GALGOTIAS UNIVERSITY

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

HEART DISEASE PREDICTION

A Report for the Evaluation 3 of Project 2

Submitted by

VAIBHAV

GUPTA

(1613101802)

in partial fulfilment for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

Under the Supervision of

Dr. PALLAVI MURGHAI GOEL ,

Assosiate Professor

APRIL / MAY- 2020

SCHOOL OF COMPUTING AND SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

Certified that this project report **"HEART DISEASE PREDICTION"** is the bonafide work of "**VAIBHAV GUPTA (1613101802)"** who carried out the project work under my supervision.

SIGNATURE OF HEAD

Dr. MUNISH SHABARWAL,
PhD (Management), PhD (CS)
Professor & Dean,
School of Computing Science & Engineering

SIGNATURE OF SUPERVISOR

Dr. PALLAVI MURGHAI GOEL,
Associate Professor
School of Computing Science & Engineering

TABLE OF CONTENTS

ABSTRACT

Heart disease is a major cause of death throughout the world. It is difficult to predict by medical practitioners as it requires expertise and higher knowledge of prediction. The environment in healthcare sector is information rich but lacks knowledge. A lot of data is available in healthcare systems over the internet but there is a lack of effective analysis tool to discover hidden patterns in data. An automated system will enhance medical efficiency and reduce cost and time. This software intends to predict the occurrence of a disease based on the data which is gathered from kaggle. The objective is to extract the hidden patterns by applying data mining techniques on the dataset and to predict the presence value on a scale. The prediction of heart disease requires a huge size of data which is too massive and complex to process and analyse by conventional technique. Our aim is to find out an suitable technique that is efficient and accurate for prediction of cardiac disease.

Keywords- prediction, heart disease, machine learning, algorithms, analysis

1. INTRODUCTION

1.1 General

Among various life-threatening diseases, heart disease has garnered a great deal of attention in medical research. The diagnosis of heart disease is a challenging task, which can offer automated prediction about the heart condition of patient so that further treatment can be made effective. The diagnosis of heart disease is usually based on signs, symptoms and physical examination of the patient. There are several factors that increase the risk of heart disease, such as smoking habit, body cholesterol level, family history of heart disease, obesity, high blood pressure, and lack of physical exercise.

A major challenge faced by health care organizations, such as hospitals and medical centers, is the provision of quality services at affordable costs.1 The quality service implies diagnosing patients properly and administering effective treatments. The available heart disease database consists of both numerical and categorical data. Before further processing, cleaning and filtering are applied on these records in order to filter the irrelevant data from the database.2 The proposed system can determine an exact hidden knowledge, ie, patterns and relationships associated with heart disease from a historical heart disease database. It can also answer the complex queries for diagnosing heart disease; therefore, it can be helpful to health care practitioners to make intelligent clinical decisions. Results showed that the proposed system has its unique potency in realizing the objectives of the defined machine learning goals.

1.2 MACHINE LEARNING

1.2.1 General

Machine Learning is the field of study that gives computers the capability to learn without being explicitly programmed. ML is one of the most exciting technologies that one would have ever come across. As it is evident from the name, it gives the computer that makes it more similar to humans: The ability to learn. Machine learning is actively being used today, perhaps in many more places than one would expect.

Fig 1. Machine Learning

The term Machine Learning was coined by Arthur Samuel in 1959, an American pioneer in the field of computer gaming and artificial intelligence and stated that "it gives computers the ability to learn without being explicitly programmed".

And in 1997, Tom Mitchell gave a "well-posed" mathematical and relational definition that "A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E.

Machine learning involves a computer to be trained using a given data set, and use this training to predict the properties of a given new data. For example, we can train a computer by feeding it 1000 images of cats and 1000 more images which are not of a cat, and tell each time to the computer whether a picture is cat or not. Then if we show the computer a new image, then from the above training, the computer should be able to tell whether this new image is a cat or not.

Let’s try to understand Machine Learning in layman terms. Consider you are trying to

toss a paper to a dustbin.

After first attempt, you realize that you have put too much force in it. After second attempt, you realize you are closer to target but you need to increase your throw angle. What is happening here is basically after every throw we are learning something and improving the end result. We are programmed to learn from our experience.

This implies that the tasks in which machine learning is concerned offers a fundamentally operational definition rather than defining the field in cognitive terms. This follows Alan Turing’s proposal in his paper “Computing Machinery and Intelligence”, in which the question “Can machines think?” is replaced with the question “Can machines do what we (as thinking entities) can do?” Within the field of data analytics, machine learning is used to devise complex models and algorithms that lend themselves to prediction; in commercial use, this is known as predictive analytics. These analytical models allow researchers, data scientists, engineers, and analysts to “produce reliable, repeatable decisions and results” and uncover “hidden insights” through learning from historical relationships and trends in the data set(input).

Suppose that you decide to check out that offer for a vacation . You browse through the travel agency website and search for a hotel. When you look at a specific hotel, just below the hotel description there is a section titled “You might also like these hotels”. This is a common use case of Machine Learning called “Recommendation Engine”. Again, many data points were used to train a model in order to predict what will be the best hotels to show you under that section, based on a lot of information they already know about you.

So if you want your program to predict, for example, traffic patterns at a busy intersection (task T), you can run it through a machine learning algorithm with data

about past traffic patterns (experience E) and, if it has successfully "learned", it will then do better at predicting future traffic patterns (performance measure P). The highly complex nature of many real-world problems, though, often means that inventing specialized algorithms that will solve them perfectly every time is impractical, if not impossible. Examples of machine learning problems include, "Is this cancer?", "Which of these people are good friends with each other?", "Will this person like this movie?" such problems are excellent targets for Machine Learning, and in fact machine learning has been applied such problems with great success.

When do we need Machine Learning?

When do we need machine learning rather than directly program our computers to carry out the task at hand? Two aspects of a given problem may call for the use of programs that learn and improve on the basis of their "experience": the problem's complexity and the need for adaptivity.

Tasks That Are Too Complex to Program.

• Tasks Performed by Animals/Humans: There are numerous tasks that we human beings perform routinely, yet our introspection concerning how we do them is not sufficiently elaborate to extract a well-defined program. Examples of such tasks include driving, speech recognition, and image understanding. In all of these tasks, state of the art machine learning programs, programs that "learn from their experience," achieve quite satisfactory results, once exposed to sufficiently many training examples.

• Tasks beyond Human Capabilities: Another wide family of tasks that benefit from machine learning techniques are related to the analysis of very large and complex data sets: astronomical data, turning medical archives into medical knowledge, weather prediction, analysis of genomic data, Web search engines, and electronic commerce. With more and more available digitally recorded data, it becomes obvious that there are treasures of meaningful information buried in data archives that are way too large and too complex for humans to make sense of. Learning to

detect meaningful patterns in large and complex data sets is a promising domain in which the combination of programs that learn with the almost unlimited memory capacity and ever increasing processing speed of computers opens up new horizons. Adaptivity. One limiting feature of programmed tools is their rigidity – once the program has been written down n and installed, it stays unchanged. However, many tasks change over time or from one user to another. Machine learning tools – programs whose behavior adapts to their input data – offer a solution to such issues; they are, by nature, adaptive to changes in the environment they interact with. Typical successful applications of machine learning to such problems include programs that decode handwritten text, where a fixed program can adapt to variations between the handwriting of different users; spam detection programs, adapting automatically to changes in the nature of spam e-mails; and speech recognition programs.

Terminologies of Machine Learning

- **Model**

  A model is a specific representation learned from data by applying some machine learning algorithm. A model is also called hypothesis.

- **Feature**

  A feature is an individual measurable property of our data. A set of numeric features can be conveniently described by a feature vector. Feature vectors are fed as input to the model. For example, in order to predict a fruit, there may be features like color, smell, taste, etc.

  Note: Choosing informative, discriminating and independent features is a crucial step for effective algorithms. We generally employ a feature extractor to extract the relevant features from the raw data.

- **Target (Label)**

  A target variable or label is the value to be predicted by our model. For the fruit example discussed in the features section, the label with each set of input would be the name of the fruit like apple, orange, banana, etc.

- **Training**

  The idea is to give a set of inputs(features) and it's expected outputs(labels), so after training, we will have a model (hypothesis) that will then map new data to one of the categories trained on.

- **Prediction**

  Once our model is ready, it can be fed a set of inputs to which it will provide a predicted output(label).

The figure shown below clears the above concepts:

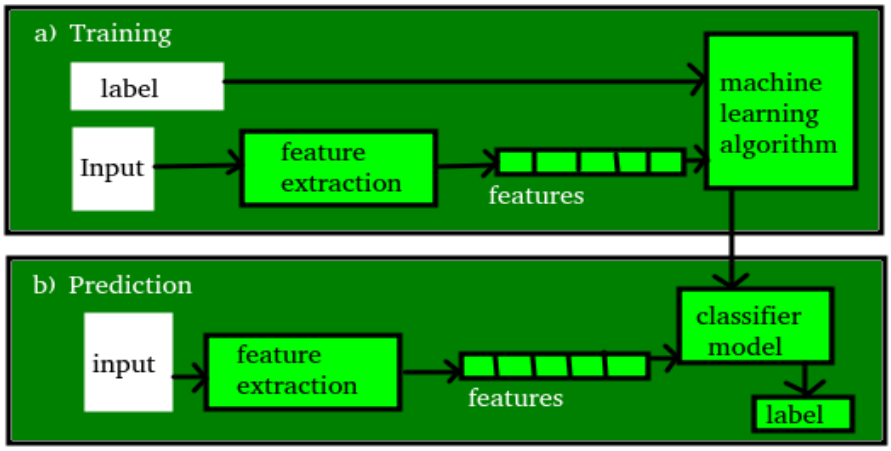


Fig-2 Training and Prediction

1.2.2 Machine Learning Types

1.2.2.1 General

Learning is, of course, a very wide domain. Consequently, the field of machine learning has branched into several subfields dealing with different types of learning tasks. We give a rough taxonomy of learning paradigms, aiming to provide some perspective of where the content sits within the wide field of machine learning.

Terms frequently used are:

- Labeled data: Data consisting of a set of training examples, where each example is a pair consisting of an input and a desired output value (also called the supervisory signal, labels, etc)

- Classification**:** The goal is to predict discrete values, e.g. {1,0}, {True, False}, {spam, not spam}.
- Regression**:** The goal is to predict continuous values, e.g. home prices.

There some variations of how to define the types of Machine Learning Algorithms but commonly they can be divided into categories according to their purpose and the main categories are the following:

- Supervised learning
- Unsupervised Learning
- Semi-supervised Learning
- Reinforcement Learning

1.2.2.2 Supervised Learning

- I like to think of supervised learning with the concept of function approximation, where basically we train an algorithm and in the end of the process we pick the function that best describes the input data, the one that for a given X makes the best estimation of y (X -> y). Most of the time we are not able to figure out the true function that always make the correct predictions and other reason is that the algorithm rely upon an assumption made by humans about how the computer should learn and this assumptions introduce a bias.
- Here the human experts act as the teacher where we feed the computer with training data containing the input/predictors and we show it the correct answers (output) and from the data the computer should be able to learn the patterns.

- Supervised learning algorithms try to model relationships and dependencies between the target prediction output and the input features such that we can predict the output values for new data based on those relationships which it learned from the previous data sets.

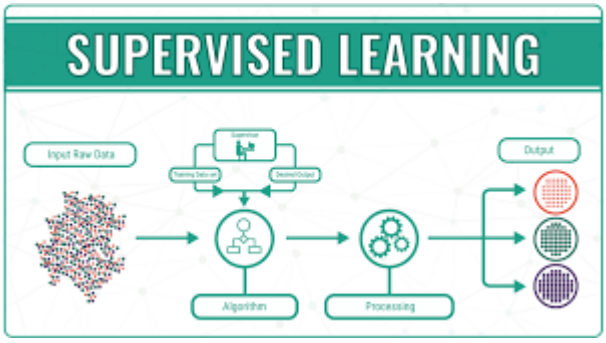


Fig 3: Supervised Learning

Draft

- Predictive Model
- we have labelled data
- The main types of supervised learning problems include regression and classification problems

List of Common Algorithms

- Nearest Neighbour
- Naive Bayes
- Decision Trees
- Linear Regression
- Support Vector Machines (SVM)
- Neural Networks

1.2.2.3 Unsupervised Learning

- The computer is trained with unlabelled data.
- Here there's no teacher at all, actually the computer might be able to teach you new things after it learns patterns in data, these algorithms a particularly useful in cases where the human expert doesn't know what to look for in the data.
- are the family of machine learning algorithms which are mainly used in pattern detection and descriptive modelling. However, there are no output categories or labels here based on which the algorithm can try to model relationships. These algorithms try to use techniques on the input data to mine for rules, detect patterns, and summarize and group the data points which help in deriving meaningful insights and describe the data better to the users.

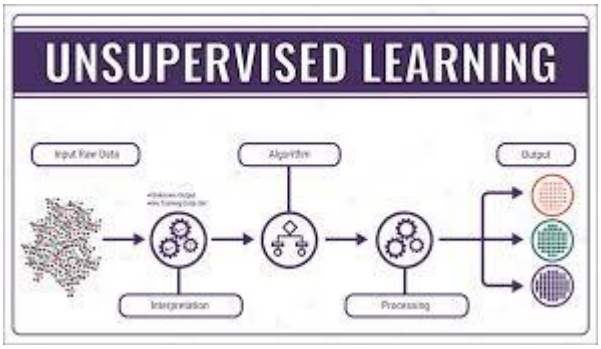


Fig 4: Unsupervised Learning

Draft

- Descriptive Model
- The main types of unsupervised learning algorithms include Clustering algorithms and Association rule learning algorithms.

List of Common Algorithms

- k-means clustering, Association Rules

1.2.2.4 Semi-Supervised Learning

In the previous two types, either there are no labels for all the observation in the dataset or labels are present for all the observations. Semi-supervised learning falls in between these two. In many practical situations, the cost to label is quite high, since it requires skilled human experts to do that. So, in the absence of labels in the majority of the observations but present in few, semi-supervised algorithms are the best candidates for the model building. These methods exploit the idea that even though the group memberships of the unlabeled data are unknown, this data carries important information about the group parameters.

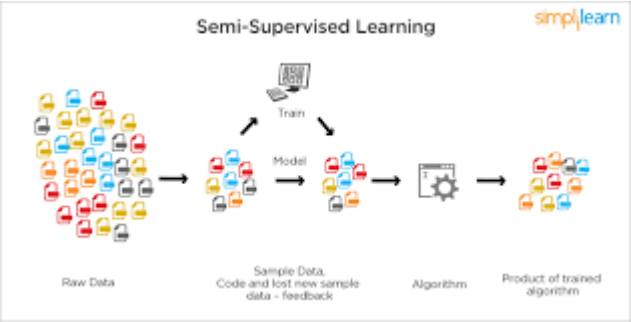


Fig 5: Semi-Supervised Learning

1.2.2.5 Reinforcement Learning

method aims at using observations gathered from the interaction with the environment to take actions that would maximize the reward or minimize the risk. Reinforcement learning algorithm (called the agent) continuously learns from the environment in an iterative fashion. In the process, the agent learns from its experiences of the environment until it explores the full range of possible states.

Reinforcement Learning is a type of Machine Learning, and thereby also a branch of Artificial Intelligence. It allows machines and software agents to automatically determine the ideal behaviour within a specific context, in order to maximize its performance. Simple reward feedback is required for the agent to learn its behaviour; this is known as the reinforcement signal.

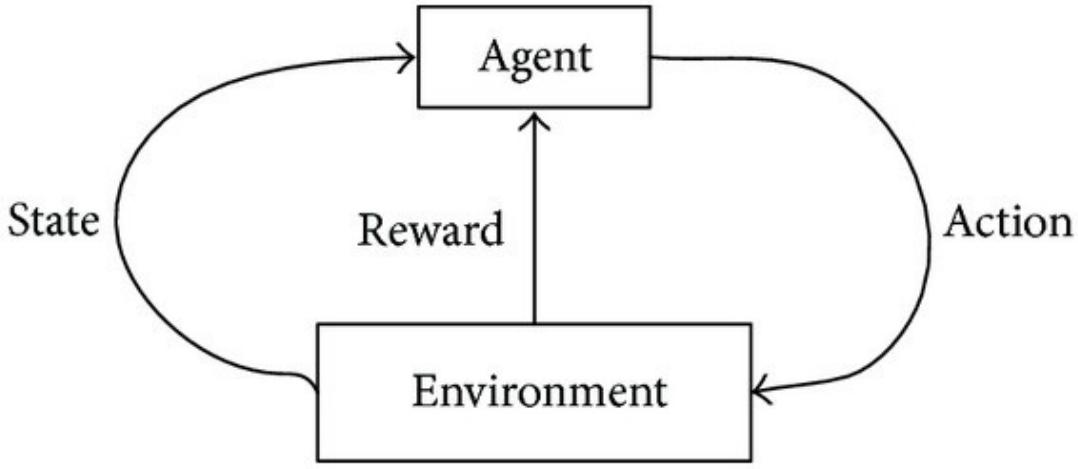


Fig 6: Reinforcement Learning

There are many different algorithms that tackle this issue. As a matter of fact, Reinforcement Learning is defined by a specific type of problem, and all its solutions are classed as Reinforcement Learning algorithms. In the problem, an agent is supposed decide the best action to select based on his current state. When this step is repeated, the problem is known as a Markov Decision Process.

In order to produce intelligent programs (also called agents), reinforcement learning goes through the following steps:

1. Input state is observed by the agent.
2. Decision making function is used to make the agent perform an action.
3. After the action is performed, the agent receives reward or reinforcement from the environment.
4. The state-action pair information about the reward is stored.

List of Common Algorithms

- Q-Learning

- Temporal Difference (TD)
- Deep Adversarial Networks

Use cases:

Some applications of the reinforcement learning algorithms are computer played board games (Chess, Go), robotic hands, and self-driving cars.

1.3 DECISION TREE

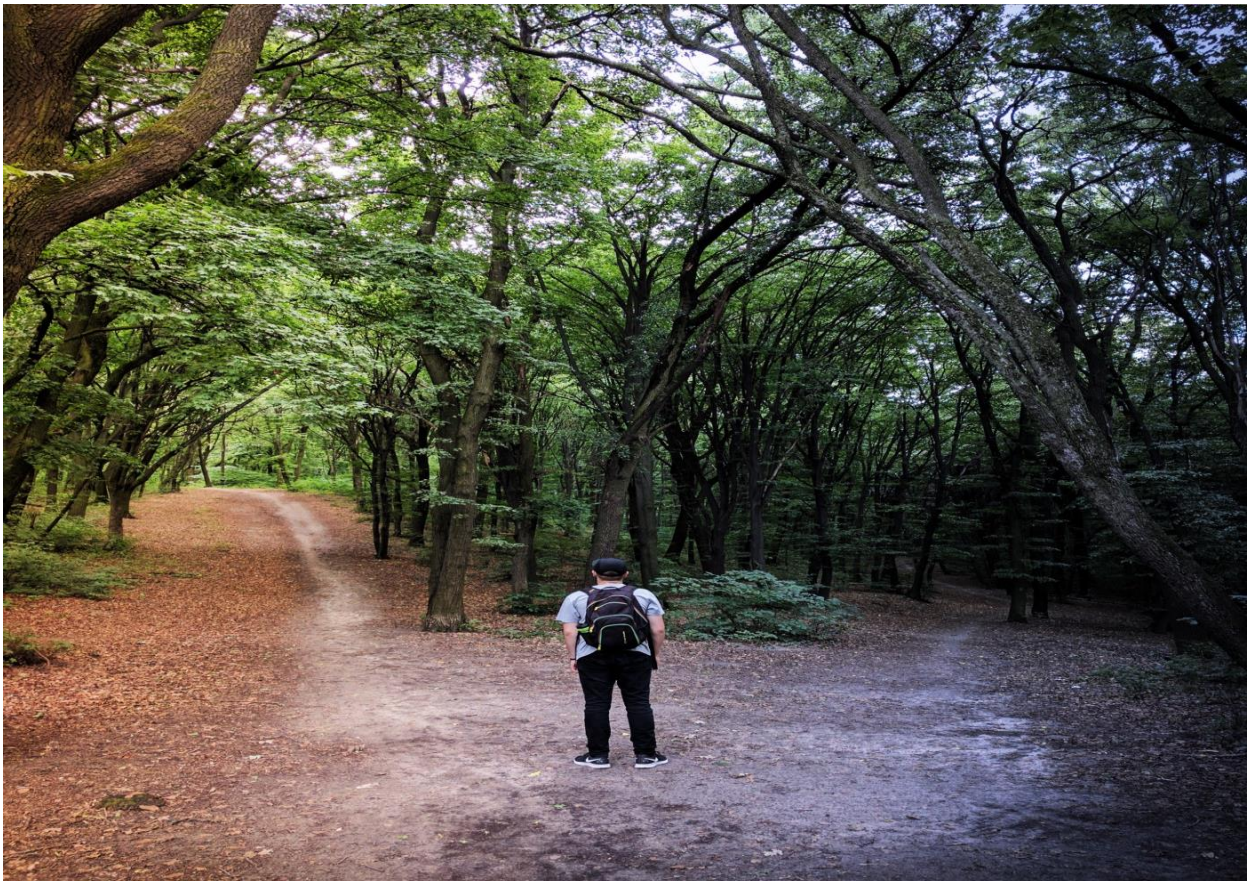


Fig 7: Decide whether to go?

A decision tree is a flowchart-like structure in which each internal node represents a test on a feature (e.g. whether a coin flip comes up heads or tails) , each leaf node represents a class label (decision taken after computing all features) and branches represent conjunctions of features that lead to those class labels. The paths from root to leaf represent classification rules. Below diagram illustrate the basic flow of decision tree for decision making with labels (Rain (Yes), No Rain (No)).

Decision tree is one of the predictive modelling approaches used in statistics, data mining and machine learning.

Decision trees are constructed via an algorithmic approach that identifies ways to split a data set based on different conditions. It is one of the most widely used and practical methods for supervised learning. Decision Trees are a non-parametric supervised learning method used for both classification and regression tasks.

Tree models where the target variable can take a discrete set of values are called classification trees. Decision trees where the target variable can take continuous values (typically real numbers) are called regression trees. Classification and Regression Tree (CART) is general term for this.

Data Format

Data comes in records of forms.
(x,Y)=(x1,x2,x3,....,xk,Y)

The dependent variable, Y, is the target variable that we are trying to understand, classify or generalize. The vector x is composed of the features, x1, x2, x3 etc., that are used for that task.

Example

```
training_data = [
        ['Green', 3, 'Apple'],
        ['Yellow', 3, 'Apple'],
        ['Red', 1, 'Grape'],
        ['Red', 1, 'Grape'],
        ['Yellow', 3, 'Lemon'],
         ]
    # Header = ["Color", "diameter", "Label"]
    # The last column is the label.
    # The first two columns are features.
```

In Decision Tree the major challenge is to identification of the attribute for the root node in each level. This process is known as attribute selection. We have two popular attribute selection measures:

1. Information Gain
2. Gini Index

1. Information Gain

When we use a node in a decision tree to partition the training instances into smaller subsets the entropy changes. Information gain is a measure of this change in entropy.

Definition: Suppose S is a set of instances, A is an attribute, S_v is the subset of S with A = v, and Values (A) is the set of all possible values of A, then

$$Gain(S, A) = Entropy(S) - \sum_{v \epsilon Values(A)} \frac{|S_v|}{|S|}.Entropy(S_v)$$

Entropy

Entropy is the measure of uncertainty of a random variable, it characterizes the impurity of an arbitrary collection of examples. The higher the entropy more the information content.

Definition: Suppose S is a set of instances, A is an attribute, S_v is the subset of S with A = v, and Values (A) is the set of all possible values of A, then

$$Gain(S, A) = Entropy(S) - \sum_{v \epsilon Values(A)} \frac{|S_v|}{|S|}.Entropy(S_v)$$

2. Gini Index

- Gini Index is a metric to measure how often a randomly chosen element would be incorrectly identified.
- It means an attribute with lower Gini index should be preferred.
- Sklearn supports “Gini” criteria for Gini Index and by default, it takes “gini” value.
- The Formula for the calculation of the of the Gini Index is given below.

$$GiniIndex = 1 - \sum_j p_j^2$$

The most notable types of decision tree algorithms are: -

1. Iterative Dichotomiser 3 (ID3): This algorithm uses Information Gain to decide which attribute is to be used classify the current subset of the data. For each level of the tree, information gain is calculated for the remaining data recursively.
2. C4.5: This algorithm is the successor of the ID3 algorithm. This algorithm uses either Information gain or Gain ratio to decide upon the classifying attribute. It is a direct improvement from the ID3 algorithm as it can handle both continuous and missing attribute values.
3. Classification and Regression Tree(CART): It is a dynamic learning algorithm which can produce a regression tree as well as a classification tree depending upon the dependent variable.

Steps for Making decision tree

- Get list of rows (dataset) which are taken into consideration for making decision tree (recursively at each nodes).
- Calculate uncertainty of our dataset or Gini impurity or how much our data is mixed up etc.

- Generate list of all question which needs to be asked at that node.
- Partition rows into True rows and False rows based on each question asked.
- Calculate information gain based on Gini impurity and partition of data from previous step.
- Update highest information gain based on each question asked.
- Update best question based on information gain (higher information gain).
- Divide the node on best question. Repeat again from step 1 again until we get pure node (leaf nodes).

Building Decision Tree

Let’s build decision tree based on training data.

```
training_data = [
                ['Green', 3, 'Apple'],
                ['Yellow', 3, 'Apple'],
                ['Red', 1, 'Grape'],
                ['Red', 1, 'Grape'],
                ['Yellow', 3, 'Lemon'],
                ]
  # Header = ["Color", "diameter", "Label"]
  # The last column is the label.
  # The first two columns are features.

 my_tree = build_tree(training_data)

print_tree(my_tree)
```

Output

```
Is diameter >= 3?
 --> True:
Is color == Yellow?
```

--> True:

Predict {'Lemon': 1, 'Apple': 1}

--> False:

Predict {'Apple': 1}

--> False:

Predict {'Grape': 2}

From above output we can see that at each steps data is divided into True and False rows. This process keeps repeated until we reach leaf node where information gain is 0 and further split of data is not possible as nodes are Pure.

How to avoid overfitting the Decision tree model

Overfitting is one of the major problems for every model in machine learning. If model is overfitted it will poorly be generalized to new samples. To avoid decision tree from overfitting we remove the branches that make use of features having low importance. This method is called as Pruning or post-pruning. This way we will reduce the complexity of tree, and hence improves predictive accuracy by the reduction of overfitting.

Pruning should reduce the size of a learning tree without reducing predictive accuracy as measured by cross-validation set. There are 2 major Pruning techniques.

- Minimum Error*:* The tree is pruned back to the point where the cross-validated error is a minimum.
- Smallest Tree*:* The tree is pruned back slightly further than the minimum error. Technically the pruning creates a decision tree with cross-validation error within 1 standard error of the minimum error.

Early Stop or Pre-pruning

An alternative method to prevent overfitting is to try and stop the tree-building process early, before it produces leaves with very small samples. This heuristic is known as early stopping but is also sometimes known as pre-pruning decision trees.

At each stage of splitting the tree, we check the cross-validation error. If the error does not decrease significantly enough then we stop. Early stopping may underfit by stopping too early. The current split may be of little benefit, but having made it, subsequent splits more significantly reduce the error.

Early stopping and pruning can be used together, separately, or not at all. Post pruning decision trees is more mathematically rigorous, finding a tree at least as good as early stopping. Early stopping is a quick fix heuristic. If used together with pruning, early stopping may save time. After all, why build a tree only to prune it back again?

Important Terminology related to Decision Trees

1. **Root Node:** It represents the entire population or sample and this further gets divided into two or more homogeneous sets.
2. **Splitting:** It is a process of dividing a node into two or more sub-nodes.
3. **Decision Node:** When a sub-node splits into further sub-nodes, then it is called the decision node.
4. **Leaf / Terminal Node:** Nodes do not split is called Leaf or Terminal node.
5. **Pruning:** When we remove sub-nodes of a decision node, this process is called pruning. You can say the opposite process of splitting.
6. **Branch / Sub-Tree:** A subsection of the entire tree is called branch or sub-tree.
7. **Parent and Child Node:** A node, which is divided into sub-nodes is called a parent node of sub-nodes whereas sub-nodes are the child of a parent node.

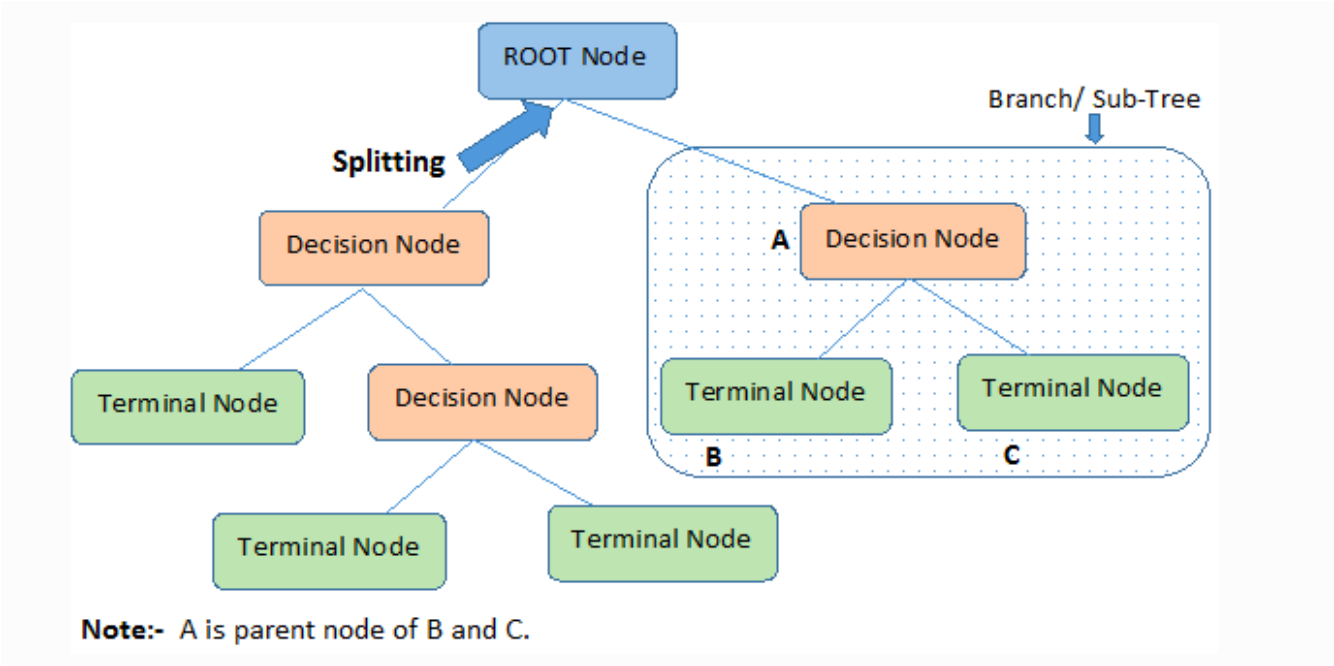


Fig 8: Decision Tree Terminlogy Representation

Decision trees classify the examples by sorting them down the tree from the root to some leaf/terminal node, with the leaf/terminal node providing the classification of the example.

Each node in the tree acts as a test case for some attribute, and each edge descending from the node corresponds to the possible answers to the test case. This process is recursive in nature and is repeated for every subtree rooted at the new node.

Optimizing the Decision Tree Classifier

- **criterion**: optional (default=”gini”) or Choose attribute selection measure: This parameter allows us to use the different-different attribute selection measure. Supported criteria are “gini” for the Gini index and “entropy” for the information gain.
- **splitter**: string, optional (default=”best”) or Split Strategy: This parameter allows us to choose the split strategy. Supported strategies are “best” to choose the best split and “random” to choose the best random split.
- **max_depth**: int or None, optional (default=None) or Maximum Depth of a Tree: The maximum depth of the tree. If None, then nodes are expanded until all the leaves contain less than min_samples_split samples. The higher value of maximum depth causes overfitting, and a lower value causes underfitting (Source).

In Scikit-learn, optimization of decision tree classifier performed by only pre-pruning. The maximum depth of the tree can be used as a control variable for pre-pruning.

```
# Create Decision Tree classifer object
classifier = DecisionTreeClassifier (criterion="entropy", max_depth=3)
# Train Decision Tree Classifer
classifier = classifier.fit(X_train,y_train)
#Predict the response for test dataset
y_pred = classifier.predict(X_test)
# Model Accuracy, how often is the classifier correct?
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

Advantage of Decision Tree

- Easy to use and understand.
- Can handle both categorical and numerical data.
- Resistant to outliers, hence require little data pre-processing.

Disadvantage of Decision Tree

- Prone to overfitting.
- Require some kind of measurement as to how well they are doing.
- Need to be careful with parameter tuning.
- Can create biased learned trees if some classes dominate.

Naïve Bayes

Bayes' Theorem is stated as:

P(h|d) = (P(d|h) * P(h)) / P(d) P(h|d) is the probability of hypothesis h given the data d. This is called the posterior probability. P(d|h) is the probability of data d given that the hypothesis h was true. P(h) is the probability of hypothesis h being true (regardless of the data). This is called the prior probability of h. P(d) is the probability of the data (regardless of the hypothesis).we are interested in calculating the posterior probability of P(h|d) from the prior probability p(h) with P(D) and P(d|h). After calculating the posterior probability for a number of different hypotheses, we will select the hypothesis with the highest probability. This is the maximum probable hypothesis and may formally be called the (MAP) hypothesis. This can be written as: MAP(h) = max(P(h|d)) or

MAP(h) = max((P(d|h) * P(h)) / P(d))

Or

MAP(h) = max(P(d|h) * P(h))

The P(d) is a normalizing term which allows us to calculate the probability. We can drop it when we are interested in the most probable hypothesis as it is constant and only used to normalize. Back to classification, if we have an even number of instances in each class in our training data, then the probability of each class (e.g. P(h)) will be equal. Again, this would be a constant term in our equation, and we could drop it so that we end up with: MAP(h) = max(P(d|h)) Naive Bayes is a classification algorithm for binary (two-class) and multi-class classification problems. The technique is easiest to understand when described using binary or categorical input values. It is called naive Bayes or idiot Bayes because the calculation of the probabilities for each hypothesis are simplified to make their calculation tractable. Rather than attempting to calculate the values of each attribute value P (d1, d2, d3|h), they are assumed to be conditionally independent given the

target value and calculated as P(d1|h) * P(d2|H) and so on. This is a very strong assumption that is most unlikely in real data, i.e. that the attributes do not interact. Nevertheless, the approach performs surprisingly well on data where this assumption does not hold.

MAP(h) = max(P(d|h) * P(h))

Gaussian Naïve Bayes

mean(x) = 1/n * sum(x)

Where n is the number of instances and x are the values for an input variable in your training data. We can calculate the standard deviation using the following equation:

standard deviation(x) = sqrt (1/n * sum(xi-mean(x)^2))

This is the square root of the average squared difference of each value of x from the mean value of x, where n is the number of instances, sqrt() is the square root function, sum() is the sum function, xi is a specific value of the x variable for the i'th instance and mean(x) is described above, and ^2 is the square. Gaussian PDF with a new input for the variable, and in return the Gaussian PDF will provide an estimate of the probability of that new input value for that class.

pdf (x, mean, sd) = (1 / (sqrt (2 * PI) * sd)) * exp (-((x-mean^2)/(2*sd^2)))

Where pdf(x) is the Gaussian Probability Density Function (PDF), sqrt () is the square root, mean and sd are the mean and standard deviation calculated above, Pi is the numerical constant, exp () is the numerical constant e or Euler's number raised to power.

2. LITERATURE SURVEY

2.1 General

Monika Gandhi et.al, [1] used Naïve Bayes, Decision tree and neural network algorithms and analysed the medical dataset. There are an enormous number of features involved. So, there's a requirement to scale back the amount of features. this will be done by feature selection. On doing this, they assert that point is reduced. They made use of decision tree and neural networks.

J Thomas, R Theresa Princy [2] made use of K nearest neighbour algorithm, neural network, naïve Bayes and decision tree for heart condition prediction. They made use of knowledge mining techniques to detect the guts disease risk rate.

Sana Bharti, Shailendra Narayan Singh [3] made use of Particle Swarm Optimization, Artificial neural network, Genetic algorithm for prediction. Associative classification may be a new and efficient technique which integrates

association rule mining and classification to a model for prediction and achieved good accuracy.

Purushottam et.al, [4] proposed “An automated system in diagnosis would enhance medical aid and it also can reduce costs. during this study, we've designed a system which will efficiently discover the principles to predict the danger level of patients supported the given parameter about their health. the principles are often prioritized supported the user's requirement. The performance of the system is evaluated in terms of classification accuracy and therefore the results shows that the system has great potential in predicting the guts disease risk level more accurately”.

Sellappan Palaniyappan, Rafiah Awang [5] made use of decision tree Naïve Bayes, Decision tree, Artificial Neural Networks to create Intelligent heart condition Prediction Systems (IHDPS).To enhance visualization and simple interpretation, it displays the results both in tabular and graphical forms. By providing effective treatments, it also helps to scale back treatment costs. Discovery of hidden patterns and relationships often has gone unexploited. Advanced data processing techniques helped remedy this example.

Himanshu Sharma,M A Rizvi [6] made use of Decision tree, support vector machine, deep learning, K nearest neighbour algorithms. Since the datasets contain noise, they tried to scale back the noise by cleaning and pre-processing the dataset and also tried to scale back the dimensionality of the dataset. They found that good accuracy are often achieved with neural networks.

Animesh Hazra et.al, [7] discussed intimately the disorder and different symptoms of attack. the various sorts of classification and clustering algorithms and tools were used.

V.Krishnaiah, G.Narsimha, N.Subhash Chandra [8] presented an analysis using data processing. The analysis showed that using different techniques and taking different number of attributes gives different accuracies for predicting heart diseases.

Ramandeep Kaur, Er.Prabhsharn Kaur [9] have showed that the guts disease data contains unnecessary, duplicate information. This has got to be pre processed. Also, they assert that feature selection has got to be done on the dataset for achieving better

results.

J.Vijayashree and N.Ch.SrimanNarayanaIyengar [10] used data processing. an enormous amount of knowledge is produced on a day to day. As such, it can't be interpreted manually. data processing are often effectively wont to predict diseases from these datasets. during this paper, different data processing techniques are analysed on heart condition database. last, this paper analyses and compares how different classification algorithms work on a heart condition database.

Benjamin EJ et.al [11] says that there are seven key factors for heart condition like smoking, physical inactivity, nutrition, obesity, cholesterol, diabetes and high vital sign. They also discussed the statistics of heart condition including stroke and cardio vascular disease.

Abhay Kishore et.al [12] on their experimentation showed that recurrent neural network gives good accuracy in comparison to other algorithms like CNN, Naïve Bayes and SVM. Hence, neural networks perform well in heart condition prediction. They also achieved a system that would predict silent heart attacks and inform the user as earliest possible.

M.Nikhil Kumar et.al [13] used various algorithms – Decision tree, random forest, Naïve Bayes, KNN, Support vector machine, logistic model tree algorithm. Naïve Bayes algorithm gave good results in comparison to other algorithms. They made use of UCI repository of heart condition dataset. Also, J48 algorithm took less time to create and gave good results.

Amandeep Kaur et.al [14] compared various algorithms like artificial neural network, K – nearest neighbour, Naïve Bayes, Support vector machine on heart condition prediction.

3. IMPLEMENTATION OF MODEL

3.1 EXISTING SYSTEM

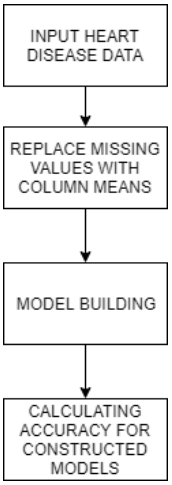
Heart disease can be managed effectively with a combination of lifestyle changes, medicine and in some cases, surgery. With the right treatment, the symptoms of heart disease can be reduced and the functioning of the heart improved. The predicted results can be used to prevent and thus reduce cost for surgical treatment and other expensive.

The overall objective of my work will be to predict accurately with few tests and attributes the presence of heart disease. Attributes considered form the primary basis for tests and give accurate results more or less. Many more input attributes can be taken but our goal is to predictwith few attributes and faster efficiency the risk of having heart disease. Decisions are often made based on doctors' intuition and experience rather than on the knowledge rich data hidden in the data set and databases. This practice leads to unwanted biases, errors and excessive medical costs which affects the quality of service provided to patients.

The health care environment is still „information rich" but „knowledge poor" Their is a wealth of data available within the healthcare systems. However, there is a lack of effective analysis tools to discover hidden relationships and trends in the data for African .

3.2 PROPOSED SYSTEM

In this paper, comparison of various machine learning methods is done for predicting the 10 year risk of coronary heart disease of the patients from their medical data. The following is the flowchart for proposed methodology:



The heart disease data set is taken as input. It is then pre-processed by replacing non-available values with column means. Four different methods were used in this paper. The different methods used are depicted in the output is the accuracy metrics of the machine learning models. The model can then be used in prediction

K-Nearest Neighbours (KNN)

KNN is a non-parametric machine learning algorithm. The KNN algorithm is a supervised learning method. This means that all the data is labelled and the algorithm learns to predict the output from the input data. It performs well even if the training data is large and contains noisy values. The data is divided into training and test sets. The train set is used for model building and training. A k- value is decided which is often the square root of the number of observations. Now the test data is predicted on the model built. There are different distance measures. For continuous variables, Euclidean distance, Manhattan distance and Minkowski distance measures can be used. However, the commonly used measure is Euclidean distance. The formula for Euclidean distance is as follows:

$$d = \sqrt{\sum_{i=1}^{k} (x_i - y_i)^2}$$

Naive Bayes algorithm (NB)

This is a classification algorithm which is used when the dimensionality of the input is very high. A Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. It is based on Bayes theorem. The Bayes theorem is as follows:

$$P(Y/X) = P(X/Y)\ P(X)$$

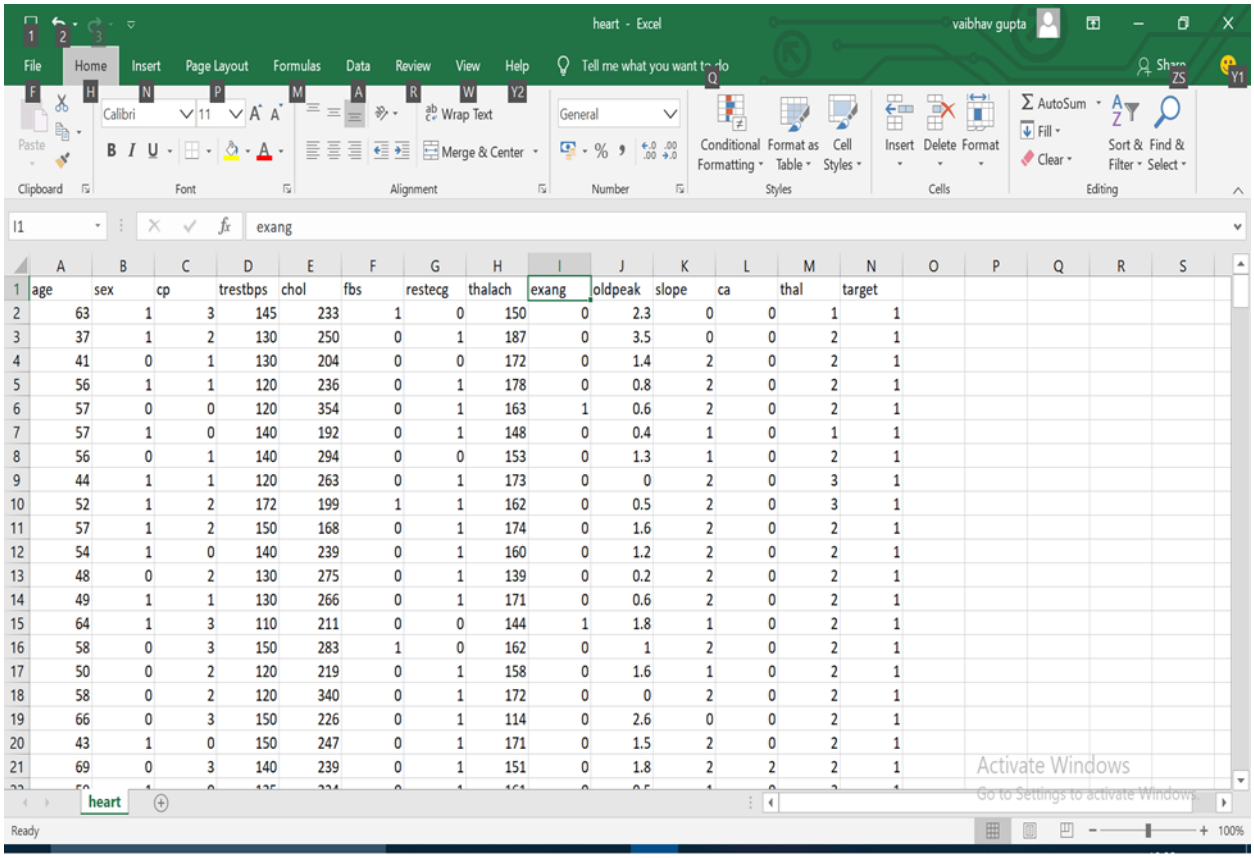
This calculates the probability of Y given X where X is the prior event and Y is the dependence event. It needs less training data. It can be used for binary classification problems and is very simple.

Decision trees

Decision trees is one of the ways to display an algorithm. It is a classic machine learning algorithm. In heart disease, there are several factors such as cigarette, BP, Hypertension, age etc. The challenge of the decision tree lies in the selection of the root node. This factor used in root node must clearly classify the data. We make use of age as the root node. The decision tree is easy to interpret. They are non-parametric and they implicitly do feature selection.

3.2.1 DATA SET:

Datasets are gathered from Kaggle. Data set is now provided to Machine learning models on the basis of this facts this version is trained.



	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	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
3	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
4	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
5	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
6	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
7	57	1	0	140	192	0	1	148	0	0.4	1	0	1	1
8	56	0	1	140	294	0	0	153	0	1.3	1	0	2	1
9	44	1	1	120	263	0	1	173	0	0	2	0	3	1
10	52	1	2	172	199	1	1	162	0	0.5	2	0	3	1
11	57	1	2	150	168	0	1	174	0	1.6	2	0	2	1
12	54	1	0	140	239	0	1	160	0	1.2	2	0	2	1
13	48	0	2	130	275	0	1	139	0	0.2	2	0	2	1
14	49	1	1	130	266	0	1	171	0	0.6	2	0	2	1
15	64	1	3	110	211	0	0	144	1	1.8	1	0	2	1
16	58	0	3	150	283	1	0	162	0	1	2	0	2	1
17	50	0	2	120	219	0	1	158	0	1.6	1	0	2	1
18	58	0	2	120	340	0	1	172	0	0	2	0	2	1
19	66	0	3	150	226	0	1	114	0	2.6	0	0	2	1
20	43	1	0	150	247	0	1	171	0	1.5	2	0	2	1
21	69	0	3	140	239	0	1	151	0	1.8	2	2	2	1

3.3 IMPLEMENTATION OF MODEL

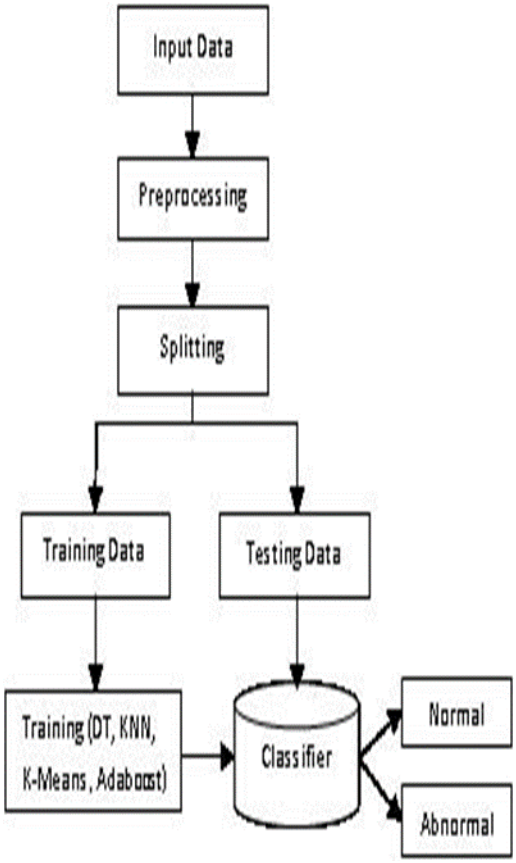


Fig 11: Diagram of Prediction Model

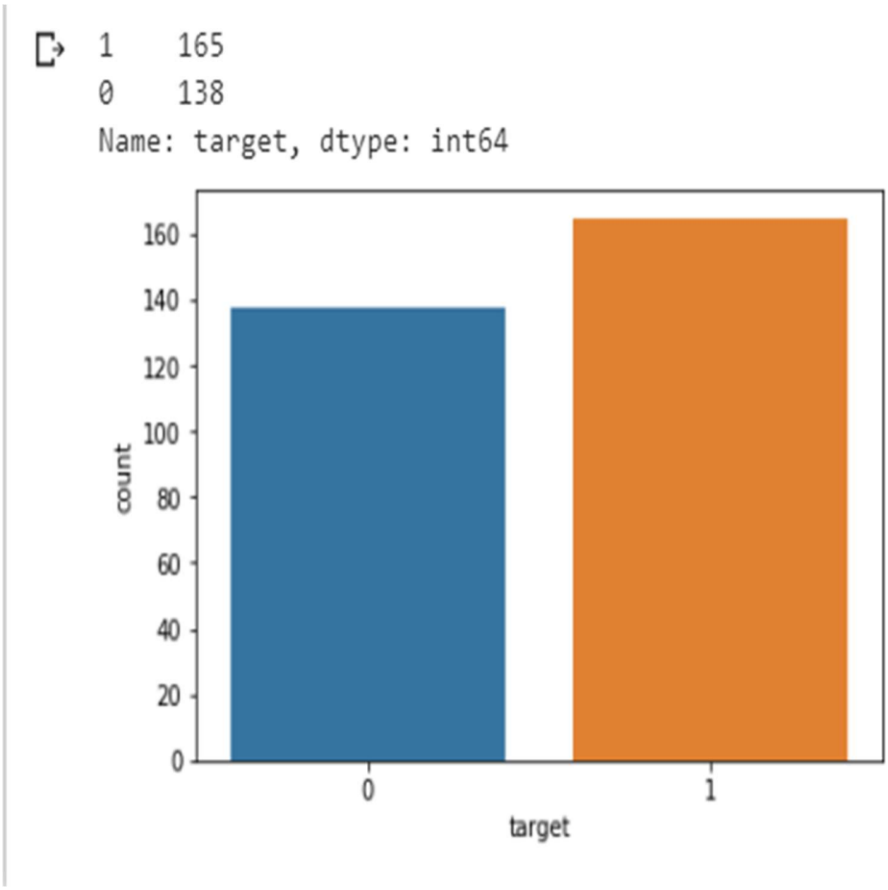
3.3.1 **Exploratory Data Analysis (EDA)**

(1 is who have Heart Disease and 0 is who don’t have Heart Disease)

```
y = data["target"]

sns.countplot(y)


target_temp = data.target.value_counts()

print(target_temp)
```

```
1    165
0    138
Name: target, dtype: int64
```

No. of Heart Disease patients is 165. No. of patients who don't have a heart disease is 138. [Which is a good balance of target data.]



Decision Tree:

Decision Tree is a greedy algorithm it searches the entire space of possible decision

trees. so, we need to find an optimum parameter(s) or criteria for stopping the decision tree at some point. We use the hyperparameters to prune the decision tree.

Decision tree algorithm used in model:

We have built a classification model which learn decision rules gathered from data features will make predictions and will generate a tree structure with the help of decision nodes corresponding to input variables. Following processes are followed to classify the data:

1)For selection of attribute we will be using splitting criterion Gini index. The best score of the attribute will be chosen as deciding node.

2)Root split node is created with other subsets and then first process is repeated until the next best attribute is selected as deciding node.

3)Process 2 is continued until reaching a leaf node.

4)In the last step, pruning is applied to avoid the overfitting by removing that sections of tree which has little classification power and determine the optimum size of tree.

3.3.2 Generated Decision Tree for Model

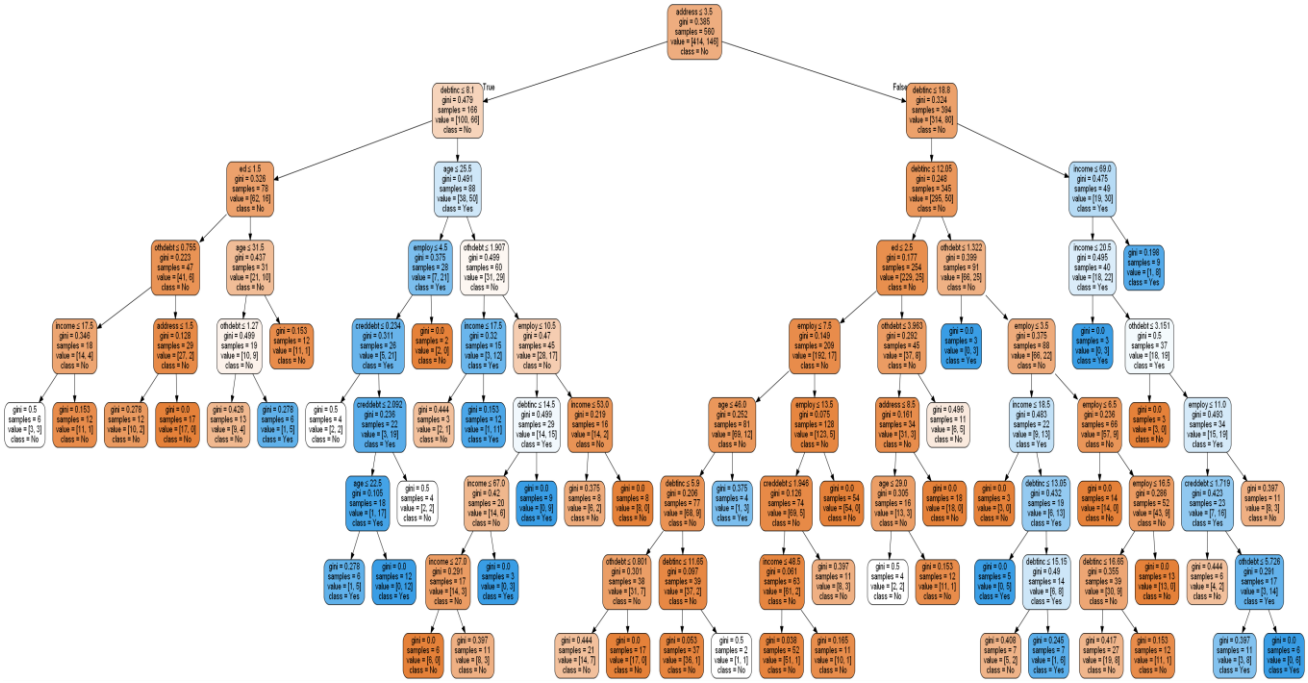


Fig 19: Decision tree for the model

Visualization of decision tree is with the help of graphviz package.

3.3.3 Source Code

```
# coding: utf-8

# In[2]:



from numpy import genfromtxt
import numpy as np
from numpy import *
import matplotlib
#matplotlib.use('TKAgg') # matplotlib renderer for windows

import matplotlib.pyplot as plt



from sklearn.svm import LinearSVC
from sklearn.decomposition import PCA
import pylab as pl
```

```
from itertools import cycle
from sklearn import cross_validation
from sklearn.svm import SVC

# After imorting all the necessary packages now
#Loading the data and pruning it
dataset = genfromtxt('Desktop/Heart/data.csv',delimiter=',')

#Printing the datasetd
X = dataset[:,0:12] #Feature set
Y = dataset[:,13]    #label Set

#Replacing 1-4 by label 1  tesko arthaat   # Item with 0 value is already indexed as
0 , so rest are indexed as 1
for index, item in enumerate(Y):   # Last row gives 4 diff types of output , so
convert them to 0  or 1
        if not (item == 0.0):       # that is either Yes or No
                Y[index] = 1
print(Y)
target_names = ['0','1']


# PLOTTING part starts

#Method to plot the graph for reduced Dimensions
def plot_2D(data,target,target_names):
        colors = cycle('rgbcmykw')
        target_ids = range(len(target_names))
        plt.figure()
        for i,c, label in zip(target_ids, colors, target_names):
```

```
                plt.scatter(data[target == i, 0], data[target == i, 1], c=c,
label=label)
        plt.legend()
        plt.savefig('Problem 2 Graph')

# TIME FOR SVM
# Classifying the data using a linear SVM and perdicting the probabilities of
disease belonging to a particular classs

modelSVM = LinearSVC(C=0.1)
pca = PCA(n_components=2, whiten=True).fit(X)   # n denotes number of
components to keep after Dimensionality Reduction
X_new = pca.transform(X)

#Calling the above defined function plot_2D
plot_2D(X_new, Y, target_names)

# Applying cross validation on the training and test set for validating our linear
SVM model
X_train,X_test,Y_train,Y_test = cross_validation.train_test_split(X_new, Y,
test_size = 0.2, train_size=0.8, random_state=0)
modelSVM = modelSVM.fit(X_train,Y_train)
print("Linear SVC values with Split")
print(modelSVM.score(X_test, Y_test))



modelSVMRaw = LinearSVC(C = 0.1)
modelSVMRaw = modelSVMRaw.fit(X_new, Y)
cnt = 0
```

```
for i in modelSVMRaw.predict(X_new):
        if(i == Y[1]):
                cnt = cnt+1
print("Linear SVC score without split")
print(float(cnt)/101)

# Applying the PCA on the data features
modelSVM2 = SVC(C = 0.1,kernel='rbf')

# Applying the cross validation on training and the test set for validating our linear
SVM model
X_train1,X_test1,Y_train1,Y_test1 = cross_validation.train_test_split(X_new, Y,
test_size = 0.2, train_size=0.1, random_state=0)
modelSVM2 = modelSVM2.fit(X_train1,Y_train1)
print("RBF score with split")
print(modelSVM2.score(X_test1,Y_test1))



modelSVMRaw2 = SVC(C=0.1, kernel='rbf')
modelSVMRaw2 = modelSVMRaw2.fit(X_new,Y)
cnt1 = 0
for i in modelSVMRaw2.predict(X_new):
        if i == Y[1]:
                cnt1 = cnt1 + 1
print("RBF score without split")
print(float(cnt1)/298)



# Only perform 2 algorithms
```

```
# creating the mest plots
X_min, X_max = X_new[:,0].min() - 1, X_new[:,0].max() + 1
Y_min, Y_max = X_new[:,1].min() - 1, X_new[:,1].max() + 1
xx, yy = np.meshgrid(np.arange(X_min, X_max,0.2),
                     np.arange(Y_min, Y_max,0.2))

#Titles for the plot
titles = "SVC ( RBF kernel) - Plotting highest varied 2 PCA values"

# PLot the decision boundary . For that we'l; assign a color to each
# point in the mesh
plt .subplot(2,2, i + 1)
plt.subplots_adjust(wspace = 0.4, hspace=0.4)
Z = modelSVM2.predict(np.c_[xx.ravel(), yy.ravel()])


#Put the result into a color plot
Z = Z.reshape(xx.shape)
plt.contourf(xx,yy,Z,cmap=plt.cm.Paired, alpha=0.1)

#plot also the color points
plt.scatter(X_new[:,0], X_new[:,1], c=Y, cmap = plt.cm.Paired)
plt.xlabel("PCA1")
plt.ylabel("PCA2")
plt.xlim(xx.min(),xx.max())
plt.ylim(yy.min(),yy.max())
plt.xticks(())
plt.yticks(())
plt.title(titles)
```

```
plt.show()
```

4. RESULTS

The dataset used is Framingham taken from Kaggle [17].

There were 16 attributes were as follows: Male – gender 0 for female and 1 for male, Age – age of the patient, Education – values 1-5, education of the patient. Current smoker – 1 if current smoker and 0 otherwise, Cigarette per day – if current smoker then number of cigarette per day, BP Meds – vital sign, Prevalent BP – prevalent vital sign, Prevalent Hyp – prevalent hyper tension, Diabetes – 1 if diabetes 0 otherwise, Total cholesterol – cholesterol level, Sys BP – systolic vital sign, Dia BP – diastolic vital sign, BMI – body mass index, pulse – pulse or pulse of the patient, Glucose – glucose level, Ten Year CHD – has chronic heart condition or not.

The machine learning models is evaluated using the AUC-ROC metric. This will be used to understand the model performance.

The ROC curve of the algorithms is as follows:

FIGURE 3: ROC CURVE FOR DECISION TREE

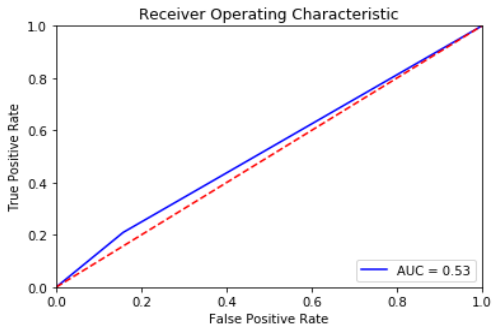
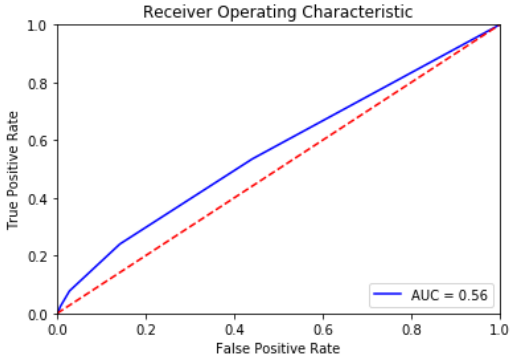


FIGURE 4: ROC CURVE FOR KNN



5. Conclusion

This discusses the various machine learning algorithm, decision tree and k- nearest neighbour which were applied to the data set. It utilizes the data such as blood pressure, cholesterol, diabetes and then tries to predict the possible coronary heart disease patient in next 10 years.

Family history of heart disease can also be a reason for developing a heart disease as mentioned earlier. So, this data of the patient can also be included for further increasing the accuracy of the model.

This work will be useful in identifying the possible patients who may suffer from heart disease in the next 10 years. This may help in taking preventive measures and hence try to avoid the possibility of heart disease for the patient. So when a patient is predicted as positive for heart disease, then the medical data for the patient can be closely analysed by the doctors. An example would be - suppose the patient has diabetes which may be the cause for heart disease in future and then the patient can be given treatment to have diabetes in control which in turn may prevent the heart disease.

The heart disease prediction can be done using other machine learning algorithms. Logistic regression can also perform well in case of binary classification problems such as heart disease prediction. Random forests can perform well than decision trees. Also, the ensemble methods and artificial neural networks can be applied to the data set. The results can be compared and improvised.

5. REFERENCES

[1]Gandhi, Monika & Singh, Shailendra. (2015). Predictions in heart disease using techniques of data mining. 2015 1st International Conference on Futuristic Trends in Computational Analysis and Knowledge Management, ABLAZE 2015. 520-525. 10.1109/ABLAZE.2015.7154917.

[2] Thomas, J. & Princy, R. (2016). Human heart disease prediction system using data mining techniques. 1-5. 10.1109/ICCPCT.2016.7530265.

[3] S. Bharti and S. N. Singh, "Analytical study of heart disease prediction comparing with different algorithms," International Conference on Computing, Communication & Automation, Noida, 2015, pp. 78-82.

[4] Purushottam, & Saxena, Kanak & Sharma, Richa. (2015). Efficient heart disease prediction system using decision tree. International Conference on Computing, Communication and Automation, ICCCA 2015. 72-77. 10.1109/CCAA.2015.7148346.

[5] Mat Ghani, Mohd & Awang, Raflah. (2008). Intelligent heart disease prediction system using data mining techniques. 8. 108 - 115. 10.1109/AICCSA.2008.4493524.

[6] Sharma, Himanshu. "Prediction of Heart Disease using Machine Learning Algorithms: A Survey." (2017).

[7] Hazra, Animesh & Mandal, Subrata & Gupta, Amit & Mukherjee, Arkomita & Mukherjee, Asmita. (2017). Heart Disease Diagnosis and Prediction Using Machine Learning and Data Mining Techniques: A Review. Advances in Computational Sciences and Technology. 10. 2137-2159.

[8] V Krishnaiah, G Narsimha and Subhash N Chandra. Article: Heart Disease Prediction System using Data Mining Techniques and Intelligent Fuzzy Approach: A Review. *International Journal of Computer Applications* 136(2):43-51, February 2016. Published by Foundation of Computer Science (FCS), NY, USA

[9] Kaur, Ramandeep and Er. Prabhsharn Kaur. "A Review-Heart Disease Forecasting Pattern using Various Data Mining Techniques." (2016).

[10] Vijayashree, J. & Iyenger, N Ch Sriman Narayana. (2016). Heart Disease Prediction System Using Data Mining and Hybrid Intelligent Techniques: A Review. International Journal of Bio-Science and Bio-Technology. 8. 139-148. 10.14257/ijbsbt.2016.8.4.16.

[11] Benjamin EJ et.al heart condition and Stroke Statistics 2018 At-a-Glance (2018)

[12] Kishore, A. G. Ravi et al. “Heart Attack Prediction Using Deep Learning.” (2018).

[13]Mutyala, Nikhil Kumar & Koushik, K.V.s & Krishna, K.. (2018). Prediction of Heart Diseases Using Data Mining and Machine Learning Algorithms and Tools. 10.13140/RG.2.2.28488.83203.

[14] Razia, Shaik & M, Shaik. (2019). Heart Disease Prediction using Machine Learning Techniques. International Journal of Recent Technology and Engineering. 8. 10.35940/ijrte.D4537.118419.