

Application of machine learning in medical diagnosis

Project report submitted in partial fulfillment of the requirement for the degree of

Bachelor of Technology

By

Dhruv Garg (1710110110)
Harshit Khandelwal (1710110137)

Under supervision

Of

Dr. Madan Gopal
(Distinguished Professor, Department of Electrical Engineering)



DEPARTMENT OF ELECTRICAL ENGINEERING
SCHOOL OF ENGINEERING
SHIV NADAR UNIVERSITY
(October 2020)

Candidate Declaration

I hereby declare that the thesis entitled “Application of machine learning in medical diagnosis” submitted for the B Tech Degree program. This thesis has been written in my own words. I have adequately cited and referenced the original sources.

1.

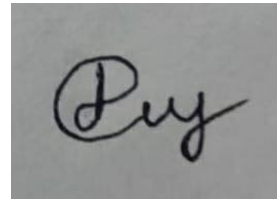


(Signature)

(Dhruv Garg)

(1710110110)

Date: 18/11/2020



(Signature)

(Harshit Khandelwal)

(1710110137)

Date: 18/11/2020

CERTIFICATE

It is certified that the work contained in the project report titled “Application of Machine learning in medical diagnosis” by “Dhruv Garg and Harshit Khandelwal” has been carried out under my/our supervision and that this work has not been submitted elsewhere for a degree.

Signature of Supervisor(s)

Dr. Madan Gopal

Department of Electrical Engineering

School of Engineering

Shiv Nadar University

November, 2020

Abstract

Machine Learning is a modern and highly sophisticated technological application which became a massive trend in various industries. It is utilised in diverse fields to predict outcomes by identifying multiple patterns which are not noticeable to humans. One of the notable implementations of machine learning is to provide excellent capabilities to estimate the probability of need to diagnose the specific disease by predicting the pattern from medical data sources. Numerous techniques are required to process large amounts of data. Data Mining is the most often used technique. To process these data, Data mining combines traditional data analysis with sophisticated algorithms. Medical data mining is an important area of Data Mining and considered as one of the important research fields due to its application in the healthcare domain. Cardiovascular disease (CVD) is a big reason for morbidity and mortality in the current living style. Identification of Cardiovascular disease is an important but a complex task that needs to be performed very minutely. An automated system in medical diagnosis would enhance medical care and it can also reduce costs. In this project, we will review various datasets which are currently available on open source platforms. The data ranges from details of clinical symptoms to various types of biochemical data and outputs of imaging devices for developing efficient decision support for healthcare applications. In this study, we have designed a system that can efficiently discover the rules to predict the risk level of patients based on the given parameter about their health. The rules can be prioritized based on the user's requirement. We have used sophisticated machine learning techniques such as Neural networks and decision trees. This project aims to explore the field of medical diagnosis with neural networks to understand its pipeline and performance. The challenge lies in the complexity of the data and correlations when it comes to prediction using conventional techniques. We are using a South African Heart Disease dataset of 462 instances and UCI dataset of 303 instances derived by the considered ML techniques using 10-fold cross-validation.

Table of Contents

List of figures	(pg no. 6)
List of Tables	(pg no. 7)
1. Introduction	(pg no. 8)
a. Motivation	(pg no. 8)
b. Problem Statement	(pg no. 8)
c. Methodology.....	(pg. no. 9)
2. Literature Review	(pg no. 10)
3. Work done	(pg no. 11)
3.1 Data Understanding and preparation.....	(pg no. 11)
3.1.1 South African Dataset	(pg no. 11)
3.1.2 UCI Dataset	(pg no. 14)
3.2 Modelling and Evaluation	(pg no. 19)
3.2.1 Neural Networks.....	(pg no. 19)
3.2.1.1 NN for SA dataset	(pg no. 21)
3.2.1.2 NN for UCI dataset	(pg no. 26)
3.2.2 Decision Trees	(pg no. 29)
3.2.2.1 DT for SA dataset	(pg no. 30)
3.2.2.2 DT for UCI dataset	(pg no. 34)
4. Conclusion	(pg no. 37)
5. Future Prospects	(pg no. 38)
6. Conclusion.....	(pg no. 39)

List of Figures

3.1 Data distribution in dataset CHD (present-> 1, absent ->0)	(pg no. 12)
3.2 Correlation matrix for South African dataset	(pg no.12)
3.3 Famhist attribute dependency	(pg no. 13)
3.4 Distplot for age of CHD patients	(pg no. 13)
3.5 Data distribution in cleveland CHD (present-> 1, absent ->0).....	(pg no. 16)
3.6 Correlation matrix for cleveland dataset.....	(pg no. 16)
3.7 Gender vs Target	(pg no. 18)
3.8 Chest pain vs target.....	(pg no. 18)
3.9 An example of ST segment from electrocardiography.....	(pg no. 18)
3.10 Weights, bias and gradient descent in neural network	(pg no. 20)
3.11 Tan and sigmoid functions.....	(pg no. 21)
3.12 Cost vs iteration.....	(pg no. 22)
3.13 Xavier Initialization	(pg no. 22)
3.14 A single neuron representation.....	(pg no. 23)
3.15 Neural Network model for dataset 1	(pg no. 23)
3.16 Confusion matrix for 0.5 cut-off	(pg no. 24)
3.17 ROC curve	(pg no. 25)
3.18 Confusion matrix for a general case from ROC	(pg no. 26)
3.19 Neural Network model for cleveland dataset.....	(pg no. 28)
3.20 Softmax layer probability output.....	(pg no. 28)
3.21 Confusion matrix for cleveland dataset.....	(pg no. 29)
3.22 General decision tree structure.....	(pg no. 30)
3.23 Preliminary Tree for South African data.....	(pg no. 31)
3.24 Confusion metric for Preliminary tree.....	(pg no. 31)
3.25 Accuracy vs Alpha for south african data.....	(pg no. 32)
3.26. Mean and Std dev of accuracy over cross validation sets.....	(pg no. 32)
3.27. Pruned tree after ccp.....	(pg no. 33)
3.28. Confusion matrix for Pruned tree.....	(pg no. 33)
3.29 Preliminary Tree for UCI data	(pg no. 34)
3.30 Confusion matrix for Preliminary tree.....	(pg no. 34)
3.31 Accuracy vs alpha for UCI data.....	(pg no. 35)
3.32 Mean and std deviation of all cross validation set.....	(pg no. 35)
3.33 Pruned tree for Cleveland data.....	(pg no. 35)
3.34 Confusion matrix for pruned tree.....	(pg no. 36)

List of Tables

3.1 Attributes details of South african heart disease dataset.....	(pg no. 11)
3.2 Attributes details of South african heart disease dataset	(pg no. 15)
3.3 Missing values in 6 rows... ..	(pg no. 16)
3.4 Example of label to one-hot encoding conversion.....	(pg no. 17)
3.5 Converting categorical values to encoded data	(pg no. 17)
3.6 Evaluation metric for 0.5 cut-off	(pg no. 24)
3.7 Evaluation metric for a general case from ROC	(pg no. 26)
3.8 Evaluation Metric from Cleveland Database	(pg no. 29)
3.9 Evaluation Metric for preliminary tree for SA Dataset	(pg no. 31)
3.10 Evaluation Metric for pruned tree for SA Dataset	(pg no. 33)
3.11 Evaluation Metric for preliminary tree for UCI Dataset.....	(pg no. 34)
3.12 Evaluation Metric for pruned tree for UCI Dataset.....	(pg no. 36)
3.13 Performance comparison for SA Dataset	(pg no. 37)
3.14 Performance comparison for UCI Dataset	(pg no. 37)

Chapter 1

Introduction

1.1 Motivation:

Medical diagnosis is the process of determining the cause of a patient's illness or condition by investigating information acquired from various sources including physical examination, patient interview, laboratory tests, patient's and the patient's family medical record, and existing medical knowledge of the cause of observed signs and symptoms. In this age of technology and digitalization, data has proven to be the fuel for organizations and industries. The healthcare industry is not far behind in this respect. However, this wealth of data is often left untapped due to the lack of adequate analytical tools, methods and personnel to discover insights and hidden relationships in this data. Getting a correct diagnosis is the most crucial step in treating a patient as it allows physicians to find the best treatment for the patient's condition. Furthermore, physicians are usually prone to cognitive bias and incorrect applications of heuristics during the diagnosis stage. They are more biased towards disease or conditions which they have diagnosed in the past. They often trust the initial diagnostic impression, even though further information might not support that initial assumption. Now, ML has been applied in a variety of areas within the healthcare industry such as diagnosis, personalized treatment, drug discovery, clinical trial research, radiology and radiotherapy, smart electronic health record, and epidemic outbreak prediction. The field of medical diagnosis in medical systems is quite rich with possibilities and its advantages such as cost-cutting, early diagnosis, and potentially saving human life. Furthermore, it can also aid in devising a monitory and preventive program for those who might be susceptible to suffering from CHD(Coronary heart diseases), based on their medical records and family history.

1.2 Problem Statement:

Heart disease is one of the biggest causes of morbidity and mortality among the population of the world. Prediction of cardiovascular disease is regarded as one of the most important subjects in the section of clinical data analysis. This makes heart disease a major concern to be dealt with. But it is difficult to identify heart disease because of several contributory risk factors such as diabetes, high blood pressure, high cholesterol, abnormal pulse rate, and many other factors. Due to such constraints, scientists have turned towards modern approaches like Data Mining and Machine Learning for predicting the disease. ML techniques allow the use of intelligent methods across different datasets to reveal useful insights. This reprogrammable ability of ML in exploring, processing and interpreting data sets makes it favourable for decision makers in domains such as medical diagnosis. Since detecting CHD involves training a model based on historical dataset, ML seems to be an appropriate technology to deal with this problem.

1.3 Methodology:

The methodology used can be defined in six phases from understanding the dataset to deployment. The first phase deals with a specification business goal and its transformation to the data mining context. The second phase focuses on detailed understanding through various graphical and statistical methods. Data preparation is usually the most complex and most time consuming phase including data aggregation, cleaning, reduction or transformation. In modelling, Neural networks are applied with various optimisation techniques preferable for a given preprocessed dataset. The dataset is used for training and testing using a 10 cross validation. The obtained results are evaluated by traditional metrics like accuracy, ROC depending on the type of distribution of data(balanced or unbalanced). Next, the model is optimised using hyperparameters optimisation. The last phase is devoted to the deployment of the best results in real usage and identification of the best or worst practices for the next analytical processes. Similarly decision trees are also employed. Decision Trees are supervised learning algorithms which have a predefined target variable & they are mostly used in non-linear decision making with simple linear decision surface. In other words, they are adaptable for solving any kind of problem at hand (classification or regression). One of the best and mostly used supervised learning methods are tree-based algorithms. They empower predictive modeling with higher accuracy, better stability and provide ease of interpretation. Unlike linear modeling techniques, they map nonlinear relationships quite well. Pruning is a technique in machine learning that reduces the size of decision trees by removing sections of the tree. It also reduces the complexity of the final classifier, and hence improves predictive accuracy by the reduction of overfitting. Post pruning can be done by first allowing the tree to grow to its full potential and then pruning the tree at each level after calculating the cross-validation accuracy at each level. Calculation of Entropy and Gini method behaves in the same way but entropy involves a log computation and Gini impurity involves a square computation. Since computing square is cheaper than logarithmic function- prefer Gini impurity.

Chapter 2

Literature Survey

Various research initiatives have been undertaken by experts, academic scholars and data science community in predicting and screening of medical data for various diseases. Multiple ML algorithms have been used in past researches to carry out these predictions. We will be reviewing relevant research works before we go ahead with our analysis on the dataset.

Apte et al [1] carried out prediction of heart disease by using a dataset having 13 attributes, which included main attributes like sex, blood pressure and cholesterol. The authors added two more attributes: smoking and obesity. ANN, Decision Tree and Naive Bayes classification techniques were used, and the results obtained pinpointed that ANN had the highest predictive accuracy on the utilized dataset.

As Heart Diseases are very serious diseases, many researchers tried to predict it or to extract crucial risk factors. The relatively known data sets are Cleveland, Hungarian, and Long Beach VA freely available on UCI machine learning repository. El-Bialy et al. used all these datasets in their study [2]. At first, authors selected five common variables for each dataset (Cleveland, Hungarian, Long Beach VA and Statlog project) and applied two data mining techniques: decision tree C4.5 and Fast Decision Tree (improved C4.5 by Jiang SU and Harry Zhang).

In [3], authors prepared the datasets for modeling phase, in which they applied some selected methods as decision trees, Naive Bayes (NB), Support Vector Machine (SVM) or Apriori algorithm, ANN on datasets such as south african heart diseases, UCI machine learning from Cleveland, Hungarian, and Long Beach VA, and Z-Alizadeh Sani Dataset using some statistical methods to investigate possible relations between input variables themselves or between them and target diagnostics.

H. Gonsalves et al [4] used classification techniques such as SVM , NB to classify the positive diagnosis cases from the dataset based on the performance of the classification models derived by the ML is measured using the confusion matrix. The confusion matrix is a contingency table that displays the number of instances assigned to each class thus allowing us to calculate the classification accuracy, sensitivity, specificity, true positives (TPs), true negatives (TNs), false positives (FPs), and false negatives (FNs) among others.

Chapter 3

Work Done

3.1 Data Understanding and Preparation:

3.1.1 Dataset 1. South African heart disease

The dataset for this research has been obtained from South African Heart Disease which is a subset of a larger dataset. It contains 462 instances (observations) and 10 attributes in all (shown in Table 1), of which 9 are independent factors and 1 variable, i.e. CHD is the dependent variable or labelled class. The dataset is a retrospective sample of males in a heart-disease high-risk region of the Western Cape in South Africa-KEEL where the labelled class CHD has two predictive outcomes: positive (1) and negative (0). There are no missing values in the dataset.

Each high-risk patient was monitored in this study and the attributes obtained were as follows: systolic blood pressure (sbp), cumulative tobacco in kg (tobacco), bad cholesterol also known as low density lipoprotein cholesterol (ldl), adiposity, family history of heart disease (famhist), type-A behaviour (typea), obesity, current alcohol consumption (alcohol), and age at onset (age).

Table 3.1 Attribute details of South African heart disease dataset

S.no.	Attribute	Domain	Data Type
1	Row	Index number	Integer
2	SBP	Systolic blood pressure [101 - 218]	Integer
3	Tobacco	Cumulative tobacco (Kg) [0 - 31.2]	Float
4	LDL	Low density lipoprotein cholesterol [0.98 - 15.33]	Float
5	Adiposity	Measure of % body fat [6.74 - 42.49]	Float
6	Famhist	Family history of heart disease [Present, Absent]	Categorical
7	typea	Characterized by an excessive competitive drive, impatience and anger/hostility [13- 78]	Integer
8	Obesity	Measured weight to height ratio [14.70 - 46.58]	Float
9	alcohol	Current alcohol consumption (0.00 – 147.19)	Float
10	Age	Age in years [15 - 54]	Integer
11	CHD	Coronary heart disease – target attribute 0 - Negative diagnosis 1 - Positive diagnosis	Categorical

Link to dataset file:

https://drive.google.com/file/d/1qmvq6775jVOLc-9FHX0Sj_Sv8Eb1GHAZ/view?usp=sharing

Pre-processing: The South African Heart Disease dataset obtained from Stanford datasets is available in .dat file format. As a result, we first converted the data from .dat to .csv. Moreover, in order to avoid confusion and gain clarity during analysis, we used the map feature in Python to convert the data type of ‘famhist’ from Integer to Factor. Thus, all the values ‘1’ were replaced by ‘Yes’ and all values ‘0’ were replaced by ‘No’. There were no missing values in the dataset. Also as the categorical variables such as Famhist are in binary form, we do not have to do anything special to them. Also we need to drop the first column of “Row” as it is just the index no. of the dataset.

The data has an unbalanced **nature**, as while visualising we observe 302 healthy and 160 positive diagnosis. (Total 462).

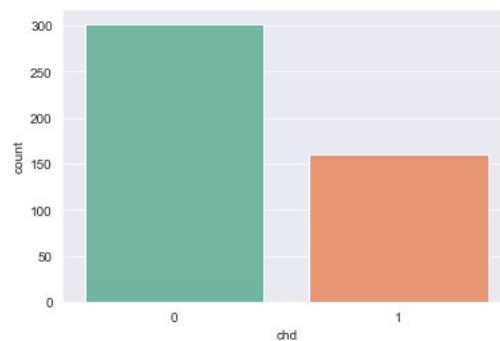


Fig3.1 Data distribution of CHD(Present -1, absent - 0)

Correlation analysis :

Correlation shows whether the characteristics are related to each other or to the target variable. Correlation can be positive (increase in one value, the value of the objective variable increases) or negative (increase in one value, the value of the target variable decreased).

From this heatmap we can observe that the ‘adiposity’ is highly correlated to obesity and age. However there’s not any attribute directly affecting the target variable.

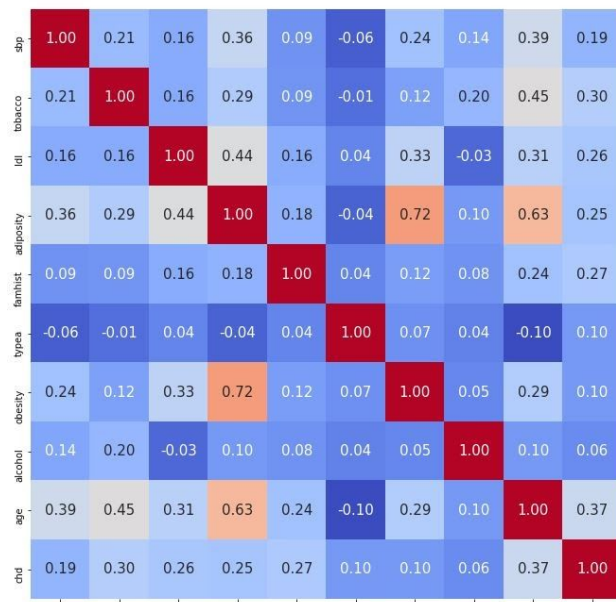


Fig3.2 Correlation matrix for South african dataset

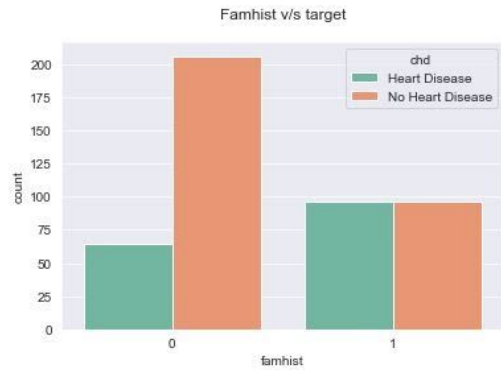


Fig3.3 Famhist attribute dependency

Visualising a relation between family history of heart disease (famhist) and target variable, we can say that a chance to be positively diagnosed is around 50% if some heart disease occurred in the family history.

Hence fam hist increases a chance for positive diagnosis.

Dependence on age:

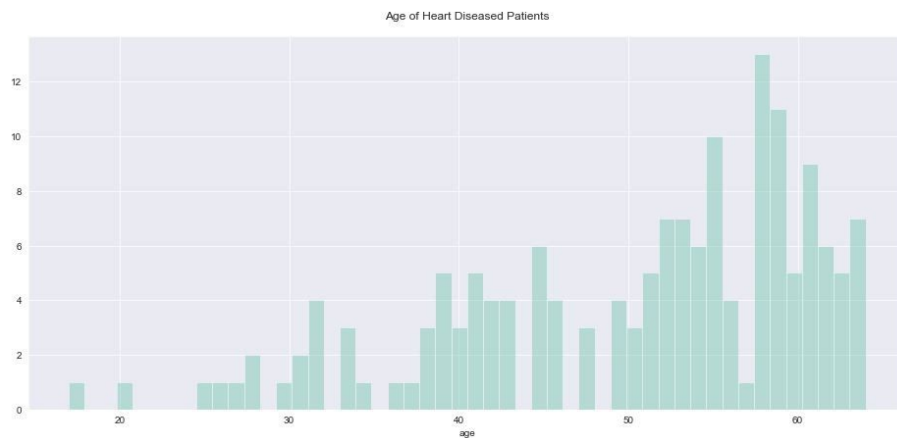


Fig 3.4 Distplot for age of CHD patients

It's highly visible that the chances of CHD are higher for those having age > 50 . Similarly According to [3] confirmed the importance of famhist = present, age , tobacco and LDL . All four variables increase a chance for the positive diagnosis of the HD.

Hence, the features are given by,

input attribute -

x^1 - sbp

x^2 - tobacco

x^3 - ldl

x^4 - adiposity

x^5 - famhist

x^6 - typea

x^7 - obesity

x^8 - alcohol

x^9 - age

y - chd

Min-max normalization is one of the most common ways to normalize data. For every feature, the minimum value of that feature gets transformed into a 0, the maximum value gets transformed into a 1, and every other value gets transformed into a decimal between 0 and 1.

$$\frac{value - min}{max - min}$$

3.1.2 Dataset 2. UCI Machine learning repository Datasets from Cleveland database

This dataset is composed of 14 attributes which are age, sex, chest pain type, resting blood pressure, serum cholesterol, fasting blood sugar, resting electrocardiographic results, maximum heart rate achieved, exercise induced angina, oldpeak. This database includes 76 attributes, but all published studies relate to the use of a subset of 14 of them.

The Cleveland database is the only one used by ML researchers to date. One of the major tasks on this dataset is to predict based on the given attributes of a patient that whether that particular person has a heart disease or not and other is the experimental task to diagnose and find out various insights from this dataset which could help in understanding the problem more. There are 303 instances in the dataset with 6 missing values.

The dataset was created by: -

1. Hungarian Institute of Cardiology. Budapest: Andras Janosi, M.D.
2. University Hospital, Zurich, Switzerland: William Steinbrunn, M.D.
3. University Hospital, Basel, Switzerland: Matthias Pfisterer, M.D.
4. V.A. Medical Center, Long Beach and Cleveland Clinic Foundation: Robert Detrano, M.D., Ph.D.

Link :

<https://drive.google.com/file/d/1NNl8rDIilwtwK8WhUukFbo80cswfbT-/view?usp=sharing>

S. no	Attribute	Domain	Data Type
1	Age	Age in years [28 - 77]	Integer
2	Sex	0 - Female , 1 - Male	Binary
3	CP	Chest Pain type 1 - Typical Angina 2 - Atypical Angina 3 - Non-Anginal pain 4 - Asymptotic	Categorical

4	restbp	Resting Blood pressure (mm/Hg) : [0, 200]	Decimal
5	chol	Serum Cholesterol(mg/dl) [0 - 603]	Decimal
6	fbs	Fasting blood pressure: Compares the fasting blood sugar value of an individual with 120mg/dl if fbs > 120 mg/dl : 1(true) if fbs < 120 mg/dl : 0 (false)	Binary
7	restecg	displays resting electrocardiographic results: 0 - Normal 1- having ST-T wave abnormality 2- showing probable or definite left ventricular hypertrophy by Este's criteria	Categorical
8	thalach	max heart achieved [60 , 202]	Decimal
9	exang	Exercise induced angina, [0: yes, 1: No]	Binary
10	oldpeak	ST depression induced by exercise relative to rest(mm) [-2.6, -6.2]	Decimal
11	slope	The slope of the peak exercise ST segment 1-upsloping 2-flat 3-downsloping	Categorical
12	ca	Number of major vessels colored by fluoroscopy [0, 3]	Integer
13	thal	displays the thalassemia: 0 - Null 1 - Fixed defect(no blood flow in some parts of heart) 2 - Normal blood flow 3 - Reversible defect (A blood flow is observed but it is not normal)	Categorical
14	Target	Coronary heart disease – target attribute 0 - Negative diagnosis 1 - Positive diagnosis	Binary

Table 3.2. Attributes details of Cleveland heart disease dataset

Pre Processing:

The dataset is available in csv format with all the values in decimals format with 6 missing values. The dataset needs to be applied to a softmax function as a testing layer for probability of both the possible classes 1- CHD positive, 0- Negative.

The data has a balanced nature, as while visualisation we observe 165 Positive diagnosis and 138 Negative diagnosis.

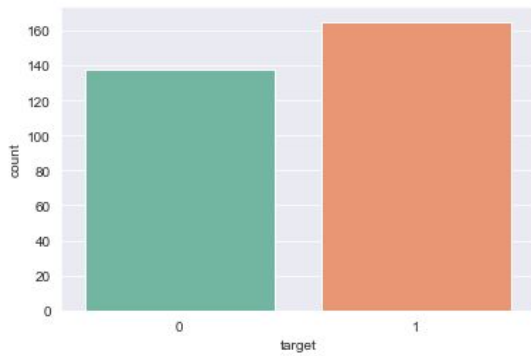


Fig 3.5. Data distribution in cleveland CHD (present-> 1, absent ->0)

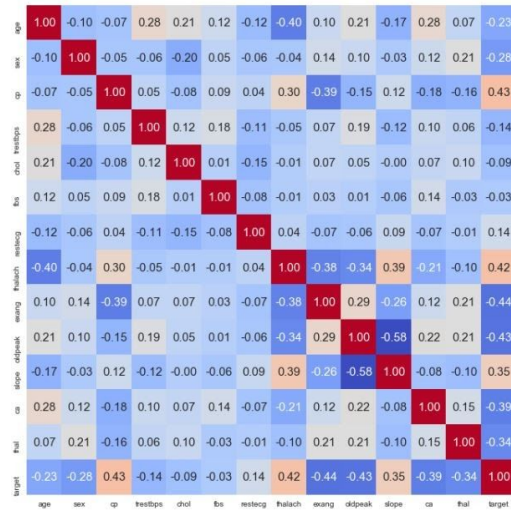


Fig 3. 6. Correlation matrix for cleveland dataset

Correlation analysis : Correlation shows whether the characteristics are related to each other or to the target variable. Correlation can be positive (increase in one value, the value of the objective variable increases) or negative (increase in one value, the value of the target variable decreased). **From this heatmap we can observe that the ‘cp’ chest pain has the highest relation to the target variable mere 0.43 which is not enough to directly relate the attributes.** Medical emergency is a heart attack. Tissue loses oxygen without blood and dies causing chest pain. **However there isn’t much correlation between other attributes.**

Missing Values -

The two attributes ca (No. of vessels in fluoroscopy) and thal (thalassemia) has a total of 6 instances with missing value. Here (?) means a missing value.

Table 3.3 Missing values in 6 rows

	age	sex	cp	restbp	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	hd
87	53.0	0.0	3.0	128.0	216.0	0.0	2.0	115.0	0.0	0.0	1.0	0.0	?	0
166	52.0	1.0	3.0	138.0	223.0	0.0	0.0	169.0	0.0	0.0	1.0	?	3.0	0
192	43.0	1.0	4.0	132.0	247.0	1.0	2.0	143.0	1.0	0.1	2.0	?	7.0	1
266	52.0	1.0	4.0	128.0	204.0	1.0	0.0	156.0	1.0	1.0	2.0	0.0	?	2
287	58.0	1.0	2.0	125.0	220.0	0.0	0.0	144.0	0.0	0.4	2.0	?	7.0	0
302	38.0	1.0	3.0	138.0	175.0	0.0	0.0	173.0	0.0	0.0	1.0	?	3.0	0

So 6 of 303 rows, or 2% contain missing values. Since $303-6 = 297$, and 297 is still plenty of data to build classifiers, we will remove the rows with missing values, rather than to try to

impute the values.

Now the total data length is **297 instances**.

In case of categorical data for features - cp, restecg, slope, thal, we need to change the values as because for example in For the cp (chest pain) column, we have 4 options:

1. typical angina
2. atypical angina
3. non-anginal pain
4. asymptomatic

If we treated these values, 1, 2, 3 and 4, like continuous data, then we would assume that 4 which means "asymptomatic", is mere similar to 3, which means "non-anginal pain", than it is to 1 or 2, which are other types of chest pain. That means the classifier would be more likely to cluster the patients with 4s and 3s together than the patients with 4s and 1s together In contrast, if we treat these numbers like categorical data, then we treat each one as a separate category that is no more or less similar to any of the other categories. Thus, the likelihood of clustering patients with 4s with 3s is the same as clustering 4s with 1s, and that approach is more reasonable.

Hence, we use one hot-encoding using the `get_dummies()` method. **One hot encoding** is a process by which categorical variables are converted into a form that could be provided to ML algorithms to do a better job in prediction. Rather than labeling things as a number starting from 1 and then increasing for each category, we'll go for more of a **binary** style of categorizing.

cp	cp_1.0	cp_2.0	cp_3.0	cp_4.0
1.0	1	0	0	0
4.0	0	0	0	1
4.0	0	0	0	1
3.0	0	0	1	0
2.0	0	1	0	0

[Table 3.4 example of label to one hot encoding conversion]

We will convert these 4 feature into their encoded form -

	age	sex	resttbp	chol	fbs	thalach	exang	oldpeak	ca	cp_1.0	...	cp_4.0	restecg_0.0	restecg_1.0	restecg_2.0	slope_1.0	slope_2.0	slope_3.0	thal_3.0	thal_6.0	thal_7.0
0	63.0	1.0	145.0	233.0	1.0	150.0	0.0	2.3	0.0	1	...	0	0	0	1	0	0	1	0	1	0
1	67.0	1.0	160.0	286.0	0.0	108.0	1.0	1.5	3.0	0	...	1	0	0	1	0	1	0	1	0	0
2	67.0	1.0	120.0	229.0	0.0	129.0	1.0	2.6	2.0	0	...	1	0	0	1	0	1	0	0	0	1
3	37.0	1.0	130.0	250.0	0.0	187.0	0.0	3.5	0.0	0	...	0	1	0	0	0	0	1	1	0	0
4	41.0	0.0	130.0	204.0	0.0	172.0	0.0	1.4	0.0	0	...	0	0	0	1	1	0	0	1	0	0

5 rows × 22 columns

Table 3.5 Converting categorical values to encoded data

This gives us 22 independent separate features which can be visualised as

x^1 - age	x^2 - sex	x^3 - restbp
x^4 - chol	x^5 - fbs	x^6 - thalach
x^7 - exang	x^8 - oldpeak	x^9 - ca
x^{10} - cp_1.0	x^{11} - cp_2.0	x^{12} - cp_3.0
x^{13} - cp_4.0	x^{14} - restecg_0.0	x^{15} - restecg_1.0
x^{16} - restecg_2.0	x^{17} - slope_1.0	x^{18} - slope_2.0
x^{19} - slope_3.0	x^{20} - thal_3.0	x^{21} - thal_6.0
x^{22} - thal_7.0	y - hd	

According to this Cleveland dataset **males are more susceptible** to get Heart Disease than females.

Sudden Heart Attacks are experienced by men between 70% — 89%.

There are four types of chest pain, asymptomatic, atypical angina, non-anginal pain and typical angina.

Most of the Heart Disease patients are found to have asymptomatic chest pain.

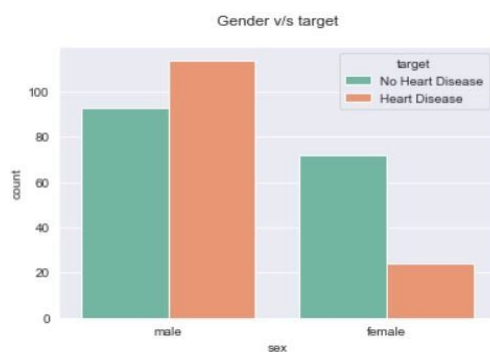


Fig 3.7. Gender vs Target

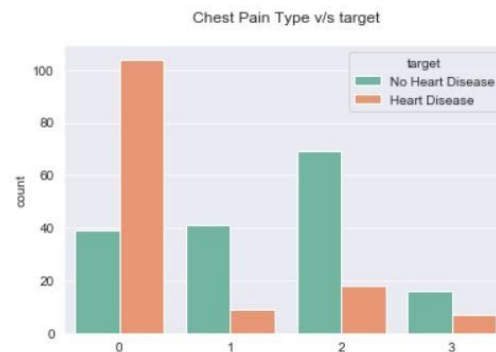


Fig 3.8. Chest pain vs Target

Slope: The only way to tell If you had an asymptomatic attack is by an **Electrocardiogram or Echocardiogram**. These tests can reveal changes that signal a heart attack.

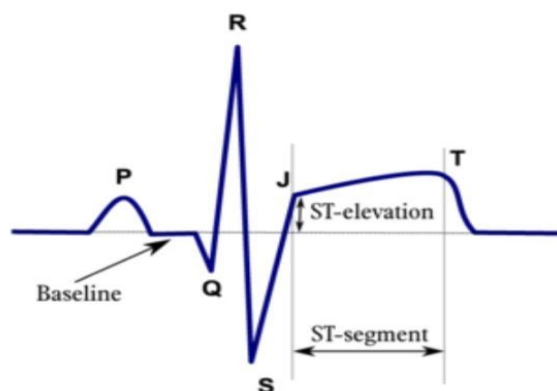


Fig 3.9. An example of ST segment from electrocardiography

For a better understanding of variables describing the EKG results, we present an example of ST segments.

- 1-upsloping
- 2-flat
- 3-downsloping

Flat, downsloping or depressed ST segments may indicate coronary ischemia.

3.2 Modelling and Evaluation

We used python's Jupyter notebook to build two machine learning classifiers to evaluate both the dataset's performance, **Python's** simple syntax and readability promote rapid testing of complex algorithms, and make the language accessible easily. We applied selected data mining methods to predict the Heart Disease and to extract relevant rules. For the prediction models, we verified experimentally three data divisions to training and testing sets. The estimation methodology used for testing the ML technique against the considered dataset is k-fold cross validation.

3.2.1 Neural Networks

In neural network application a 10 cross fold validation approach is used, In this, the entire dataset is split into 10 folds and at each iteration, 9 folds are used for training the ML models and the remaining 1-fold is used for evaluation. This process is repeated 10 times and the error rate is recorded at each iteration. The final error of the model is the average of all errors produced during the 10 runs. It maps a set of input data onto a set of appropriate output data. It consists of 3 layers input layer, hidden layer & output layer. There is connection between each layer & weights are assigned to each connection. The primary function of neurons of the input layer is to divide input x_i into neurons in a hidden layer. Neuron of the hidden layer adds input signal x_i with weights w_{ji} of respective connections from the input layer.

$$Y_j = f\left(\sum w_{ji} x_i\right)$$

Where f is a simple threshold function such as sigmoid or hyperbolic tangent function.

Parameter initialisation:

Training your neural network requires specifying an initial value of the weights. A well chosen initialization method will help learning. A well chosen initialization can:

1. Speed up the convergence of gradient descent
2. Increase the odds of gradient descent converging to a lower training (and generalization) error

However as using zero as initializer of weights ultimately fails the derivative with respect to loss function is the same for every w in $W[l]$, thus all **weights** have the same value in subsequent iterations. However, An important thing to keep in mind is that **biases** have no effect whatsoever when **initialized** with 0.

Here we used a special initialization, "Xavier initialization", which uses a scaling factor for the weights of $\sqrt{1/\text{layers_dims}[l-1]+\text{layers_dims}[l]}$ where l represent the current layer no. and layer dims represent the dimension of the layer, this scales the random generation of weights, we use the fixed random initialisation function in python call `np.seed()` to ensure similar results.

Loss function and Computing cost :

The intuition behind calculating cost function is to see how the error is reduced over an entire training set or in general a cost function is a measure of "how good" a model did for its training set. In mathematical/statistical terms though this is represented as, getting closer to the solution by way of 'minimising the error rate' at each step. Cost function which we used for both of our neural network is:

$$J(W, b) = \frac{1}{m} \sum_{i=1}^m L(\hat{y}^{(i)}, y^{(i)})$$

$$L(\hat{y}, y) = -(y \log \hat{y} + (1 - y) \log(1 - \hat{y}))$$

How do neural networks learn: The learning process is about changing the values of the W and b parameters so that the loss function is minimized. In order to achieve this goal, we will turn to for help with calculus and use gradient descent method to find a function minimum. In each iteration we will calculate the values of the loss function partial derivatives with respect to each of the parameters of our neural network.

Backpropagation: Back-propagation is the essence of neural net training. It is the method of fine-tuning the weights of a neural net based on the error rate obtained in the previous epoch (i.e., iteration). Proper tuning of the weights allows you to reduce error rates and to make the model reliable by increasing its generalization. In the equations above, α represents learning rate - a hyperparameter which allows you to control the value of performed adjustment. Choosing a learning rate is crucial — we set it too low, our NN will be learning very slowly, we set it too high and we will not be able to hit the minimum.

$$\begin{aligned} \mathbf{W}^{[l]} &= \mathbf{W}^{[l]} - \alpha \mathbf{dW}^{[l]} \\ \mathbf{b}^{[l]} &= \mathbf{b}^{[l]} - \alpha \mathbf{db}^{[l]} \end{aligned}$$

$$\begin{aligned} \mathbf{dW}^{[l]} &= \frac{\partial L}{\partial \mathbf{W}^{[l]}} = \frac{1}{m} \mathbf{dZ}^{[l]} \mathbf{A}^{[l-1]T} \\ \mathbf{db}^{[l]} &= \frac{\partial L}{\partial \mathbf{b}^{[l]}} = \frac{1}{m} \sum_{i=1}^m \mathbf{dZ}^{[l](i)} \\ \mathbf{dA}^{[l-1]} &= \frac{\partial L}{\partial \mathbf{A}^{[l-1]}} = \mathbf{W}^{[l]T} \mathbf{dZ}^{[l]} \\ \mathbf{dZ}^{[l]} &= \mathbf{dA}^{[l]} * g'(\mathbf{Z}^{[l]}) \end{aligned}$$

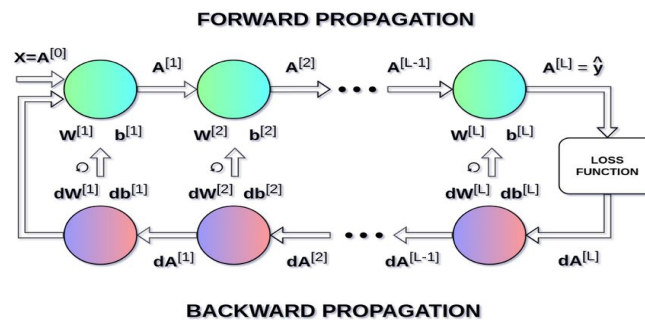


Fig 3.10. Weights, bias and gradient descent in neural network

3.2.1.1 Neural Network Model for South African dataset

The given model can be represented using the following equations, consisting of **an input, one hidden layer and output layer**, where i is the no. of instances (size of training set in cross validation).

The dataset is divided into the ratio of 80%/20% (370/92) where 10 cross validation is applied to the 80% of the dataset and rest 20% is left for evaluating the classifier. While checking for the possible divisions, this gave the best result:

- First layer consists of 9 input attributes represented by $x^{(i)}$ which has dimension of (9,1).
- In the hidden layer the main challenge is to determine the number of neurons required for best results, in the 10 cross validation method there are two sets and iterating over various possible number of neurons ranging [3,4,5,6,7,8] such that overfitting is avoided and the model is more generalised. The activation function used in this layer is **tanh function** a.k.a. hyperbolic **tangent** function, is a rescaling of the logistic sigmoid, such that its outputs **range** from -1 to 1, which is fed with the linear combination of input variables, weights and biases and returns a non linear output $a^{[1]}$ through the tanh function.
- The linear combination $z^{[2]}$ of this layer is fed to the output activation function, sigmoid. The main reason why we **use sigmoid function** is because it exists between (0 to 1). Therefore, it is especially **used** for models where we have to predict the probability as an output. Since probability of anything exists only between the range of 0 and 1, **sigmoid** is the right choice. For making our model simple we chose only one hidden layer and after performing 10 cross validation on different sizes of neurons in the hidden layer, the most appropriate size of neurons came out to be **n_h = 5**.

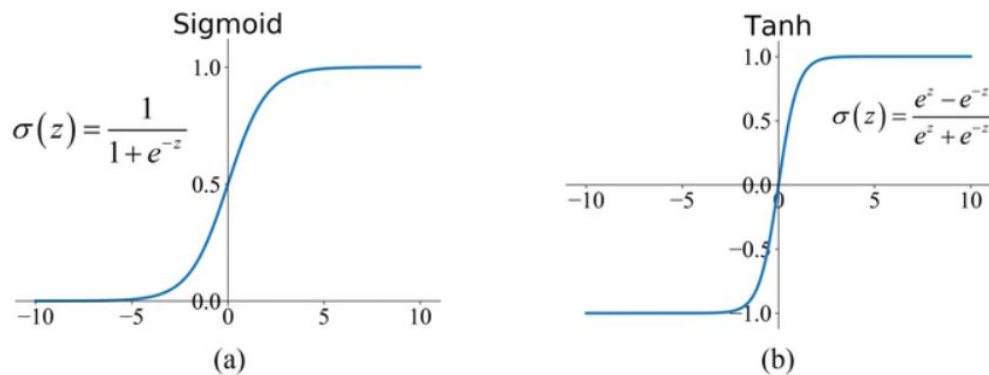


Fig 3.11. Tan and sigmoid functions

As you can see in the graph that the cost function is decreasing as the number of iterations is increasing. Thus our gradient descent is running smoothly. The lowest cost associated with our neural network is 0.34% which can be reduced by tuning hyperparameters.

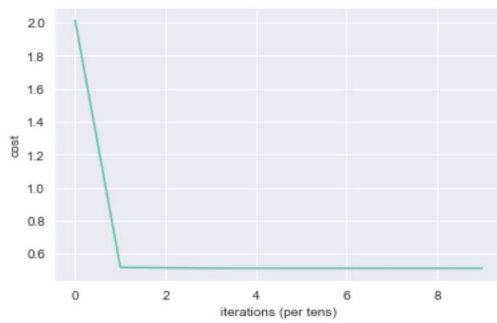


Fig 3.12. Cost vs iteration in thousands

Learning Rate taken into account is 1.5(alpha).

As the cross validation set resulted in the best case for the number of neurons = 5 in the hidden layer based on the accuracy measure, the average of all the 10 validation steps was considered and the model nearest to the value of mean was considered to be final, the cross validation being of size 10 results in 10 random selection of 1/10(37) of the whole training set for validation whereas the other 9/10(333) set is used to train this allows us to train several ML models on subsets of the available input data and evaluating them on the complementary subset of the data. Using cross-validation allows to detect overfitting, ie, failing to generalize a pattern.

The model parameters -

W1(The weight matrix hidden layer which receives input from input layer)**dimension -> [9,5]**

B1(The bias matrix for hidden layer) - **dimension -> [5,1]**

W2(The weight matrix for the output layer from hidden layer) - **dimension -> [5,1]**

B2 (The bias matrix for the output layer)- **dimension -> [1,1]**

- While the weights were initialised using the Xavier initialization , It tries to make sure the distribution of the inputs to each activation function is zero mean and unit variance resulting in smarter weights avoiding saturation.

```
W1 = np.random.randn(n_h,n_x)*np.sqrt(1/n_h+n_x)
b1 = np.zeros((n_h,1))
W2 = np.random.randn(n_y,n_h)*np.sqrt(1/n_h+n_y)
b2 = np.zeros((n_y,1))
```

These random weights are hence scaled by a factor dependent on the layer sizes between which the weights are present.

Fig 3.13. Xavier Initialisation

While updating the weights during backpropagation while training the model overfitting were avoided using L2 regularisation techniques to avoid overfitting.

- L2 Neural network regularization is a technique used to reduce the likelihood of model overfitting. Model overfitting can occur when you train a neural network excessively. This leads to a situation where the trained neural network model predicts the output values in the training data very well, with little error and high accuracy, but when the trained model is applied to new, previously unseen data, the model predicts poorly. **L2 regularization** works by adding a term to the error function used by the training algorithm. The additional term penalizes large weight values. In L2 regularization we add a fraction of the sum of the squared weight values to the base error of the sum of the squared weight values to the base error.

Z1 (The linear combination of weight and bias) - $\text{np.dot}(W1, X.T) + B1 \rightarrow \text{dimension } [5,1]$

A1(Output of tanh function) - $\text{np.tanh}(Z1) \rightarrow \text{dimension } [5,1]$

Z2(The linear combination of weight and bias in output layer) $\rightarrow \text{dimension } [1,1]$

A2(The probability output of sigmoid function) $\rightarrow \text{dimension } [1,1]$

These dimensions remain the same for every instance of the data.

Fig 3.15. Neural Network model for dataset 1

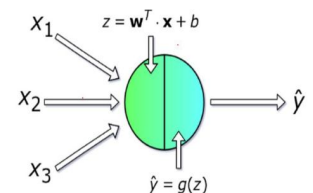
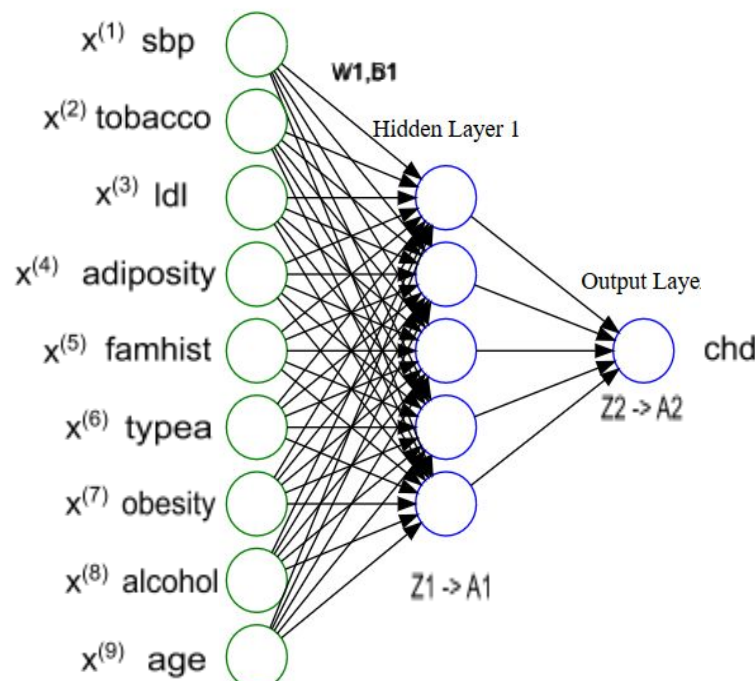


Fig 3.14. A single neuron representation

Training set - (x_train) 372 instances where in each fold 37 were in the validation set and 335 were used to train.

Testing set : 92 observations available after spilt.

Evaluation: As the data is highly unbalanced the test set of 92 observation has a high certainty of being biased towards a particular class, which might lead a false metric to evaluate to judge the classifier as because if the probability of sigmoid to consider classes if

$$\text{Class} = \begin{cases} 0 & \text{if } A2 < 0.5 \\ 1 & \text{if } A2 \geq 0.5 \end{cases}$$

This might lead to huge misclassification loss and a huge bias might appear, if we draw the Confusion matrix for this consideration we get, The quality and character of ML is exposed in the nature of the performance metrics chosen.

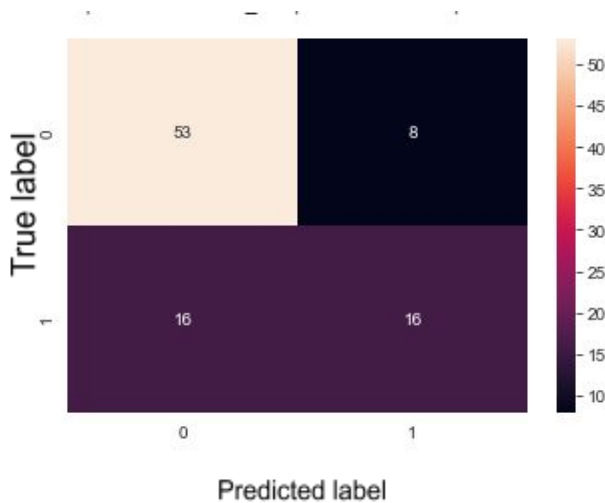


Fig 3.16. Confusion matrix for 0.5 cut-off

here, TP - 16, FP - 8, FN - 16, TN - 53

1. **True Positive (TP):** Number of patients that are predicted to have CHD and do actually have CHD.
2. **False Positive (FP):** Number of patients that are predicted to have CHD and do not actually have CHD.
3. **False Negative (FN):** Number of patients that are predicted to not have CHD but do actually have CHD.
4. **True Negative (TN):** Number of patients that are predicted to not have CHD and do not actually have CHD.

Here the most costliest case for us are the False negatives as a person might die if he is predicted healthy when actually he was suffering from disease, hence our aim should be to reduce this.

- One of the most common measures of performance comparison in classification analysis is **Accuracy**. It is the number of predictions made correctly out of the total predictions made by the model. However it **does not allow us to evaluate specifically according to the need of the problem**.

- The next measure of performance is **Sensitivity(True positive rate)**. It measures the number of instances correctly predicted as positive by the classifier out of the total number of instances that are actually positive. Sensitivity is also known as Recall or True Positive Rate.

As this includes both TP and FN a higher value will result in better prediction of those who are actually suffering from disease and would try to not make any fatal prediction.

$$\text{Sensitivity} = \frac{TP}{(TP+FN)} \quad \text{Specificity} = \frac{TN}{(TN+FP)}$$

- Another measure is **specificity(1-false positive rate)** It measures the number of correct negative predictions by the classifier out of the total number of instances that are actually negative. It is also known as True Negative Rate. But as it predicts related to negative class which is of healthy population, it is not as costly as the Sensitivity is. As even if a person is falsely predicted that he has the disease, he can go for further tests, as these algorithms are for help of medical practitioners whose main aim should be to avoid the possibility of falsely predicting when someone actually has a disease.

Evaluation Metric	Value
Accuracy	74.19
Sensitivity(TPR)	0.5
Specificity(1-FPR)	0.86
FPR	0.13

Table 3.6. Evaluation metrics for 0.5 cut-off

- Hence our main focus is to improve sensitivity of the model, here in case when the prediction had a cutoff of 0.5, we can observe that the sensitivity of the model is quite low, as the no. of false negatives are high. Hence, to achieve the best case we need to change the cut-off for prediction, as the output for the 97 instances of test set from sigmoidal layer **A2** lies between 0 and 1(Posterior probabilities).

- This is since the bi-variable response of the labelled class is unequal. Out of the original 462 instances, only 160 patients are said to have CHD whereas the remaining 302 individuals do not suffer from CHD. This discrepancy might sublimely influence the accuracy rate as the model can predict all values of the majority class and thus achieve an overall high accuracy while blinding out the mis-predictions occurring in the minority class.

- Hence, we plot a **Receiver operating curve(ROC)** based on the posterior probability given to the predict function in which a range of probabilities for sigmoid function are considered and by calculating the values of **TPR and FPR** from their respective parameters(TP, TN, FP, FN) coming from each case (probability) the TPR and FPR are plotted. This graph helps us to find the optimum threshold point specific to our application as in this case we need TPR to be high, but it highly depends on the practitioners requirement as to how much TPR and FPR they need from the model to predict disease without compromising for any potential fatal scenario.

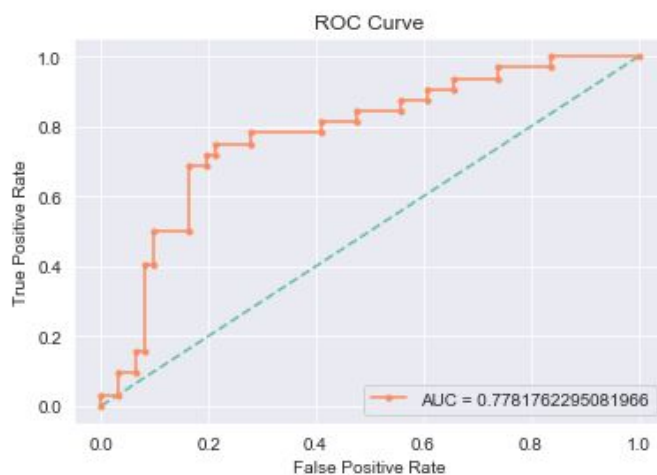


Fig 3.17. ROC curve

As observed from the ROC CURVE it now depends on the application of the medical practitioner as to how high TPR and FPR he wants for his application and the corresponding value of threshold can be determined from the graph. However the as mentioned in the paper[4] that the best case scenario in medical diagnosis is to achieve sensitivity or specificity rates greater than 0.8 but in the model the best case achieved was a little less this can be resolved **Increasing the number of instances under study and balancing the bi-variable responses of the labelled class might help to improve the performance of the ML models.**

This optimal case was achieved by the threshold value of **0.32180** on the ROC curve as the value of TPR and FPR here were 0.75(sensitivity) and 0.78 these threshold values can be easily found out through the ROC curve for any value of TPR and FPR according to requirement. In this specific case as well we can see that the accuracy and sensitivity has increased.



Fig 3.18. Confusion matrix for a general case from ROC

Evaluation metric	Value
Accuracy	77.14
Tpr(Sensitivity)	0.75
Specificity	0.78
Fpr(1-Specificity)	0.213

Table 3.7. Evaluation Metric for a general case from ROC

Also the No. of false negative values have reduced. However this is just an example of roc application it can provide user specific output and even better cases if more instances of data is provided.

3.2.1.2 Neural Network Model for Cleveland heart disease dataset(UCI repository)

As in this case we had $303 - 6 = 297$ instances as 6 instances had a missing value in either ca or thal removing these values did not affect the whole model as these contribute only 2% of the data, this data. The data is split into 80% /20 % (236/61) for modelling.

Here as 4 variables had more than 2 classes they have been encoded to a binary format Because neural networks work internally with numeric data, binary data (such as sex, which can be male or female) and categorical data (such as a cp, which can be typical angina, atypical angina, non anginal pain and asymptomatic) must be encoded in numeric form.

- Hence the size of **x(Input layer) is 236 x 22.(while training as 236 examples are available.)**
- Similar to the previous model the hidden layer performed best for the tanh hyperbolic function.
- The output layer here consists of a softmax function instead of sigmoid as the dataset

here is somewhat balanced. As while computing probability of both the classes a considerable amount of difference was observed in the resulting probability of each example allowing us to easily compute the best class for the examples. The main reason why we **use softmax function** is because it gives us probability of each class. In a balanced dataset, it is often used to find out the probability of occurrence of each class.

The softmax function outputs are probabilities of each class for e.x. in a cross validation while training we get probabilities of every instance for class [0(negative) , 1(positive)] which helps determine the class of any particular instance.

MODEL PARAMETERS -

- The number of iterations for back propagation **5000**
- As the cross validation set resulted in the best case for the **number of neurons = 5** in the hidden layer based on the bias - variance trade-off, the average of all the 10 validation steps was considered and the model nearest to the value of mean was considered to be final. the cross validation being of size 10 results in 10 random selection of 1/10(24) of the whole training set for validation whereas the other 9/10(214) set is used to train
- **W1**(The weight matrix of hidden layer which receives input from input layer)**dimension [22,5]**
- **B1**(The bias matrix for hidden layer) - **dimension -> [5,1]**
- **W2**(The weight matrix for the output layer from hidden layer) - **dimension -> [5,1]**
- **B2** (The bias matrix for the output layer)- **dimension -> [1,1]**
- Learning rate used along with L2 regularisation to prevent overfitting where learning rate was **0.4**.
- **Z1** (The linear combination of weight and bias) - **np.dot(W1,X.T)+B1 -> dimension [5,1]**
- **A1**(Output of tanh function) - **np.tanh(Z1) -> dimension [5,1]**
- **Z2**(The linear combination of weight and bias in output layer) -> **np.dot(W2,A1.T)+B2 dimension [2,1]**
- **A2**(The probability output of sigmoid function)-> **dimension [2,1]**
- These dimensions remain the same for every instance of the data.

Softmax function can be visualised as

Softmax
activation function

$$\frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}}$$

It states that we need to apply a standard exponential function to each element of the output layer, and then normalize these values by dividing by the sum of all the exponentials. Doing so ensures the sum of all exponentiated values adds up to 1.

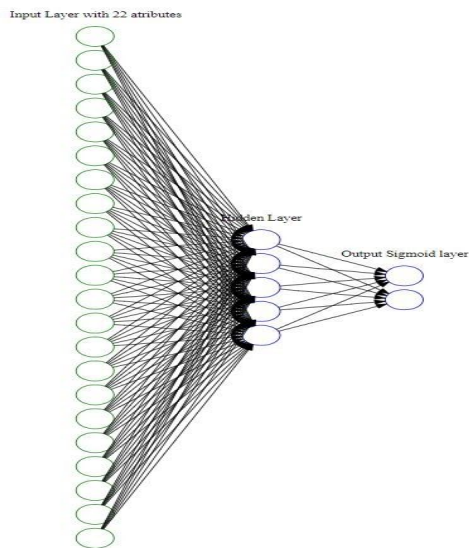


Fig 3.19. Neural Network model for Cleveland dataset

The first group of experiments resulted in the best model generated by a neural network visualized. This model contained only a reduced set of input variables. The best Neural network model was generated by, 10 cross-validation, and all original variables. While training the model with the train and cross validation set the Average accuracy of the best model was with hidden layers size $n_h=5$.

As we used Softmax function on the output layer the prediction of each instance comes to be as probability of each class which can be compared and an special function `argmax` allows us to return the higher probability class as output.

```
[[0.93465059 0.06308683 0.73595938 0.0224373 0.06532665 0.04305326
 0.08805488 0.98348327 0.89425315 0.9790334 0.20491415 0.08419246
 0.99134014 0.6661379 0.13348793 0.92105669 0.44903885 0.98190499
 0.78997441 0.00507148 0.76075868 0.9950977 0.26083065 0.99261708
 0.08205789]
[0.06534941 0.93691317 0.26404062 0.9775627 0.93467335 0.95694674
 0.91194512 0.01651673 0.10574685 0.0209666 0.79508585 0.91580754
 0.00865986 0.3338621 0.86651207 0.07894331 0.55096115 0.01809501
 0.21002559 0.99492852 0.23924132 0.0049023 0.73916935 0.00738292
 0.91794211]]
```

Fig 3.20. Softmax layer probability output

As these both array represent the values of probability of both class of each example, in each case the higher probability is **near to 1** and the probability is the complement to it as the sum of

both probabilities is **1 in a softmax function**. These output values can easily be compared with the original dataset as it's a balanced dataset with a ratio of roughly half, hence their **softmax function** is used because it gives us probability of each class. In a balanced dataset, it is often used to find out the probability of occurrence of each class.

The accuracy achieved in this model is substantially near to models suggested by [3] in literature reviews.

Accuracy test: 83.01666666666668 - Accuracy train: 87.73961019743794%

Fig 3.21 Confusion matrix for cleveland data

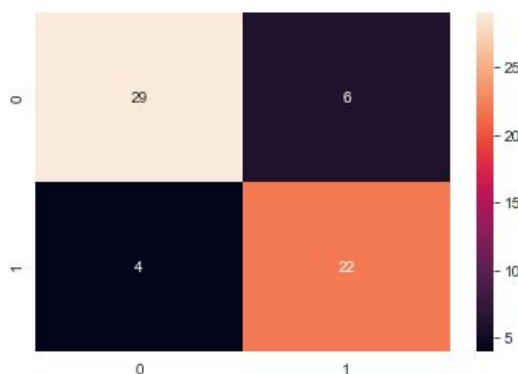


Table 3.8. Evaluation Metric for Cleveland dataset

Evaluation Metric	Value
Accuracy	83.016
Tpr(Sensitivity)	0.846
Specificity	0.828
Fpr(1-specificity)	0.171

3.2.2 Decision Trees

A Decision tree is a flowchart like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label.

Working of a Decision tree:

1. The root node feature is selected based on the results from the Attribute Selection Measure(ASM).
2. The ASM is repeated until a leaf node, or a terminal node cannot be split into sub-nodes.

We used the CART type decision tree which employs the Gini Index as its ASM.

Gini Index: The measure of the degree of probability of a particular variable being wrongly classified when it is randomly chosen is called the Gini index or Gini impurity.[19] Let's perceive the criterion of the Gini Index, the **Gini index varies between values 0 and 1**, where 0 expresses the purity of classification, i.e. all the elements belong to a specified class or only one class exists there. And 1 indicates the random distribution of elements across various classes. The value of 0.5 of the Gini Index shows an equal distribution of elements over some classes. While designing the decision tree, the features possessing the least value of the Gini Index would

get preferred.[20]

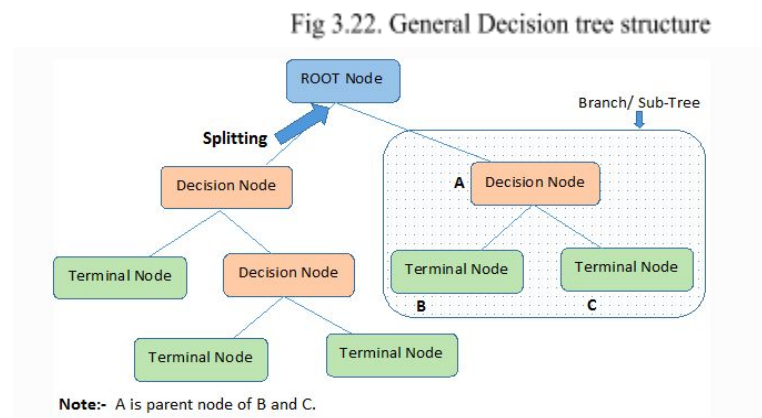
Mathematical Formula :

$$\text{Gini} = 1 - \sum_{i=1}^n (p_i)^2$$

P_i = probability of an object being classified into a particular class.

Decision Tree Representation:

Decision trees classify instances by sorting them down the tree from the root to some leaf node, which provides the classification of the instance. An instance is classified by starting at the root node of the tree, testing the attribute specified by this node, then moving down the tree branch corresponding to the value of the attribute. This process is then repeated for the subtree rooted at the new node.[18]



Source- <https://www.kdnuggets.com/>

3.2.2.1 Decision Tree Model for South African dataset

As decision trees are not able to handle categorical data so we converted all categorical data to numerical through one hot encoding and then we provided our data to the classifier. The train and test split is the same as mentioned in the Neural Networks model. Refer to the section 3.2.1.1 for more details.

Initially we built a preliminary decision tree without any hyperparameter tuning. In this case our training data was overfitting as expected because it built a node for every feature. The training and test accuracy we achieved in this classifier is:-

Accuracy on training set: 100 %

Accuracy on test set: 56.9 %

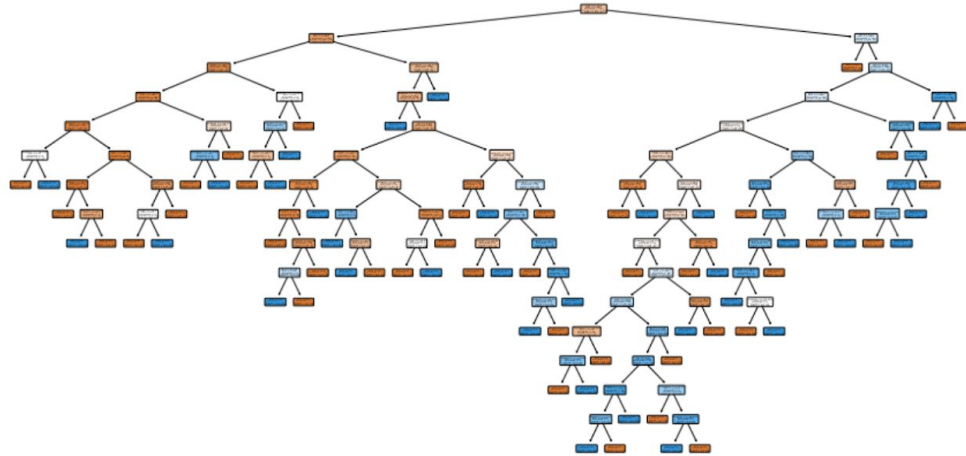


Fig 3.23 Preliminary Tree for South African data

Confusion Matrix:

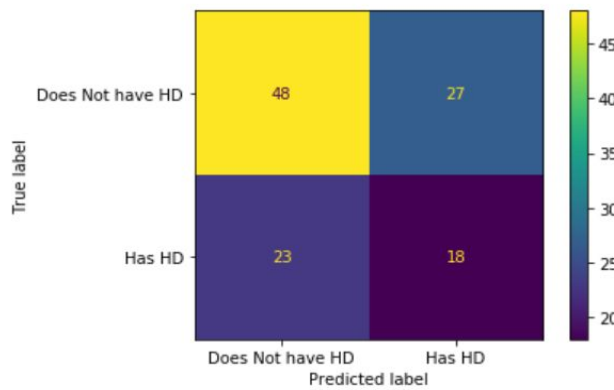


Table 3.9. Evaluation metric for preliminary tree

Evaluation Metric	Value
Train Accuracy	100%
Test Accuracy	56.9%
Sensitivity(Tpr)	0.43
Specificity	0.64
Fpr(1-Specificity)	0.36

Fig 3.24 Confusion metric for Preliminary tree

It can be observed that of the $48 + 27 = 75$ people that did not have Heart Disease, (64%) were correctly classified. And of the $23 + 18 = 41$ people that have Heart Disease, (43%) were correctly classified.

As it is clear from the above figure that this classifier is not generalized well and has overfit the training dataset. Now our main concern was to reduce to the variance and generalise the classifier. It can be done by pruning the tree. The sensitivity and specificity of the model is very low, which may result in fatal cases as False negatives are high.

Minimal Cost Complexity Pruning:

Minimal cost-complexity pruning is an algorithm used to prune a tree to avoid over-fitting. This algorithm is parameterized by $\alpha \geq 0$ known as the complexity parameter. The complexity parameter is used to define the cost-complexity measure, $R_\alpha(T)$ of a given tree T :

$$R_\alpha(T) = R(T) + \alpha|T|$$

where $|T|$ is the number of terminal nodes in T and $R(T)$ is traditionally defined as the total misclassification rate of the terminal nodes. We used the total sample weighted impurity of the terminal nodes for $R(T)$. As shown above, the impurity of a node depends on the criterion.

Minimal cost-complexity pruning finds the subtree of T that minimizes $R_\alpha(T)$.

The cost complexity measure of a single node is $R_\alpha(t) = R(t) + \alpha$. The branch, T_t , is defined to be a tree where node t is its root. In general, the impurity of a node is greater than the sum of impurities of its terminal nodes, $R(T_t) < R(t)$.

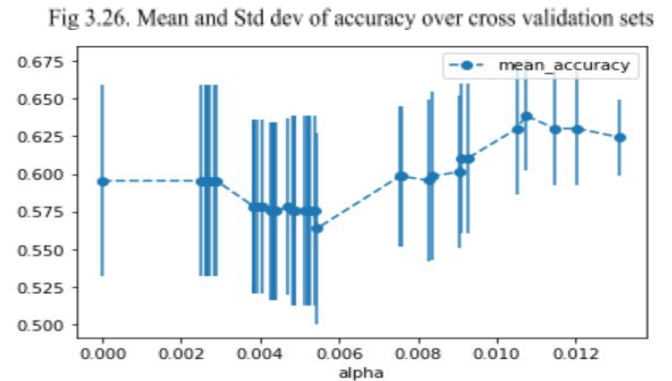
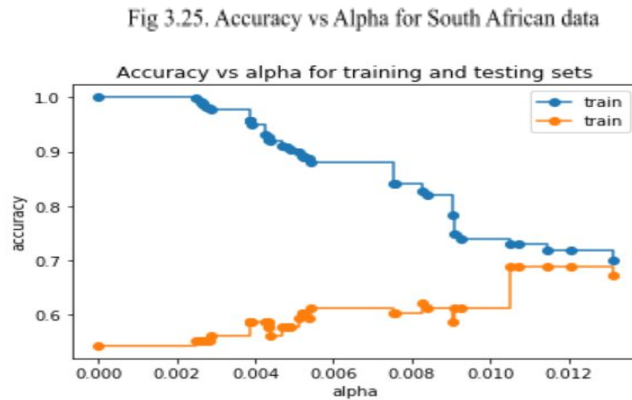
However, the cost complexity measure of a node, t , and its branch, T_t , can be equal depending on α . We define the effective α of a node to be the value where they are equal, $R_\alpha(T_t) = R_\alpha(t)$ or $\alpha_{eff}(t) = \frac{R(t) - R(T_t)}{|T| - 1}$. A non-terminal node with

the smallest value of α_{eff} is the weakest link and will be pruned.

Different values of alpha after extracting is:-

```
[0.          0.00247729 0.00262743 0.00262743 0.00264933 0.00266079
 0.00272961 0.00285112 0.00289017 0.00385356 0.00385356 0.00385356
 0.00385356 0.00392363 0.0040328  0.00425863 0.00432681 0.00433526
 0.00433526 0.00440407 0.004693   0.00481696 0.00488456 0.00488824
 0.00513809 0.00520231 0.00520231 0.00525486 0.00536746 0.00544016
 0.00753868 0.00758707 0.00826782 0.00837353 0.00905707 0.0091007
 0.00924739 0.01052065 0.01072934 0.01146676 0.01204746 0.01312246]
```

Thus creating one decision tree for every alpha. The resultant plot for accuracy of training and test data set ranging on different values of alpha is given below:-



Now for each candidate value of alpha, a 10 fold cross validation is run and we have stored mean and standard deviation for each value of alpha..

Ideal alpha from above two observations is row 38 for which our classifier is generalising well:-

Ideal alpha is:- 0.010729

	alpha	mean_accuracy	std
37	0.010521	0.630186	0.044207
38	0.010729	0.638799	0.036993
39	0.011467	0.630104	0.037925

We used this value of alpha in our optimised tree. Figure for our final pruned tree is:-

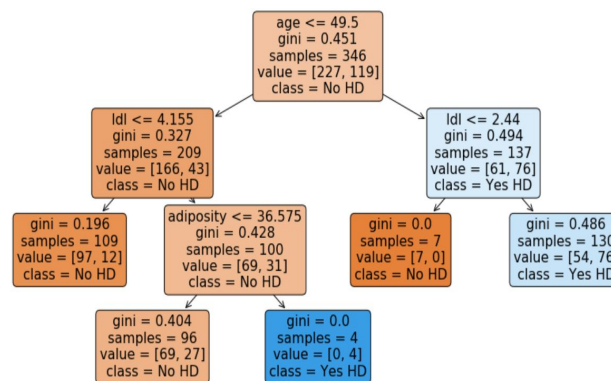


Fig 3.27 Pruned tree after CCP

Confusion Matrix :

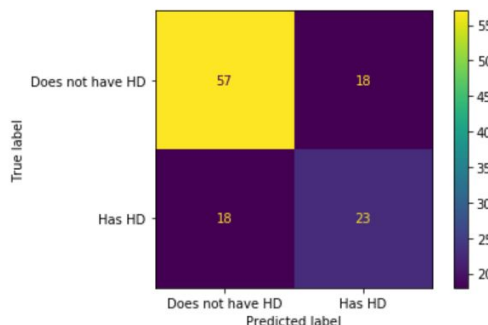


Fig 3.28 Confusion matrix for Pruned tree

Evaluation Metric	Value
Train Accuracy	73.1%
Test Accuracy	72.1%
Sensitivity (Tpr)	0.56
Specificity	0.76
Fpr(1-Specificity)	0.24

Table 3.10 Evaluation metric for pruned tree

We see that the pruned tree is better at classifying patients than the full sized tree. Of the 57+18=75 people that did not have heart disease, 57 (76%) were correctly classified. This is an important improvement over the full sized tree, which only correctly classifies 31(74%) of the patients without heart disease. Of the 18+23 = 41 people with the heart disease , 23(56%) were correctly classified. Again, this is an improvement over the full size tree, which only correctly classified 18(43%) of the patients with the heart disease. We could have achieved more accuracy by tuning the hyperparameters more. But our accuracy was in the range of that of blogs. **Increasing the number of instances under study and balancing the bi-variable responses of the labelled class might help to improve the performance of the ML models.**

given below:-

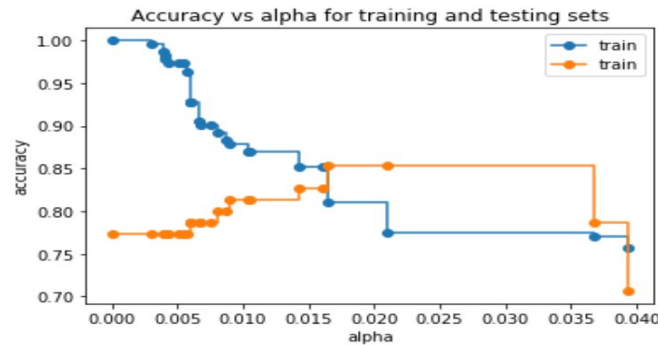


Fig 3.31 Accuracy vs alpha for UCI data

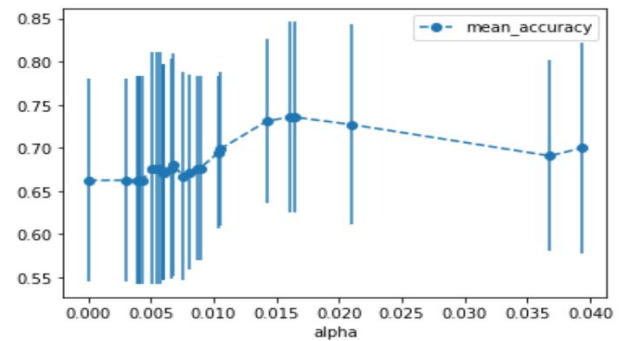


Fig 3.32 Mean and std deviation of all cross validation set

Now for each candidate value of alpha, a 10 fold cross validation is run and we have stored mean and standard deviation for each value of alpha. By using the above two methods we have found our ideal value of **alpha to be:-**

0.014. Now we can use this value of alpha to draw the final classification tree.

	alpha	mean_accuracy	std
20	0.014225	0.747778	0.091395

Pruned Tree:-

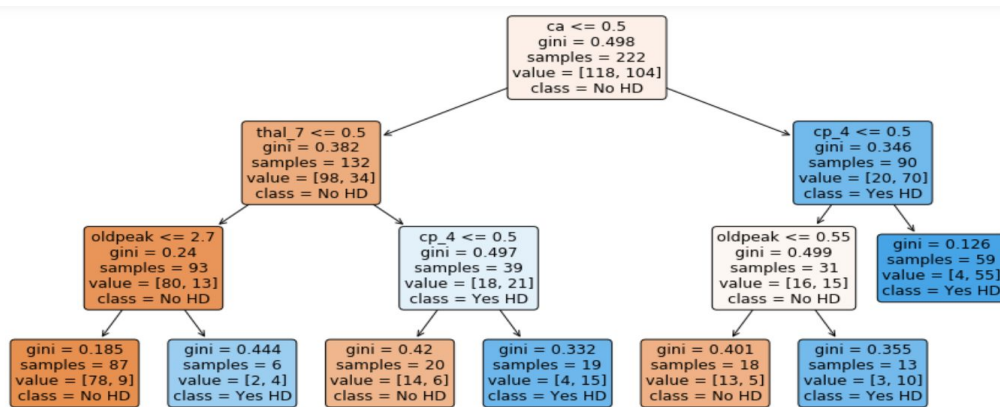


Fig 3.33 Pruned tree for Cleveland data

Here's how to interpret the tree:-

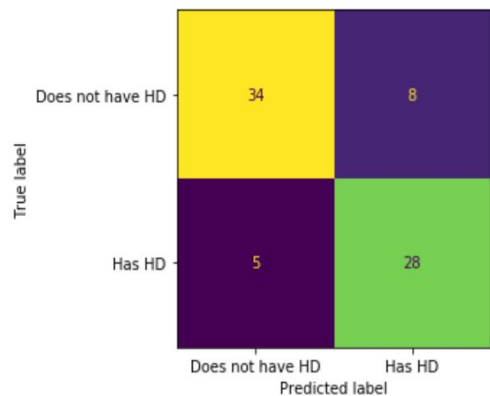
- The variable (column name) and the threshold for splitting the observations. For example, in the tree's root, we use ca to split the observations. All observations with $ca \leq 0.5$ go to the left and all observations with $ca > 0.5$ go to the right
- Gini is the gini index or score for that node
- Samples tell us how many samples are in that node
- Value tells us how many samples in the node are in each category. In this example, we have two categories, No and Yes, referring to whether or not a patient has heart disease. The number of patients with No comes first because the categories are in alphabetical

order. Thus, in the root, 118 patients have No and 104 patients have Yes

- Class tells us whichever category is represented most in the node, In the root, since 118 people have No and only 104 people have Yes, The class is set to No.

The leaves are just like the nodes, except that they do not contain a variable and threshold for splitting the observations. Lastly, the nodes and leaves are colored by the class. In this case No is different shades of orange-ish and Yes is different shades of blue. The darker the shade, the lower the gini score, and that tells us how much the node or leaf is skewed towards one class.

Confusion Matrix:-



Evaluation Metric	Value
Train Accuracy	85.1%
Test Accuracy	82.7%
Sensitivity(Tpr)	0.848
Specificity	0.809
Fpr(1-Specificity)	0.1904

Fig 3.34 Confusion matrix for pruned tree

Table 3.12 Evaluation metric for pruned tree

We see that the pruned tree is better at classifying patients than the full sized tree. Of the $34+8=42$ people that did not have heart disease, 34 (81%) were correctly classified. This is an important improvement over the full sized tree, which only correctly classifies 31(74%) of the patients without heart disease. Of the $5+28 = 33$ people with the heart disease , 28(85%) were correctly classified. Again, this is an improvement over the full size tree, which only correctly classified 26(79%) of the patients with heart disease.

Chapter 4 - Conclusion

In this research, we restricted our study to using Decision Tree and Neural Network algorithms. These algorithms were decided based on literature review and attributes of the techniques. The performance of the obtained models was analysed using evaluation measures of Accuracy, Sensitivity, Specificity, TPs, TNs, FPs and FNs (i.e. Confusion Matrix). Our main evaluation is based on the **Sensitivity(True positive rate)**. As It measures the number of instances correctly predicted as positive by the classifier out of the total number of instances that are actually positive. Sensitivity is also known as Recall or True Positive Rate. As this includes both TP and FN a higher value will result in better prediction of those who are actually suffering from disease and would try to not make any fatal prediction.

SOUTH AFRICAN DATASET :

Although in neural network we used a ROC curve to allow the medical practitioner requirement from the model and provide with desired Tpr and Fpr values from the model using the appropriate probability, still just for observation we have a considered a case with considerably good amount of sensitivity as to reduce chance of any fatal prediction, we can observe that this is much higher than an optimised case of decision tree. As the dataset is highly unbalanced, even the cost complexity pruning couldn't improve the decision tree's performance enough to be comparable to the literature review while the neural network performance is closer to the literature.

Evaluation Metric	Neural Network	Decision Tree
Accuracy	77.14%	72.1%
Sensitivity	0.75	0.56
Specificity	0.76	0.78

Table 3.13 Performance comparison for SA dataset

Cleveland Dataset :

The UCI dataset is balanced hence the neural network and decision tree models performed significantly well as compared to the literature [3]. The softmax function in this balanced dataset is often used to find out the probability of occurrence of each class which performed significantly well.

Evaluation Metric	Neural Network	Decision Tree
Accuracy	83%	82.7%
Sensitivity	0.84	0.84
Specificity	0.82	0.809

Table 3.14 Performance comparison for cleveland dataset

Based on the obtained ML results from the CHD dataset, future research needs to be carried out to improve the performance of the model, especially to increase the Sensitivity and Specificity rates. As the current datasets have a limited number of observations from just one hospital from a certain region, hence to obtain better results a more detailed, researched and widespread dataset is required. Thereafter, the prediction model obtained through the research conducted can be used to develop a mobile application which will help people to track their health and thereby lead to early detection for CHD.

Chapter 5

Future Prospects

Classification of Electrocardiograph (ECG) signals plays an important role in diagnosis of heart disease. An accurate ECG classification is a challenging problem. Surveying the ECG classification into arrhythmia types. Early and accurate arrhythmia types are important in detecting heart diseases and choosing appropriate treatment for a patient. Different classifiers are available for ECG Classification. Amongst all classifiers, artificial neural networks (ANN's) have become very popular and most widely used for ECG classification. These classifications involve feature extraction from the raw graphical data of ECG recorded for hours which might follow a unique pattern for each patient based on the patient's health condition. Any irregularity in the heart rhythm or damage to the heart muscles can change the electrical activity of the heart, so the ECG changes. This might occur due to various conditions such as family history of heart disease, smoking, overweight, diabetes, high cholesterol or high blood pressure. The main advantage of using the deep learning CNN model is that it automatically detects the important features without any human supervision. Being computationally efficient it uses special convolution and pooling operations and performs parameter sharing. This enables CNN models to run on any device making them universally attractive hence the model developed for ECG classification can be deployed to the hospitals machines for easy assistance of practitioners.

References

- [1] Apte, C. S. (2012). Improve study of Heart Disease prediction systems using Data Mining Classification techniques. International journal of computer application, 47(10), 44-48. doi:10.5120/7228-0076
- [2] R. El-Bialy, M. A. Salama, O. H. Karam, and M. E. Khalifa, "Feature Analysis of Coronary Artery Heart Disease Data Sets", Procedia Computer Science, ICCMIT 2015, vol. 65, pp. 459–468, doi: 10.1016/j.procs.2015.09.132
- [3] František Babič, Jaroslav Olejá, Zuzana Vantová, Ján Paralič, “ Predictive and Descriptive Analysis for Heart Disease Diagnosis”, Technical university of Košice, Slovakia, <https://annals-csis.org/proceedings/2017/drp/pdf/219.pdf>
- [4] Amanda H. Gonsalves , Fadi Thabtah, Rami Mustafa A. Mohammad, Gurpreet Singh, "Prediction of Coronary Heart Disease using Machine Learning: An Experimental Analysis", the 2019 3rd International Conference, July 2019, doi: [10.1145/3342999.3343015](https://doi.org/10.1145/3342999.3343015)
- [5] <https://towardsdatascience.com/heart-disease-prediction-73468d630cfc>
- [6] <https://towardsdatascience.com/heart-disease-uci-diagnosis-prediction-b1943ee835a7#5a5ada>
- [7] <http://www.adeveloperdiary.com/data-science/deep-learning/neural-network-with-softmax/>

- [x-in-python/](#)
- [8] Renato Cuocolo, Maria Brunella Cipullo, Arnaldo Stanzione*, Lorenzo Ugga, Valeria Romeo, Leonardo Radice, Arturo Brunetti and Massimo Imbriaco, "Machine learning applications in prostate cancer magnetic resonance imaging", Cuocolo et al. European Radiology Experimental (2019) 3:35, <https://doi.org/10.1186/s41747-019-0109-2>
 - [9] <https://towardsdatascience.com/https-medium-com-piotr-skalski92-deep-dive-into-deep-n-etworks-math-17660bc376ba>
 - [10] <https://stats.stackexchange.com/questions/154879/a-list-of-cost-functions-used-in-neural-networks-alongside-applications>
 - [11] <https://www.codegrepper.com/code-examples/python/plot+roc+curve+with+threshold+scores+python>
 - [12] <http://www.webgraphviz.com/>
 - [13] <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7148346>
 - [14] https://www.researchgate.net/publication/315023624_Early_Prediction_of_Heart_Disease_Using_Decision_Tree_Algorithm#:~:text=The%20heart%20disease%20accounts%20to%20be%20the%20leading%20cause%20of%20death%20worldwide.&text=Data%20mining%20algorithms%20such%20as,predict%20patterns%20in%20the%20dataset.
 - [15] <https://stats.stackexchange.com/questions/105501/understanding-roc-curve/105577#105577>
 - [16] <https://www.fharrell.com/post/mlconfusion/>
 - [17] <https://towardsdatascience.com/decision-trees-d07e0f420175>
 - [18] <https://www.geeksforgeeks.org/decision-tree/>
 - [19] <https://towardsai.net/p/programming/decision-trees-explained-with-a-practical-example-fe47872d3b53>
 - [20] <https://medium.com/analytics-steps/understanding-the-gini-index-and-information-gain-in-decision-trees-ab4720518ba8>