 200 to 239 mg/dL is borderline high.\nA total cholesterol level of 240 mg/dL or greater is high.\n\n\n")

    img_path = r'C:\Users\naga_\3. Coding\Heart Disease Code Implementation\Imgs\cholesterol.png'

    display(Image(filename=img_path, width=500, height=400))

  button.on_click(print_text)

display(button)

  print("[5].ST depression induced by exercise relative to rest (oldpeak): Requires exercise stress test\n")

  button = widgets.Button(description="Click Me!")

  def print_text(b):

    print("[5].Oldpeak value measures the extent of ST segment depression during exercise compared to rest, as recorded\n on an electrocardiogram (ECG) test. The ST segment of the ECG reflects the period between the depolarization \nand repolarization of the ventricles. During exercise, the heart needs more oxygen, which may lead to changes in the \nST segment.Oldpeak value is typically measured in millimeters (mm) and represents the difference between the lowest \npoint of the ST segment during exercise and the baseline measurement taken during rest. A higher Oldpeak value \nindicates a greater degree of ST segment depression and may be indicative of the presence of \nsignificant coronary artery disease.\n\n\n")

  button.on_click(print_text)

display(button)

  print("[6].Resting blood pressure (trestbps): Required test sphygmomanometer(BP Machine)\n")

  button = widgets.Button(description="Click Me!")

  def print_text(b):

    print("[6]The trestbps (resting blood pressure): It represents the patient's resting blood pressure measured in \nmillimeters of mercury (mm Hg) when they were admitted to the hospital. Blood pressure is the force of blood against \nthe walls of the arteries as the heart pumps it around the body, and it is an important indicator of cardiovascular health.\n\nHigh blood pressure, or hypertension, is a major risk factor for heart disease and other cardiovascular conditions.\nBlood pressure is usually measured using a sphygmomanometer, which consists of an inflatable cuff that is wrapped around the upper arm and a mercury or aneroid manometer that measures the pressure. Normal resting blood pressure is usually considered \nto be around 120 by 80 mm Hg, with values above 140 by
```

```
  90 mm Hg indicating hypertension.\n\n\n")
  button.on_click(print_text)
display(button)
  print("[7].Maximum heart rate achieved (thalach): Requires exercise stress test\n")
  button = widgets.Button(description="Click Me!")
  def print_text(b):
    print("[7].The Maximum heart rate achieved (thalach): It represents the maximum heart
  rate achieved by the patient during exercise.\nIt is measured in beats per minute (BPM).
  The maximum heart rate is an important indicator of cardiovascular fitness &
  health.\n\n\n")
  button.on_click(print_text)
display(button)
  print("[8].Exercise-induced angina (exang): Requires exercise stress test\n")
  button = widgets.Button(description="Click Me!")
  def print_text(b):
    print("[8].The exang represents the presence or absence of exercise-induced angina,
  which is chest pain or discomfort that occurs \nduring physical activity. Angina is a
  common symptom of coronary artery disease, which is caused by the narrowing or blockage
  \nof the arteries that supply blood to the heart.\n\nExercise-induced angina occurs when the
  heart is not receiving enough oxygen-rich blood during physical activity. \nThis can cause
  chest pain or discomfort, as well as shortness of breath, sweating, and fatigue. The severity
  and frequency \nof exercise-induced angina can vary depending on the degree of blockage
  in the coronary arteries.\n\nIt is important to note that exercise-induced angina is just one of
  many symptoms of heart disease, and not all patients with \nheart disease experience this
  symptom. Other symptoms of heart disease include chest pain or discomfort at rest,
  shortness of \nbreath, fatigue, dizziness, and palpitations. Therefore, a comprehensive
  evaluation of a patient's cardiovascular health status \nshould take into account all of these
  factors.\n\n\n")
  button.on_click(print_text)
display(button)
  print("[9].Resting electrocardiographic results (restecg): Requires electrocardiogram (ECG)
  test\n")
  button = widgets.Button(description="Click Me!")
  def print_text(b):
    print("[9].The ST segment is a section of the ECG waveform that occurs between the end
```

```
    of the QRS complex \n(which represents ventricular depolarization) and the beginning of
    the T wave (which represents \nventricular repolarization). The slope of the ST segment
    refers to whether the ECG waveform in \nthis section is sloping upwards (positive
    slope(High)), sloping downwards (negative slope(Low)), or is \nhorizontal (zero
    slope(Normal)).\n\nThe slope variable takes on one of three values:\n\nUpsloping: This
    indicates a positive slope of the ST segment, which is typically associated with \na better
    prognosis for patients with suspected or confirmed heart disease.\n\nFlat: This indicates a
    zero slope of the ST segment, which may indicate that the patient has a mild\n form of heart
    disease or is in the early stages of the disease.\n\nDownsloping: This indicates a negative
    slope of the ST segment, which is typically associated with\na more severe form of heart
    disease and a higher risk of adverse outcomes.\n\n\n")

        img_path = r'C:\Users\naga_\3. Coding\Heart Disease Code
    Implementation\Imgs\Low1.jpg'

        display(Image(filename=img_path, width=170, height=170))

        img_path = r'C:\Users\naga_\3. Coding\Heart Disease Code
    Implementation\Imgs\Normal1.jpg'

        display(Image(filename=img_path, width=170, height=170))

        img_path = r'C:\Users\naga_\3. Coding\Heart Disease Code
    Implementation\Imgs\High1.jpg'

        display(Image(filename=img_path, width=170, height=170))

    button.on_click(print_text)

display(button)

    print("[10].Fasting blood sugar (fbs): Requires The fasting plasma glucose (FPG) test\n")

    button = widgets.Button(description="Click Me!")

    def print_text(b):

        print("[10].The Fasting blood sugar levels refer to the amount of glucose present in the
    blood after overnight fast, and high levels \nof fasting blood sugar are a known risk factor
    for diabetes and cardiovascular disease.\nFasting blood sugar is a binary variable, meaning
    it can take on one of two values: 0 or 1. \nA value of 0 indicates that the fasting blood sugar
    level was less than 120 mg/dl\nA value of 1 indicates that the fasting blood sugar level was
    greater than or equal to 120 mg/dl\n\n\n")

    button.on_click(print_text)

    display(button)

print("\n\n\n")
```

8.7.1 Input

```
p_age = int(input("\nEnter Patient Age(in number) :"))
p_trestbps = int(input("\nEnter Patient Blood Pressure value [Number ex:121]: "))
p_chol = int(input("\nEnter Patient Cholesterol value: "))
p_thalach = int(input("\nEnter maximum Heart rate: "))
p_oldpeak = float(input("\nEnter patient oldpeak value: "))
p_fbs = int(input("Enter fasting blood sugar value\n0 is for normal value\n1 is for abnormal value: "))
p_cp = int(input("\nEnter the Chest pain type:\n1 is for Typical\n2 is for Atypical\n3 is for Non-Anginal pain\n4 is for Asymptomatic :"))
p_exang = int(input("\nEnter exercise indused angina value \n1 is for YES\n0 is for NO : "))
p_sex = int(input("\nEnter gender of patient \n1 is for Male\n2 is for Female : "))
p_restecg = int(input("\nEnter Resting Electrocardiogram value:\n 0 is for normal\n 1 is for havig ST\n 2 is for hypertrophy :" ))
p_numerical_cols=[[p_age,p_trestbps,p_chol,p_thalach,p_oldpeak]]
if p_cp==1:
    p_cp_1=True
    p_cp_2=False
    p_cp_3=False
    p_cp_4=False
elif p_cp==2:
    p_cp_1=False
    p_cp_2=True
    p_cp_3=False
    p_cp_4=False
elif p_cp==3:
    p_cp_1=False
    p_cp_2=False
    p_cp_3=True
    p_cp_4=False
```

```
elif p_cp==4:
    p_cp_1=False
    p_cp_2=False
    p_cp_3=True
    p_cp_4=False
else:
    p_cp_1=False
    p_cp_2=False
    p_cp_3=False
    p_cp_4=False
if p_restecg==0:
    p_restecg_0=True
    p_restecg_1=False
    p_restecg_2=False
elif p_restecg==1:
    p_restecg_0=False
    p_restecg_1=True
    p_restecg_2=False
elif p_restecg==2:
    p_restecg_0=False
    p_restecg_1=False
    p_restecg_2=True
p_cat_cols=[[p_fbs, p_cp_2, p_cp_3, p_exang, p_cp_1, p_sex, p_cp_4, p_restecg_0,
p_restecg_1, p_restecg_2]]
dummy_cat1=[[0,False,False,0,True,0,False,False,True,False]]
dummy_cat2=[[0,True,False,0,False,0,False,True,False,False]]
dummy_num1=[[30,170,237,170,0]]
dummy_num2=[[32,105,198,165,0]]
p_cate_cols=p_cat_cols+dummy_cat1+dummy_cat2
p_numeri_cols=p_numerical_cols+dummy_num1+dummy_num2
```

```
p_numeri_cols=np.array(p_numeri_cols)
p_cate_cols=np.array(p_cate_cols)
p_cate_cols.reshape(3,10)
p_numeri_cols.reshape(3,5)
def my_fun(p_numeri_cols,p_cate_cols,scaler):
    p_x_scaled = scaler.fit_transform(p_numeri_cols)
    p_x_cat = p_cate_cols
    p_x = np.hstack((p_x_cat,p_x_scaled))
    return p_x
p_data_x = my_fun(p_numeri_cols,p_cate_cols,scaler)

predicted_op_of_Random_forest=rf_clf.predict(p_data_x)
predicted_op_of_TabNet=tn_clf.predict(p_data_x)
predicted_op_of_Support_Vector_Machine=svm_clas.predict(p_data_x)
count1 = 0
if(predicted_op_of_Support_Vector_Machine[0]!=None):
    if predicted_op_of_Support_Vector_Machine[0]==0:
        pass
    else:
        count1+=1
if(predicted_op_of_Random_forest[0]!=None):
    if predicted_op_of_Random_forest[0]==0:
        pass
    else:
        count1+=1
if(predicted_op_of_TabNet[0]!=None):
    if predicted_op_of_TabNet[0]==0:
        pass
    else:
        count1+=1
```

```
print("Result:")
if count1>=2:
    print("The patirnt have RISK to get Heart Disease")
else:
  print("The patient seems to be NORMAL,Take care of your Health")
```

8.7.2 Output

```
Enter Patient Age(in number) :38

Enter Patient Blood Pressure value [Number ex:121]: 130

Enter Patient Cholesterol value: 140

Enter maximum Heart rate: 90

Enter patient oldpeak value: 0
Enter fasting blood sugar value
0 is for normal value
1 is for abnormal value: 0

Enter the Chest pain type:
1 is for Typical
2 is for Atypical
3 is for Non-Anginal pain
4 is for Asymptomatic :2

Enter exercise indused angina value
1 is for YES
0 is for NO : 0

Enter gender of patient
1 is for Male
2 is for Female : 1

Enter Resting Electrocardiogram value:
 0 is for normal
 1 is for havig ST
 2 is for hypertrophy :2



Result:
The patient seems to be NORMAL, Take care of your Health
```

Chapter 9

Screenshots Of Project

The test values for predicting Heart Disease were derived from various medical tests(the names of tests are given below) and examinations.Here are the tests and measurements that the person has to obtain inorder to know whether they had heart disease or not:

[1].Age: The age of a person who is going to take the test
[2].Sex: The gender of the person

[3].Chest pain type (cp): Determined from the person's medical history and symptoms
There are 4 types of chest pains are there:
1 = typical angina -- occurs when the person is working hard

2 = atypical angina -- This Usually feels like a stabbing or burning pain in your chest and may sometimes have characteristics similar to indigestion. If the pain also spreads to your arms, back, or neck, or if you are feeling sick or have breathing difficulties, you should immediately look for medical attention.

3 = non-anginal pain -- Happens due to small blood vessels bringing oxygenated blood to heart muscles (cause damage to muscle due to lack of blood flow)

4 = asymptomatic -- A silent heart attack is a heart attack, a silent heart attack might not cause chest pain or shortness of breath, which are typically associated with a heart attack.

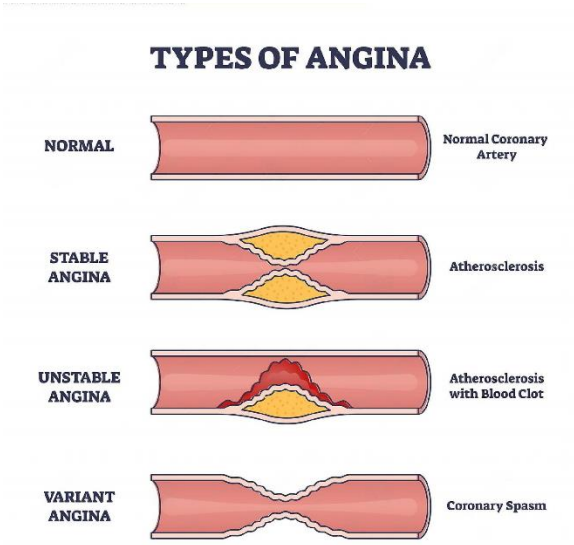


Figure 9.1: Types of Anginas

[4].Cholesterol: serum cholesterol levels in mg/dl (milligrams per deciliter). Serum cholesterol levels refer to the amount of cholesterol present in the blood, and high levels of cholesterol are a known risk factor for heart disease.

A total cholesterol level of less than 200 mg/dL is normal.
A total cholesterol level of 200 to 239 mg/dL is borderline high.
A total cholesterol level of 240 mg/dL or greater is high.

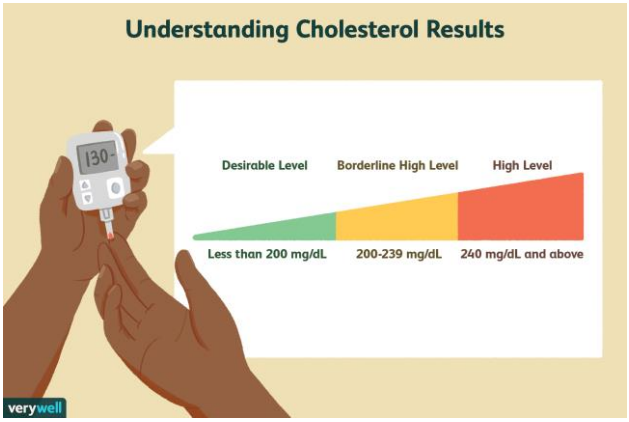


Figure 9.2: Cholesterol level

[5].Oldpeak value measures the extent of ST segment depression during exercise compared to rest, as recorded on an electrocardiogram (ECG) test. The ST segment of the ECG reflects the period between the depolarization and repolarization of the ventricles. During exercise, the heart needs more oxygen, which may lead to changes in the ST segment.Oldpeak value is typically measured in millimeters (mm) and represents the difference between the lowest point of the ST segment during exercise and the baseline measurement taken during rest. A higher Oldpeak value indicates a greater degree of ST segment depression and may be indicative of the presence of significant coronary artery disease.

[6]The trestbps (resting blood pressure): It represents the patient's resting blood pressure measured in millimeters of mercury (mm Hg) when they were admitted to the hospital. Blood pressure is the force of blood against the walls of the arteries as the heart pumps it around the body, and it is an important indicator of cardiovascular health.

High blood pressure, or hypertension, is a major risk factor for heart disease and other cardiovascular conditions. Blood pressure is usually measured using a sphygmomanometer, which consists of an inflatable cuff that is wrapped around the up per arm and a mercury or aneroid manometer that measures the pressure. Normal resting blood pressure is usually considered to be around 120 by 80 mm Hg, with values above 140 by 90 mm Hg indicating hypertension.

[7].The Maximum heart rate achieved (thalach): It represents the maximum heart rate achieved by the patient during exercise. It is measured in beats per minute (BPM). The maximum heart rate is an important indicator of cardiovascular fitness & health.

[8].The exang represents the presence or absence of exercise-induced angina, which is chest pain or discomfort that occurs during physical activity. Angina is a common symptom of coronary artery disease, which is caused by the narrowing or blockage of the arteries that supply blood to the heart.

Exercise-induced angina occurs when the heart is not receiving enough oxygen-rich blood during physical activity. This can cause chest pain or discomfort, as well as shortness of breath, sweating, and fatigue. The severity and frequency of exercise-induced angina can vary depending on the degree of blockage in the coronary arteries.

It is important to note that exercise-induced angina is just one of many symptoms of heart disease, and not all patients with heart disease experience this symptom. Other symptoms of heart disease include chest pain or discomfort at rest, shortness of breath, fatigue, dizziness, and palpitations. Therefore, a comprehensive evaluation of a patient's cardiovascular health status should take into account all of these factors.

[9].The ST segment is a section of the ECG waveform that occurs between the end of the QRS complex (which represents ventricular depolarization) and the beginning of the T wave (which represents ventricular repolarization). The slope of the ST segment refers to whether the ECG waveform in this section is sloping upwards (positive slope(High)), sloping downwards (negative slope(Low)), or is horizontal (zero slope(Normal)).

The slope variable takes on one of three values:

Upsloping: This indicates a positive slope of the ST segment, which is typically associated with a better prognosis for patients with suspected or confirmed heart disease.

Flat: This indicates a zero slope of the ST segment, which may indicate that the patient has a mild form of heart disease or is in the early stages of the disease.

Downsloping: This indicates a negative slope of the ST segment, which is typically associated with a more severe form of heart disease and a higher risk of adverse outcomes.

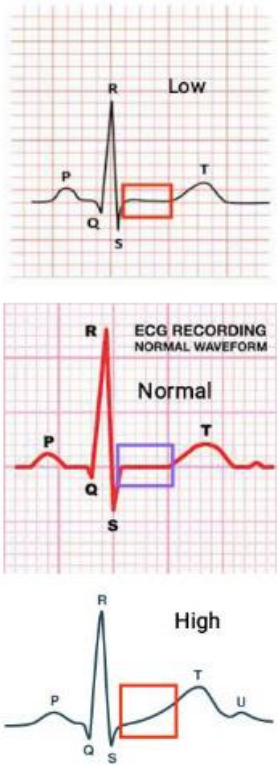


Figure 9.3:Values of Slope variable

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1020	59	1	1	140	221	0	1	164	1	0.0	2	0	2	1
1021	60	1	0	125	258	0	0	141	1	2.8	1	1	3	0
1022	47	1	0	110	275	0	0	118	1	1.0	1	1	2	0
1023	50	0	0	110	254	0	0	159	0	0.0	2	0	2	1
1024	54	1	0	120	188	0	1	113	0	1.4	1	1	3	0

1025 rows × 14 columns

Figure 9.4: Dataset

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	num
0	28	1	2	130.0	132.0	0.0	2.0	185.0	0.0	0.0	NaN	NaN	NaN	0
1	29	1	2	120.0	243.0	0.0	0.0	160.0	0.0	0.0	NaN	NaN	NaN	0
2	29	1	2	140.0	NaN	0.0	0.0	170.0	0.0	0.0	NaN	NaN	NaN	0
3	30	0	1	170.0	237.0	0.0	1.0	170.0	0.0	0.0	NaN	NaN	6.0	0
4	31	0	2	100.0	219.0	0.0	1.0	150.0	0.0	0.0	NaN	NaN	NaN	0

Figure 9.5: First five records of dataset

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	num
289	52	1	4	160.0	331.0	0.0	0.0	94.0	1.0	2.5	NaN	NaN	NaN	1
290	54	0	3	130.0	294.0	0.0	1.0	100.0	1.0	0.0	2.0	NaN	NaN	1
291	56	1	4	155.0	342.0	1.0	0.0	150.0	1.0	3.0	2.0	NaN	NaN	1
292	58	0	2	180.0	393.0	0.0	0.0	110.0	1.0	1.0	2.0	NaN	7.0	1
293	65	1	4	130.0	275.0	0.0	1.0	115.0	1.0	1.0	2.0	NaN	NaN	1

Figure 9.6: Last five records of dataset

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 294 entries, 0 to 293
Data columns (total 14 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   age         294 non-null    int64
 1   sex         294 non-null    int64
 2   cp          294 non-null    int64
 3   trestbps    293 non-null    float64
 4   chol        271 non-null    float64
 5   fbs         286 non-null    float64
 6   restecg     293 non-null    float64
 7   thalach     293 non-null    float64
 8   exang       293 non-null    float64
 9   oldpeak     294 non-null    float64
 10  slope       104 non-null    float64
 11  ca          3 non-null      float64
 12  thal        28 non-null     float64
 13  num         294 non-null    int64
dtypes: float64(10), int64(4)
memory usage: 32.3 KB
```

Figure 9.7: Attributes and their datatypes

```
Index  Existence of duplicate value

0      False
1      False
3      False
4      False
5      False
       ...
289    False
290    False
291    False
292    False
293    False
Length: 261, dtype: bool
Count of Duplicate rows :  0
Count of Non-Duplicate rows :  261
```

Figure 9.8: Count of duplicates

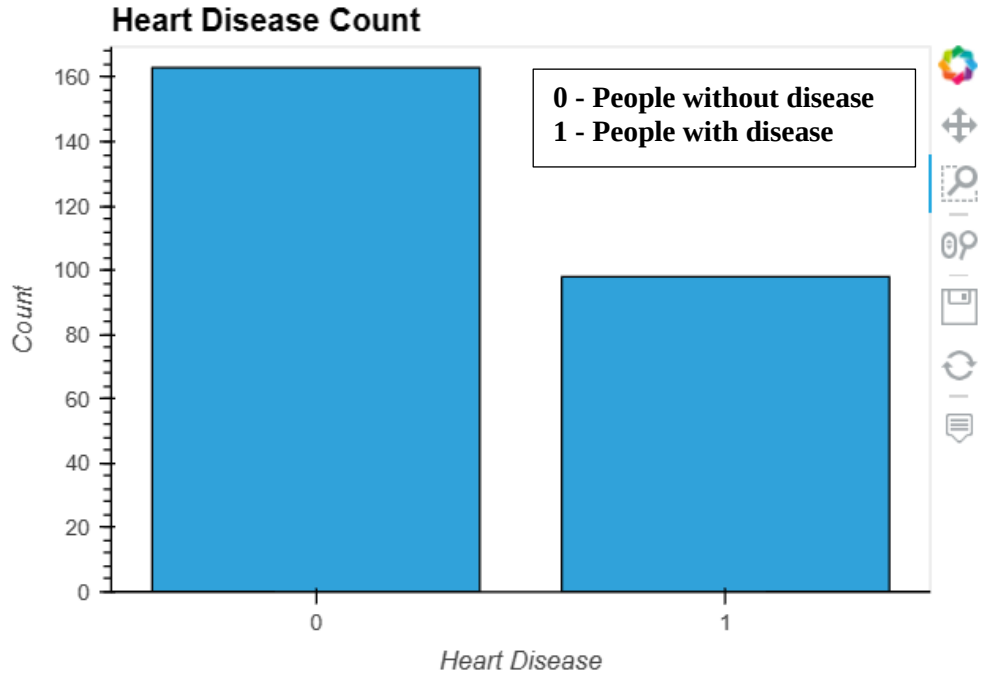


Figure 9.9: Count plot of patients with (1) & without (0) heart disease

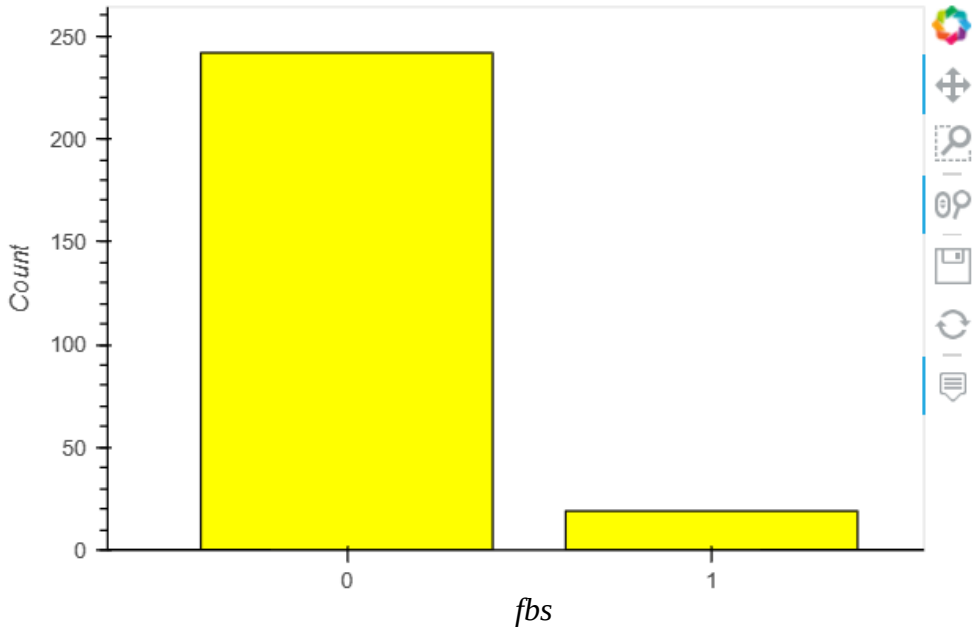


Figure 9.10: Count plot of patients with fbs(1) & without fbs(0)

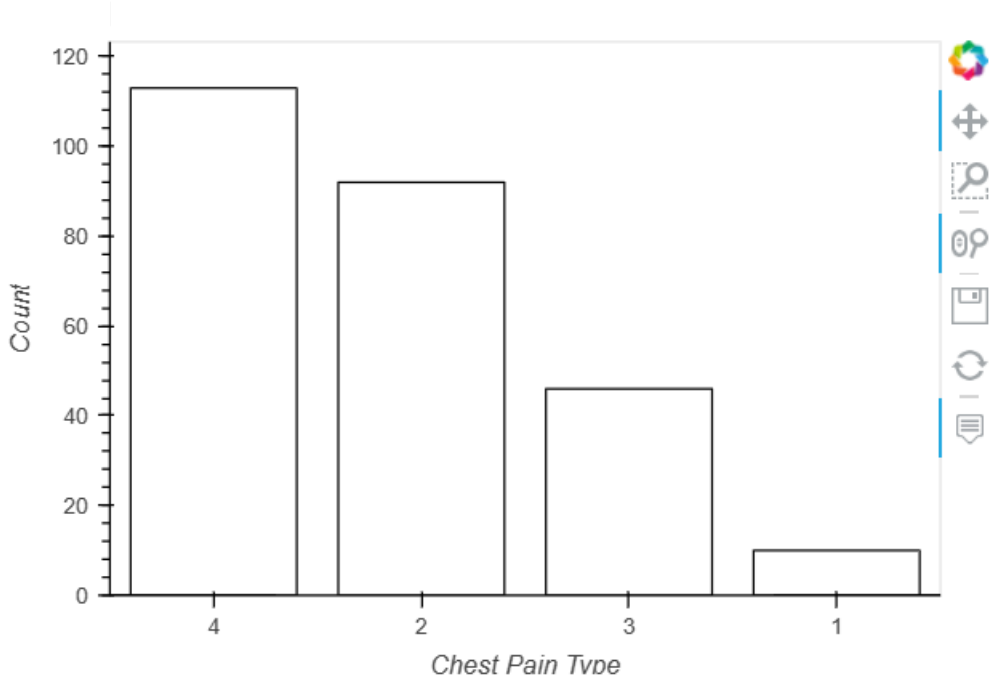


Figure 9.11: Count plot for chest pain type

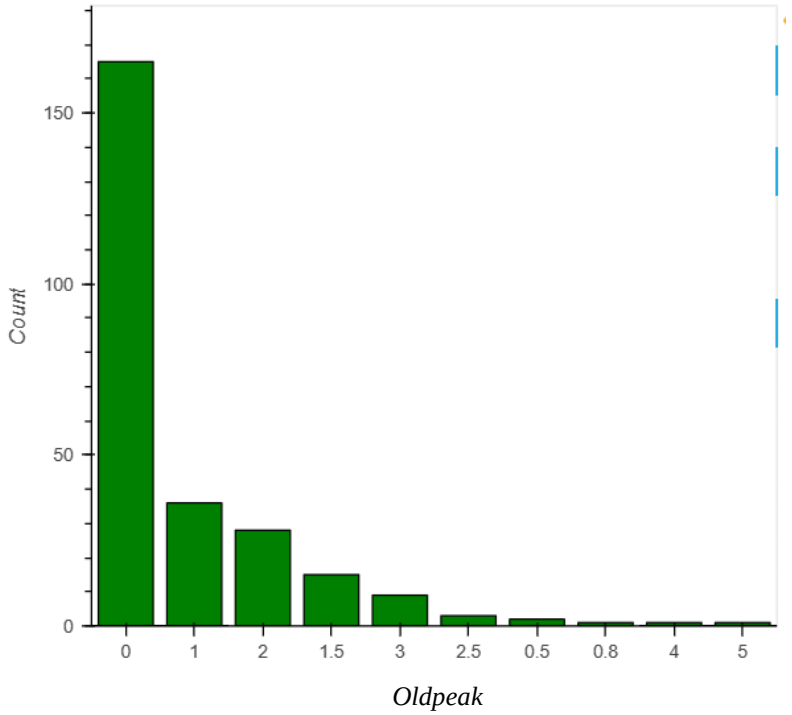


Figure 9.12: Count plot for Oldpeak

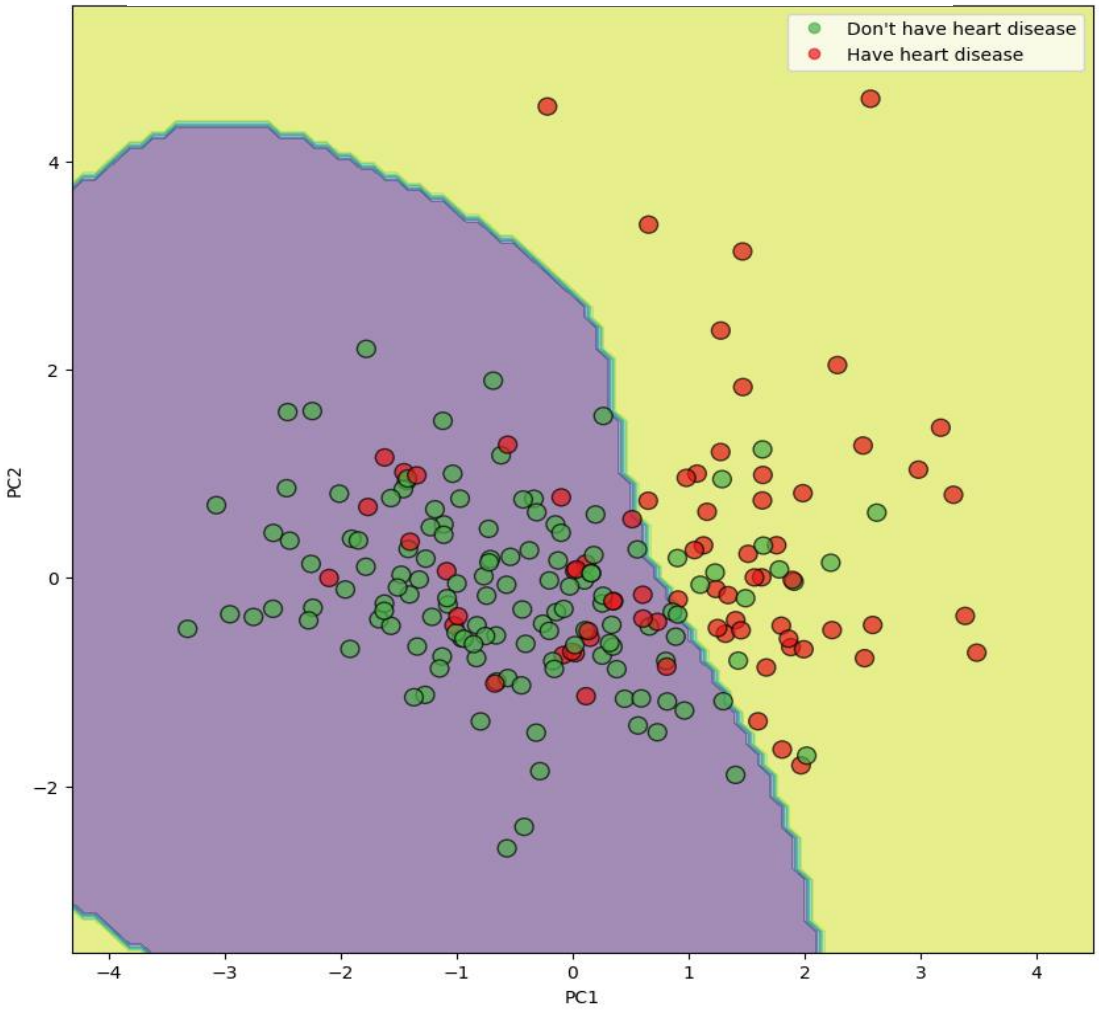


Figure 9.13: Scatter plot for svm classifer

```
a = mean_squared_error(y_test,y_predict)
print('Mean squred error is : ',a)
svm_accuracy_1 = accuracy_score(y_test,y_predict)
print("Accuracy of the SVM model is : {0:.2f}".format(svm_accuracy_1*100),'%')
```

```
Mean squred error is :  0.1509433962264151
Accuracy of the SVM model is : 84.91 %
```

```
print("Accuracy of Random Forest: ",rf_max_accuracy,"%")
print("Used Random state ",best_x)
print("Number of Decision Trees used :",best_y)
```

```
Accuracy of Random Forest:  90.57 %
Used Random state  6
Number of Decision Trees used : 9
```

```
error=0
error = mean_squared_error(y_test,tn_predicted)
print('Mean squred error is : ',error)
tn_accuracy = accuracy_score(y_test,tn_predicted)
print("Accuracy of the TabNet model is : {0:.2f}".format(tn_accuracy*100),'%')
```

```
Mean squred error is :  0.1320754716981132
Accuracy of the TabNet model is : 86.79 %
```

```
print("Accuracy of the SVM model is : {0:.2f}".format(svm_accuracy*100),'%')
print("Accuracy of Random Forest: ",rf_max_accuracy,"%")
print("Accuracy of the TabNet model is : {0:.2f}".format(tn_accuracy*100),'%')
```

```
Accuracy of the SVM model is : 84.91 %
Accuracy of Random Forest:  90.57 %
Accuracy of the TabNet model is : 86.79 %
```

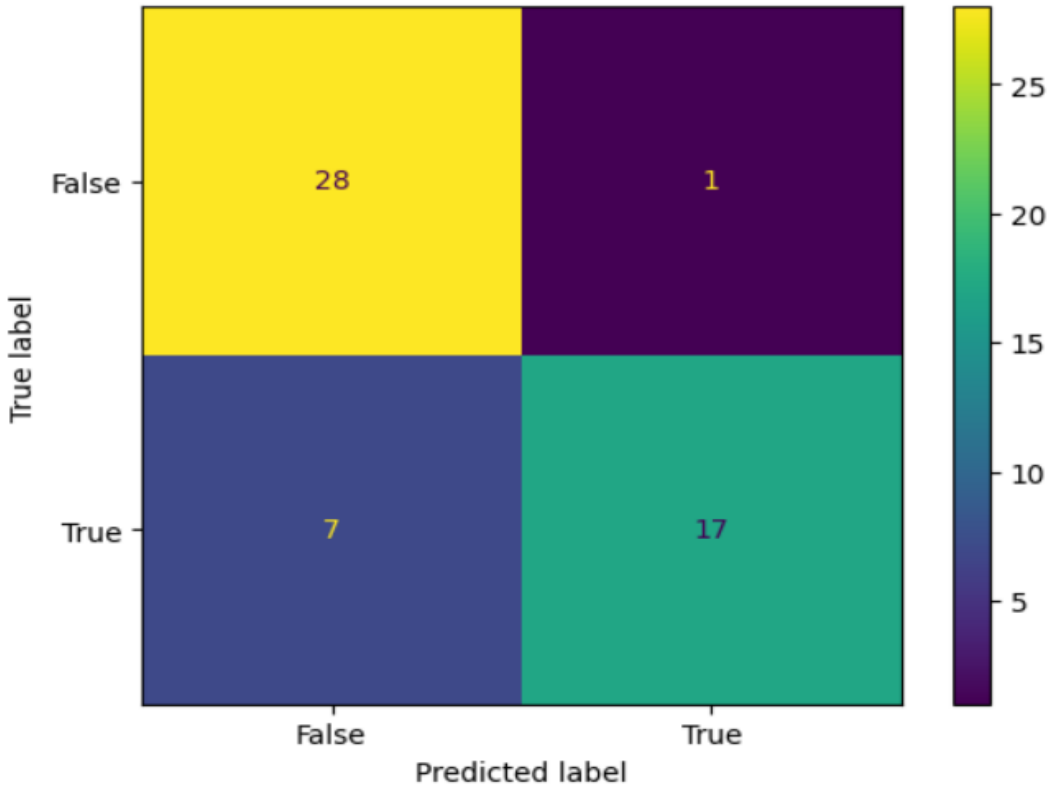


Figure 9.14: Confusion matrix for SVM

SVM Classification Report :

	precision	recall	f1-score	support
0	0.80	0.97	0.88	29
1	0.94	0.71	0.81	24
accuracy			0.85	53
macro avg	0.87	0.84	0.84	53
weighted avg	0.87	0.85	0.85	53

Figure 9.15: Classification report for SVM

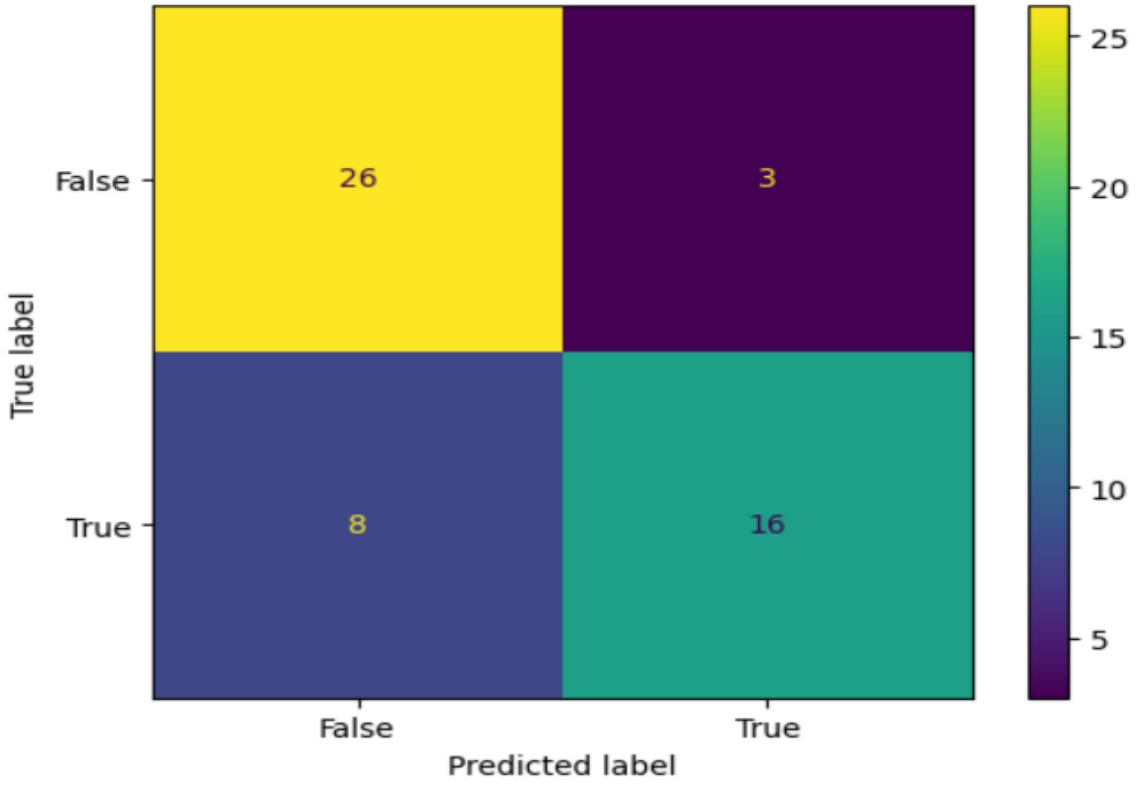


Figure 9.16: Confusion matrix for Random Forest

Random Forest Classification Report :

	precision	recall	f1-score	support
0	0.79	0.90	0.84	29
1	0.85	0.71	0.77	24
accuracy			0.81	53
macro avg	0.82	0.80	0.81	53
weighted avg	0.82	0.81	0.81	53

Figure 9.17: Classification report for Random Forest

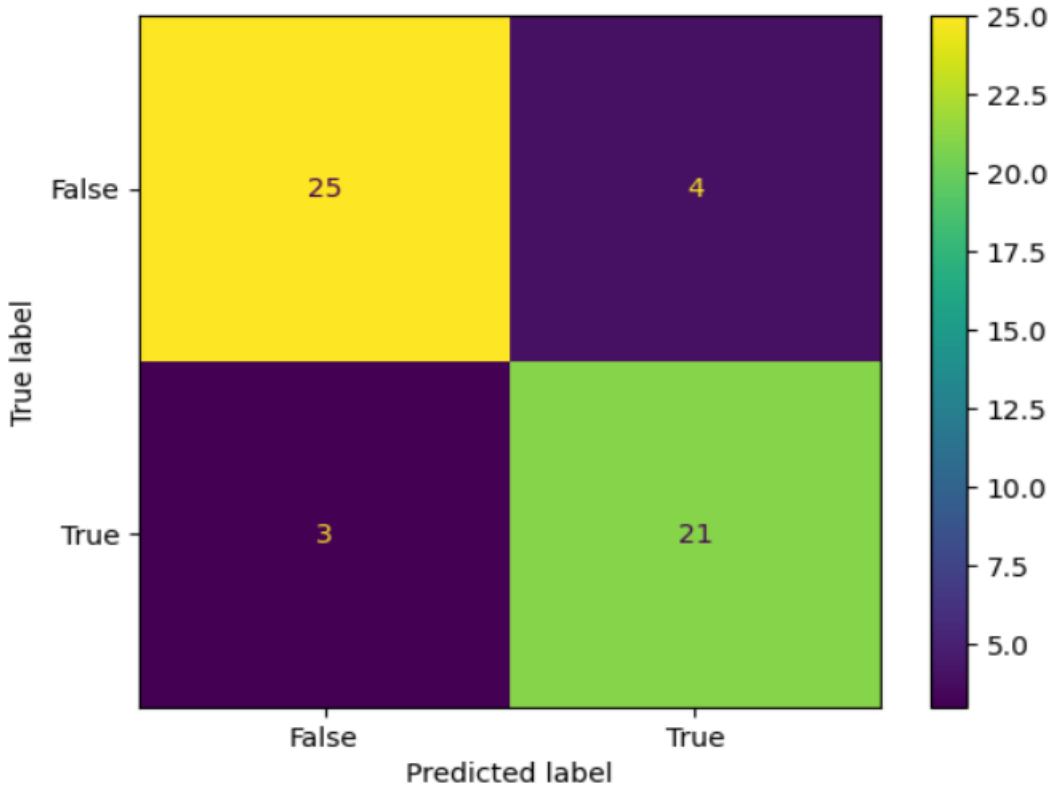


Figure 9.18: Confusion matrix for TabNet

TabNet Classification Report :

	precision	recall	f1-score	support
0	0.79	0.93	0.86	29
1	0.89	0.71	0.79	24
accuracy			0.83	53
macro avg	0.84	0.82	0.82	53
weighted avg	0.84	0.83	0.83	53

Figure 9.19: Classification report for TabNet

Chapter 10

Conclusion and Future Scope

10.1 Conclusion

Heart diseases are a major killer in India and throughout the world, application of promising technology like machine learning to the initial prediction of heart diseases will have a profound impact on society. The early detection of heart disease can aid in making decisions on lifestyle changes in high-risk patients and in turn reduce complications, which can be a great milestone in the field of medicine. The number of people facing heart disease is on the rise each year. This prompts its early diagnosis and treatment. In this paper, the three different machine learning algorithms used to measure the performance of Support Vector Machine (SVM), Random Forest, and Tabnet Model are applied to the dataset. The expected attributes leading to heart disease in patients are available in the dataset which contains 14 important features that are useful to evaluate the system are selected. If all the attributes are taken into consideration, then the efficiency of the system the author gets is less. To increase efficiency, attribute selection is done. In this, 14 features are selected for evaluating the model which gives more accuracy.

10.2 Future Scope

By making this project a cloud-based application users can access it from anywhere in the world with the help of the internet. Collecting more data for the dataset will help in increasing the performance of the models.

References

[1] Mr. ChalaBeyene, Prof. Pooja Kamat, "Survey on Prediction and Analysis the Occurrence of Heart Disease Using Data Mining Technique", International Journal of Pure and Applied Mathematics, 2018.

[2] Mohan, Senthilkumar, Chandrasegar Thirumalai, and Gautam Srivastava, "Effective heart disease prediction using hybrid machine learning techniques" IEEE Access 7 (2019): 81542-81554.

[3] Ali, Liaqat, et al, "An optimized stacked support vector machines based expert system for the effective prediction of heart failure" IEEE Access 7 (2019): 54007-54014.

[4] Singh Yeshvendra K., Nikhil Sinha, and Sanjay K. Singh, "Heart Disease Prediction System Using Random Forest", International Conference on Advances in Computing and Data Sciences. Springer, Singapore, 2016.

[5] C.-L. Chang and C.-H. Chen, "Applying decision tree and neural network to increase quality of dermatologic diagnosis," Expert Syst. Appl., vol. 36, no. 2, Part 2, pp. 4035–4041, Mar. 2009.

[6] S. Shilaskar and A.Ghatol, "Feature selection for medical diagnosis :Evaluation for cardiovascular diseases," Expert Syst. Appl., vol. 40, no. 10, pp. 4146–4153, Aug. 2013.

[7] Yangzi Mu, Mengxing Huang "Diagnosis prediction via Recurrent Neural Networks," International Journal of Machine Learning and Computing, Vol. 8, No. 2, April 2018.

[8] Madhumita Pal, Smita Parija "Risk prediction of cardiovascular disease using machine learning classifiers," Open Medicine 2022; 1100-1113.