Identification of Heart Disease from Common Indicators

Three levels of models have been produced. Predictors were eliminated by Backwards Selection to minimize AIC score.

Fine tuning ideas: 1. split observations by sex and run logistic models for each of these sets 2. Split observations by first N (1,2,3) discriminating predictors from decision tree and run logistic models for each combination.

Data downloaded from Kaggle: https://www.kaggle.com/ronitf/heart-disease-uci

Call libraries and read in dataset

```
library(readr)
library(MASS)
library(class)

## Warning: package 'class' was built under R version 4.0.3

library(rpart)

## Warning: package 'rpart' was built under R version 4.0.3

library(rpart.plot)

## Warning: package 'rpart.plot' was built under R version 4.0.3

library(e1071)

## Warning: package 'e1071' was built under R version 4.0.3

heart <-
read_csv("C:\\Users\\mapet\\Documents\\WayneState_classes\\DSA600_DataScience
AndAnalytics\\R files\\FinalProject_2\\heart.csv")

## Parsed with column specification:
## cols(
##   age = col_double(),
##   sex = col_double(),
##   cp = col_double(),
##   trestbps = col_double(),
##   chol = col_double(),
##   fbs = col_double(),
##   restecg = col_double(),
##   thalach = col_double(),
##   exang = col_double(),
##   oldpeak = col_double(),
##   slope = col_double(),
##   ca = col_double(),
##   thal = col_double(),
```

```
##   target = col_double()
## )
```

Quick look at data

```
summary(heart)
```

```
##       age             sex               cp            trestbps    
##  Min.   :29.00   Min.   :0.0000   Min.   :0.0000   Min.   : 94.0  
##  1st Qu.:48.00   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:120.0  
##  Median :56.00   Median :1.0000   Median :1.0000   Median :130.0  
##  Mean   :54.52   Mean   :0.6791   Mean   :0.9595   Mean   :131.6  
##  3rd Qu.:61.00   3rd Qu.:1.0000   3rd Qu.:2.0000   3rd Qu.:140.0  
##  Max.   :77.00   Max.   :1.0000   Max.   :3.0000   Max.   :200.0  
##       chol             fbs            restecg          thalach     
##  Min.   :126.0   Min.   :0.0000   Min.   :0.0000   Min.   : 71.0  
##  1st Qu.:211.0   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:133.0  
##  Median :242.5   Median :0.0000   Median :1.0000   Median :152.5  
##  Mean   :247.2   Mean   :0.1453   Mean   :0.5236   Mean   :149.6  
##  3rd Qu.:275.2   3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.:166.0  
##  Max.   :564.0   Max.   :1.0000   Max.   :2.0000   Max.   :202.0  
##      exang           oldpeak          slope             ca        
##  Min.   :0.0000   Min.   :0.000   Min.   :0.000   Min.   :0.0000  
##  1st Qu.:0.0000   1st Qu.:0.000   1st Qu.:1.000   1st Qu.:0.0000  
##  Median :0.0000   Median :0.800   Median :1.000   Median :0.0000  
##  Mean   :0.3277   Mean   :1.059   Mean   :1.395   Mean   :0.6791  
##  3rd Qu.:1.0000   3rd Qu.:1.650   3rd Qu.:2.000   3rd Qu.:1.0000  
##  Max.   :1.0000   Max.   :6.200   Max.   :2.000   Max.   :3.0000  
##       thal           target      
##  Min.   :1.000   Min.   :0.0000  
##  1st Qu.:2.000   1st Qu.:0.0000  
##  Median :2.000   Median :1.0000  
##  Mean   :2.328   Mean   :0.5405  
##  3rd Qu.:3.000   3rd Qu.:1.0000  
##  Max.   :3.000   Max.   :1.0000
```

```
head(heart)
```

```
## # A tibble: 6 x 14
##     age   sex    cp trestbps  chol   fbs restecg thalach exang oldpeak
slope
##   <dbl> <dbl> <dbl>    <dbl> <dbl> <dbl>   <dbl>   <dbl> <dbl>   <dbl>
<dbl>
## 1    63     1     3      145   233     1       0     150     0     2.3
0
## 2    37     1     2      130   250     0       1     187     0     3.5
0
## 3    41     0     1      130   204     0       0     172     0     1.4
2
## 4    56     1     1      120   236     0       1     178     0     0.8
2
```

```
## 5     57      0      0       120    354      0        1      163      1      0.6
2
## 6     57      1      0       140    192      0        1      148      0      0.4
1
## # ... with 3 more variables: ca <dbl>, thal <dbl>, target <dbl>
```

Create Functions

```
ROC_func <- function(df, label_colnum, score_colnum, add_on = F, color =
"black"){
  # Sort by score (high to low)
  df <- df[order(-df[,score_colnum]),]
  rownames(df) <- NULL  # Reset the row number to 1,2,3,...
  n <- nrow(df)
  # Total # of positive and negative cases in the data set
  P <- sum(df[,label_colnum] == 1)
  N <- sum(df[,label_colnum] == 0)

  # Vectors to hold the coordinates of points on the ROC curve
  TPR <- c(0,vector(mode="numeric", length=n))
  FPR <- c(0,vector(mode="numeric", length=n))

  # Calculate the coordinates from one point to the next
  AUC = 0
  for(k in 1:n){
    if(df[k,label_colnum] == 1){
      TPR[k+1] = TPR[k] + 1/P
      FPR[k+1] = FPR[k]
    } else{
      TPR[k+1] = TPR[k]
      FPR[k+1] = FPR[k] + 1/N
      AUC = AUC + TPR[k+1]*(1/N)
    }
  }

  # Plot the ROC curve
  if(add_on){
    points(FPR, TPR, main=paste0("ROC curve"," (n = ", n, ")"), type = 'l',
col=color, cex.lab = 1.2, cex.axis = 1.2, cex.main = 1.2)
  } else{
    plot(FPR, TPR, main=paste0("ROC curve"," (n = ", n, ")"), type = 'l',
col=color, cex.lab = 1.2, cex.axis = 1.2, cex.main = 1.2)
  }
  return(AUC)
}

resetSets <- function(){
  trainset <<- sample(1:nrow(heart), round(nrow(heart)*0.7))
  validset <<- setdiff(1:nrow(heart),trainset)
}
```

```
errorRate <- function(newData){
  pred <- func2(newData)
  correct <- sum(ifelse(pred==newData$target,1,0))
  return(1-(correct/nrow(newData)))
}

errorMatrix <- function(newData){
  pred <- func2(newData)
  err1 <- sum(ifelse(pred==1,ifelse(newData$target==1,1,0),0))
  err2 <- sum(ifelse(pred==1,ifelse(newData$target==0,1,0),0))
  err3 <- sum(ifelse(pred==0,ifelse(newData$target==1,1,0),0))
  err4 <- sum(ifelse(pred==0,ifelse(newData$target==0,1,0),0))

  totalerror <- (err2+err3)/nrow(pred)

cat(sprintf("PTrue-LTrue: %s\n", err1))
cat(sprintf("PTrue-LFalse: %s\n", err2))
cat(sprintf("PFalse-LTrue: %s\n", err3))
cat(sprintf("PFalse-LFalse: %s\n", err4))
cat(sprintf("Total Error: %s\n", totalerror))
}

resetSets()
```

Logistic Model

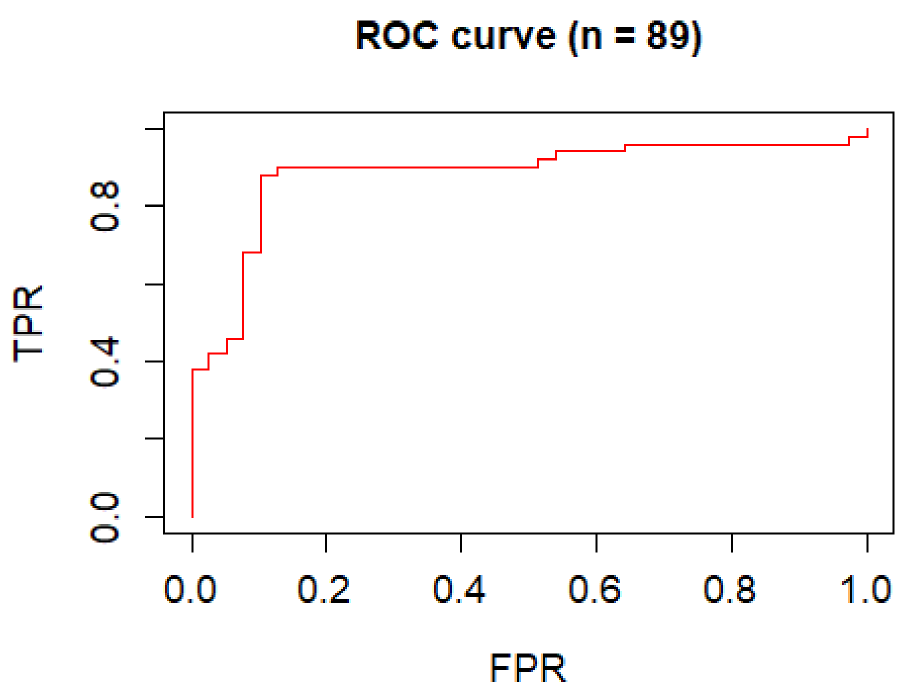
Confirmation Model

```
#Model
df.log.confirmation <- glm(target ~ sex + cp + exang + thalach + oldpeak + ca
+ thal, data = heart, subset = trainset, family = "binomial")

#Summary of model
summary(df.log.confirmation)

##
## Call:
## glm(formula = target ~ sex + cp + exang + thalach + oldpeak +
##     ca + thal, family = "binomial", data = heart, subset = trainset)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4008  -0.3156   0.1559   0.4360   2.8309
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  4.036357   2.123683   1.901  0.05735 .
## sex         -1.326119   0.531645  -2.494  0.01262 *
```

```
## cp           0.997429   0.245729   4.059 4.93e-05 ***
## exang       -1.323050   0.523179  -2.529  0.01144 *
## thalach      0.006757   0.012598   0.536  0.59170
## oldpeak     -0.759674   0.271454  -2.799  0.00513 **
## ca          -1.361726   0.297225  -4.581 4.62e-06 ***
## thal        -1.177043   0.377840  -3.115  0.00184 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 284.83  on 206  degrees of freedom
## Residual deviance: 130.78  on 199  degrees of freedom
## AIC: 146.78
##
## Number of Fisher Scoring iterations: 6
```

```
#Scoring Functions

#Function containing model

func1 <- function(newData)  {
  len <- nrow(newData)
  scores <- numeric(len)
  for (i in 1:len)  {
    val <- 0.37171  +  newData[i,'sex']*(-1.67767) +
newData[i,'cp']*(0.98436) +  newData[i,'exang']*(-1.10948) +
newData[i,'thalach']*(0.03018) +  newData[i,'oldpeak']*(-1.05497) +
newData[i,'ca']*(-0.85910) +  newData[i,'thal']*(-1.18683)
    scores[i] <- exp(val)/(1+exp(val))
  }
  return(scores)
}

#Manually chose optimal threshold, that optimized AUC
threshold <- 0.55

#Function turns score values into categorical predictions using threshold
func2 <- function(newData)   {
  scores <- func1(newData)

  prediction <- numeric(nrow(newData))
  prediction <- ifelse(scores>threshold,1,0)
  return(prediction)
}

#Calculate ROC curve, AUC value, confusion matrix and error Rate
df_score_truelabel <- data.frame(func2(heart[validset,]),true.label =
```

```
heart[validset,'target'])
head(df_score_truelabel)

##   func2.heart.validset.... target
## 1                        1      1
## 2                        1      1
## 3                        1      1
## 4                        1      1
## 5                        1      1
## 6                        1      1

df.log.confirmation.AUC <- ROC_func(df_score_truelabel, 1, 2, color = 'red')
```



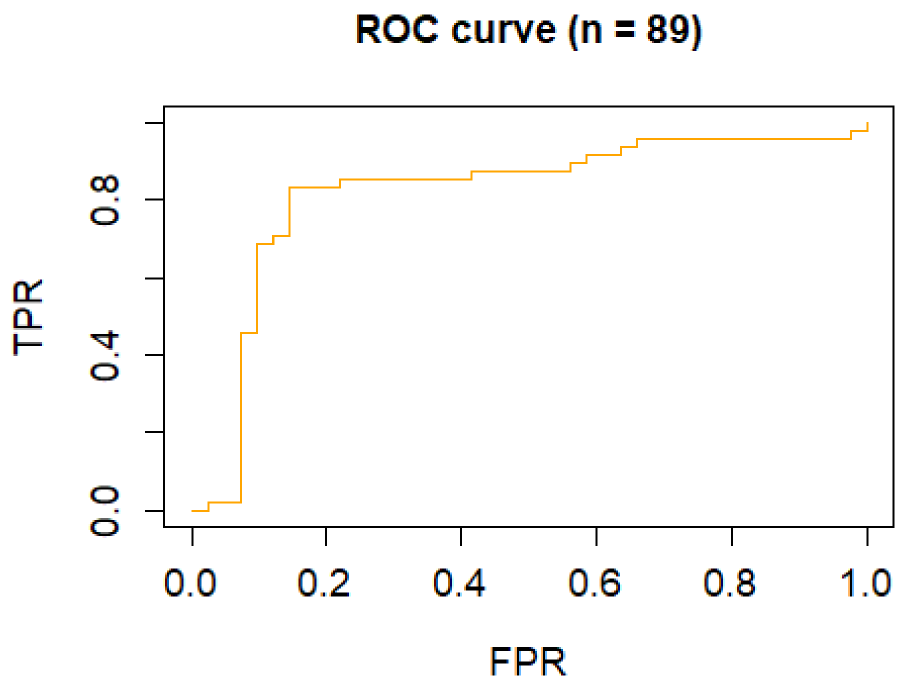

```
df.log.confirmation.ErrorR <- errorRate(newData = heart[validset,])

cat(sprintf("Threshold: %s\n", threshold))

## Threshold: 0.55

cat(sprintf("Total Error: %s\n", df.log.confirmation.ErrorR))

## Total Error: 0.134831460674157

cat(sprintf("AUC: %s\n", df.log.confirmation.AUC))

## AUC: 0.883589743589744

errorMatrix(heart[validset,])
```

```
## PTrue-LTrue: 42
## PTrue-LFalse: 8
## PFalse-LTrue: 4
## PFalse-LFalse: 35
```

Classification Model

```
#Modelling
df.log.classification <- glm(target ~ sex + cp + thalach + exang + oldpeak,
data = heart, subset = trainset, family = "binomial")
summary(df.log.classification )

##
## Call:
## glm(formula = target ~ sex + cp + thalach + exang + oldpeak,
##     family = "binomial", data = heart, subset = trainset)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3971  -0.4805   0.2391   0.6483   2.7330
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.24441    1.61392  -0.151 0.879629
## sex         -1.61173    0.45522  -3.541 0.000399 ***
## cp           0.94784    0.21761   4.356 1.33e-05 ***
## thalach      0.01389    0.01011   1.374 0.169540
## exang       -1.20462    0.44071  -2.733 0.006269 **
## oldpeak     -0.90182    0.22019  -4.096 4.21e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 284.83  on 206  degrees of freedom
## Residual deviance: 168.99  on 201  degrees of freedom
## AIC: 180.99
##
## Number of Fisher Scoring iterations: 5

#Scoring Functions
# Function for model
func1 <- function(newData)  {
  len <- nrow(newData)
  scores <- numeric(len)

  for (i in 1:len)  {
    val <- -3.14787  +  newData[i,'sex']*(-1.97844)  +
newData[i,'cp']*(1.04538)  +  newData[i,'thalach']*(0.033864)  +
newData[i,'exang']*(-1.21676)  +  newData[i,'oldpeak']*(-1.11956)
    scores[i] <- exp(val)/(1+exp(val))
```

```
  }
  return(scores)
}

#Function turns score values into categorical predictions using threshold
func2 <- function(newData)   {
  scores <- func1(newData)
  threshold <- 0.6 #Manually chose optimal threshold, that optimized AUC
  prediction <- numeric(nrow(newData))
  prediction <- ifelse(scores>threshold,1,0)
  return(prediction)
}

#Calculate ROC curve, AUC value, confusion matrix and error Rate
df_score_truelabel <- data.frame(func2(heart[validset,]),true.label =
heart[validset,'target'])
head(df_score_truelabel)

##   func2.heart.validset.... target
## 1                        0      1
## 2                        1      1
## 3                        0      1
## 4                        0      1
## 5                        1      1
## 6                        1      1

df.log.classification.AUC <- ROC_func(df_score_truelabel, 1, 2, color =
'orange')
```




```
df.log.classification.ErrorR <- errorRate(newData = heart[validset,])


cat(sprintf("Total Error: %s\n", df.log.classification.ErrorR))
```

```
## Total Error: 0.157303370786517
```

```
cat(sprintf("AUC: %s\n", df.log.classification.AUC))
```

```
## AUC: 0.819105691056912
```

```
errorMatrix(heart[validset,])
```

```
## PTrue-LTrue: 40
## PTrue-LFalse: 8
## PFalse-LTrue: 6
## PFalse-LFalse: 35
```

Early Warning Model

```
#Modelling

df.log.alarm <- glm(target ~ age + sex + cp + trestbps, data = heart, subset
= trainset, family = "binomial")
summary(df.log.alarm)
```

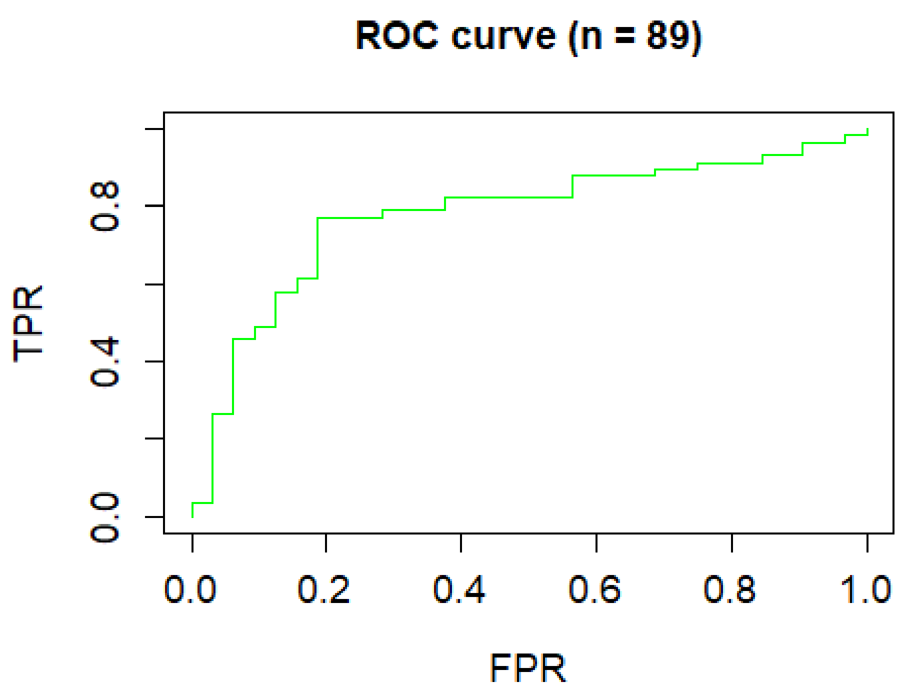
```
##
## Call:
```

```
## glm(formula = target ~ age + sex + cp + trestbps, family = "binomial",
##     data = heart, subset = trainset)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1493  -0.7477   0.2465   0.7357   2.4587
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  7.85524    1.78854   4.392 1.12e-05 ***
## age         -0.06150    0.02212  -2.780  0.00543 **
## sex         -1.88307    0.42218  -4.460 8.18e-06 ***
## cp           1.31780    0.20915   6.301 2.96e-10 ***
## trestbps    -0.03054    0.01100  -2.776  0.00550 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 284.83  on 206  degrees of freedom
## Residual deviance: 196.72  on 202  degrees of freedom
## AIC: 206.72
##
## Number of Fisher Scoring iterations: 5

#Scoring Functions
#Function for model
func1 <- function(newData)  {
  len <- nrow(newData)
  scores <- numeric(len)

  for (i in 1:len)  {
    val <- 6.56547  +  newData[i,'age']*(-0.06335)  +  newData[i,'sex']*(-
1.75071)  +  newData[i,'cp']*(1.08806)  +  newData[i,'trestbps']*(-0.02126)
    scores[i] <- exp(val)/(1+exp(val))
  }
  return(scores)
}

#Function turns score values into categorical predictions using threshold
func2 <- function(newData)   {
  scores <- func1(newData)
  threshold <- 0.4 #Manually chose optimal threshold, that optimized AUC
  prediction <- numeric(nrow(newData))
  prediction <- ifelse(scores>threshold,1,0)
  return(prediction)
}

#Calculate ROC curve, AUC value, confusion matrix and error Rate
df_score_truelabel <- data.frame(func2(heart[validset,]),true.label =
```

```
heart[validset,'target'])
head(df_score_truelabel)
```

```
##   func2.heart.validset.... target
## 1                        1      1
## 2                        1      1
## 3                        0      1
## 4                        1      1
## 5                        1      1
## 6                        1      1
```

```
df.log.alarm.AUC <- ROC_func(df_score_truelabel, 1, 2 , color = 'green')
```




```
df.log.alarm.ErrorR <- errorRate(newData = heart[validset,])


cat(sprintf("Total Error: %s\n", df.log.alarm.ErrorR))
```

```
## Total Error: 0.258426966292135
```

```
cat(sprintf("AUC: %s\n", df.log.alarm.AUC))
```

```
## AUC: 0.777412280701755
```

```
errorMatrix(heart[validset,])
```

```
## PTrue-LTrue: 40
## PTrue-LFalse: 17
```

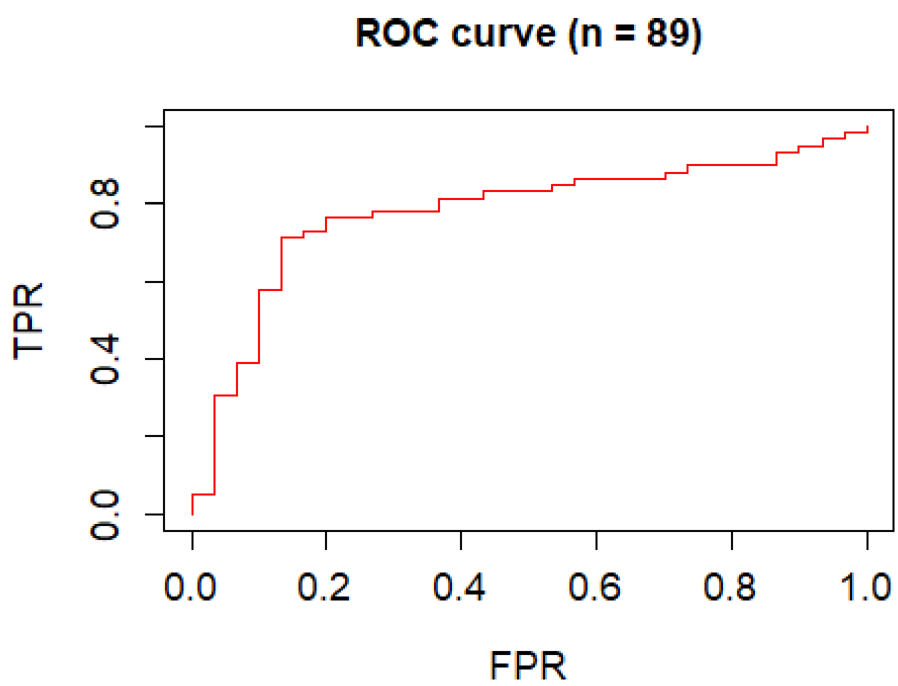
```
## PFalse-LTrue: 6
## PFalse-LFalse: 26
```

KNN Model

Confirmation Model

```
#subsets as s
earlywarning <- 6 #earlywarning
classification <- 11 #Classification
confirmation <- 13 #Confirmation


s = confirmation

#This sets all three predictor subsets: Early Alarm, Classification,
Confirmation
traindata <- heart[trainset, c(1:s,14) ]
testdata <- heart[validset, c(1:s,14)]
hrt <- heart[, c(1:s,14)]
('-----------------------------------------------------')

## [1] "-----------------------------------------------------"

(s)

## [1] 13

#Normalizing the data makes the accuracy jump higher in calculations below
normalize <- function(x){return ((x - min(x)) / (max(x) - min(x))) }
heart_n <- as.data.frame(lapply(hrt[,], normalize))
n <- length(heart_n)-1

#KNN model
#Manually tested different K values for the best accuracy
pred_knn <- knn(train= heart_n[trainset, 1:n], test=heart_n[validset, 1:n],
cl=heart[trainset,]$target, k=25)

#Confusion Matrix table
table( predictions = pred_knn, target = heart[validset,]$target)

##            target
## predictions  0  1
##           0 26  4
##           1 17 42

#Error Rate
( sum(ifelse(pred_knn == heart[validset,]$target, 1, 0))/
length(heart[validset,]$target))
```

```
## [1] 0.7640449

#create an AUC curve
knn_pred_truelabel <- data.frame(pred_knn, heart[validset,]$target)
heart.knn.AUC <- ROC_func(knn_pred_truelabel, 1, 2 , color = 'red')
```




```
(heart.knn.AUC)

## [1] 0.7785311
```

Classification Model

```
s =classification

#This sets all three predictor subsets: Early Alarm, Classification,
Confirmation
traindata <- heart[trainset, c(1:s,14) ]
testdata <- heart[validset, c(1:s,14)]
hrt <- heart[, c(1:s,14)]
('-----------------------------------------------------')

## [1] "-----------------------------------------------------"

(s)

## [1] 11
```

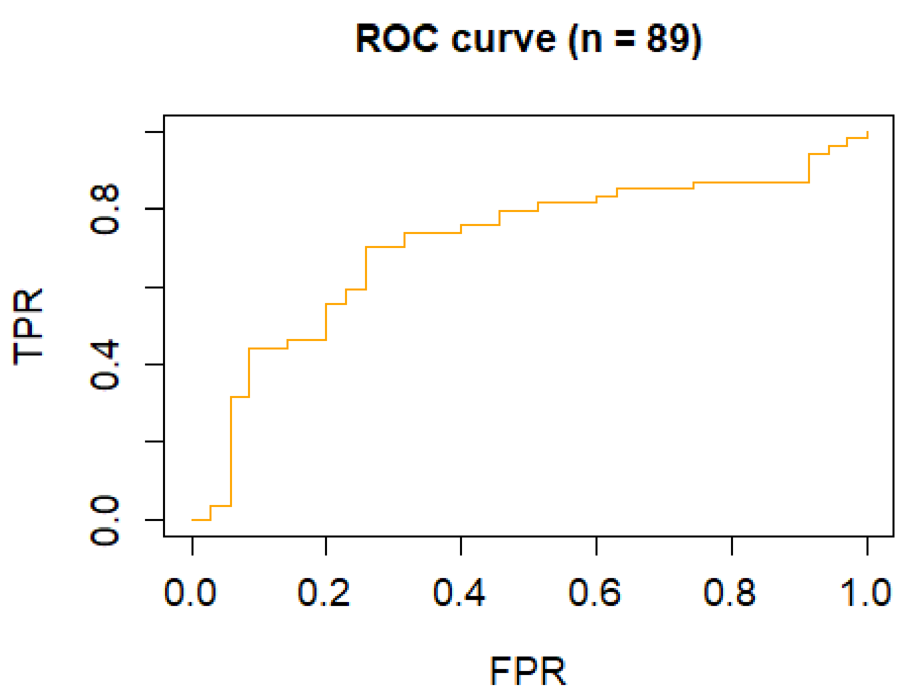
```r
#Normalizing the data makes the accuracy jump higher in calculations below
normalize <- function(x){return ((x - min(x)) / (max(x) - min(x))) }
heart_n <- as.data.frame(lapply(hrt[,], normalize))
n <- length(heart_n)-1

#KNN model
#Manually tested different K values for the best accuracy
pred_knn <- knn(train= heart_n[trainset, 1:n], test=heart_n[validset, 1:n],
cl=heart[trainset,]$target, k=20)

#Confusion Matrix table
table( predictions = pred_knn, target = heart[validset,]$target)
```

```
##              target
## predictions  0  1
##            0 26  9
##            1 17 37
```

```r
#Error Rate
( sum(ifelse(pred_knn == heart[validset,]$target, 1, 0))/
length(heart[validset,]$target))
```

```
## [1] 0.7078652
```

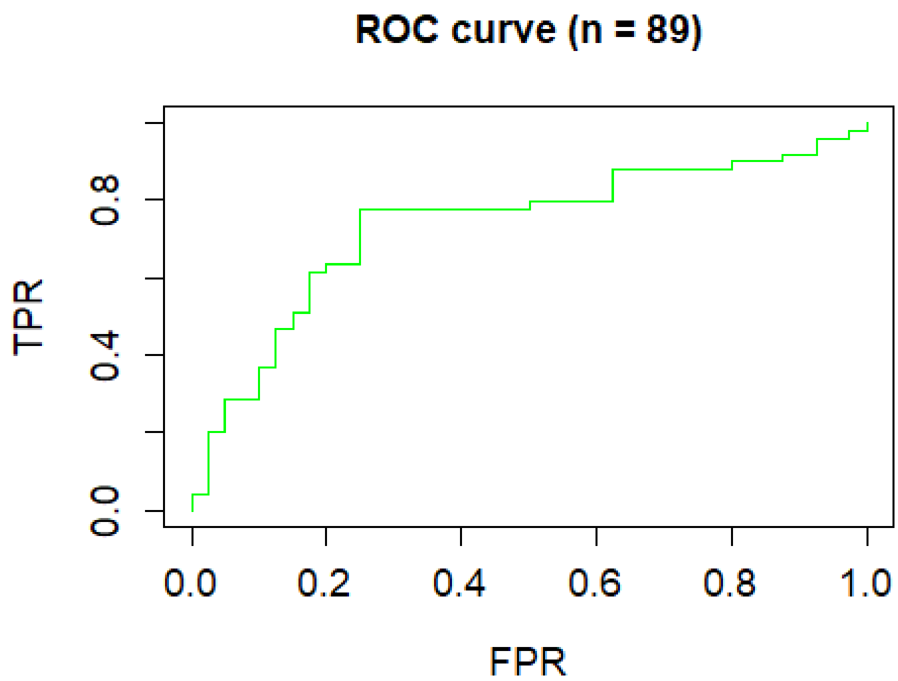
```r
#create an AUC curve
knn_pred_truelabel <- data.frame(pred_knn, heart[validset,]$target)
heart.knn.AUC <- ROC_func(knn_pred_truelabel, 1, 2 , color = 'orange')
```



```
(heart.knn.AUC)

## [1] 0.7100529
```

Early Warning Model

```
s = earlywarning

#This sets all three predictor subsets: Early Alarm, Classification,
Confirmation
traindata <- heart[trainset, c(1:s,14) ]
testdata <- heart[validset, c(1:s,14)]
hrt <- heart[, c(1:s,14)]
('-----------------------------------------------------')

## [1] "-----------------------------------------------------"

(s)

## [1] 6

#Normalizing the data makes the accuracy jump higher in calculations below
normalize <- function(x){return ((x - min(x)) / (max(x) - min(x))) }
heart_n <- as.data.frame(lapply(hrt[,], normalize))
n <- length(heart_n)-1

#KNN model
#Manually tested different K values for the best accuracy
pred_knn <- knn(train= heart_n[trainset, 1:n], test=heart_n[validset, 1:n],
cl=heart[trainset,]$target, k=15)

#Confusion Matrix table
table( predictions = pred_knn, target = heart[validset,]$target)

##             target
## predictions  0  1
##            0 30 10
##            1 13 36

#Error Rate
( sum(ifelse(pred_knn == heart[validset,]$target, 1, 0))/
length(heart[validset,]$target))

## [1] 0.741573

#create an AUC curve
knn_pred_truelabel <- data.frame(pred_knn, heart[validset,]$target)
heart.knn.AUC <- ROC_func(knn_pred_truelabel, 1, 2, color = 'green')
```

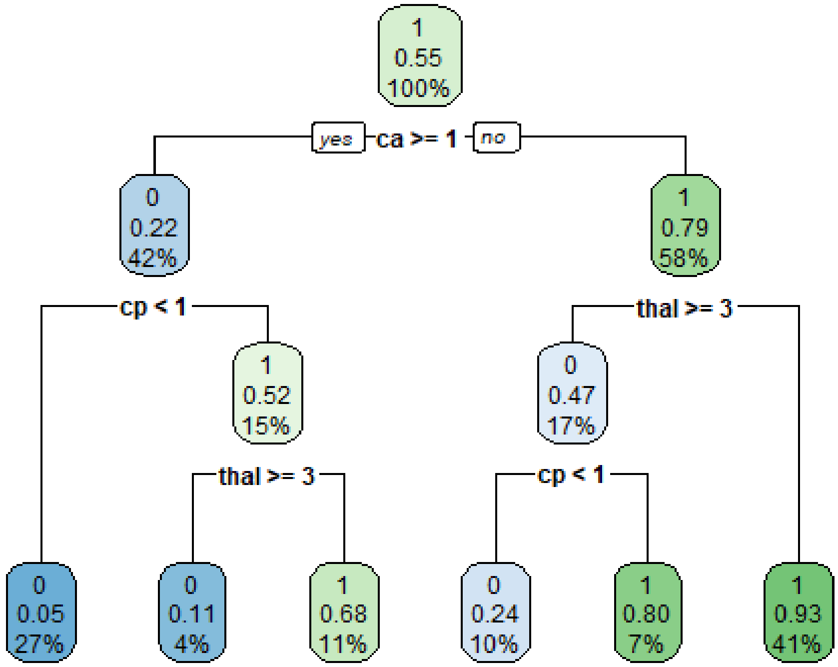



```
(heart.knn.AUC)

## [1] 0.7336735
```

Decision Tree

Confirmation Model

```
tree_hrt_conf <- rpart(target ~ . -trestbps -slope -restecg -exang -chol -sex
-age -thalach -oldpeak -fbs, data = heart,method = 'class',  subset =
trainset)
#tree_flight
rpart.plot(tree_hrt_conf)
```



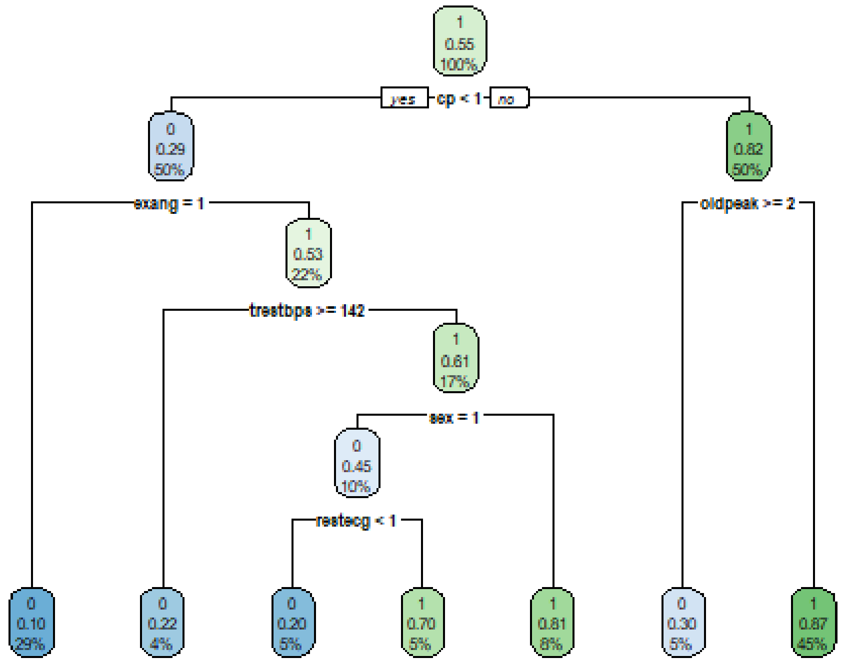

```
summary(tree_hrt_conf)

## Call:
## rpart(formula = target ~ . - trestbps - slope - restecg - exang -
##     chol - sex - age - thalach - oldpeak - fbs, data = heart,
##     subset = trainset, method = "class")
##   n= 207
##
##           CP nsplit rel error    xerror       xstd
## 1 0.52688172      0 1.0000000 1.0000000 0.07695304
## 2 0.05913978      1 0.4731183 0.4731183 0.06329247
## 3 0.04301075      3 0.3548387 0.4193548 0.06049491
## 4 0.01000000      5 0.2688172 0.2795699 0.05126935
##
## Variable importance
##   ca thal   cp
##   49   26   25
##
## Node number 1: 207 observations,    complexity param=0.5268817
##   predicted class=1  expected loss=0.4492754  P(node) =1
##     class counts:    93   114
##    probabilities: 0.449 0.551
##   left son=2 (87 obs) right son=3 (120 obs)
##   Primary splits:
##       ca   < 0.5 to the right, improve=33.15030, (0 missing)
##       cp   < 0.5 to the left,  improve=28.75218, (0 missing)
##       thal < 2.5 to the right, improve=24.58965, (0 missing)
```

```
##   Surrogate splits:
##       cp   < 0.5 to the left,  agree=0.618, adj=0.092, (0 split)
##       thal < 2.5 to the right, agree=0.614, adj=0.080, (0 split)
##
## Node number 2: 87 observations,    complexity param=0.04301075
##   predicted class=0  expected loss=0.2183908  P(node) =0.4202899
##     class counts:    68    19
##    probabilities: 0.782 0.218
##   left son=4 (56 obs) right son=5 (31 obs)
##   Primary splits:
##       cp   < 0.5 to the left,  improve=8.5387070, (0 missing)
##       thal < 2.5 to the right, improve=5.0235600, (0 missing)
##       ca   < 1.5 to the right, improve=0.3074986, (0 missing)
##
## Node number 3: 120 observations,    complexity param=0.05913978
##   predicted class=1  expected loss=0.2083333  P(node) =0.5797101
##     class counts:    25    95
##    probabilities: 0.208 0.792
##   left son=6 (36 obs) right son=7 (84 obs)
##   Primary splits:
##       thal < 2.5 to the right, improve=10.496030, (0 missing)
##       cp   < 0.5 to the left,  improve= 8.402778, (0 missing)
##
## Node number 4: 56 observations
##   predicted class=0  expected loss=0.05357143  P(node) =0.2705314
##     class counts:    53     3
##    probabilities: 0.946 0.054
##
## Node number 5: 31 observations,    complexity param=0.04301075
##   predicted class=1  expected loss=0.483871  P(node) =0.1497585
##     class counts:    15    16
##    probabilities: 0.484 0.516
##   left son=10 (9 obs) right son=11 (22 obs)
##   Primary splits:
##       thal < 2.5 to the right, improve=4.160639000, (0 missing)
##       ca   < 1.5 to the right, improve=0.387379700, (0 missing)
##       cp   < 1.5 to the left,  improve=0.005610098, (0 missing)
##
## Node number 6: 36 observations,    complexity param=0.05913978
##   predicted class=0  expected loss=0.4722222  P(node) =0.173913
##     class counts:    19    17
##    probabilities: 0.528 0.472
##   left son=12 (21 obs) right son=13 (15 obs)
##   Primary splits:
##       cp < 0.5 to the left,  improve=5.525397, (0 missing)
##
## Node number 7: 84 observations
##   predicted class=1  expected loss=0.07142857  P(node) =0.4057971
##     class counts:     6    78
##    probabilities: 0.071 0.929
```

```
##
## Node number 10: 9 observations
##   predicted class=0  expected loss=0.1111111  P(node) =0.04347826
##     class counts:     8     1
##    probabilities: 0.889 0.111
##
## Node number 11: 22 observations
##   predicted class=1  expected loss=0.3181818  P(node) =0.1062802
##     class counts:     7    15
##    probabilities: 0.318 0.682
##
## Node number 12: 21 observations
##   predicted class=0  expected loss=0.2380952  P(node) =0.1014493
##     class counts:    16     5
##    probabilities: 0.762 0.238
##
## Node number 13: 15 observations
##   predicted class=1  expected loss=0.2  P(node) =0.07246377
##     class counts:     3    12
##    probabilities: 0.200 0.800
```

Classification Model

```
tree_hrt_class <- rpart(target
~age+sex+cp+trestbps+chol+fbs+restecg+thalach+exang+oldpeak+slope, data =
heart,method = 'class',  subset = trainset)

rpart.plot(tree_hrt_class)
```




```
summary(tree_hrt_class)

## Call:
## rpart(formula = target ~ age + sex + cp + trestbps + chol + fbs +
##     restecg + thalach + exang + oldpeak + slope, data = heart,
##     subset = trainset, method = "class")
##   n= 207
##
##           CP nsplit rel error    xerror       xstd
## 1 0.47311828      0 1.0000000 1.0000000 0.07695304
## 2 0.04301075      1 0.5268817 0.5698925 0.06751965
## 3 0.03225806      4 0.3978495 0.6666667 0.07086173
## 4 0.01000000      6 0.3333333 0.6021505 0.06872494
##
## Variable importance
##       cp    exang  oldpeak  thalach    slope      age trestbps      sex
##       25       19       16       14        8        5        4        3
##     chol  restecg
##        2        2
##
## Node number 1: 207 observations,    complexity param=0.4731183
##   predicted class=1  expected loss=0.4492754  P(node) =1
##     class counts:    93   114
##    probabilities: 0.449 0.551
##   left son=2 (104 obs) right son=3 (103 obs)
##   Primary splits:
##       cp       < 0.5   to the left,  improve=28.75218, (0 missing)
```

```
##       exang   < 0.5   to the right, improve=24.90516, (0 missing)
##       oldpeak < 1.7   to the right, improve=18.67590, (0 missing)
##       slope   < 1.5   to the left,  improve=16.23781, (0 missing)
##       thalach < 147.5 to the left,  improve=15.79129, (0 missing)
##   Surrogate splits:
##       exang   < 0.5   to the right, agree=0.725, adj=0.447, (0 split)
##       thalach < 147.5 to the left,  agree=0.705, adj=0.408, (0 split)
##       oldpeak < 0.85  to the right, agree=0.643, adj=0.282, (0 split)
##       slope   < 1.5   to the left,  agree=0.623, adj=0.243, (0 split)
##       age     < 52.5  to the right, agree=0.585, adj=0.165, (0 split)
##
## Node number 2: 104 observations,    complexity param=0.04301075
##   predicted class=0  expected loss=0.2884615  P(node) =0.5024155
##     class counts:    74    30
##    probabilities: 0.712 0.288
##   left son=4 (59 obs) right son=5 (45 obs)
##   Primary splits:
##       exang   < 0.5   to the right, improve=9.512647, (0 missing)
##       oldpeak < 0.7   to the right, improve=6.282051, (0 missing)
##       slope   < 1.5   to the left,  improve=5.817762, (0 missing)
##       thalach < 158.5 to the left,  improve=3.860177, (0 missing)
##       chol    < 265.5 to the right, improve=3.473558, (0 missing)
##   Surrogate splits:
##       thalach  < 150.5 to the left,  agree=0.702, adj=0.311, (0 split)
##       oldpeak  < 0.85  to the right, agree=0.683, adj=0.267, (0 split)
##       slope    < 1.5   to the left,  agree=0.673, adj=0.244, (0 split)
##       sex      < 0.5   to the right, agree=0.635, adj=0.156, (0 split)
##       trestbps < 116   to the right, agree=0.635, adj=0.156, (0 split)
##
## Node number 3: 103 observations,    complexity param=0.04301075
##   predicted class=1  expected loss=0.184466  P(node) =0.4975845
##     class counts:    19    84
##    probabilities: 0.184 0.816
##   left son=6 (10 obs) right son=7 (93 obs)
##   Primary splits:
##       oldpeak  < 1.95  to the right, improve=5.887065, (0 missing)
##       sex      < 0.5   to the right, improve=2.862527, (0 missing)
##       age      < 56.5  to the right, improve=2.781958, (0 missing)
##       trestbps < 153   to the right, improve=2.716060, (0 missing)
##       slope    < 1.5   to the left,  improve=2.395484, (0 missing)
##
## Node number 4: 59 observations
##   predicted class=0  expected loss=0.1016949  P(node) =0.2850242
##     class counts:    53     6
##    probabilities: 0.898 0.102
##
## Node number 5: 45 observations,    complexity param=0.04301075
##   predicted class=1  expected loss=0.4666667  P(node) =0.2173913
##     class counts:    21    24
##    probabilities: 0.467 0.533
```

```
##   left son=10 (9 obs) right son=11 (36 obs)
##   Primary splits:
##       trestbps < 142   to the right, improve=2.1777780, (0 missing)
##       sex      < 0.5   to the right, improve=2.1407410, (0 missing)
##       restecg  < 0.5   to the left,  improve=1.8980240, (0 missing)
##       oldpeak  < 1.75  to the right, improve=1.5621620, (0 missing)
##       age      < 63.5  to the left,  improve=0.9135135, (0 missing)
##   Surrogate splits:
##       oldpeak < 1.95  to the right, agree=0.867, adj=0.333, (0 split)
##       thalach < 105.5 to the left,  agree=0.844, adj=0.222, (0 split)
##       fbs     < 0.5   to the right, agree=0.822, adj=0.111, (0 split)
##
## Node number 6: 10 observations
##   predicted class=0  expected loss=0.3  P(node) =0.04830918
##     class counts:     7     3
##    probabilities: 0.700 0.300
##
## Node number 7: 93 observations
##   predicted class=1  expected loss=0.1290323  P(node) =0.4492754
##     class counts:    12    81
##    probabilities: 0.129 0.871
##
## Node number 10: 9 observations
##   predicted class=0  expected loss=0.2222222  P(node) =0.04347826
##     class counts:     7     2
##    probabilities: 0.778 0.222
##
## Node number 11: 36 observations,    complexity param=0.03225806
##   predicted class=1  expected loss=0.3888889  P(node) =0.173913
##     class counts:    14    22
##    probabilities: 0.389 0.611
##   left son=22 (20 obs) right son=23 (16 obs)
##   Primary splits:
##       sex     < 0.5   to the right, improve=2.3361110, (0 missing)
##       restecg < 0.5   to the left,  improve=2.0000000, (0 missing)
##       thalach < 143.5 to the right, improve=1.3583840, (0 missing)
##       chol    < 266.5 to the right, improve=1.2341880, (0 missing)
##       oldpeak < 1.3   to the right, improve=0.5790914, (0 missing)
##   Surrogate splits:
##       trestbps < 122   to the left,  agree=0.694, adj=0.312, (0 split)
##       thalach  < 135.5 to the right, agree=0.694, adj=0.312, (0 split)
##       age      < 60.5  to the left,  agree=0.667, adj=0.250, (0 split)
##       chol     < 262   to the left,  agree=0.667, adj=0.250, (0 split)
##       oldpeak  < 1.5   to the left,  agree=0.667, adj=0.250, (0 split)
##
## Node number 22: 20 observations,    complexity param=0.03225806
##   predicted class=0  expected loss=0.45  P(node) =0.09661836
##     class counts:    11     9
##    probabilities: 0.550 0.450
##   left son=44 (10 obs) right son=45 (10 obs)
```

```
##   Primary splits:
##       restecg  < 0.5   to the left,  improve=2.5000000, (0 missing)
##       chol     < 243.5 to the right, improve=1.0666670, (0 missing)
##       oldpeak  < 0.45  to the right, improve=1.0666670, (0 missing)
##       trestbps < 113.5 to the left,  improve=0.4454545, (0 missing)
##       thalach  < 159   to the left,  improve=0.3646465, (0 missing)
##   Surrogate splits:
##       chol     < 243.5 to the right, agree=0.90, adj=0.8, (0 split)
##       age      < 46.5  to the left,  agree=0.65, adj=0.3, (0 split)
##       thalach  < 143.5 to the right, agree=0.65, adj=0.3, (0 split)
##       trestbps < 109   to the right, agree=0.60, adj=0.2, (0 split)
##       oldpeak  < 0.25  to the right, agree=0.60, adj=0.2, (0 split)
##
## Node number 23: 16 observations
##   predicted class=1  expected loss=0.1875  P(node) =0.07729469
##     class counts:     3    13
##    probabilities: 0.188 0.813
##
## Node number 44: 10 observations
##   predicted class=0  expected loss=0.2  P(node) =0.04830918
##     class counts:     8     2
##    probabilities: 0.800 0.200
##
## Node number 45: 10 observations
##   predicted class=1  expected loss=0.3  P(node) =0.04830918
##     class counts:     3     7
##    probabilities: 0.300 0.700
```

```
t_pred = predict(tree_hrt_class,heart[validset,],type="class")
(confMat <- table(heart[validset,]$target,t_pred))
```

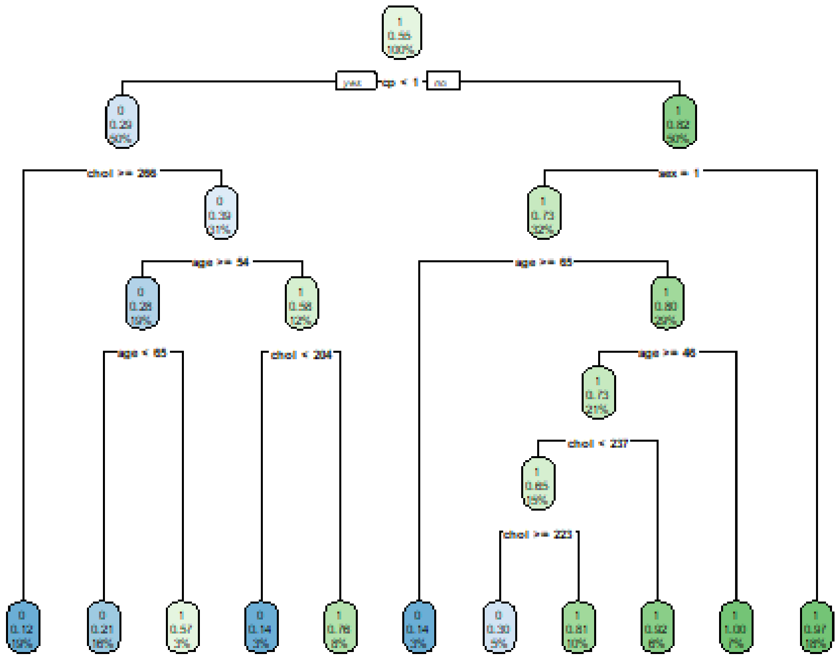
```
##    t_pred
##      0  1
##   0 26 17
##   1  9 37
```

```
(accuracy <- sum(diag(confMat))/sum(confMat))
```

```
## [1] 0.7078652
```

Early Warning Model

```
tree_hrt_early <- rpart(target ~age+sex+cp+chol, data = heart,method =
'class',  subset = trainset)
#tree_flight
rpart.plot(tree_hrt_early)
```



```r
summary(tree_hrt_early)

## Call:
## rpart(formula = target ~ age + sex + cp + chol, data = heart,
##     subset = trainset, method = "class")
##   n= 207
##
##           CP nsplit rel error    xerror       xstd
## 1 0.47311828      0 1.0000000 1.0000000 0.07695304
## 2 0.03225806      1 0.5268817 0.5268817 0.06575949
## 3 0.02688172      4 0.4301075 0.6021505 0.06872494
## 4 0.01433692      6 0.3763441 0.6129032 0.06910587
## 5 0.01075269      9 0.3333333 0.6021505 0.06872494
## 6 0.01000000     10 0.3225806 0.5698925 0.06751965
##
## Variable importance
##   cp chol  age  sex
##   41   26   26    7
##
## Node number 1: 207 observations,    complexity param=0.4731183
##   predicted class=1  expected loss=0.4492754  P(node) =1
##     class counts:    93   114
##    probabilities: 0.449 0.551
##   left son=2 (104 obs) right son=3 (103 obs)
##   Primary splits:
##       cp   < 0.5   to the left,  improve=28.752180, (0 missing)
##       sex  < 0.5   to the right, improve= 8.291649, (0 missing)
```

```
##       age  < 55.5  to the right, improve= 6.257785, (0 missing)
##       chol < 273.5 to the right, improve= 4.741074, (0 missing)
##   Surrogate splits:
##       age  < 52.5  to the right, agree=0.585, adj=0.165, (0 split)
##       chol < 246.5 to the right, agree=0.585, adj=0.165, (0 split)
##       sex  < 0.5   to the right, agree=0.541, adj=0.078, (0 split)
##
## Node number 2: 104 observations,    complexity param=0.03225806
##   predicted class=0  expected loss=0.2884615  P(node) =0.5024155
##     class counts:    74    30
##    probabilities: 0.712 0.288
##   left son=4 (40 obs) right son=5 (64 obs)
##   Primary splits:
##       chol < 265.5 to the right, improve=3.473558, (0 missing)
##       sex  < 0.5   to the right, improve=3.036216, (0 missing)
##       age  < 53.5  to the right, improve=1.570888, (0 missing)
##
## Node number 3: 103 observations,    complexity param=0.02688172
##   predicted class=1  expected loss=0.184466  P(node) =0.4975845
##     class counts:    19    84
##    probabilities: 0.184 0.816
##   left son=6 (66 obs) right son=7 (37 obs)
##   Primary splits:
##       sex  < 0.5   to the right, improve=2.8625270, (0 missing)
##       age  < 56.5  to the right, improve=2.7819580, (0 missing)
##       chol < 223   to the right, improve=0.6864153, (0 missing)
##       cp   < 1.5   to the right, improve=0.2810366, (0 missing)
##   Surrogate splits:
##       chol < 264   to the left,  agree=0.718, adj=0.216, (0 split)
##       age  < 62.5  to the left,  agree=0.670, adj=0.081, (0 split)
##
## Node number 4: 40 observations
##   predicted class=0  expected loss=0.125  P(node) =0.1932367
##     class counts:    35     5
##    probabilities: 0.875 0.125
##
## Node number 5: 64 observations,    complexity param=0.03225806
##   predicted class=0  expected loss=0.390625  P(node) =0.3091787
##     class counts:    39    25
##    probabilities: 0.609 0.391
##   left son=10 (40 obs) right son=11 (24 obs)
##   Primary splits:
##       age  < 53.5  to the right, improve=2.852083, (0 missing)
##       sex  < 0.5   to the right, improve=2.343750, (0 missing)
##       chol < 199.5 to the left,  improve=1.420441, (0 missing)
##   Surrogate splits:
##       chol < 259.5 to the left,  agree=0.688, adj=0.167, (0 split)
##
## Node number 6: 66 observations,    complexity param=0.02688172
##   predicted class=1  expected loss=0.2727273  P(node) =0.3188406
```

```
##     class counts:    18    48
##    probabilities: 0.273 0.727
##   left son=12 (7 obs) right son=13 (59 obs)
##   Primary splits:
##       age  < 64.5  to the right, improve=5.3488880, (0 missing)
##       chol < 223   to the right, improve=1.4518640, (0 missing)
##       cp   < 1.5   to the right, improve=0.5454545, (0 missing)
##
## Node number 7: 37 observations
##   predicted class=1  expected loss=0.02702703  P(node) =0.178744
##     class counts:     1    36
##    probabilities: 0.027 0.973
##
## Node number 10: 40 observations,    complexity param=0.01075269
##   predicted class=0  expected loss=0.275  P(node) =0.1932367
##     class counts:    29    11
##    probabilities: 0.725 0.275
##   left son=20 (33 obs) right son=21 (7 obs)
##   Primary splits:
##       age  < 64.5  to the left,  improve=1.4911260, (0 missing)
##       sex  < 0.5   to the right, improve=0.9782132, (0 missing)
##       chol < 240   to the right, improve=0.8166667, (0 missing)
##
## Node number 11: 24 observations,    complexity param=0.03225806
##   predicted class=1  expected loss=0.4166667  P(node) =0.115942
##     class counts:    10    14
##    probabilities: 0.417 0.583
##   left son=22 (7 obs) right son=23 (17 obs)
##   Primary splits:
##       chol < 203.5 to the left,  improve=3.8347340, (0 missing)
##       age  < 44.5  to the left,  improve=0.6736597, (0 missing)
##   Surrogate splits:
##       age < 37.5  to the left,  agree=0.792, adj=0.286, (0 split)
##
## Node number 12: 7 observations
##   predicted class=0  expected loss=0.1428571  P(node) =0.03381643
##     class counts:     6     1
##    probabilities: 0.857 0.143
##
## Node number 13: 59 observations,    complexity param=0.01433692
##   predicted class=1  expected loss=0.2033898  P(node) =0.2850242
##     class counts:    12    47
##    probabilities: 0.203 0.797
##   left son=26 (44 obs) right son=27 (15 obs)
##   Primary splits:
##       age  < 45.5  to the right, improve=1.6640990, (0 missing)
##       chol < 236.5 to the left,  improve=1.2739070, (0 missing)
##       cp   < 2.5   to the left,  improve=0.2574196, (0 missing)
##   Surrogate splits:
##       chol < 303   to the left,  agree=0.78, adj=0.133, (0 split)
```

```
##
## Node number 20: 33 observations
##   predicted class=0  expected loss=0.2121212  P(node) =0.1594203
##     class counts:    26     7
##    probabilities: 0.788 0.212
##
## Node number 21: 7 observations
##   predicted class=1  expected loss=0.4285714  P(node) =0.03381643
##     class counts:     3     4
##    probabilities: 0.429 0.571
##
## Node number 22: 7 observations
##   predicted class=0  expected loss=0.1428571  P(node) =0.03381643
##     class counts:     6     1
##    probabilities: 0.857 0.143
##
## Node number 23: 17 observations
##   predicted class=1  expected loss=0.2352941  P(node) =0.0821256
##     class counts:     4    13
##    probabilities: 0.235 0.765
##
## Node number 26: 44 observations,    complexity param=0.01433692
##   predicted class=1  expected loss=0.2727273  P(node) =0.2125604
##     class counts:    12    32
##    probabilities: 0.273 0.727
##   left son=52 (31 obs) right son=53 (13 obs)
##   Primary splits:
##       chol < 236.5 to the left,  improve=1.4148430, (0 missing)
##       age  < 50.5  to the left,  improve=1.3368980, (0 missing)
##       cp   < 2.5   to the left,  improve=0.2808003, (0 missing)
##
## Node number 27: 15 observations
##   predicted class=1  expected loss=0  P(node) =0.07246377
##     class counts:     0    15
##    probabilities: 0.000 1.000
##
## Node number 52: 31 observations,    complexity param=0.01433692
##   predicted class=1  expected loss=0.3548387  P(node) =0.1497585
##     class counts:    11    20
##    probabilities: 0.355 0.645
##   left son=104 (10 obs) right son=105 (21 obs)
##   Primary splits:
##       chol < 223   to the right, improve=3.5173580, (0 missing)
##       age  < 50.5  to the left,  improve=2.3364060, (0 missing)
##       cp   < 1.5   to the right, improve=0.2370266, (0 missing)
##   Surrogate splits:
##       age < 60.5  to the right, agree=0.742, adj=0.2, (0 split)
##
## Node number 53: 13 observations
##   predicted class=1  expected loss=0.07692308  P(node) =0.06280193
```

```
##     class counts:     1    12
##    probabilities: 0.077 0.923
##
## Node number 104: 10 observations
##   predicted class=0  expected loss=0.3  P(node) =0.04830918
##     class counts:     7     3
##    probabilities: 0.700 0.300
##
## Node number 105: 21 observations
##   predicted class=1  expected loss=0.1904762  P(node) =0.1014493
##     class counts:     4    17
##    probabilities: 0.190 0.810
```

```
t_pred = predict(tree_hrt_early,heart[validset,],type="class")
(confMat <- table(heart[validset,]$target,t_pred))
```

```
##    t_pred
##      0  1
##   0 18 25
##   1 11 35
```

```
(accuracy <- sum(diag(confMat))/sum(confMat))
```

```
## [1] 0.5955056
```

Naive Bayes

```
#Converting to categorical type
heart$sex <- as.factor(heart$sex)
heart$cp <- as.factor(heart$cp)
heart$fbs <- as.factor(heart$fbs)
heart$restecg <- as.factor(heart$restecg)
heart$exang <- as.factor(heart$exang)
heart$slope <- as.factor(heart$slope)
heart$ca <- as.factor(heart$ca)
heart$thal <- as.factor(heart$thal)
heart$target <- as.factor(heart$target)
```

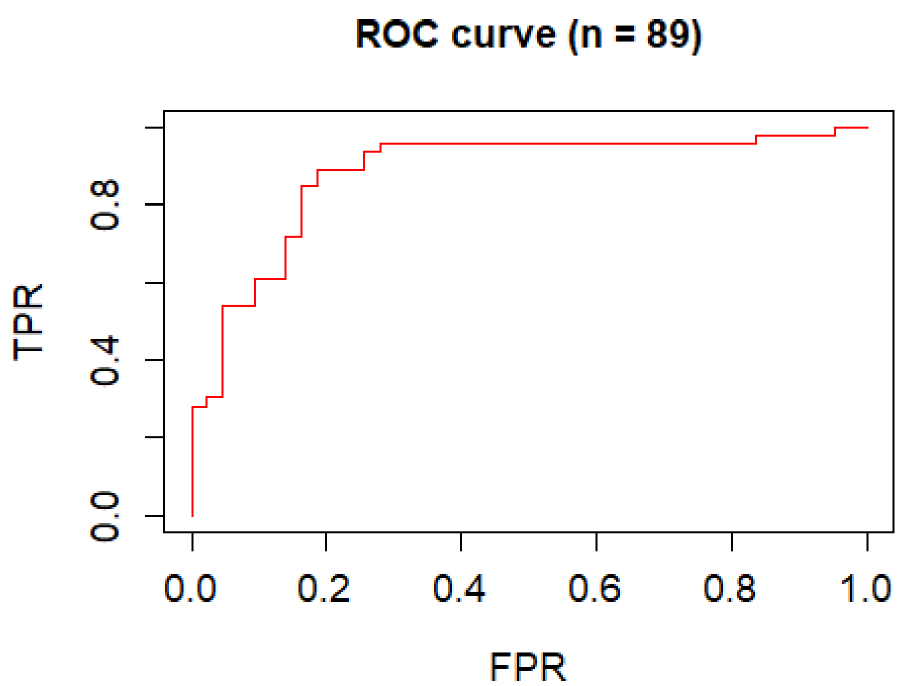
Confirmation Model

```
#Naive Bayes Model
nb_full <- naiveBayes(target~., data = heart,  subset = trainset)

#Calculate predictions
pred1 <- predict(nb_full, heart[validset,])

#Model accuracy
table(pred1, heart[validset,]$target,dnn = c('Pred','Actual'))
```

```
##     Actual
## Pred  0  1
```

```
##    0 34  5
##    1  9 41
```

```
pred1_raw <- predict(nb_full, heart[validset,],type='raw')
pred1_df <- data.frame(score = pred1_raw[,'1'], true.class =
ifelse(heart[validset,]$target == '1',1,0))
ROC_func(pred1_df,2,1, color = 'red')
```

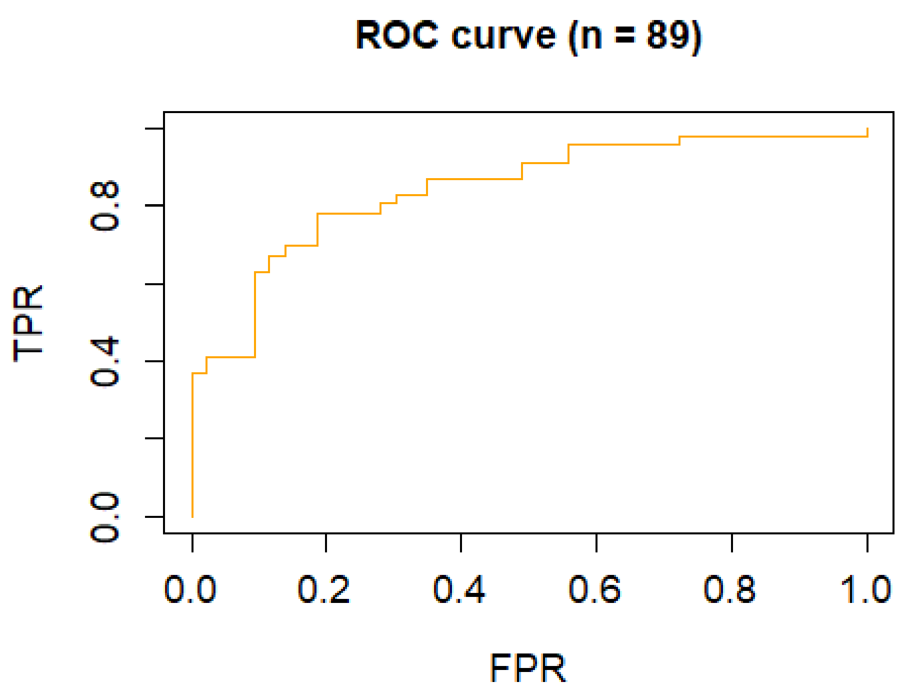



```
## [1] 0.8816987
```

Classification Model

```
#Create model
nb_classification <- naiveBayes(target~ age + sex + cp + trestbps + chol +
fbs + restecg + thalach + exang + oldpeak + slope, data = heart, subset =
trainset)

#calculate predictions
pred3 <- predict(nb_classification, heart[validset,])

#Model Accuracy
table(pred3, heart[validset,]$target, dnn = c('Pred','Actual'))

##     Actual
## Pred  0  1
##    0 30  8
##    1 13 38
```

```r
pred3_raw <- predict(nb_classification, heart[validset,],type='raw')
pred3_df <- data.frame(score = pred3_raw[,'1'], true.class =
ifelse(heart[validset,'target'] == '1',1,0))
ROC_func(pred3_df,2,1, color = 'orange') #removed add_on = T
```

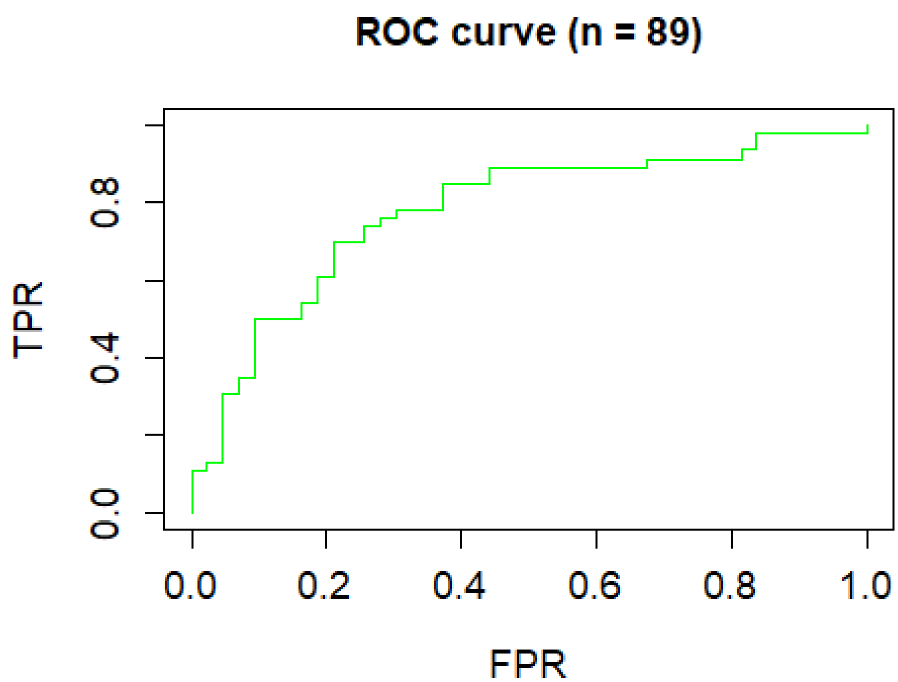



```
## [1] 0.8437816

#legend(
#   "bottomright",
#   lty=c(1,1,1),
#   col=c("red", "orange", "green"),
#   legend = c("Confirmation Subset", "Classification Subset", "Early Alarm
Subset")
#)
```

Early Warning Model

```r
#Create model
nb_early_alarm <- naiveBayes(target~ age + sex + cp + trestbps + chol + fbs,
data = heart, subset = trainset)

#calculate predictions
pred2 <- predict(nb_early_alarm, heart[validset,])

#Model Accuracy
table(pred2, heart[validset,]$target,dnn = c('Pred','Actual'))
```

```
##      Actual
## Pred  0  1
##    0 27  8
##    1 16 38

pred2_raw <- predict(nb_early_alarm, heart[validset,],type='raw')
pred2_df <- data.frame(score = pred2_raw[,'1'], true.class =
ifelse(heart[validset,'target'] == '1',1,0))
ROC_func(pred2_df,2,1,  color = 'green') #removed: add_on = T because it
didn't work
```




```
## [1] 0.7790698
```

Add something comparing all models together