MACHINE LEARNING MODEL FOR ARTIFICIAL NEURAL NETWORK ON HEART DISEASE DATA

1.INTRODUCTION

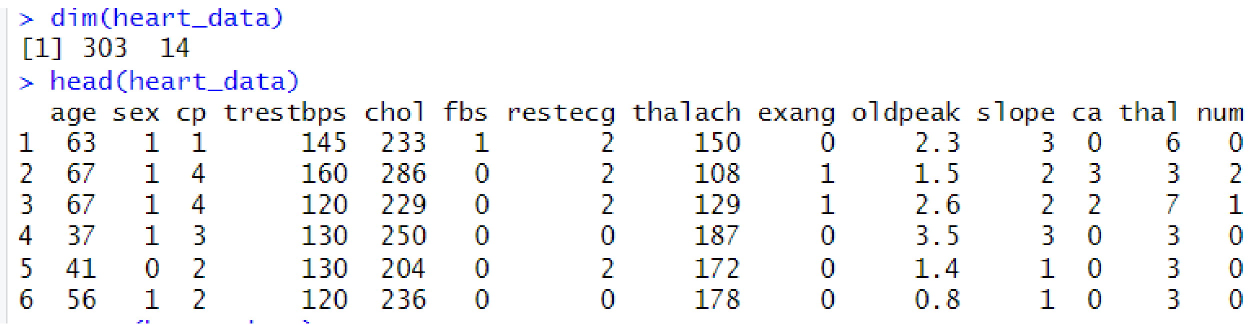
This project looked at the use of Artificial Neural Network (ANN) algorithms to make predictions on heart disease data. The Heart data was taken from the UCI Machine Learning Repository, it has patient attributes with the target variable been “num,” showing the rate of coronary artery narrowing. The main aim of this model building is to make predictions on heart diseases. The steps involved are data preparation, exploratory data analysis, data partitioning, and model building. We also emphasized evaluation metrics such as Miss classification rate, Mean Squared Error (MSE) and confusion matrix. Furthermore, ROC curves were also analyzed toascertain model performance.

2.DATA COLLECTION

The data set was read into R first to enable us to develop a machine learning model for Artificial Neural Network on Heart disease data.

# Read the data

```
heart_data <- read.csv(file = "https://archive.ics.uci.edu/ml/machine-learning-databases/heart-disease/processed.cleveland.data",header = FALSE, na.strings = c("NA", "", " ", "?"),   col.names = c("age", "sex", "cp", "trestbps", "chol", "fbs",    "restecg", "thalach", "exang", "oldpeak", "slope", "ca", "thal", "num"))
```



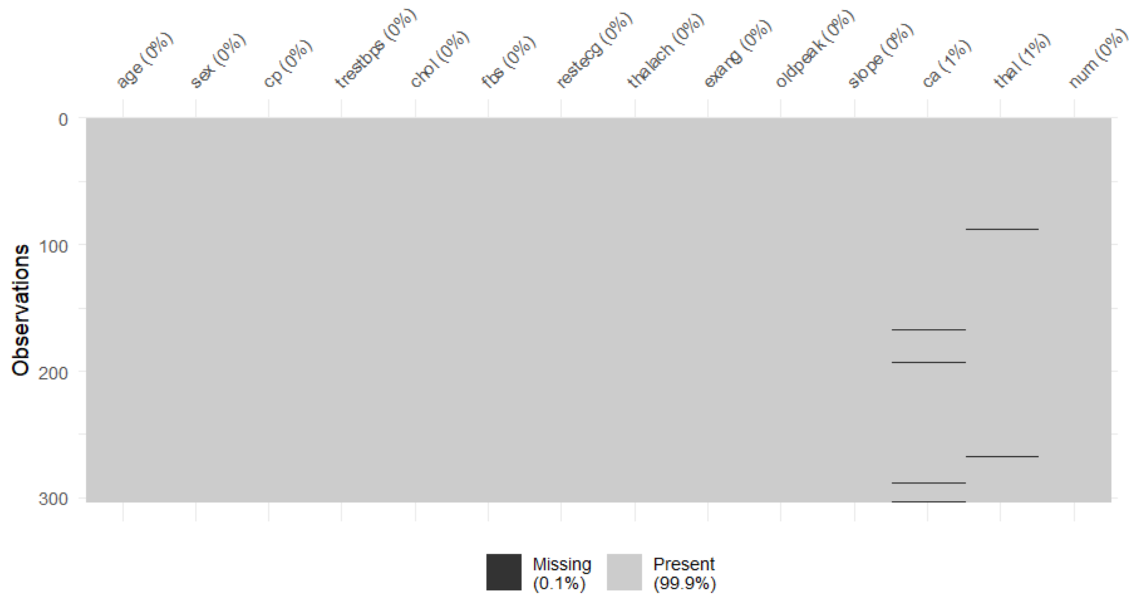
```
> dim(heart_data)
[1] 303  14
> head(heart_data)
  age sex cp trestbps chol fbs restecg thalach exang oldpeak slope ca thal num
1  63   1  1      145  233   1       2     150     0     2.3     3  0    6   0
2  67   1  4      160  286   0       2     108     1     1.5     2  3    3   2
3  67   1  4      120  229   0       2     129     1     2.6     2  2    7   1
4  37   1  3      130  250   0       0     187     0     3.5     3  0    3   0
5  41   0  2      130  204   0       2     172     0     1.4     1  0    3   0
6  56   1  2      120  236   0       0     178     0     0.8     1  0    3   0
```

The dimension shows we have 304 data size (in Rows) and 14 variables (in Columns). That means 1 target variable and 14 predictors variables. The first 6 rows of the data set was displayed above.

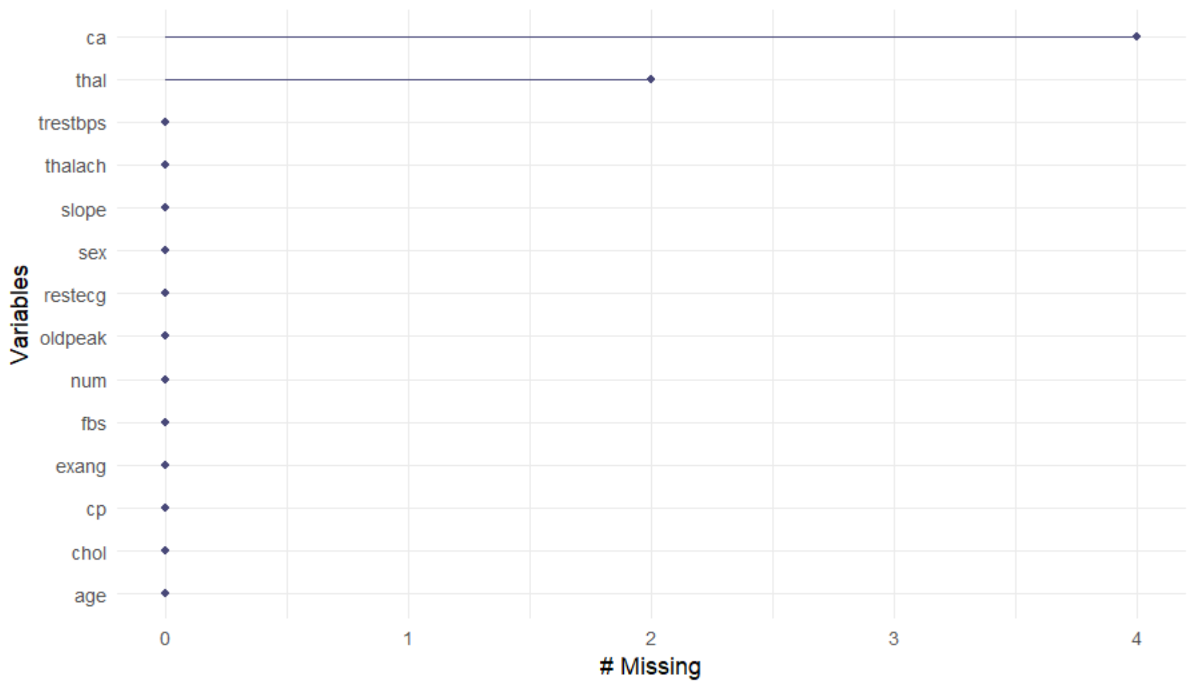
3. DATA PREPARATION

Inspecting the variable types, missing values and possibly wrong records, and other issues.

```
library(naniar)
vis_miss(data)
```



```
gg_miss_var(heart_data)
```



```
heart_data = heart_data
vnames <- colnames(heart_data)
n <- nrow(heart_data)
out <- NULL
for (j in 1:ncol(heart_data)){
 vname <- colnames(heart_data)[j]
 x <- as.vector(heart_data[,j])
 n1 <- sum(is.na(x), na.rm=TRUE)  # NA
 n2 <- sum(x=="NA", na.rm=TRUE) # "NA"
 n3 <- sum(x==" ", na.rm=TRUE)  # missing
 nmiss <- n1 + n2 + n3
 nmiss <- sum(is.na(x))
 ncomplete <- n-nmiss
 out <- rbind(out, c(col.num=j, v.name=vname, mode=mode(x), n.level=length(unique(x)),
             ncom=ncomplete, nmiss= nmiss, miss.prop=nmiss/n))
}
out <- as.data.frame(out)
```

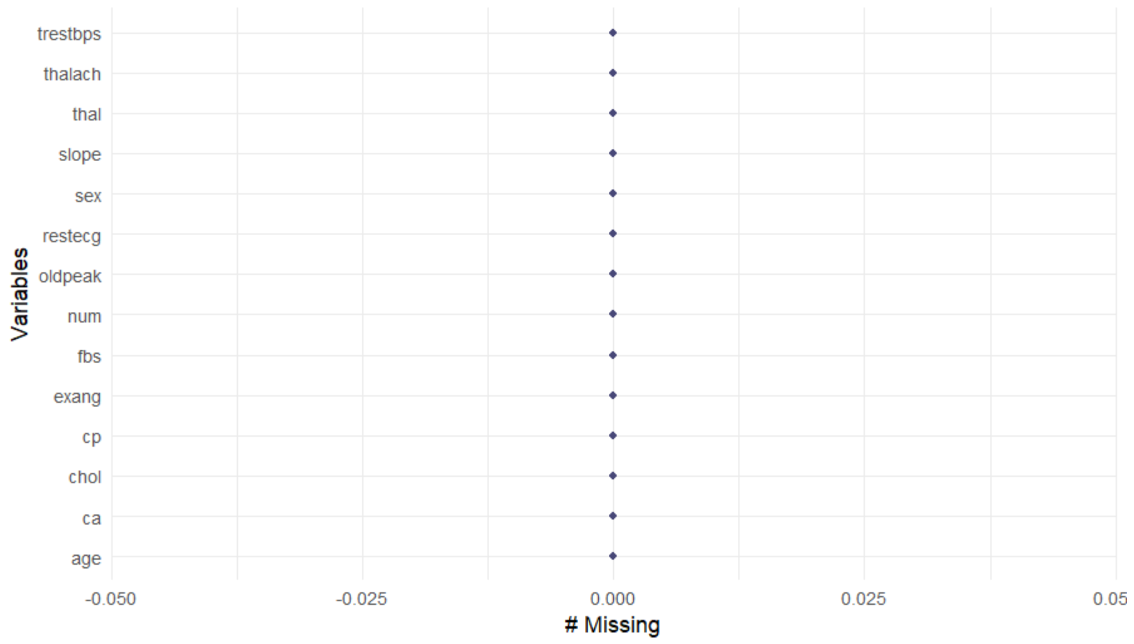
```
row.names(out) <- NULL

out

for (j in 1:NCOL(heart_data)){

  print(colnames(heart_data)[j])

  print(table(heart_data[,j], useNA="ifany"))}
```

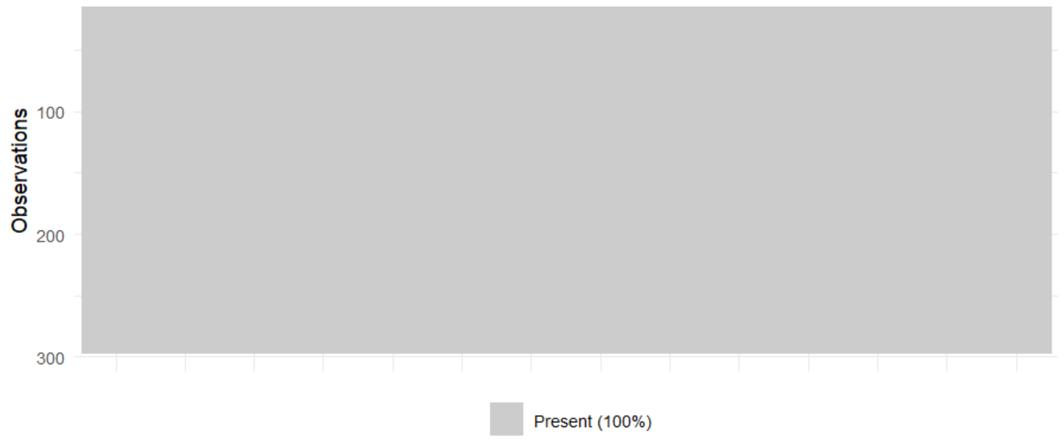
```
   col.num   v.name      mode n.level ncom nmiss miss.prop
1        1      age   numeric      41  297     0         0
2        2      sex   numeric       2  297     0         0
3        3       cp   numeric       4  297     0         0
4        4 trestbps   numeric      50  297     0         0
5        5     chol   numeric     152  297     0         0
6        6      fbs   numeric       2  297     0         0
7        7  restecg   numeric       3  297     0         0
8        8  thalach   numeric      91  297     0         0
9        9    exang   numeric       2  297     0         0
10      10  oldpeak   numeric      40  297     0         0
11      11    slope   numeric       3  297     0         0
12      12       ca   numeric       4  297     0         0
13      13     thal   numeric       3  297     0         0
14      14      num character       2  297     0         0
```

```
> for (j in 1:NCOL(heart_data)){
+   print(colnames(heart_data)[j])
+   print(table(heart_data[,j], useNA="ifany"))
+ }

[1] "age"

29 34 35 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60
61 62 63 64
 1  2  4  2  2  4  3 10  8  8 11  8  7  5  7  5  7 12 13  8 16  8 11 17 19 14 12
8 11  9 10
65 66 67 68 69 70 71 74 76 77
 8  7  9  4  3  4  3  1  1  1
[1] "sex"

  0   1
 97 206
[1] "cp"

  1   2   3   4
 23  50  86 144
[1] "trestbps"

 94 100 101 102 104 105 106 108 110 112 114 115 117 118 120 122 123 124 125 126
128 129 130
  2   4   1   2   1   3   1   6  19   9   1   3   1   7  37   4   1   6  11   3
12   1  36
132 134 135 136 138 140 142 144 145 146 148 150 152 154 155 156 158 160 164 165
170 172 174
  8   5   6   3  12  32   3   2   5   2   2  17   5   1   1   1   1  11   1   1
4   1   1
178 180 192 200
  2   3   1   1
[1] "chol"
```

```
126 131 141 149 157 160 164 166 167 168 169 172 174 175 176 177 178 180 182 183
184 185 186
  1   1   1   2   1   1   1   1   1   1   1   1   1   2   1   4   1   1   1   1
1   1   1
187 188 192 193 195 196 197 198 199 200 201 203 204 205 206 207 208 209 210 211
212 213 214
  1   2   2   2   1   2   6   2   3   1   3   3   6   2   2   2   2   2   1   4
5   2   2
215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234
235 236 237
  1   2   1   2   3   3   2   2   3   1   2   4   2   2   3   3   3   2   4   6
2   3   1
239 240 241 242 243 244 245 246 247 248 249 250 252 253 254 255 256 257 258 259
260 261 262
  4   4   1   1   4   3   3   3   2   2   3   3   1   2   5   2   3   1   3   1
2   2   1
263 264 265 266 267 268 269 270 271 273 274 275 276 277 278 281 282 283 284 286
288 289 290
  3   2   2   2   2   2   5   2   2   2   3   2   1   2   1   1   4   3   1   2
3   2   1
293 294 295 298 299 300 302 303 304 305 306 307 308 309 311 313 315 318 319 321
322 325 326
  1   2   2   2   2   1   2   3   2   2   1   1   2   3   1   1   2   2   1   1
1   2   1
327 330 335 340 341 342 353 354 360 394 407 409 417 564
  1   2   2   1   1   1   1   1   1   1   1   1   1   1
[1] "fbs"

  0   1
258  45
[1] "restecg"

  0   1   2
151   4 148
[1] "thalach"

 71  88  90  95  96  97  99 103 105 106 108 109 111 112 113 114 115 116 117 118
120 121 122
  1   1   1   1   2   1   1   2   3   1   2   2   3   2   1   3   3   2   1   1
3   1   4
123 124 125 126 127 128 129 130 131 132 133 134 136 137 138 139 140 141 142 143
144 145 146
  2   1   7   4   1   1   1   4   4   7   2   1   2   1   3   2   6   3   6   7
7   4   4
147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166
167 168 169
  5   3   2   7   4   8   3   5   4   6   5   6   4   9   6  11   9   2   5   3
1   5   6
170 171 172 173 174 175 177 178 179 180 181 182 184 185 186 187 188 190 192 194
195 202
  5   4   7   7   5   3   1   5   5   2   2   5   1   1   2   1   1   1   1   1
1   1
[1] "exang"

  0   1
204  99
[1] "oldpeak"

  0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9   1 1.1 1.2 1.3 1.4 1.5 1.6 1.8 1.9   2
2.1 2.2 2.3
 99   7  12   3   9   5  14   1  13   3  14   2  17   1  13   5  11  10   5   9
1   4   2
2.4 2.5 2.6 2.8 2.9   3 3.1 3.2 3.4 3.5 3.6 3.8   4 4.2 4.4 5.6 6.2
  3   2   6   6   1   5   1   2   3   1   4   1   3   2   1   1   1
[1] "slope"

  1   2   3
142 140  21
```

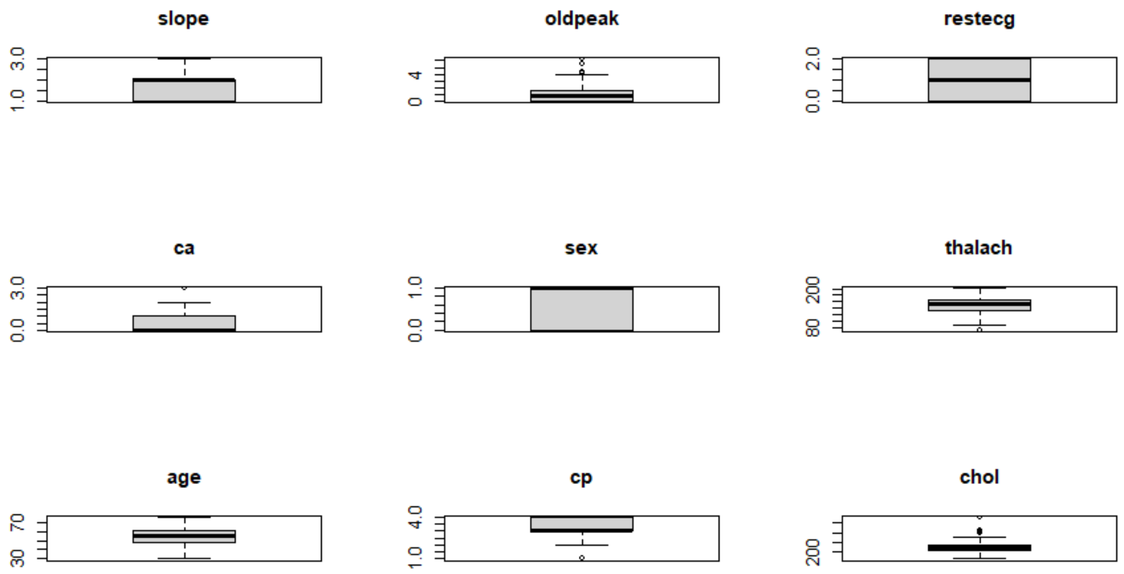
```
[1] "ca"

   0    1    2    3 <NA> 
 176   65   38   20    4 
[1] "thal"

   3    6    7 <NA>
```





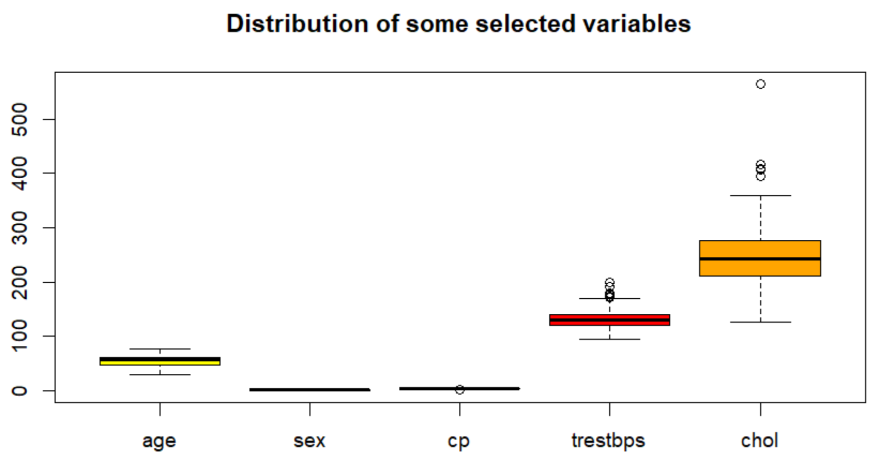
```
Plot box plots for the selected features
> for (col in selected_features) {
+   boxplot(cleaned_data[[col]], main = col)
+ }
```

<u>BOXPLOTS FOR SOME SELECTED FEATURES</u>



There are a few outliers in some of the selected variables. The above box plot shows proof of some selected variables.

```
boxplot(cleaned_data[,1:5],col=c("yellow","blue","green","red","orange"),main="D
istribution of some selected variables")
```



Few of the predictor variables contain an outlier. An evident of that is the above box plot of some selected variables.

4. EXPLORATORY DATA ANALYSIS

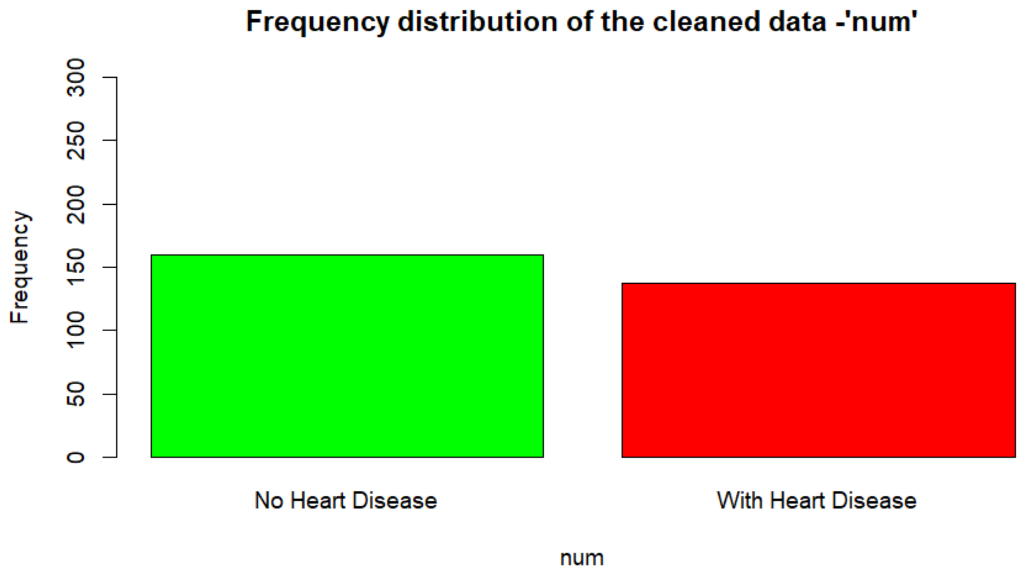
In other to explore the frequency distribution of the output variable in the data set. We obtain the Bar plot of the target variable outcome. The association of the Target variable on some selected predicted variables will also be observed.

```
class_table = table(cleaned_data$num)
> class_table

No Heart Disease(NHD)      Heart Disease (HD)

  0                            1
 160                           137
```

```
> class_percentage <- prop.table(class_table) * 100
> print(class_percentage) ### cleaned_data is Imbalance
```

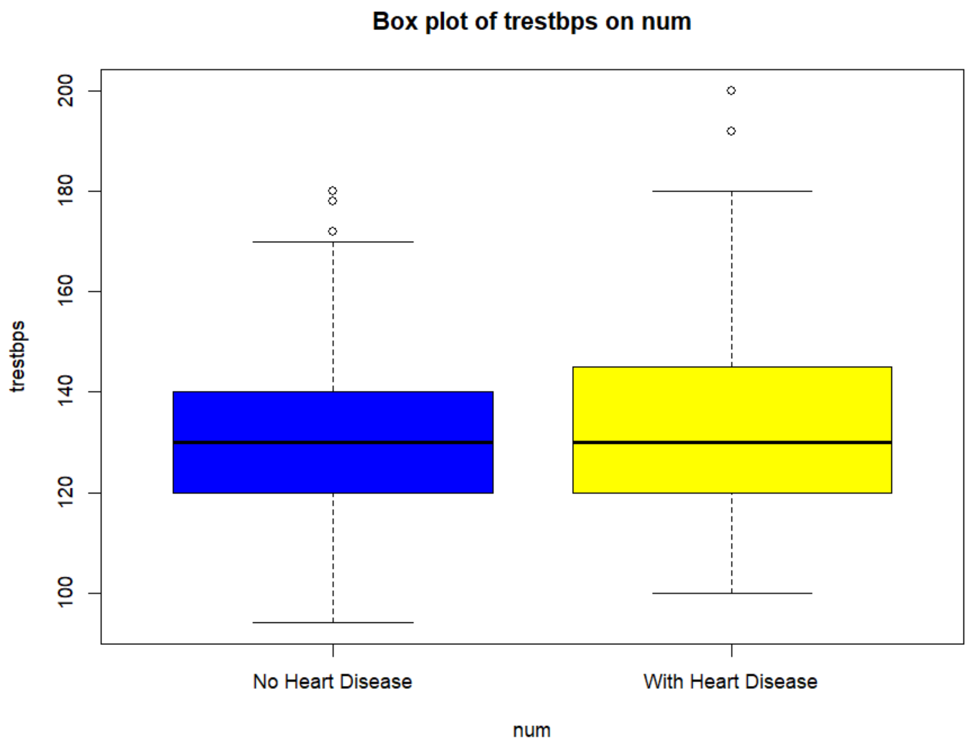
```
    NHD          HD
       0        1
 53.87205    46.12795
```

```
cleaned_data_num= cleaned_data$num
> cleaned_data_num<- ifelse(cleaned_data_num==1, "With Heart Disease", "No Heart
Disease")
> tab=table(cleaned_data_num,useNA="no")
> barplot(tab,names.arg=row.names(tab),col=c("green","red"),ylab="Frequency",
+         xlab="num",main="Frequency distribution of the cleaned data -'num'",yl
im=c(0,300))
```



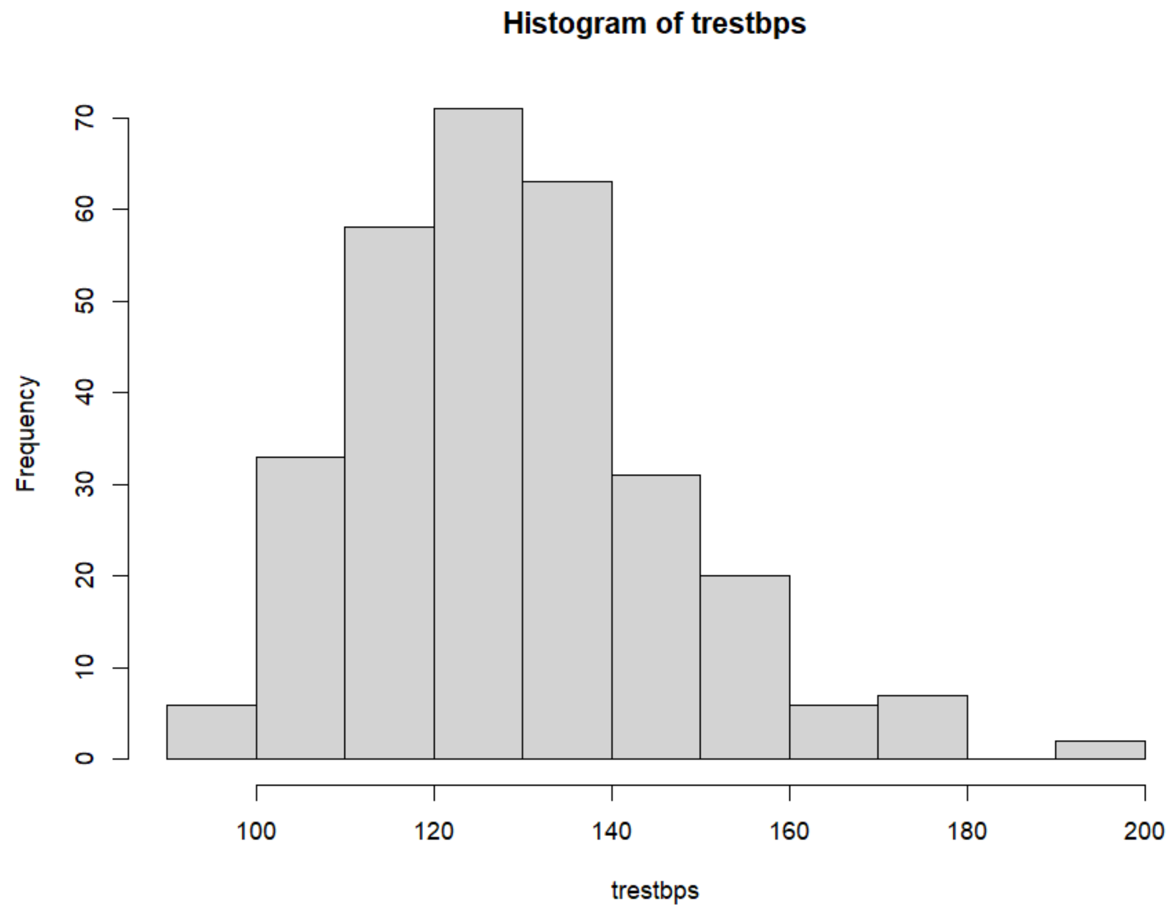
The plot above shows that the frequency distribution of the target variable "num". We have 160 frequencies of class with "No Heart Disease" and 137 frequencies of class "With Heart Disease".

```
boxplot<-boxplot(cleaned_data$trestbps~cleaned_data_num,xlab="num", ylab="trestb
ps"
+                  ,main="Box plot of trestbps on num",col=c("blue","yellow"))
```



The Boxplot above shows the distribution of the predictor variable trestbps on num, which indicate the presence of association between predictor variable trestbps on target variable "num".

```
hist(cleaned_data$trestbps,main="Histogram of trestbps",xlab="trestbps")
```



The plot shows the distribution of the trestbps is symmetrical. Further evidence will be presented using the two-normality test below.

```
ad.test(cleaned_data$trestbps)

        Anderson-Darling normality test

data:  cleaned_data$trestbps
A = 2.5048, p-value = 2.435e-06
```

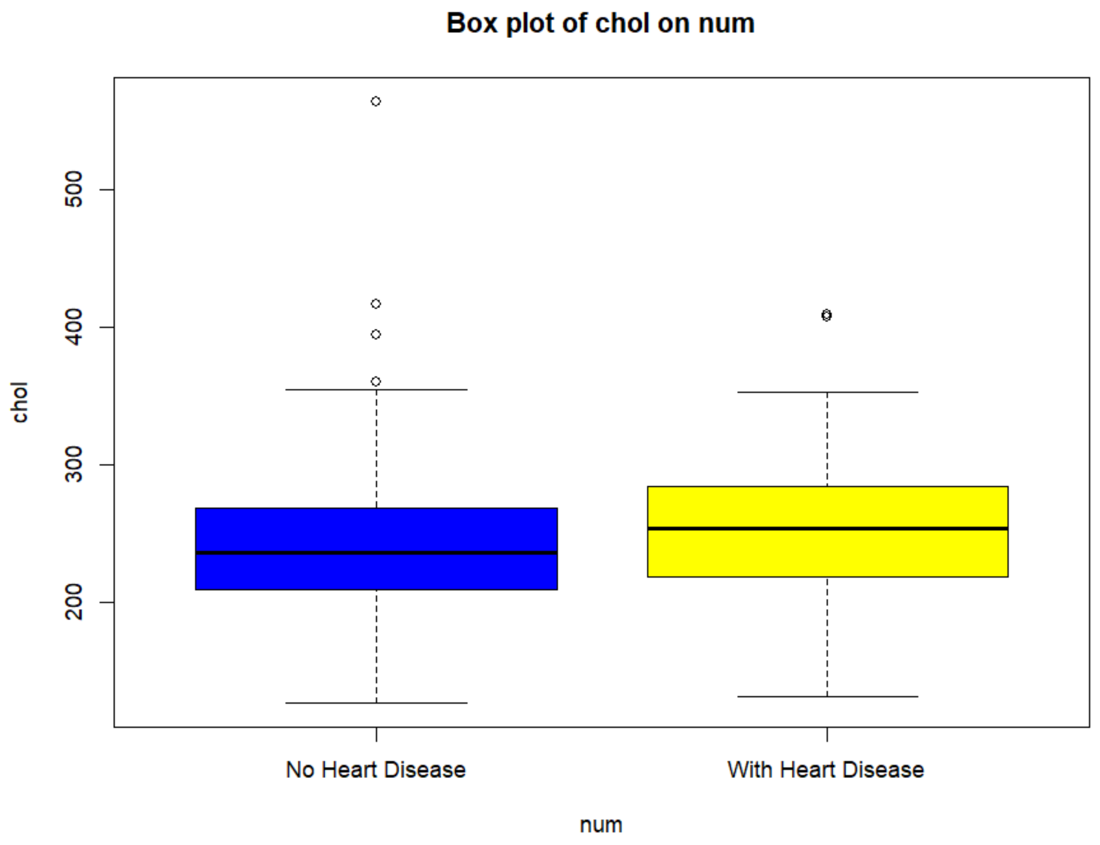
```
shapiro.test(cleaned_data$trestbps)

        Shapiro-Wilk normality test

data:  cleaned_data$trestbps
W = 0.96676, p-value = 2.416e-06
```

The p-value of both tests are all less than alpha level of significance 0.05, which indicate that the distribution is not normal. Therefore, the assumption of t-test is violated, then we use the non-parametric alternative approach called wilcoxon rank sum test to examine the association between trestbps on num.

```
wilcox.test(cleaned_data$trestbps~cleaned_data_num, cleaned_data=cleaned_data, a
lternative = "two.sided"
+              ,na.action = na.omit)

        Wilcoxon rank sum test with continuity correction

data:  cleaned_data$trestbps by cleaned_data_num
W = 9292.5, p-value = 0.02346
alternative hypothesis: true location shift is not equal to 0
```

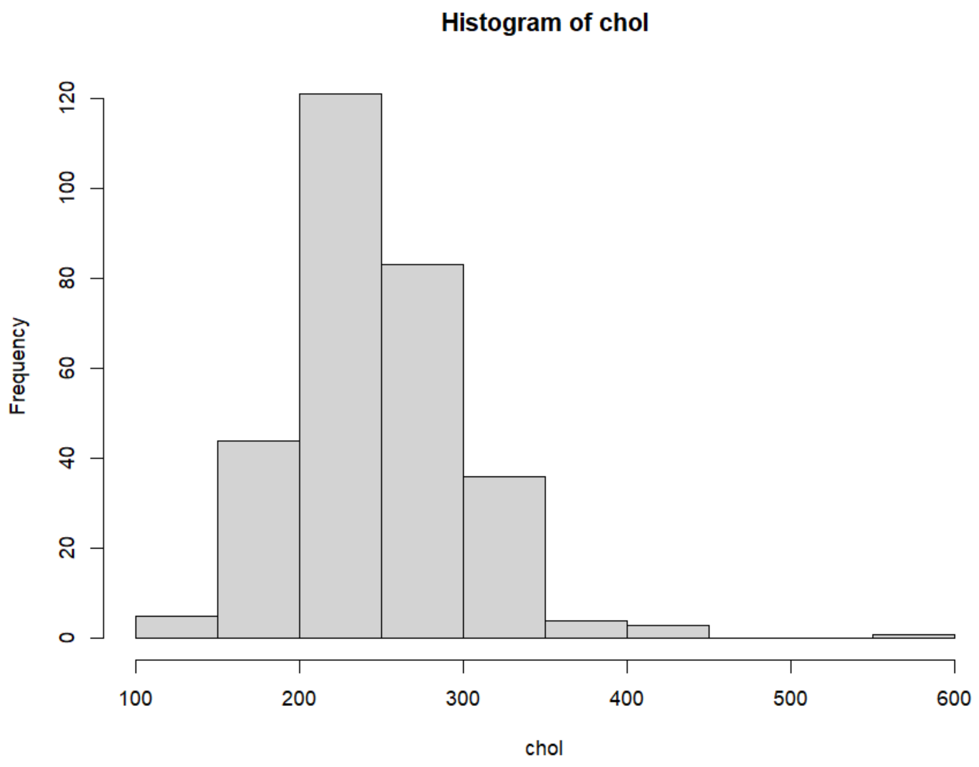
The p-value of the test above which is less than alpha level of significance 0.05 indicate there exist a significance relationship between predictors variable trestbps on target variable “num”.

```
boxplot<-boxplot(cleaned_data$chol~cleaned_data_num,xlab="num", ylab="chol"
+                  ,main="Box plot of chol on num",col=c("blue","yellow"))
```



The Boxplot above shows the distribution of the predictor variable Chol on num, which indicate the presence of association between predictor variable Chol on target variable “num”.

```
hist(cleaned_data$chol,main="Histogram of chol",xlab="chol")
```



The plot shows the distribution of the Chol is symmetrical. Further evidence will be presented using the two-normality test below.

```
Anderson-Darling normality test

data:  cleaned_data$chol
A = 1.5818, p-value = 0.0004469


shapiro.test(cleaned_data$chol)

        Shapiro-Wilk normality test

data:  cleaned_data$chol
W = 0.94829, p-value = 1.019e-08
```

The p-value of the test above which is less than alpha level of significance 0.05
indicate there exist a significance relationship between predictors variable
Chol on target variable “num”.

```
wilcox.test(cleaned_data$chol~cleaned_data_num, cleaned_data=cleaned_data, alter
native = "two.sided"
+              ,na.action = na.omit)

        Wilcoxon rank sum test with continuity correction

data:  cleaned_data$chol by cleaned_data_num
W = 9492, p-value = 0.04669
alternative hypothesis: true location shift is not equal to 0
```

The p-value of the test above which is less than alpha level of significance 0.05
indicate there exist a significance relationship between predictors variable
Chol on target variable “num”.

5.DATA PARTITIONING

Partitioning our data set into train and test with a ratio of approximately 2:1
in sample size. The train data takes 67% of the whole data set and the test takes
33% of the whole data set.

```
split_data <- sample(x=1:2, size = n, replace=TRUE, prob=c(0.67, 0.33))
> D1<- dat[split_data == 1, ]
> D2 <- dat[split_data == 2, ]
> yobs <- D2$num
> D2<- D2[ , -14]
```

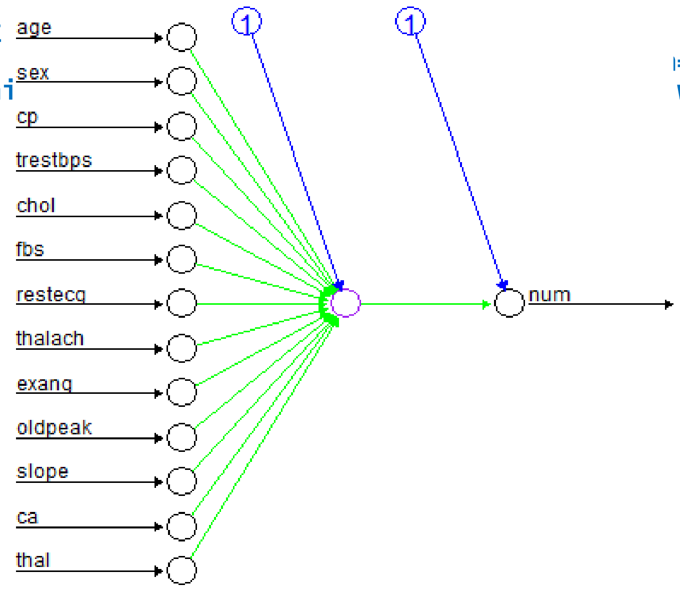
We have the training data set with “D1” as training input and “y.train” as training
output. Finally, we have the testing data set with “D2” as testing input and “yobs”
as testing output.

6. MODEL BUILDING

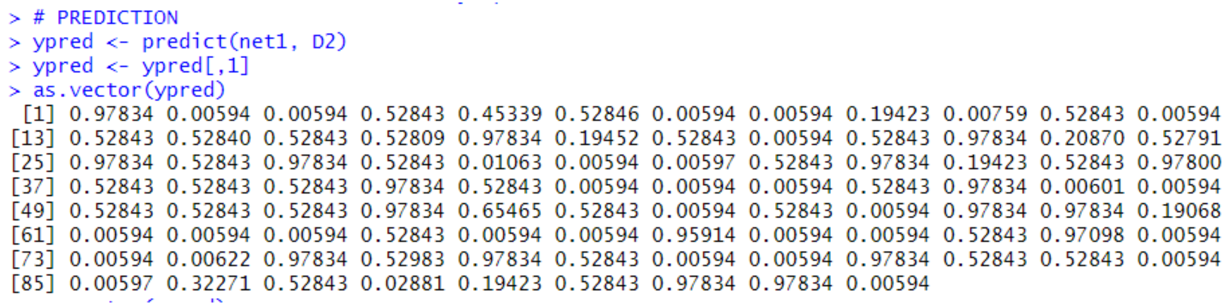
We are going to fit at least three different artificial neural network (ANN) models, with
different numbers of layers and different number of units.

```
library(neuralnet)
#https://www.rdocumentation.org/packages/neuralnet/versions/1.44.2/topics/neural
net
options(digits=3)
net1 <- neuralnet(num ~ .,
                  data=D1,
                  hidden=3, #1 hidden layer, 4 neurons
                  act.fct='logistic', err.fct="sse", linear.output=F,
likelihood=TRUE)
```

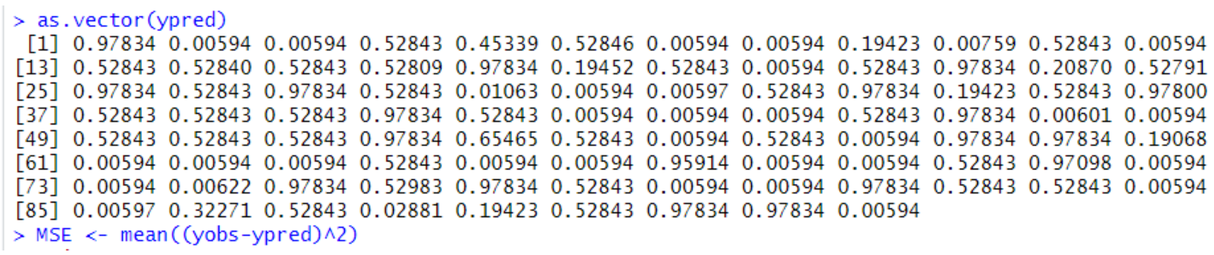
```
# PLOT THE
plot(net1,                                                         =6.5, radius=.15,
      col.hi                                                       wd=1, fontsize=9)
```



The weights of the built ANN model across neurons and hidden layers are shown in the plot above. It shows how the model makes predictions using different neurons and hidden layers. The weights distribution helps to get a clear picture of neural network and identify which neurons and hidden layers that helps the model’s decision-making process.



```
> # PREDICTION
> ypred <- predict(net1, D2)
> ypred <- ypred[,1]
> as.vector(ypred)
 [1] 0.97834 0.00594 0.00594 0.52843 0.45339 0.52846 0.00594 0.00594 0.19423 0.00759 0.52843 0.00594
[13] 0.52843 0.52840 0.52843 0.52809 0.97834 0.19452 0.52843 0.00594 0.52843 0.97834 0.20870 0.52791
[25] 0.97834 0.52843 0.97834 0.52843 0.01063 0.00594 0.00597 0.52843 0.97834 0.19423 0.52843 0.97800
[37] 0.52843 0.52843 0.52843 0.97834 0.52843 0.00594 0.00594 0.00594 0.52843 0.97834 0.00601 0.00594
[49] 0.52843 0.52843 0.52843 0.97834 0.65465 0.52843 0.00594 0.52843 0.00594 0.97834 0.97834 0.19068
[61] 0.00594 0.00594 0.00594 0.52843 0.00594 0.00594 0.95914 0.00594 0.00594 0.52843 0.97098 0.00594
[73] 0.00594 0.00622 0.97834 0.52983 0.97834 0.52843 0.00594 0.00594 0.97834 0.52843 0.52843 0.00594
[85] 0.00597 0.32271 0.52843 0.02881 0.19423 0.52843 0.97834 0.97834 0.00594
```



```
> as.vector(ypred)
 [1] 0.97834 0.00594 0.00594 0.52843 0.45339 0.52846 0.00594 0.00594 0.19423 0.00759 0.52843 0.00594
[13] 0.52843 0.52840 0.52843 0.52809 0.97834 0.19452 0.52843 0.00594 0.52843 0.97834 0.20870 0.52791
[25] 0.97834 0.52843 0.97834 0.52843 0.01063 0.00594 0.00597 0.52843 0.97834 0.19423 0.52843 0.97800
[37] 0.52843 0.52843 0.52843 0.97834 0.52843 0.00594 0.00594 0.00594 0.52843 0.97834 0.00601 0.00594
[49] 0.52843 0.52843 0.52843 0.97834 0.65465 0.52843 0.00594 0.52843 0.00594 0.97834 0.97834 0.19068
[61] 0.00594 0.00594 0.00594 0.52843 0.00594 0.00594 0.95914 0.00594 0.00594 0.52843 0.97098 0.00594
[73] 0.00594 0.00622 0.97834 0.52983 0.97834 0.52843 0.00594 0.00594 0.97834 0.52843 0.52843 0.00594
[85] 0.00597 0.32271 0.52843 0.02881 0.19423 0.52843 0.97834 0.97834 0.00594
```

```
> MSE <- mean((yobs-ypred)^2)
```

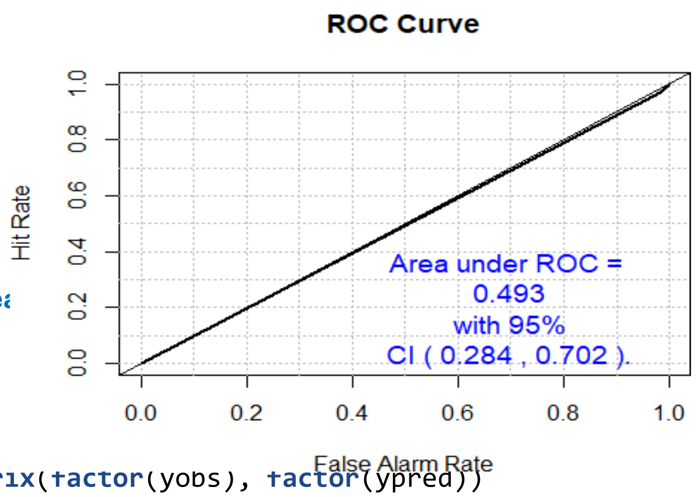
```
MSE
```

```
[1] 0.256
```

An MSE of 0.256 (26%) shows the performance of the model in predicting heart disease in data used. This percentage (26%) is very low and not acceptable. Hence, a better model or more data exploration needs to be done to build a model that can help reduce misclassification.

We will use various evaluation methodologies to assess models and the performance that would be obtained. The use of different evaluation methods would help us to determine the strength and weaknesses of the models.

```
roc.plot(mod.nn, plot.thres=NULL)
text(x=0.7, y=0.2, paste("Area under ROC = \n", round(rf_AUC$cvAUC, digits = 3),
"\n with 95% \nCI (",
rf_auc.ci[1], ",", rf_auc.ci[2], ").", sep = " "),
col="blue", cex =1.2)
```



```
miss.rate <- mea

[1] 0.64
```

```
confusionMatrix(factor(yobs), factor(ypred))

## Confusion Matrix and Statistics##
##             Reference

Prediction  0  1
```

```
        0  1 56
        1  1 31
```

```
Accuracy :                  0.36
95% CI :                    (0.261, 0.468)
No Information Rate :       0.978
P-Value [Acc > NIR] :       1

Kappa :                     -0.01

Mcnemar's Test P-Value :    8.52e-13

Sensitivity :               0.5000
Specificity :               0.3563
Pos Pred Value :            0.0175
Neg Pred Value :            0.9687
Prevalence :                0.0225
Detection Rate :            0.0112
Detection Prevalence :      0.6404
Balanced Accuracy :         0.4282

'Positive' Class :          0
```

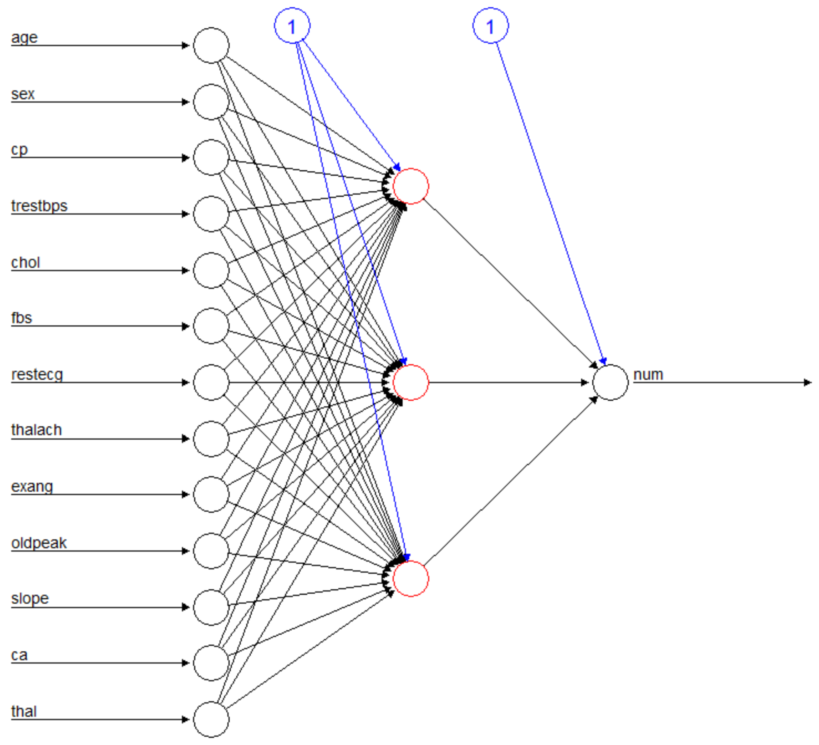
Area Under the Curve of 0.493 shows suggested ANN model is 49% accurate to predict if a person is having a heart disease or not. We had a confidence interval between 0.284 to 0.702.

The ROC curve shows the sensitivity (True Positive Rate) and False Positive Rate (1 – Specificity). The model had a misclassification rate of 64%, and a benchmark of 50%.

From the confusion matrix, the sensitivity (True Positive Rate) or “Recall” of 0.50 tells us that the model is average in terms of the percentage in predicting whether a patient is having heart disease or not.

```
 library(neuralnet)
#https://www.rdocumentation.org/packages/neuralnet/versions/1.44.2/topics/neuralnet
options(digits=3)
net1 <- neuralnet(num ~ .,
          data=D1,
          hidden=3, #1 hidden layer, 4 neurons
```

```
        act.fct='logistic', err.fct="sse", linear.output=F, likelihood=TRUE)

# PLOT THE MODEL
plot(net1, rep="best", show.weights=FALSE, dimension=6.5, radius=.15,
  col.hidden="red", col.hidden.synapse="black", lwd=1, fontsize=9)
```



The weights of the constructed ANN model spanning neurons and hidden layers are displayed in the plot above. It also explains how various neurons and hidden layers are used to make the predictions. The weights distribution identifies neurons and neural networks, which helps the model's decision-making process become more efficient.

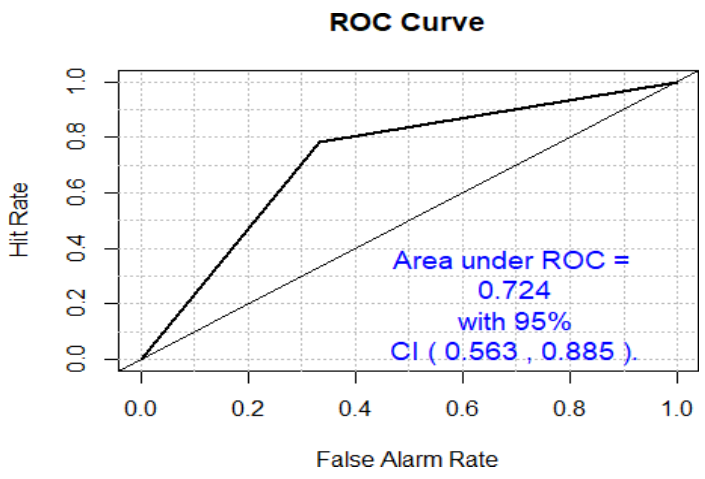
```
ypred1 <- predict(net2, D2)
ypred1 <- ypred1[,1]
as.vector(ypred1)
```

```
##  [1] 0.784 0.124 0.784 0.784 0.124 0.124 0.124 0.734 0.778 0.124 0.784
0.124
## [13] 0.784 0.784 0.124 0.124 0.124 0.124 0.124 0.783 0.124 0.784 0.784
0.784
```

```
## [25] 0.784 0.784 0.784 0.784 0.124 0.124 0.124 0.124 0.784 0.784 0.124
0.204
## [37] 0.784 0.124 0.784 0.784 0.124 0.784 0.784 0.124 0.124 0.784 0.761
0.784
## [49] 0.784 0.124 0.125 0.784 0.784 0.784 0.784 0.163 0.124 0.124 0.784
0.784
## [61] 0.124 0.784 0.784 0.124 0.784 0.784 0.784 0.124 0.124 0.124 0.784
0.124
## [73] 0.124 0.784 0.784 0.124 0.124 0.124 0.720 0.124 0.124 0.124 0.124
0.124
## [85] 0.784 0.124 0.124 0.784 0.124
```

```
MSE.c <- mean((yobs-ypred1)^2)
MSE.c
```

```
[1] 0.208
```



The MSE has improved to 21%, as can be seen above. This demonstrates the model's predictive ability for heart diseases. 21% MSE, however, is insufficient, and this suggests that the model needs to be enhanced.

```
rf_AUC <- ci.cvAUC(predictions = ypred1, labels =yobs, folds=1:NROW(D2),
confidence = 0.95)

rf_AUC
```

##$cvAUC # [1] 0.724
$se
## [1] 0.0823 ##

```
## $ci
## [1] 0.563 0.885 ##
## $confidence # [1] 0.95
```

```
roc.plot(mod.nn, plot.thres=NULL)
text(x=0.7, y=0.2, paste("Area under ROC = \n", round(rf_AUC$cvAUC, digits = 3), "\n with 95% \nCI
(",
rf_auc.ci[1], ",", rf_auc.ci[2], ").", sep = " "),
col="blue", cex =1.2)
```

```
(miss.rate <- mean(yobs != ypred1))
```

```
## [1] 0.292
```

```
confusionMatrix(factor(yobs), factor(ypred1))## Confusion Matrix and Statistics
## Prediction 0 1
            0 38 19
```

```
1           7  25

Accuracy :                     0.708
95% CI :                       (0.602,
No Information Rate :          0.506
P-Value [Acc > NIR] :          8.4e-05

Kappa :                        0.414

Mcnemar's Test P-Value :       0.031

Sensitivity :                  0.844
Specificity :                  0.568
Pos Pred Value :               0.667
Neg Pred Value :               0.781
Prevalence :                   0.506
Detection Rate :               0.427
Detection Prevalence :         0.640
Balanced Accuracy :            0.706

'Positive' Class :             0
```

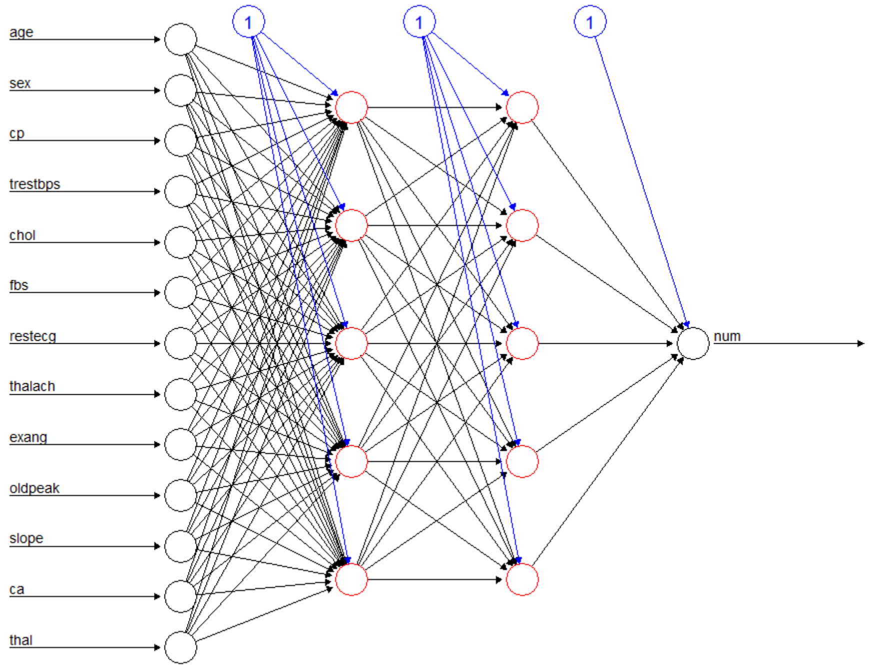
The model's improvement is indicated by the AUC of 0.724, which indicates that it can accurately categorize 72% of patients as having heart disease or not. This improvement is also shown by the confidence interval, which supports our assumption that the model performs well and that the true AUC lies in the range of 0.563 to 0.885.

The trade-off between sensitivity (True Positive Rate) and False Positive Rate (1 – Specificity) is displayed by the ROC curve. Our proposed model outperforms the 50% benchmark, as evidenced by a 29% decrease in the misclassification rate and an overall area under the curve of 0.724.

The confusion matrix, upon analysis, indicates a sensitivity (also known as the "True Positive Rate" or "Recall") of 0.84. This shows a consistent percentage of correctly identifying people as having heart disease. Moreover, the True Negative Rate (specificity) of 0.568 indicates an above-average capacity to accurately determine the absence of heart disease.

With a precision (positive predictive value) of 67% and a negative predictive value of 78%, our artificial neural network model yields an accuracy of 71%. These demonstrate low errors in the classification of positive and negative heart disease statuses, indicating a favourable false positive and negative rate.

```
set.seed(12345)
options(digits=3)
net3 <- neuralnet(num ~ .,
          data=D1,
          hidden=c(8,7,5), #3 hidden layer, 8,7,5 neurons respectively
          act.fct='logistic', err.fct="sse", linear.output=FALSE, likelihood=TRUE )


# PLOT THE MODEL
plot(net3, rep="best", show.weights=FALSE, dimension=6.5, radius=.15,
  col.hidden="red", col.hidden.synapse="black", lwd=1, fontsize=9, arrow.length=0.15)
```



The neurons and hidden layers that the neural network produced, along with the corresponding weights of the created ANN model, are displayed in the above image. The aforementioned image also illustrates the significance that the model ascribes to various neurons and hidden layers in its prediction process. Through meticulous examination of the weight distribution, we can enhance our comprehension of the neural network's internal workings. Additionally, we are able to recognize the important roles that the neurons and hidden layers play in the model's ability to make decisions. Understanding the workings of the ANN model and interpreting its behaviour requires knowledge of this information.

```
ypred2 <- predict(net3, D2)
ypred2 <- ypred2[,1]
as.vector(ypred2)
```

```
##  [1] 2.09e-01 2.09e-01 9.40e-01 2.09e-01 1.83e-25 2.09e-01 2.08e-01 2.09e-
01
##  [9] 9.40e-01 2.09e-01 9.40e-01 1.83e-25 9.40e-01 2.09e-01 9.40e-01 2.09e-
01
## [17] 2.09e-01 2.09e-01 1.83e-25 9.31e-01 2.09e-01 9.40e-01 1.86e-25 9.40e-
01
## [25] 9.40e-01 2.09e-01 9.40e-01 9.40e-01 2.09e-01 1.84e-25 2.02e-01 1.08e-
23
## [33] 9.40e-01 9.40e-01 1.87e-01 2.09e-01 2.09e-01 2.09e-01 9.29e-01 9.27e-
01
## [41] 9.27e-01 9.40e-01 9.40e-01 1.87e-25 1.83e-25 9.40e-01 2.09e-01 2.09e-
01
## [49] 2.09e-01 2.09e-01 2.09e-01 9.40e-01 9.40e-01 9.40e-01 9.27e-01 2.09e-
```

```
01
## [57] 9.40e-01 2.09e-01 9.40e-01 9.40e-01 2.09e-01 9.40e-01 2.09e-01 2.09e-
01
## [65] 2.09e-01 2.09e-01 2.09e-01 1.07e-23 1.83e-25 2.09e-01 2.09e-01 2.09e-
01
## [73] 1.83e-25 9.40e-01 2.09e-01 2.09e-01 1.83e-25 1.08e-23 2.09e-01 9.40e-
01
## [81] 1.83e-25 1.83e-25 2.09e-01 2.09e-01 9.27e-01 2.09e-01 2.09e-01 2.09e-
01
## [89] 2.09e-01

MSE.c <- mean((yobs-ypred2)^2)
MSE.c

## [1] 0.181

ypred2 <- ifelse(ypred2>0.5,1,0)
as.vector(ypred2)
```

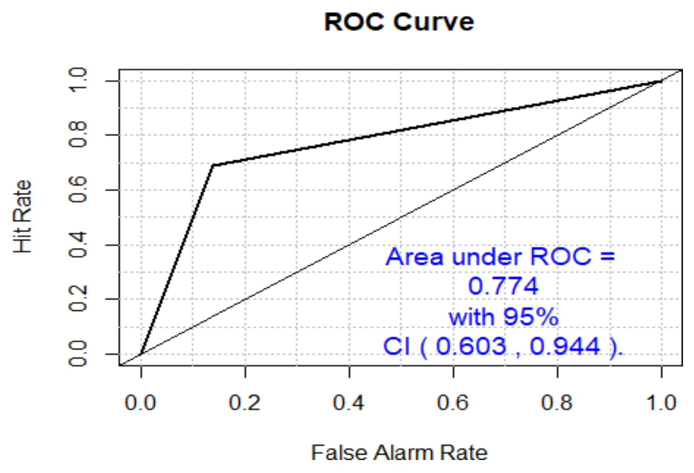
```
##  [1] 0 0 1 0 0 0 0 0 1 0 1 0 1 0 1 0 0 0 0 1 0 1 0 1 1 0 1 1 0 0 0 0 1 1 0
0 0 0
## [39] 1 1 1 1 1 0 0 1 0 0 0 0 0 1 1 1 1 0 1 0 1 1 0 1 0 0 0 0 0 0 0 0 0 0 0
1 0 0
## [77] 0 0 0 1 0 0 0 0 1 0 0 0 0
```

As a result of the aforementioned, we were able to get an enhanced MSE (Mean Squared Error) of 18%, which indicates that the model performed better in correctly predicting the heart disease class within the dataset. This enhancement highlights the value of adding more intricate layers, which enable the model to adjust and learn, leading to a lower misclassification rate. Through iteration and refinement of the model architecture, we want to improve the model's efficacy in real-world applications by improving its capacity to categorize instances of heart disease properly. The model's performance must be continuously improved, and this iterative process guarantees the model's applicability and dependability in real-world situations.

```
  rf_AUC <- ci.cvAUC(predictions = ypred2, labels =yobs,
folds=1:NROW(D2),confidence = 0.95)

rf_AUC

##  $cvAUC
## [1]
0.774##
## $se
## [1]
```

```
0.0871##
## $ci
## [1] 0.603
0.944##
##
$confidence##
[1] 0.95
```

```
roc.plot(mod.nn, plot.thres=NULL)
text(x=0.7, y=0.2,paste("Area under ROC = \n", round(rf_AUC$cvAUC, digits
=3), "\n with 95% \nCI (",
                        rf_auc.ci[1], ",",rf_auc.ci[2], ").", sep = " "),
col="blue", cex =1.2)
```



```
(miss.rate <- mean(yobs != ypred2))
```

```
## [1] 0.202
```

```
confusionMatrix(factor(yobs), factor(ypred2))
```

```
## Confusion Matrix and Statistics

        Reference
  Prediction  0  1
           0 49  8
           1 10 22
```

Accuracy :	0.798
95% CI :	(0.699, 0.876)
No Information Rate :	0.663

P-Value [Acc > NIR]	0.00377
Kappa :	0.555
Mcnemar's Test P-Value :	0.81366

```
            Sensitivity : 0.831
            Specificity : 0.733
         Pos Pred Value : 0.860
         Neg Pred Value : 0.688
             Prevalence : 0.663
         Detection Rate : 0.551
   Detection Prevalence : 0.640
      Balanced Accuracy : 0.782

       'Positive' Class : 0
```

The development in our model is evident from the AUC of 0.774, which suggests a 77% capacity to accurately estimate whether a patient would have heart disease or not. The confidence interval also supports this improvement. The fact that the genuine AUC lies between 0.603 to 0.944 demonstrates our degree of confidence in the model's performance.

The ROC curve (1 – Specificity) graphically illustrates the trade-off between True Positive Rate (TPR) and False Positive Rate (FPR). The resulting model outperforms the 50% benchmark with a lower misclassification rate of 20% and a total area under the curve of 0.774.

The sensitivity (sometimes referred to as the "True Positive Rate" or "Recall") is 0.83, which is a great proportion for correctly classifying people as having heart disease, as determined by looking at the confusion matrix. Furthermore, the above-average ability to reliably classify the absence of heart disease is indicated by the True Negative Rate (specificity) of 0.733.

Our trained artificial neural network (ANN) model achieves 80% accuracy with a precision (Positive Predictive Value) of 86% and a negative predictive value of 69%. With a favourable false positive and negative rate, these measurements show a low rate of incorrectly classifying heart disease statuses as positive or negative.

MODEL COMPARISON

We assessed different hidden layer and neuron structures in order to construct the ANN machine learning model:

Model 1: One hidden layer with 1 neuron (Perceptron). Model 2: One hidden layer with 3 neurons. Model 3: Two hidden layers with 5 neurons in the first layer and 2 neurons in the second layer. Below are the results of our analysis for each model:

Evaluation metrics	Model1	Model2	Model3
Accuracy	0.3600	0.7080	0.7980
Sensitivity	0.5000	0.8440	0.8310
Specificity	0.3563	0.5680	0.7330
Precision	0.0175	0.6670	0.8600
Neg. Pred.Val	0.9687	0.7810	0.6880

Evaluation metrics	Model 1	Model2	Model3
MSE	0.2560	0.2920	0.2020
Miss Rate	0.6400	0.2920	0.2020
AUC	0.4930	0.7240	0.7740

Three distinct artificial neural network (ANN) topologies were tested in this model comparison in order to determine whether heart disease was present or not. With only one hidden layer and one neuron, Model 1 performed the worst across the board, demonstrating its poor ability to correctly categorize cases of heart disease. With one hidden layer and three neurons, Model 2 performed better than Model 1, showing significant improvements in AUC, sensitivity, accuracy, and precision. But of the three models, Model 3, with its two hidden layers and five and two neurons in each, performed the best, obtaining the highest levels of accuracy, sensitivity, and precision while simultaneously lowering the MSE and misclassification rate. This implies that the ANN model's capacity to correctly predict the presence of heart disease is much enhanced by adding more layers and neurons, underscoring the significance of model complexity in capturing the complex patterns found in the data.

SUMMARY AND CONCLUSION

Our study's findings highlight the need of building more complicated models in order to create highly accurate predictive models, especially when it comes to classifying heart disease utilizing artificial neural networks (ANNs). We evaluated three models with different levels of complexity and found a consistent trend: predictive performance rose with model complexity. With its highest accuracy, sensitivity, precision, and area under the curve (AUC), Model 3, with its complex design, proved to be a most reliable and accurate predictor of heart disease. This demonstrates how important model complexity is to identifying the complicated patterns in the data and, in the end, building an extremely accurate prediction model. In order to maximize prediction performance and enhance patient outcomes, future machine learning applications in the healthcare industry should place a higher priority on the investigation and improvement of complicated model architectures.