CHAPTER-1

INTRODUCTION

1. INTRODUCTION

Heart diseases have emerged as one of the most prominent cause of death all around the world. It can be a cause of serious cardiovascular actions just like stroke and heart attack. It will happen when the heart ever stops the functioning and ceases to pump blood, the body will shut down and within very less time a person will die.

Heart disease risk assessment is very crucial to find prevention opportunities because this disease affects a person in such a way so that the patients can't be cured as easily as possible. It is a very heterogeneous and complex disease which is difficult to detector diagnose at the right time is the toughest work in medical field due to the variety of unusual signs and symptoms. Misunderstanding and wrong diagnosis made by the hospital leads to the loss of many lives of people. Unfortunately, there are many different factors can influence and complicate the detection of heart anomalies and can result in an inaccurate diagnosis or in a delay in a correct diagnosis. Due to many uncertain risk factors, sometimes heart disease diagnosis is difficult even for experts, who frequently consider accurate tools to find all the risk factors and give a clear result in a specific time period. Misunderstanding and wrong diagnosis made by the doctors leads to the death of the people lives.

The world health organization reports suggest that greater than 12 million deaths are happening worldwide due to cardiovascular problems mechanism. Thus, feasible and accurate prediction of heart related diseases is very important. An accurate prediction model for Heart disease can be a very useful for physicians as well as for patients. Every day, modern computer-based systems collect large amounts of data using automatic data record systems in different fields. So, medical organisations, all around the world, collect data on various health related issues. These data can be exploited using various machine learning techniques to gain useful insights. But the data collected is very massive and, many a 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.

On the basis of accurate risk prediction, a doctor can recommend a valid treatment plan, and patients can follow those treatment plans more confidently. Considering the high rate of cardiovascular induced fatalities, researchers have tried to adopt machine learning algorithms to diagnose heart disease. So, the machine learning algorithms are used to predict the heart disease in vital time by correctly identifying the disease in an initial stage by measuring perfectly and precisely.

Machine learning is an art of mastering system without being explicitly computed. Machine learning techniques gives support for forecasting the uncertainty levels of cardiovascular diseases based on the data present. The medical datasets used are taken from the medical organisations, all around the world, collect data on various health related issues thof the patients. Thus, these algorithms have become very useful to predict the presence or absence of heart related diseases accurately and also reduce the number of patients and the number of deaths from heart disease and it can also create a person's awareness of heart disease itself.

1.1 Existing Work:

The existing work dealt with different data that was collected from available sources. and then the data was further processed by deleting rows and columns which contain missing values to handle the unwanted data and noisy data.Then the pre processed data was given as an input to different classification algorithms like Logistic Regression, Naïve Bayes,Support Vector Machine. Finally the accuracy between different classification algorithms was compared .

1.2 Proposed Work:

In our proposed system, the dataset contains the details of the patients that is collected from the medical organisations.Then the data is pre-processed by using imputation methods like mean, median and standard deviation to remove all the noisy and massive data.here the pre-processing includes data cleaning, removal of noise i.e. missing data. k-Nearest Neighbors and Random Forest classification algorithms are performed on the pre-processed data for the prediction of heart disease. The performance of the both algorithms is observed by comparing the accuracy between the algorithms among imputation measures.

CHAPTER-2

LITERATURE SURVEY

2. LITERATURE SURVEY

[1]Prediction of Cardiovascular Disease Using Machine Learning Algorithms by Dinesh Kumar G, Santhosh Kumar, Arumugaraj K and Mareeswari V in 2018.

This paper suggests a prediction model to predict whether a people have a heart disease or not and to provide an awareness or diagnosis on that. For prediction of heart disease there are different machine learning algorithms like Random forest, Naive Bayes classifier, logistic regression, etc. By comparing the accuracies of applying rules to the individual results of Support Vector Machine, Gradient Boosting, Random forest, Naive Bayes classifier and logistic regression on the dataset taken in a region to present an accurate model of predicting cardiovascular disease. The performance of the diagnosis model is obtained by using methods like classification, accuracy, sensitivity and specificity analysis.This paper suggests the different machine learning techniques that are used for forecasting the uncertainty levels of cardiovascular diseases based on the attributes present. The medical datasets used are taken from the research that had been fascinated throughout the world. It contributes the correlative application and analysis of distinct machine learning algorithms in the R software which gives an immediate mechanism for the user to use the machine learning algorithms in R software for forecasting the cardiovascular diseases. This is non-ethical study aims to use available machine learning techniques in R software. Future work includes different ensemble methods of these algorithms which can advance to better performance with more parameter settings for these algorithms.

[2]Preprocessing Unbalanced Data using Weighted Support Vector Machines for Prediction of Heart Disease in Children by Thiago R. Tavares, Adriano L. I. Oliveira, George G. Cabral, Sandra S. Mattos and Renata Grigorio in 2016.

In this paper, they analysed the medical data to determine whether children patients are having heart disease or not. They collected raw data at Brazilian local hospital to be

preprocessed in order to build the classification models. Only non invasive information was used. They used four modeling techniques like Decision Tree, Support Vector Machine (SVM), Weighted SVM, Multilayer Perceptron ANN. For handling imbalanced dataset they took 3 different techniques. They are SMOTE - Synthetic Minority Over-sampling Technique, Weighted SVM for unbalancing preprocessing and SVM for unbalancing preprocessing. Cardiac diseases are highly prevalent and responsible for the highest morbidity and mortality rates, not only in Brazil, but in most modern societies. Many different tools and techniques are continuously developed to analyze cardiac performance. In many techniques, the diagnosis of a cardiac disease is often based on a combination of the patient's history and his/her physical examination. For the chosen dataset and two UCI datasets, preliminary experiments have shown that the method for dealing with imbalancing datasets recently introduced in did not perform well. So, this method was discarded. At the end (Weighted SVM + MLP) outperformed the other methods in 3 out of 4 performance metrics. The early prediction of a heart disease is very important.

[3]A Comprehensive Investigation and Comparison of Machine Learning Techniques in the Domain of Heart Disease by Seyedamin Pouriyeh, Sara Vahid, Giovanna Sanninoy, Giuseppe De Pietroy, Hamid Arabnia, Juan Gutierrez in 2017.

The main aim of this paper is to investigate and compare the accuracy of different data mining classification schemes, employing Ensemble Machine Learning Techniques, for the prediction of heart disease. Here, they used Longbech dataset as main database for testing and training. Different classifiers, namely Multilayer Perception(MLP), K-Nearest Neighbor(K-NN), Conjunctive Rule Learner (SCRL), Radial Basis Function(RBF), Decision Tree(DT), Naïve Bayes and Support Vector Machine (SVM),etc have been employed. Moreover, the ensemble prediction of classifiers like bagging, boosting and stacking, has been applied to the dataset to increase accuracy. In this paper author the Longbech data set with a focus on comparing global evolutionary computation approaches, and observed some prediction performance improvements when using a new approach. However, the

performance of his proposed technique is dependent on the attributes selected by the algorithm. The data set used in the current research contains 303 instances with a total number of 76 attributes. However, the majority of the studies use a maximum of 14 attributes as these are closely linked to heart disease. The features included are age, sex, chest pain type, resting blood pressure, cholesterol, fasting blood sugar, resting ECG, maximum heart rate, exercise induced angina, oldpeak, slope, number of vessels colored and thalassemia, respectively. The main class has two values, "False" and "True", corresponding to the absence or presence, respectively, of any heart disease.Among all the classifiers the best accuracy is given by SVM method by using boosting technique.

[4]A Review on Heart Disease Diagnosis and Prediction Using Machine Learning and Data Mining Techniques by AnimeshHazra, Subrata Kumar Mandal, Amit Gupta, Arkomita Mukherjee and Asmita Mukherjee in 2017.

This paper summarises some of the recent works done in data mining related to cardiovascular diseases. Some of the works showed that rather than applying a single mining technique on a data set, results are far better if a collection of mining techniques are used. Data mining algorithms can be effectively used to 'mine' relevant information from the huge amounts of data generated by the healthcare industry. Java is chosen in most of the research work for practical execution of the project. WEKA, Tanagra, Matlab etc. are some of the other popular tools used for data analysis. Careful selection of the combination of mining techniques and accurate implementation of those techniques on the data set yields a fast and effective implementation of a system for heart disease management. The required dataset is divided into two parts, one is used for mining and the smaller partition is used for verifying. Most of the time, 10 fold cross validation technique is used. Some of the works are about the comparison of different classification techniques on a dataset to correctly classify if a given patient has any probability of a heart disease or not. The idea of a hybrid model is to incorporate several known classification and selection techniques in a single model to give

better results. It is observed that hybrid models give very high accuracy if proper combinations of different algorithms are chosen.

[5]Identification of Heart Failure by Using Unstructured Data of Cardiac Patients by Muhammad Saqlain, Wahid Hussain, Nazar A. Saqib, Muazzam A. Khan in 2016.

The main purpose of this paper is to propose a simple approach that extracts the information's and define the mortality rate of HF patients within 1 year of the time period. They applied different data mining techniques to get useful information from medical reports of patients and using machine learning classification algorithm. To perform multi-class classification they used multinomial Naïve Bayes (NB) classification algorithm. The dataset used in this research was taken by a famous hospital in Pakistan: Armed Forces Institute of Cardiology (AFIC), in the form of unstructured patient's reports. They first extract the important features with the help of cardiologists and then divide these features into five basic classes: demographics, vital, medication, lab results and co-morbidities to decrease the complexity and heterogeneity of data, then perform state of- the-art classification algorithm Logistic Regression, Naïve Bayes, Neural Network, Support Vector Machine, Random Forest and Decision Tree, to classify the dataset into two classes survival and un-survival. These results can be helpful for medical practitioners and researchers for predictive analyses and for making intelligent decisions. Their future concern is to apply a newly introduced classification algorithm named as Contrast Pattern Added Regression model (CPXR) on the same dataset to improve the accuracy till 90%. Different text mining techniques can also be used to extract the buried information from the patient unstructured reports.

CHAPTER-3

METHODOLOGY

3. METHODOLOGY

3.1 Data Pre-processing:

Data Pre-processing is a technique that is used to convert the raw data into a clean data set. In this, process the records are gathered from exclusive sources and it is in raw format which contains all noisy and unnecessary data which is not feasible for the analysis. Data pre-processing uses techniques like the removal of noisy data, removal of missing data, filling default values if applicable etc as shown in fig1 are used during amendment of data before feeding it to the algorithm, so here the imputation methods mean, median and standard deviation are used to fill the missing values in the data set.Here the attribute wise mean, median standard deviation is calculated and it is then replaced with the missing values of that corresponding attribute.

Steps in Data Pre-processing:

Step 1: Import the libraries for the data to be pre-processed.

Step 2: The chosen data set to be imported by specifying the path which was saved in excel sheet.

Step 3: Check out the missing values in the chosen dataset.

Step 4: Splitting the data set into Training and Test Set

Step 5: Feature Scaling is processed to evaluate the data uniformly.

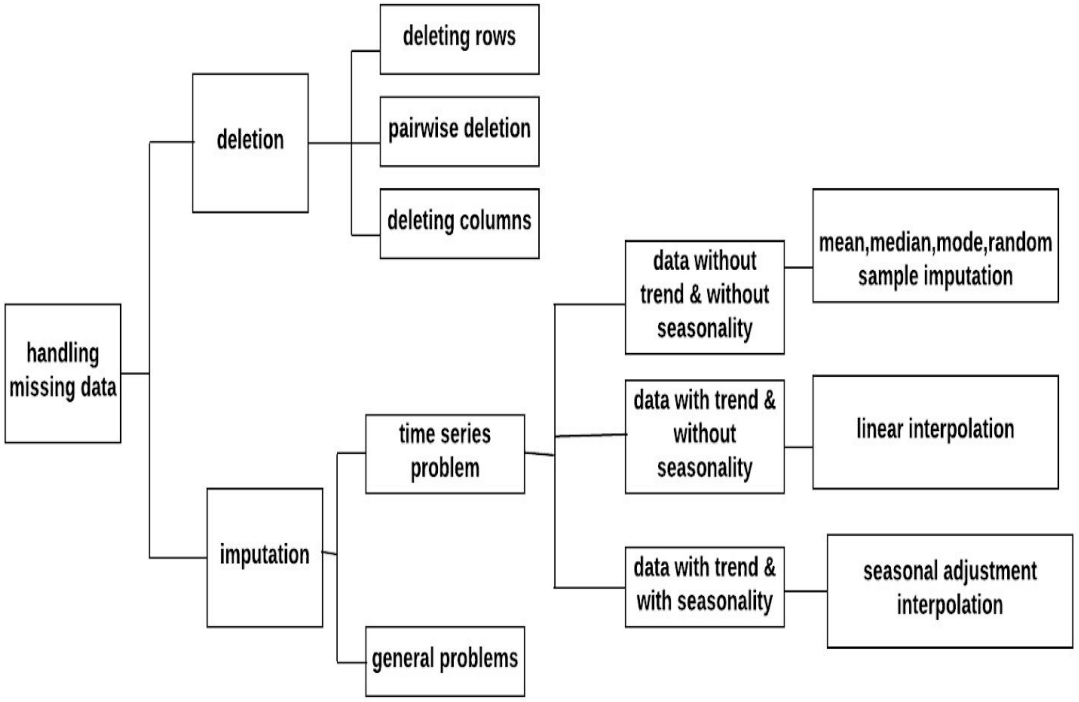

Fig 1: Methods for handling the missing data

3.2 Imputation Measures:

Imputation is the process of replacing missing data with substituted values it preserves all cases by replacing missing data with an estimated value based on other available information.Once all missing values have been imputed, the data set can then be analysed using standard techniques for complete data.A few of the well known attempts to deal with missing data include: hot deck and cold deck imputation,listwise and pairwise deletion,mean imputation, regression imputation,last observation carried forward,stochastic imputation and multiple imputation

3.2.1 Mean:

Mean imputation is a method in which the missing value on a certain variable is replaced by the mean. This method is simple and easy to use but the variability in the data is

reduced. In data pre-processing mean for each attribute is calculated by using formula and replace the mean value in place of missing values .

$$\bar{x} = (\Sigma x_i) / n$$

- $\bar{x}$ indicates for the "sample mean."
- Σ indicates "add up."
- x_i "all of the x-values."
- n means "the number of items in the sample."

3.2.2 Median:

Median imputation is a method in which the missing value on a certain variable is replaced by the median. In this method the data is arranged in numerical order and count how many numbers of numerical data is available in particular attribute. If the data in that attribute is in odd number, then result will be round up to get the position of the median number or vice versa. In data pre-processing for each attribute median is calculated and replace the median value in place of missing values. The median can be calculated by using following formula:

$$\text{Median} = (n + 1) / 2$$

n is the number of items in the set

3.2.3 Standard Deviation:

Standard Deviation imputation is a method in which the missing value on a certain variable is replaced by result of Standard deviation. In this method the average of the data set is calculated.

Take each value in the data set (x) and subtract the mean from it and Square each of the differences, and add up all of the results from to get the sum of squares. Finally divide the

sum of squares by the number of numbers in the data and set to (n – 1). The Standard Deviation can be calculated by using following formula:

$$s = \sqrt{\frac{\sum (x_i - \bar{x})^2}{n-1}}$$

- S indicates standard deviation.
- x indicates mean.
- Σ indicates "add up."
- n indicates "the number of items in the sample."

3.3 Description of dataset:

The biomedical dataset is adapted from the UCI repository. Cleveland dataset is commonly used in data mining and machine learning community. Though this dataset contains 76 attributes, but all the experiments use a subset of 14 of them. It is the integer valued from 0 (no presence) to 4. Experiments with the Cleveland data set have concentrated to distinguish presence (values 1, 2, 3, 4) from absence (value 0).

Only 14 attributes used:

1. #3 (age)
2. #4 (sex)
3. #9 (cp)
4. #10 (trestbps)
5. #12 (chol)
6. #16 (fbs)
7. #19 (restecg)
8. #32 (thalach)
9. #38 (exang)
10. #40 (oldpeak)
11. #41 (slope)
12. #44 (ca)

13. #51 (thal)
14. #58 (num)

3.4 Classification:

Classification consists of predicting a certain outcome based totally on a given input. In order to predict the outcome, the algorithm processes a training set containing a set of attributes and the respective outcome, usually called goal or prediction attribute. The algorithm tries to discover relationships between the attributes that would make it feasible to predict the outcome. During the classification process the dataset that is known as prediction set which contains the same set of attributes, except for the prediction attribute – not yet known is given to classification algorithms. The algorithm analyses the input and produces a prediction for the testing data. The prediction accuracy defines the performance of an algorithm. The classification algorithms is mainly used in a medical database which contains training set that would have relevant patient information all over the world recorded previously in the database, where the prediction attribute is to find whether or not the patient have a heart ailment or not.

3.5 Regression:

Regression analysis is a set of statistical processes for estimating the relationships among variables. It includes many techniques for modelling and analysing several variables, mainly it focuses on the relationship between a dependent variable and one or more independent variables. Regression analysis helps one understand how the typical value of the dependent variable changes when any one of the independent variables is varied, while the other independent variables are held fixed. It is mainly used for prediction and forecasting. Many techniques for carrying out regression analysis have been developed. Familiar methods such as linear regression and ordinary least squares regression are parametric, in that the regression

function is defined in terms of a finite number of unknown parameters that are estimated from the data**.**

3.6 k-Nearest Neighbor (k-NN) algorithm:

The k-Nearest Neighbors (k-NN) algorithm is a supervised machine learning algorithm that can be used to solve both classification and regression problems. It is called supervised learning classification because the process of algorithm learning from the training dataset can be thought to know the correct answers the algorithm iteratively makes predictions on the training data and stops when the algorithm achieves an acceptable level of performance.

k-NN algorithm is one of the simplest classification algorithms. Even with such simplicity, it can give highly competitive results.k-NN algorithm can also be used for regression problems. The only difference from the discussed methodology will be using averages of nearest neighbors rather than voting from nearest neighbors.

k-NN algorithm:

In k-NN algorithm dataset which is pre-processed by using imputation methods will be considered which is used to predict the class labels by the following:

- Initialize k to your chosen number of neighbors.
- Select the k entries in our database which are closest to the new sample.
- Compute the distance between the input sample and training samples by using minkowski distance.
- Now the data is sorted according to the ordered collection of distances from smallest to largest (in ascending order) by the distances.
- Now pick the first k nearest neighbors from the sorted collection of data.
- Finally the labels of the selected k entries is obtained.

Following diagram fig 2 shows the working of k-Nearest Neighbors (k-NN) algorithm and the formation of clusters by initializing k value also shows a spread of red circles (RC) and Green Squares (GS)

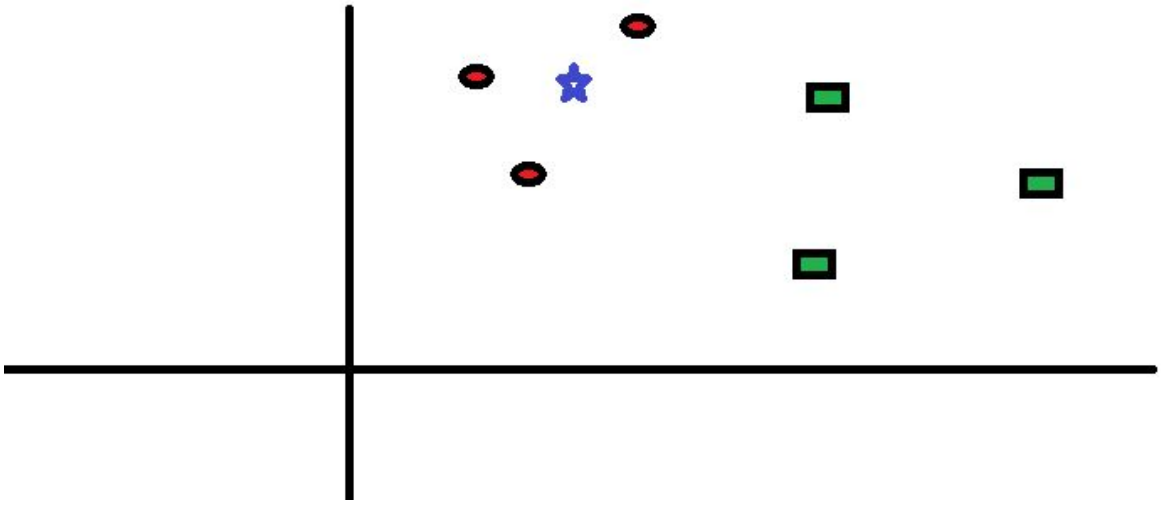

Fig 2: Working principle of k-NN

Now consider the class of the Blue star (BS). Therefore, the blue star can either be Red circle or Green Square and nothing else. The "k" in k-NN algorithm is the nearest neighbors.For example Consider k = 3. Hence, in Fig 3 shows that make a circle by considering Blue star as centre just as big as to enclose only three data points on the plane.

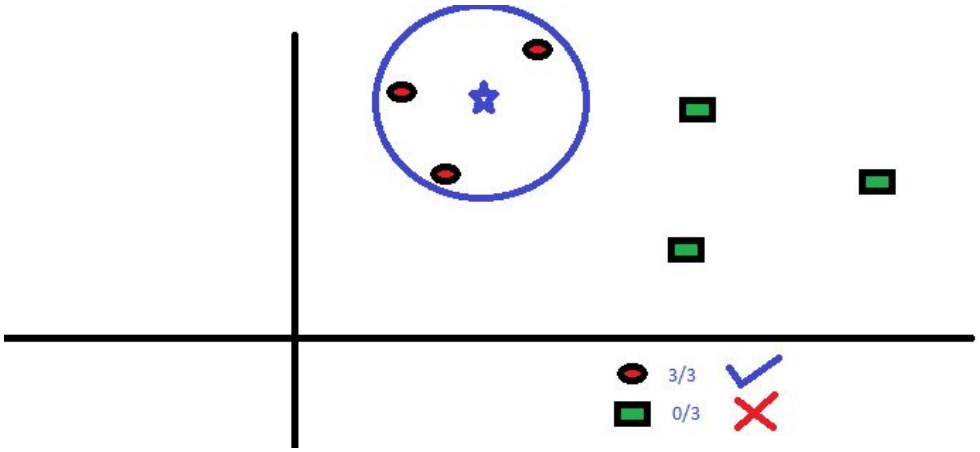

Fig 3: Cluster when k=3

The three closest points to Blue star is all Red circles. Hence,based on good confidence level the Blue star should belong to the class Red circles. Here, the choice became very prominent that all three votes from the closest neighbor went to Red circles. The choice of the parameter k is very crucial in this algorithm.

The main advantage of the k-Nearest Neighbors (k-NN) algorithm is that it is robust to noisy and training data especially when performing Inverse Square of weighted distance.This algorithm is more effective if the training data is large.

3.7 Random Forest:

Random Forest is a flexible, easy to use machine learning algorithm that produces, even without hyper-parameter tuning, a great result most of the time. It is also one of the most used algorithms because its simplicity and the fact that it can be used for both classification and regression tasks.

Working of Random Forest:

Random Forest is a supervised learning algorithm. It creates a forest and makes it somehow random. The Forest it builds is an ensemble of Decision Trees, most of the time trained with the “bagging” method. The general idea of the bagging method is that a combination of learning models increases the overall result. Random Forest builds multiple decision trees and merges them together to get a more accurate and stable prediction.

One of the big advantage of random forest is, that it can be used for both classification and regression problems, which form the majority of current machine learning systems.Classification is sometimes considered as the building block of machine learning. Hence fig 4 shows how a random forest would look like with two trees:

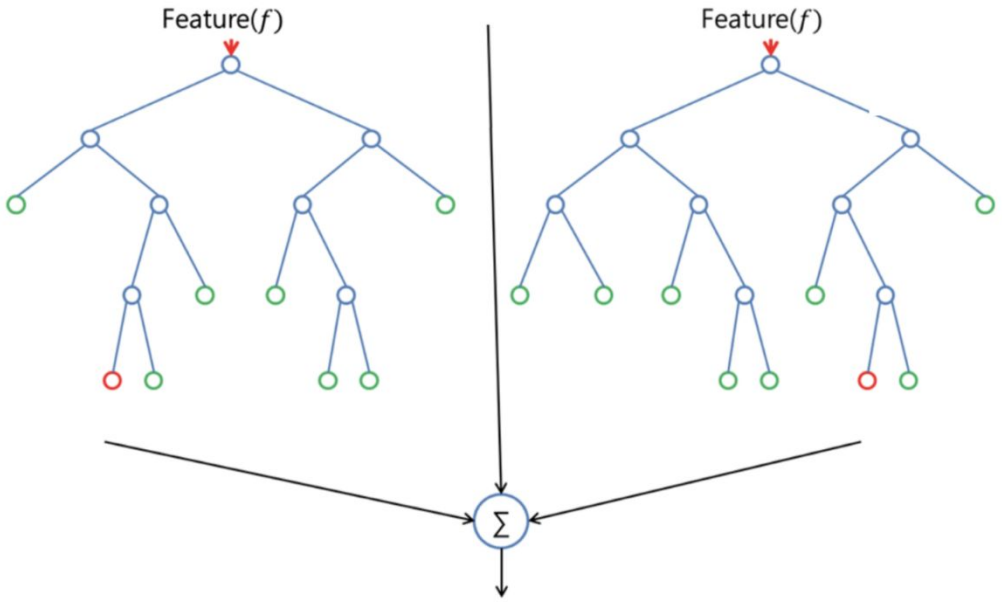

Fig 4: Random Forest with two trees

Random Forest has nearly the same hyper parameters as a decision tree or a bagging classifier. Fortunately, the decision tree with a bagging classifier and can just easily use the classifier-class of Random Forest.Random forest, one can deal with Regression tasks by using the Random Forest regressor. Random Forest adds additional randomness to the model, while growing the trees. Instead of searching for the most important feature while splitting a node, it searches for the best feature among a random subset of features. This results in a wide diversity that generally results in a better model.

Therefore, in Random Forest, only a random subset of the features is taken into consideration by the algorithm for splitting a node. You can even make trees more random, by additionally

using random thresholds for each feature rather than searching for the best possible thresholds (like a normal decision tree does).

Algorithm:

It works in four steps:

- Select random samples from a given dataset.
- Construct a decision tree for each sample and get a prediction result from each decision tree.
- Perform a vote for each predicted result.
- Select the prediction result with the most votes as the final prediction.

3.8 LIBRARIES

Numpy:

Numpy is the core library for scientific computing in Python.It provides a high-performance multidimensional array object, and tools for working with these arrays. Arrays in Numpy can be created by multiple ways, with various numbers of ranks, defining the size of the array,by defining various data types such as lists, tuples, etc. Here, the data types of arrays need not to be defined unless a specific data type is required. Numpy tries to guess the data type for arrays which are not predefined in the constructor function. Numpy arrays, basic mathematical operations are performed element-wise on the array. These operations are applied both as operator overloads and as functions. Many useful functions are provided in Numpy for performing computations on arrays such as **sum**: for addition of Array elements, **T**: for Transpose of elements, etc.

Matplotlib.pyplot:

Matplotlib.pyplot is a collection of command style functions that make matplotlib work like matlab. Each pyplot function makes some change to a figure: e.g, creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc. In matplotlib.pyplot various states are preserved across function calls, so that it keeps track of things like the current figure and plotting area, and the plotting functions are directed to the

current axes "axes" here and in most places in the documentation refers to the axes part.Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack.

Pandas:

Pandas is an open source python package that provides numerous tools for data analysis. The package comes with several data structures that can be used for many different data manipulation tasks. It also has a variety of methods that can be invoked for data analysis, which comes in handy when working on data science and machine learning problems in Python. It is a BSD(Berkeley software distribution)-licensed Python library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, statistics, analytics, etc.

Train_test_split:

The original dataset should be split up into training and testing data. For example, 80% of the data could be used for training and 20% could be used for testing. The data is split so that there is data for the model to be evaluated on to see how well the model performs on unknown data.

- **Training:** This data is used to build corresponding model. E.g. finding the optimal coefficients in a Linear Regression model or using the CART algorithm to create a Decision Tree.

- **Testing:** This data is used to see how the model performs on unknown data, as it would do in a real world situation. This data should be left completely unseen until to test corresponding model to evaluate performance.

Sci-Kit Learn:

Scikit-learn provide a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use. The scikit-learn project started as scikits.learn, a Google Summer of Code project by David Cournapeau. Its name stems from the notion that it is a "SciKit" (SciPy Toolkit), a separately-developed and distributed third-party extension to SciPy.

CHAPTER-4

IMPLEMENTATION

CHAPTER-4

IMPLEMENTATION

4.1 Data pre processing

Import libraries

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.preprocessing import Imputer
from sklearn import preprocessing
```

Importing the Dataset

To import the dataset and load it into our pandas data frame, execute the following code:

```
file="Desktop\sai.csv"
df= pd.read_csv(file,sep=',')
df.head(n=20)
```

	0	1	2	3	4	5	6	7
0	1	0	63	1	1	1	1	NaN
1	-9	4	140	0	260	0	0	0
2	0	-9	0	1	1	22	85	0
3	0	1	0	0	5	4.5	-9	5
4	112	62	160	90	140	80	1	0

5	3	2	20	19	-9	-9	0	-9
6	-9	-9	-9	-9	-9	-9	-9	2
7	27	85	2	1	1	2	1	1
8	1	1	1	2	1	1	1	1
9	1	1	0.7	5.5	name	NaN	NaN	NaN
10	2	0	44	1	1	1	1	NaN
11	-9	4	130	0	209	0	20	10
12	0	1	0	1	7	23	84	0
13	0	1	0	0	5	2.5	-9	2
14	127	73	150	80	130	70	0	0
15	0	-9	-9	-9	-9	-9	-9	-9
16	-9	-9	-9	-9	-9	-9	-9	7
17	31	84	0	1	1	1	1	1
18	1	1	1	1	1	1	1	1
19	1	1	0.5	-9	name	NaN	NaN	NaN

To replace all the 'name' records with the value '1', and to display the data, execute the following code:

```
mymap={'name':1}
m=df.applymap(lambda s: mymap.get(s) if s in mymap else s)
print (m)
```

	0	1	2	3	4	5	6	7
0	1	0	63	1	1	1	1	NaN
1	-9	4	140	0	260	0	0	0
2	0	-9	0	1	1	22	85	0
3	0	1	0	0	5	4.5	-9	5
4	112	62	160	90	140	80	1	0

5	3	2	20	19	-9	-9	0	-9
6	-9	-9	-9	-9	-9	-9	-9	2
7	27	85	2	1	1	2	1	1
8	1	1	1	2	1	1	1	1
9	1	1	0.7	5.5	1	NaN	NaN	NaN
10	2	0	44	1	1	1	1	NaN
11	-9	4	130	0	209	0	20	10
12	0	1	0	1	7	23	84	0
13	0	1	0	0	5	2.5	-9	2
14	127	73	150	80	130	70	0	0
15	0	-9	-9	-9	-9	-9	-9	-9
16	-9	-9	-9	-9	-9	-9	-9	7
17	31	84	0	1	1	1	1	1
18	1	1	1	1	1	1	1	1
19	1	1	0.5	-9	1	NaN	NaN	NaN
20	3	0	60	1	1	1	1	NaN
21	-9	4	132	1	218	1	40	40
22	0	-9	0	1	6	12	84	0
23	0	1	0	-9	5	6	-9	6
24	140	68	210	110	132	80	1	0
25	1.5	3	21	20	-9	-9	-9	-9
26	-9	-9	-9	-9	-9	-9	-9	7
27	30	84	2	1	1	1	1	2
28	1	2	1	2	1	1	1	1
29	7	2	0.52	4.1	1	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...
1970	202	0	55	1	1	1	1	NaN
1971	-9	4	122	1	223	1	20	40
1972	1	-9	0	1	5	2	86	0
1973	1	1	0	1	5	5.3	-9	5
1974	100	74	210	100	122	70	0	0
1975	0	-9	6	4	-9	-9	-9	0.39
1976	3	-9	-9	6	2	-9	-9	4
1977	17	86	2	1	2	1	1	1
1978	1	1	1	2	1	1	1	1
1979	1	1	0.69	5.6	1	NaN	NaN	NaN
1980	116	0	58	1	1	1	1	NaN
1981	-9	4	-9	0	385	0	10	20

1982	1	1	1	2	-9	-9	-9	-9
1983	-9	-9	-9	-9	-9	-9	-9	-9
1984	-9	-9	-9	-9	-9	-9	-9	-9
1985	-9	-9	-9	-9	-9	-9	-9	-9
1986	-9	-9	-9	-9	-9	-9	-9	2
1987	16	83	0	1	1	1	1	1
1988	1	1	1	1	1	1	1	1
1989	1	1	0.81	6	1	NaN	NaN	NaN
1990	160	0	62	1	1	0	0	NaN
1991	-9	2	120	1	254	0	0	0
1992	0	-9	0	2	1	24	83	0
1993	1	0	0	0	1	6.7	-9	7
1994	93	67	164	110	120	80	1	0
1995	0	-9	21	17	-9	-9	-9	-9
1996	-9	-9	-9	-9	-9	-9	-9	6
1997	20	83	1	1	1	1	1	1
1998	1	1	1	2	1	1	1	1
1999	1	1	-9	-9	1	NaN	NaN	NaN

[2000 rows x 8 columns]

To calculate the attribute wise mean, execute the following code:

```
m.mean()
```

```
0    18.677600
1    11.620050
2    26.448395
3    6.330250
5    5.707667
6    6.361056
```

7 2.478419

dtype: float64

To fill the missing values with the obtained mean use the following code:

```
m.fillna(m.mean())
```

	0	1	2	3	4	5	6	7
0	1	0	63	1	1	1	1	2.478419
1	-9	4	140	0	260	0	0	0
2	0	-9	0	1	1	22	85	0
3	0	1	0	0	5	4.5	-9	5
4	112	62	160	90	140	80	1	0
5	3	2	20	19	-9	-9	0	-9
6	-9	-9	-9	-9	-9	-9	-9	2
7	27	85	2	1	1	2	1	1
8	1	1	1	2	1	1	1	1
9	1	1	0.7	5.5	1	5.707667	6.361056	2.478419
10	2	0	44	1	1	1	1	2.478419
11	-9	4	130	0	209	0	20	10
12	0	1	0	1	7	23	84	0
13	0	1	0	0	5	2.5	-9	2
14	127	73	150	80	130	70	0	0
15	0	-9	-9	-9	-9	-9	-9	-9

16	-9	-9	-9	-9	-9	-9	-9	7
17	31	84	0	1	1	1	1	1
18	1	1	1	1	1	1	1	1
19	1	1	0.5	-9	1	5.707667	6.361056	2.478419
20	3	0	60	1	1	1	1	2.478419
21	-9	4	132	1	218	1	40	40
22	0	-9	0	1	6	12	84	0
23	0	1	0	-9	5	6	-9	6
24	140	68	210	110	132	80	1	0
25	1.5	3	21	20	-9	-9	-9	-9
26	-9	-9	-9	-9	-9	-9	-9	7
27	30	84	2	1	1	1	1	2
28	1	2	1	2	1	1	1	1
29	7	2	0.52	4.1	1	5.707667	6.361056	2.478419
...	...	...	...	...	...	...	...	...
1970	202	0	55	1	1	1	1	2.478419
1971	-9	4	122	1	223	1	20	40
1972	1	-9	0	1	5	2	86	0
1973	1	1	0	1	5	5.3	-9	5
1974	100	74	210	100	122	70	0	0
1975	0	-9	6	4	-9	-9	-9	0.39
1976	3	-9	-9	6	2	-9	-9	4
1977	17	86	2	1	2	1	1	1
1978	1	1	1	2	1	1	1	1

1979	1	1	0.69	5.6	1	5.707667	6.361056	2.478419
1980	116	0	58	1	1	1	1	2.478419
1981	-9	4	-9	0	385	0	10	20
1982	1	1	1	2	-9	-9	-9	-9
1983	-9	-9	-9	-9	-9	-9	-9	-9
1984	-9	-9	-9	-9	-9	-9	-9	-9
1985	-9	-9	-9	-9	-9	-9	-9	-9
1986	-9	-9	-9	-9	-9	-9	-9	2
1987	16	83	0	1	1	1	1	1
1988	1	1	1	1	1	1	1	1
1989	1	1	0.81	6	1	5.707667	6.361056	2.478419
1990	160	0	62	1	1	0	0	2.478419
1991	-9	2	120	1	254	0	0	0
1992	0	-9	0	2	1	24	83	0
1993	1	0	0	0	1	6.7	-9	7
1994	93	67	164	110	120	80	1	0
1995	0	-9	21	17	-9	-9	-9	-9
1996	-9	-9	-9	-9	-9	-9	-9	6
1997	20	83	1	1	1	1	1	1
1998	1	1	1	2	1	1	1	1
1999	1	1	-9	-9	1	5.707667	6.361056	2.478419

[2000 rows x 8 columns]

4.2 k-Nearest Neighbor:

Importing Libraries

```
import numpy as np

import matplotlib.pyplot as plt

import pandas as pd
```

Importing the Dataset

To import the dataset and load it into pandas dataframe, execute the following code:

```
df = pd.read_csv('Desktop\clever.csv')
```

To see what the dataset actually looks like, execute the following command:

```
df.head()
```

Executing the above script will display the first five rows of dataset as shown below:

```
Unnamed: 0 A1 A2 A3 A4 A5 A6 A7 A8 A9 A10 B1 B2 B3 B4

 0 67 14 160 286 0 2 108 1 1.5 2 3 3 2

1 67 1 4   120 229 0 2 129 12.6 2 2 7 1
```

```
2 37 1 3 130 250 0 0 187 0 3.5 3 0 3 0

 3 41 0 2 130 204 0 2 172 0 1.4 1 0 3 0

4 56 1 2 120 236 0 0 178 0 0.8 1 0 3 0
```

Pre-processing

The next step is to split the dataset into corresponding attributes and labels. To do so, use the following code

```
X = df.iloc[:, :-1].values

 y = df.iloc[:, 14].values
```

Train Test Split

To create training and test splits, execute the following script:

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20)
```

The above script splits the dataset into 80% train data and 20% test data. This means that out of total 303 records, the training set will contain 242 records and the test set contains 60 of those records.

The following script performs feature scaling:

```
from sklearn.preprocessing import Standard

Scalerscaler = StandardScaler()

scaler.fit(X_train)

X_train = scaler.transform(X_train)

X_test = scaler.transform(X_test)
```

Training and Predictions

It is extremely straight forward to train the kNN algorithm and make predictions with it, especially when using Scikit-Learn. The first step is to import the k Neighbors classifiers class from the sklearn.neighbors library. In the second line, this class is initialized with one parameter, i.e., n_neighbors. This is basically the value for the k. There is no ideal value for k and it is selected after testing and evaluation, however to start out, it seems to be the most commonly used value for KNN algorithm.

```
from sklearn.neighbors import KNeighborsClassifier

classifier = KNeighborsClassifier(n_neighbors=25)

classifier.fit(X_train, y_train)
```

The final step is to make predictions on the test data. To do so, execute the following script:

```
KNeighborsClassifier(algorithm='auto',leaf_size=30,metric='minkowski',

metric_params=None, n_jobs=1, n_neighbors=25, p=2, weights='uniform')
```

```
y_pred = classifier.predict(X_test)
```

The output of the above script looks like this:

```
Fromsklearn.metricsimportclassification_report,confusion_matrix

print(confusion_matrix(y_test, y_pred))

rint(classification_report(y_test, y_pred))

[31 4 1 0 0]

 [ 7 2 0 0 0]

[ 1 3 3 0 0]

[ 3 4 0 1 0]

[ 0 1 0 0 0]
```

precision recall f1-score support

0 0.74 0.86 0.79 36

1 0.14 0.22 0.17 9

2 0.75 0.43 0.557

31.00 0.12 0.22 8

40.00 0.000.001

avg / total 0.830.61 0.59 61

4.3 RANDOM FOREST:

Importing Libraries

```
import numpy as np

import matplotlib.pyplot as plt

import pandas as pd
```

Importing the Dataset

To import the dataset and load it into the pandas dataframe, execute the following code:

```
df = pd.read_csv('Desktop\clever.csv')
```

To see what the dataset actually looks like, execute the following command:

```
df.head()
```

```
Unnamed: 0 A1 A2 A3 A4 A5 A6 A7 A8 A9 A10B1 B2 B3 B4
0 067 1 4 160 286 0 2 108 1 1.5 2 33 2
11 67 1 4 120 229 0 2 129 1 2.6 2 2 7 1
2 2 37 1 3 130 250 0 0187 0 3.5 3 0 3 0
3 3 41 0 2 130 204 0 2 172 0 1.4 1 0 3 0
4 4 56 1 2 120 236 0 0 178 0 0.8 1 0 3 0
```

Preprocessing

The next step is to split the dataset into corresponding attributes and labels. To do so, use the following code:

```
X = df.iloc[:, :-1].values

y = df.iloc[:,13].values
```

The X variable contains the first four columns of the dataset (i.e. attributes) while y contains the labels.

To create training and test splits, execute the following script:

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.40)
```

The above script splits the dataset into 60% train data and 40% test data.

Feature Scaling

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

scaler.fit(X_train)
```

```
X_train= scaler.transform(X_train)

X_test= scaler.transform(X_test)
```

After splitting,the model is trained on the training set and perform predictions on the test set.

```
clf=RandomForestClassifier(n_estimators=100)
```

Train the model using the training sets.

```
clf.fit(X_train,y_train)

y_pred=clf.predict(X_test)
```

After training, check the accuracy using actual and predicted values.

```
from sklearn import metrics

print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.9421487603305785
```

Train Test Split

To avoid over-fitting, divide the dataset into training and test splits, which gives us a better idea as to how the algorithm performed during the testing phase. This way the corresponding algorithm is tested on un-seen data, as it would be in a production application

Feature Scaling

Before making any actual predictions, it is always a good practice to scale the features so that all of them can be uniformly evaluated.Since the range of values of raw data varies widely, in some machine learning algorithms, objective functions will not work properly without normalization. For example, the majority of classifiers calculate the distance between two points by the Euclidean distance. If one of the features has a broad range of values, the distance will be governed by this particular feature. Therefore, the range of all features should

be normalized so that each feature contributes approximately proportionately to the final distance.

The gradient descent algorithm (which is used in neural network training and other machine learning algorithms) also converges faster with normalized features.

CHAPTER-5

RESULTS

5. RESULTS

The results of experiments reveal that Random forest and K-Nearest Neighbor with 94% and 83% in mean,97% and 93% in median,93% and 90% in standard deviation have the highest and lowest accuracy of these classifiers tested respectively. Thus Random forest have highest accuracy when compared to K-Nearest Neighbor.It can be concluded that there is a huge scope of providing accuracy is more for Random forest than the k-Nearest Neighbour according to the results shown correspondingly in imputation measure as shown in the following table.

	Knn	Random Forest
Mean	83%	94%
Median	93%	97%
Standard Deviation	90%	93%

Table1:Comparison of kNN and Random Forest

CHAPTER-6

CONCLUSION

6. CONCLUSION

Heart diseases are one of the major problems of public health all over the world. An accurate prediction model for Heart disease can be a very useful to save the lives of patients. It accounts for 7.2 million deaths, i.e., 12.8% of fatalities in the world. Although heart diseases have been identified as the leading cause of death in the past decade, they are the most preventable and controllable diseases at the same time. Deaths from heart diseases show an ever-increasing trend. On the other hand, their early diagnosis plays an important role in improving patients health status .Machine learning algorithms are used for predicting the uncertainty levels of cardiovascular diseases based on the data present. Based on the above review, it can be concluded that there is a huge scope of providing accuracy is more for Random Forest than the k-Nearest Neighbor (k-NN) algorithm which is used to predict the heart disease in vital time by correctly identifying the disease in an initial stage by measuring perfectly and precisely.

6.1 Future Work:

This technique is expected to be implemented on a localized dataset with non-aggressive indices. By using PCA(Principal Component Analysis) - the attribute size can be reduced, by which the accuracy will be increased and the time complexity is reduced.This in turn, imposes lower costs on patients.

REFERENCES

1. Prediction of Cardiovascular Disease Using Machine Learning Algorithms byDinesh Kumar G, Santhosh Kumar, Arumugaraj K andMareeswari V in 2018.
2. Preprocessing Unbalanced Data using Weighted Support Vector Machines for Prediction of Heart Disease in Children by Thiago R. Tavares, Adriano L. I. Oliveira, George G. Cabral, Sandra S. Mattos and RenataGrigorio in 2016.
3. A Comprehensive Investigation and Comparison of Machine Learning Techniques in the Domain of Heart Disease by SeyedaminPouriyeh, Sara Vahid, Giovanna Sanninoy, Giuseppe De Pietroy, Hamid Arabnia, Juan Gutierrez in 2012.
4. A Review on Heart Disease Diagnosis and Prediction Using Machine Learning and Data Mining Techniques by AnimeshHazra, Subrata Kumar Mandal, Amit Gupta, Arkomita Mukherjee and Asmita Mukherjee in 2005 .
5. Identification of Heart Failure by Using Unstructured Data of Cardiac Patients by Muhammad Saqlain, Wahid Hussain, Nazar A. Saqib, Muazzam A. Khan in 2016.
6. Analytical Study of Heart Disease Prediction Comparing With Different Algorithms by Sana Bharti, Dr.Shaliendra Narayan Singh in 2015 IEEE.
7. Jaymin Patel, Prof.TejalUpadhyay, Dr.Samir Patel “Heart disease prediction using Machine learning and Data Mining Technique" Volume 7.Number1 Sept 2015March 2016.
8. G.Parthiban, S.K.Srivasta "Applying Machine learning methods in Diagnosing Heart disease for Diabetic Patients" International Journal ofApplied Information Systems (IJAIS) – ISSN: 2249-0868 Foundation of Computer Science FCS, New York, USA Volume 3– No.7, August 2012.
9. B.DhomseKanchan, M.Mahale Kishore “Study of Machine learning algorithms for special disease prediction using principal of component analysis” Global Trends in Signal Processing and information computing and Communication (ICGTSPICC),2016InternationalConference.

10. H. Ocak, "A medical decision support system based on support vector machines and the genetic algorithm for the evaluation of fetal wellbeing," Journal of medical systems, vol. 37, no. 2, pp. 1–9, 2013.
11. C. Patterson, E. Nicol, L. Bryan, T. Woodcock, J. Collinson, S. Padley, and D. Bell, "The effect of applying nice guidelines for the investigation of stable chest pain on out-patient cardiac services in the uk," QJM, vol. 104, no. 7, pp. 581–588, 2011.
12. A. Cooper, A. Timmis, and J. Skinner, "Assessment of recent onset chest pain or discomfort of suspected cardiac origin: summary of nice guidance," BMJ, vol. 340, 2010.
13. M. AkhilJabbar, B.L Deekshatulu&Priti Chandra, " Classification of Heart Disease using Artificial Neural Network and Feature Subset Selection", Global Journal of Computer Science and Technology Neural & Artificial Intelligence, Volume 13 Issue 3 Version 1.0 Year 2013.
14. Ms. Preeti Gupta ,Ms. Punam Bajaj, " Heart Disease Diagnosis System Based On Data Mining And Neural Network", international journal of engineering sciences & research technology, June, 2014.
15. T.John Peter, K. Somasundaram, " an empirical study on prediction of heart disease using classification data mining techniques", IEEE-International Conference On Advances In Engineering, Science And Management (ICAESM -2012) March 30, 31, 2012.
16. Y. Wei, T. Liu, R. Valdez, M. Gwinn and M.J. Khoury,"Application of support vector machine modeling for prediction of common diseases: the case of diabetes and pre-diabetes," BMC Medical Informatics & Decision Making, vol. 10, 2010.
17. Milan Kumari and S. Godara, "Comparative Study of Data Mining Classification Methods in Cardiovascular Disease Prediction," IJCST, vol. 2, pp. 304-309, June 2013.
18. J. Wu, J. Roy, and W.F. Stewart, "Prediction modeling using EHR data: challenges, strategies, and a comparison of machine learning approaches," Med Care, vol. 48, pp. 106-113, June 2010.

19. E. AbuKhousa, and P. Campbell, "Predictive data mining to support clinical decisions: An overview of heart disease prediction systems," Proc. IEEE, Innovations Information Technology (IIT), pp. 267-272, March 2012.
20. Vikas Chaurasia, Saurabh Pal, - "Early Prediction of Heart Diseases Using Data Mining Techniques", Carib.J.SciTech, 2013, Vol. 1, 208-2017.
21. Jyoti Soni, Ujma Ansari, Dipesh Sharma and Sunita Soni, "Predictive Data Mining for Medical Diagnosis: An Overview of Heart Disease Prediction", International Journal of Computer Applications (0975 – 8887) Volume 17– No.8, March 2011.
22. M. Pechenizkiy, A. Tsymbal and S. Puuronen, "PCA-based feature transformation for classification: issues in medical diagnostics", IEEE, Computer-Based Medical Systems, 2004. CBMS 2004. Proceedings (1063- 7125), Page No. 535 – 540, June 2004.
23. V. Chaurasia and S. Pal, "Data Mining Approach to Detect Heart Diseases", International Journal of Advanced Computer Science and Information Technology (IJACSIT), Vol. 2, No. 4, 2013, Page 56-66.
24. S , Liver Disease Prediction Using Bayesian Classification , Special Issues , 4th National Conference on Advance Computing , Application Technologies, May 2014.
25. Selzer, A. (1992). Understanding Heart Disease. Berkeley: University of California Press.