Assessed Practicals 3

Praveen Gopal Reddy

13/08/2021

Table of Contents

```
# After importing csv file of human-heart.csv ,the data frame that we going
to use is: human_heart
summary(human_heart)
```

```
##       age             sex               cp           trestbps    
##  Min.   :29.00   Min.   :0.0000   Min.   :0.000   Min.   : 94.0  
##  1st Qu.:47.50   1st Qu.:0.0000   1st Qu.:0.000   1st Qu.:120.0  
##  Median :55.00   Median :1.0000   Median :1.000   Median :130.0  
##  Mean   :54.37   Mean   :0.6832   Mean   :0.967   Mean   :131.6  
##  3rd Qu.:61.00   3rd Qu.:1.0000   3rd Qu.:2.000   3rd Qu.:140.0  
##  Max.   :77.00   Max.   :1.0000   Max.   :3.000   Max.   :200.0  
##       chol            fbs            restecg             ca        
##  Min.   :126.0   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000  
##  1st Qu.:211.0   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000  
##  Median :240.0   Median :0.0000   Median :1.0000   Median :0.0000  
##  Mean   :246.3   Mean   :0.1485   Mean   :0.5281   Mean   :0.7294  
##  3rd Qu.:274.5   3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.:1.0000  
##  Max.   :564.0   Max.   :1.0000   Max.   :2.0000   Max.   :4.0000  
##       thal           target      
##  Min.   :0.000   Min.   :0.0000  
##  1st Qu.:2.000   1st Qu.:0.0000  
##  Median :2.000   Median :1.0000  
##  Mean   :2.314   Mean   :0.5446  
##  3rd Qu.:3.000   3rd Qu.:1.0000  
##  Max.   :3.000   Max.   :1.0000
```

Task 1

Fit Random Forest models using each possible input on its own to predict Hear disease. Evaluate the quality of fit by using the function to calculate the predicted class for each target (heart disease or no heart disease) (hint, you need type='response'). Which input fits best? (i.e. which classifies the most 'targets' correctly?)

```
library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

library(randomForest)

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin

#create train and test data. we have used 70/30 train and test split
set.seed(10)
inTrainRows <- createDataPartition(human_heart$target,p=0.7,list=FALSE)
trainData <- human_heart[inTrainRows,]
testData <-  human_heart[-inTrainRows,]

################build models on each input#########
trainData$target=as.factor(trainData$target)
accuracy = rep(NA, 9)
formulas = list(target ~ age, target ~ sex, target ~ cp,
             target ~ trestbps, target ~ chol, target ~ fbs,
             target ~ restecg, target ~ ca, target ~ thal)

#use loop over formulas
for (i in 1:length(formulas)){
 model <- randomForest(formulas[[i]],
                        data=trainData)

 model_prediction = predict(model,testData,type="response")
 model_accuracy=confusionMatrix(model_prediction,
as.factor(testData$target))$overall[1]
 accuracy[i] = model_accuracy
}
print(accuracy)

## [1] 0.6000000 0.6444444 0.8000000 0.5444444 0.5222222 0.4888889 0.6000000
## [8] 0.7444444 0.7777778
```

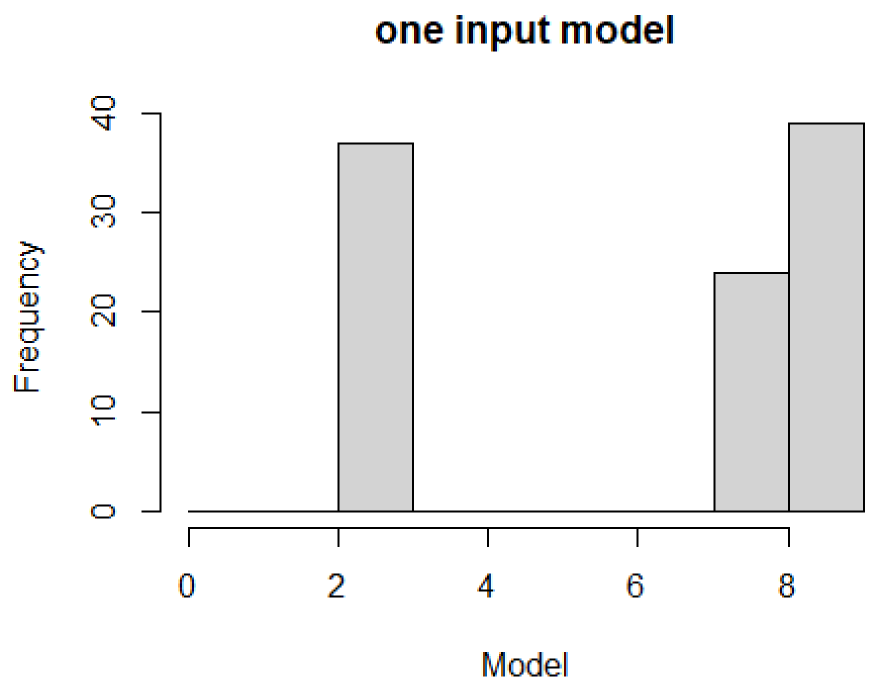
- we have built model for each individual input by using for loop. for loop will go via all individual inputs and then trains and predicts the model. we have used confusion matrix accuracy as metric to decide which input model classifies targets correctly.

Based on accuracy as you see from output, the index 9 accuracy is 77% which is 9th item in formulas variable. i.e. target ~ thal is the best predictor.

Task 2

Using cross-validation, perform a model selection to determine which features are useful for making predictions using a Random Forest. As above, use the number of 'targets' correctly classified as the criterion for deciding which model is best. You might try to find a way to loop over all possible models (ignore the possibility of no input variables. Hint: you can use in the package to generate all combinations of the numbers 1 to n). Or select features `greedily', by picking one at a time to add to the model. Present your results in the most convincing way you can.

```r
#one input models
accuracies_one_inputs = rep(NA, 9)
winner=rep(NA,100)
vec=vector("list",1)
formulas_one_inputs = list(target ~ age, target ~ sex, target ~ cp,
                target ~ trestbps, target ~ chol, target ~ fbs,
                target ~ restecg, target ~ ca, target ~ thal)

# cross-validation 100 times
for(iteration in 1:100){
  #prepare train and test set
  inTrainRows <- createDataPartition(human_heart$target,p=0.7,list=FALSE)
  trainData <- human_heart[inTrainRows,]
  testData <-  human_heart[-inTrainRows,]
  accuracies_one_inputs = rep(NA, 9)
  trainData$target=as.factor(trainData$target)
  #use for loop over formulas_one_inputs
  for (i in 1:length(formulas_one_inputs)){
    model_one_inputs <- randomForest(formulas_one_inputs[[i]],
                             data = trainData)

    prediction_one_inputs =
randomForest:::predict.randomForest(model_one_inputs,testData,type="response"
)

cf=confusionMatrix(prediction_one_inputs,as.factor(testData$target),positive=
"1")
    acc_one_input=cf$overall['Accuracy']*100
    accuracies_one_inputs[i] = acc_one_input
    #print(cf)
    #print(accuracies_one_inputs)

  }
  winner[iteration]=which.max(accuracies_one_inputs)
  vec[[iteration]]=data.frame(x=accuracies_one_inputs)
}
hist(winner, breaks=seq(0,9,1), xlab='Model', ylab='Frequency', main='one
input model')
```



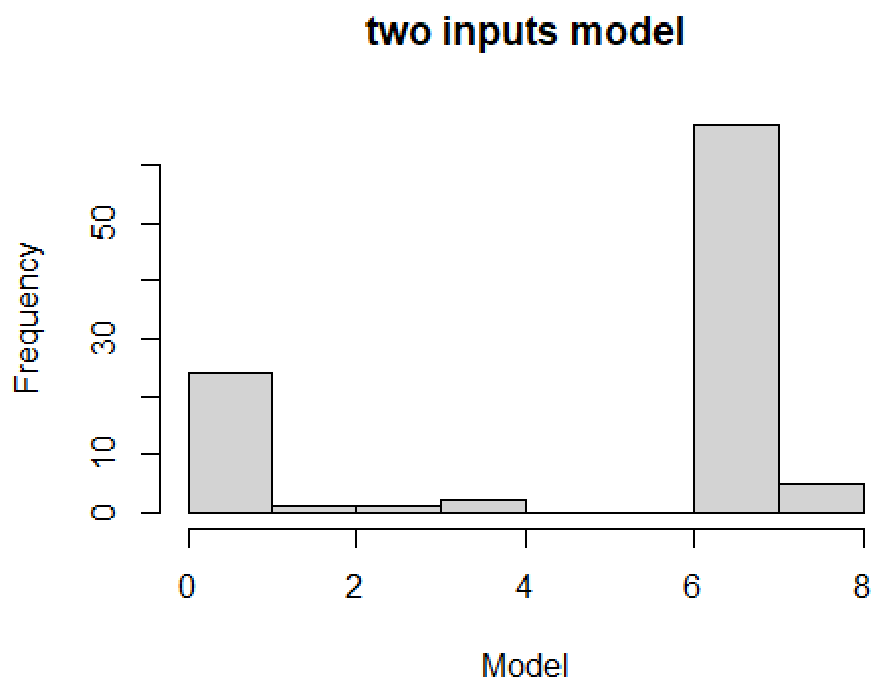
```
#model 9 is best model based on histogram

#find best model mean accuracy of 100 iterations
sum=0
for (i in 1:100)
{
  sum=sum+vec[[i]][9,1]
}
best_one_input_model_mean_accuracy=sum/100
best_one_input_model_mean_accuracy

## [1] 76.13333
```

from histogram output we can see that model 9 has won more times than other models with maximum accuracy over 100 iterations. * best model is target ~ thal. * best model mean accuracy is : 76%

so in next two input model we will keep thal as constant and add other inputs to it to create combinations.

```
#two input models
accuracies_two_inputs = rep(NA, 8)
winner=rep(NA,100)
```

```r
vec=vector("list",1)
formulas_two_inputs = list(target ~ thal+sex, target ~ thal+age,
                           target ~ thal+trestbps, target ~ thal+chol, target
~ thal+fbs,
                           target ~ thal+restecg, target ~ thal+ca, target ~
thal+cp)

# cross-validation 100 times
for(iteration in 1:100){
  #prepare train and test set
  inTrainRows <- createDataPartition(human_heart$target,p=0.7,list=FALSE)
  trainData <- human_heart[inTrainRows,]
  testData <-  human_heart[-inTrainRows,]
  accuracies_two_inputs = rep(NA, 8)
  trainData$target=as.factor(trainData$target)

  #use for loop over formulas_two_inputs
  for (i in 1:length(formulas_two_inputs)){
    model_two_inputs <- randomForest(formulas_two_inputs[[i]],
                                     data = trainData)

    prediction_two_inputs =
randomForest:::predict.randomForest(model_two_inputs,testData,type="response"
)

cf=confusionMatrix(prediction_two_inputs,as.factor(testData$target),positive=
"1")
    acc_two_input=cf$overall['Accuracy']*100
    accuracies_two_inputs[i] = acc_two_input
    #print(cf)
    #print(accuracies_two_inputs)

  }
  winner[iteration]=which.max(accuracies_two_inputs)
  vec[[iteration]]=data.frame(x=accuracies_two_inputs)
}
hist(winner, breaks=seq(0,8,1), xlab='Model', ylab='Frequency', main='two
inputs model')
```



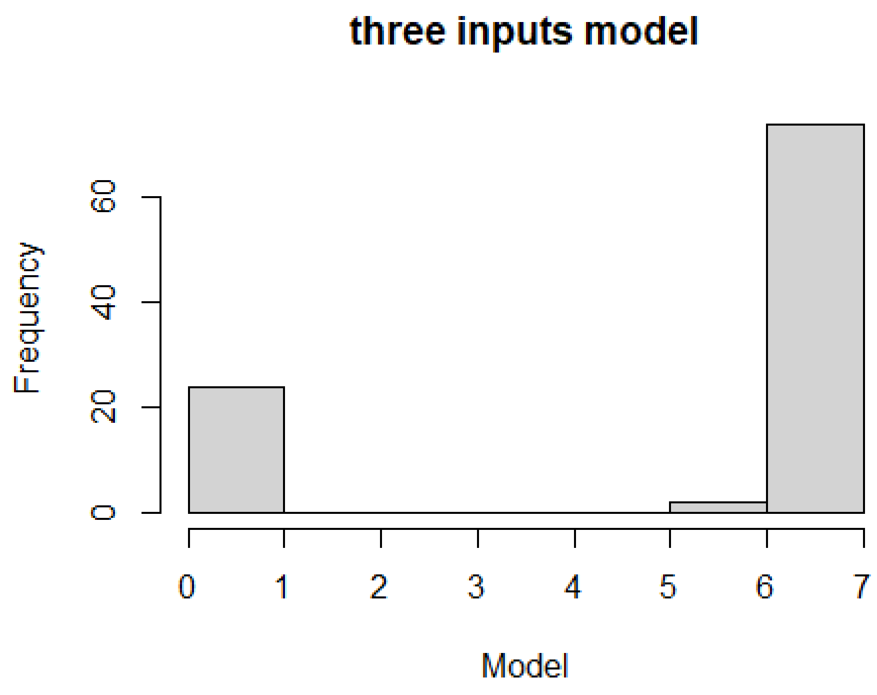
```
#model 7 is best model based on histogram
#find best model accuracy of 100 iterations
sum=0
for (i in 1:100)
{
  sum=sum+vec[[i]][7,1]
}
best_two_input_model_mean_accuracy=sum/100
best_two_input_model_mean_accuracy

## [1] 77.61111
```

- best two input model is target ~ thal+ca i.e model 7 from histogram output.
- best model mean accuracy is : 77%

In next three input model we will keep thal+ca as constant and add other inputs to it to create combinations.

```
#three input model
accuracies_three_inputs = rep(NA, 7)
winner=rep(NA,100)
vec=vector("list",1)
formulas_three_inputs = list(target ~ thal+ca+sex, target ~ thal+ca+age,
                             target ~ thal+ca+trestbps, target ~ thal+ca+chol,
target ~ thal+ca+fbs,target ~ thal+ca+restecg, target ~ thal+ca+cp)

# cross-validation 100 times
```

```
for(iteration in 1:100){
  inTrainRows <- createDataPartition(human_heart$target,p=0.7,list=FALSE)
  trainData <- human_heart[inTrainRows,]
  testData <-  human_heart[-inTrainRows,]
  accuracies_three_inputs = rep(NA, 7)
  trainData$target=as.factor(trainData$target)
  for (i in 1:length(formulas_three_inputs)){
    model_three_inputs <- randomForest(formulas_three_inputs[[i]],
                                 data = trainData)

    prediction_three_inputs =
randomForest:::predict.randomForest(model_three_inputs,testData,type="respons
e")

cf=confusionMatrix(prediction_three_inputs,as.factor(testData$target),positiv
e="1")
    acc_three_input=cf$overall['Accuracy']*100
    accuracies_three_inputs[i] = acc_three_input
    #print(cf)
    #print(accuracies_three_inputs)

  }
  winner[iteration]=which.max(accuracies_three_inputs)
  vec[[iteration]]=data.frame(x=accuracies_three_inputs)
}
hist(winner, breaks=seq(0,7,1), xlab='Model', ylab='Frequency', main='three
inputs model')
```



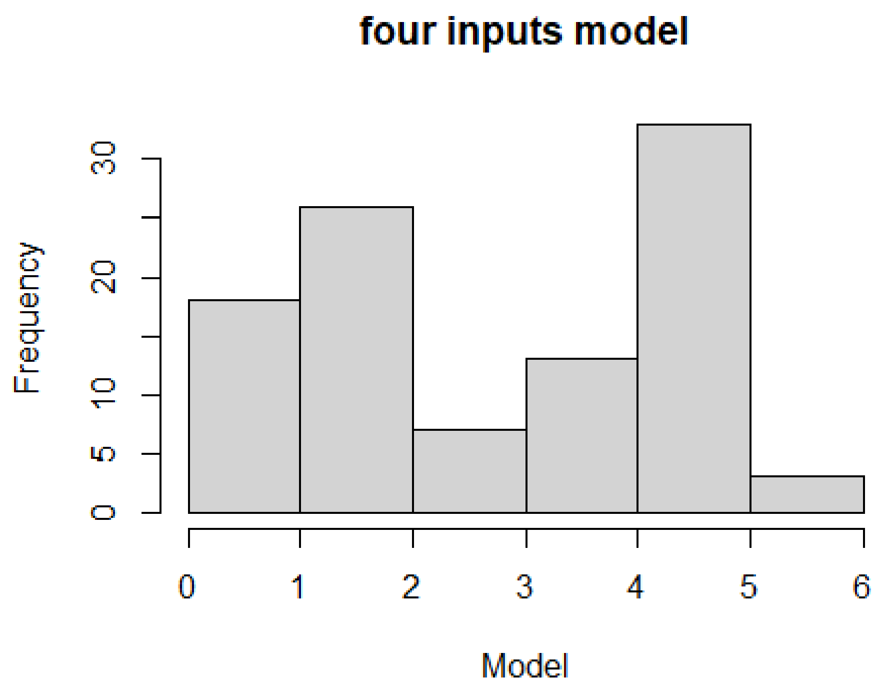
```
#model 7 is best model based on histogram
#find best model accuracy
sum=0
for (i in 1:100)
{
  sum=sum+vec[[i]][7,1]
}
best_three_input_model_mean_accuracy=sum/100
best_three_input_model_mean_accuracy

## [1] 83.27778
```

- best three input model is target ~ thal+ca+cp i.e model 7 from histogram output.
- best model mean accuracy is : 83%

In next four input model we will keep thal+ca+cp as constant and add other inputs to it to create combinations.

```
#four input model
accuracies_four_inputs = rep(NA, 6)
winner=rep(NA,100)
vec=vector("list",1)
formulas_four_inputs = list(target ~ thal+ca+cp+sex, target ~ thal+ca+cp+age,
target ~ thal+ca+cp+trestbps, target ~ thal+ca+cp+chol, target ~
thal+ca+cp+fbs, target ~ thal+ca+cp+restecg)

# cross-validation 100 times
```

```r
for(iteration in 1:100){
  inTrainRows <- createDataPartition(human_heart$target,p=0.7,list=FALSE)
  trainData <- human_heart[inTrainRows,]
  testData <-  human_heart[-inTrainRows,]
  accuracies_four_inputs = rep(NA, 6)
  trainData$target=as.factor(trainData$target)
  for (i in 1:length(formulas_four_inputs)){
    model_four_inputs <- randomForest(formulas_four_inputs[[i]],
                                      data = trainData)

    prediction_four_inputs =
randomForest:::predict.randomForest(model_four_inputs,testData,type="response
")

cf=confusionMatrix(prediction_four_inputs,as.factor(testData$target),positive
="1")
    acc_four_input=cf$overall['Accuracy']*100
    accuracies_four_inputs[i] = acc_four_input
    #print(cf)
    #print(accuracies_four_inputs)

  }
  winner[iteration]=which.max(accuracies_four_inputs)
  vec[[iteration]]=data.frame(x=accuracies_four_inputs)
}
hist(winner, breaks=seq(0,6,1), xlab='Model', ylab='Frequency', main='four
inputs model')
```



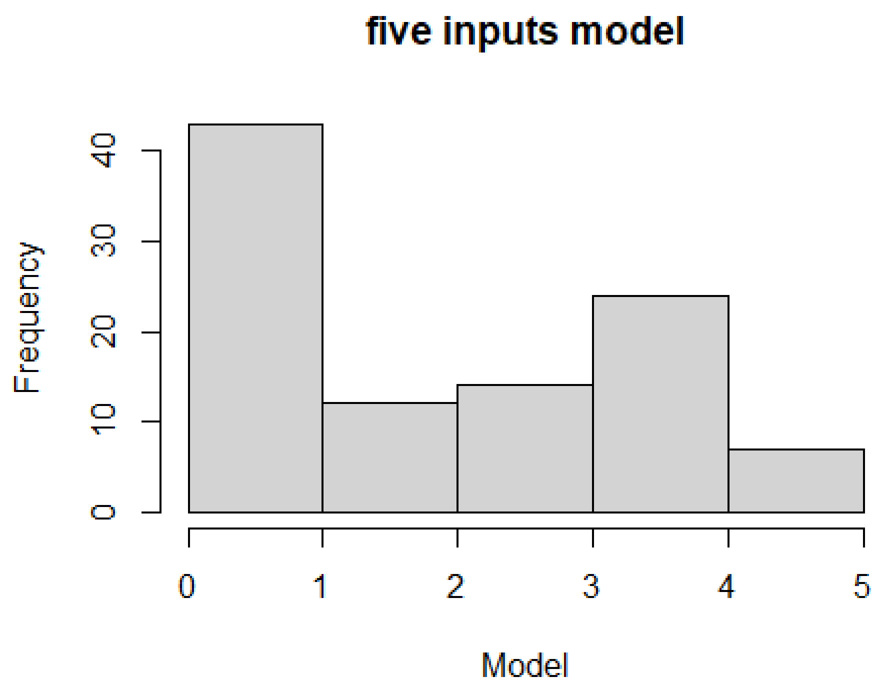
```
#model 2 is best model based on histogram
#find best model accuracy)
sum=0
for (i in 1:100)
{
  sum=sum+vec[[i]][5,1]
}
best_four_input_model_mean_accuracy=sum/100
best_four_input_model_mean_accuracy

## [1] 82.13333
```

- best four input model is target ~ thal+ca+cp+fbs i.e model 5 from histogram output.
- best model mean accuracy is : 82%

In next five input model we will keep thal+ca+cp+fbs as constant and add other inputs to it to create combinations.

```
#five input model
accuracies_five_inputs = rep(NA, 5)
winner=rep(NA,100)
vec=vector("list",1)
formulas_five_inputs = list(target ~ thal+ca+cp+fbs+age, target ~
thal+ca+cp+fbs+trestbps, target ~ thal+ca+cp+fbs+chol, target ~
thal+ca+cp+fbs+sex,target ~ thal+ca+cp+fbs+restecg)

# cross-validation 100 times
```

```
for(iteration in 1:100){
  inTrainRows <- createDataPartition(human_heart$target,p=0.7,list=FALSE)
  trainData <- human_heart[inTrainRows,]
  testData <-  human_heart[-inTrainRows,]
  accuracies_five_inputs = rep(NA, 5)
  trainData$target=as.factor(trainData$target)
  for (i in 1:length(formulas_five_inputs)){
    model_five_inputs <- randomForest(formulas_five_inputs[[i]],
                                      data = trainData)

    prediction_five_inputs =
randomForest:::predict.randomForest(model_five_inputs,testData,type="response
")

cf=confusionMatrix(prediction_five_inputs,as.factor(testData$target),positive
="1")
    acc_five_input=cf$overall['Accuracy']*100
    accuracies_five_inputs[i] = acc_five_input
    #print(cf)
    #print(accuracies_five_inputs)

  }
  winner[iteration]=which.max(accuracies_five_inputs)
  vec[[iteration]]=data.frame(x=accuracies_five_inputs)
}
hist(winner, breaks=seq(0,5,1), xlab='Model', ylab='Frequency', main='five
inputs model')
```



```
#model 1 is best model based on histogram
#find best model accuracy)
sum=0
for (i in 1:100)
{
  sum=sum+vec[[i]][1,1]
}
best_five_input_model_mean_accuracy=sum/100
best_five_input_model_mean_accuracy

## [1] 81.57778
```

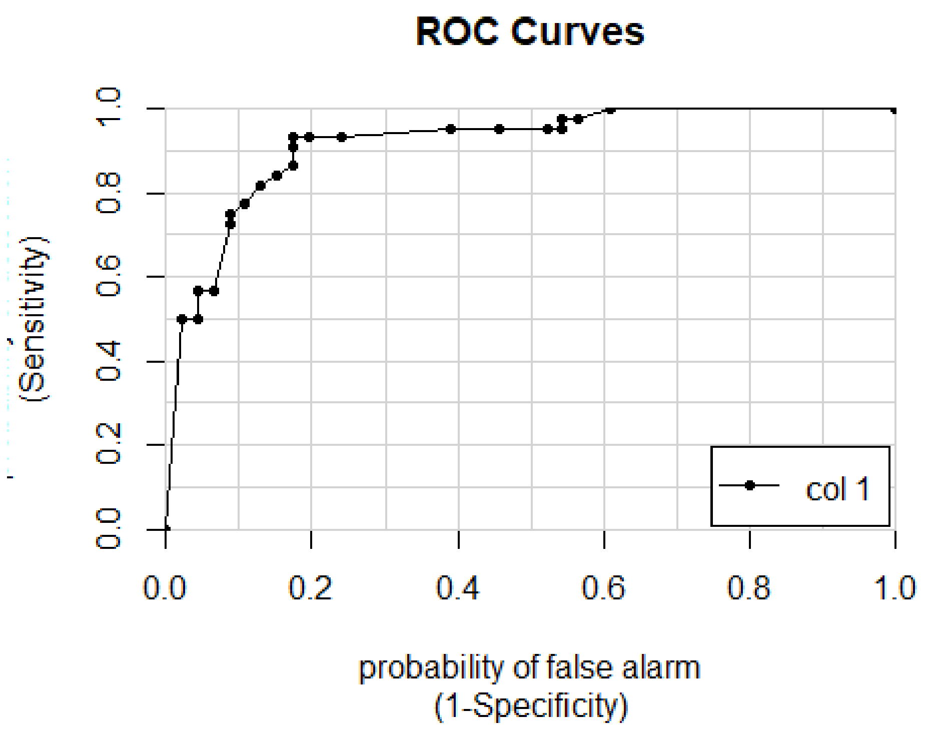
- best five input model is target ~ thal+ca+cp+fbs+age i.e model 1 from histogram output.
- best model mean accuracy is : 81%

we have stopped evaluating from six input model and further because accuracy keep reducing. we observed that target ~ thal+cp+ca input model got best mean accuracy of 83% from three input combinations.

Task 3

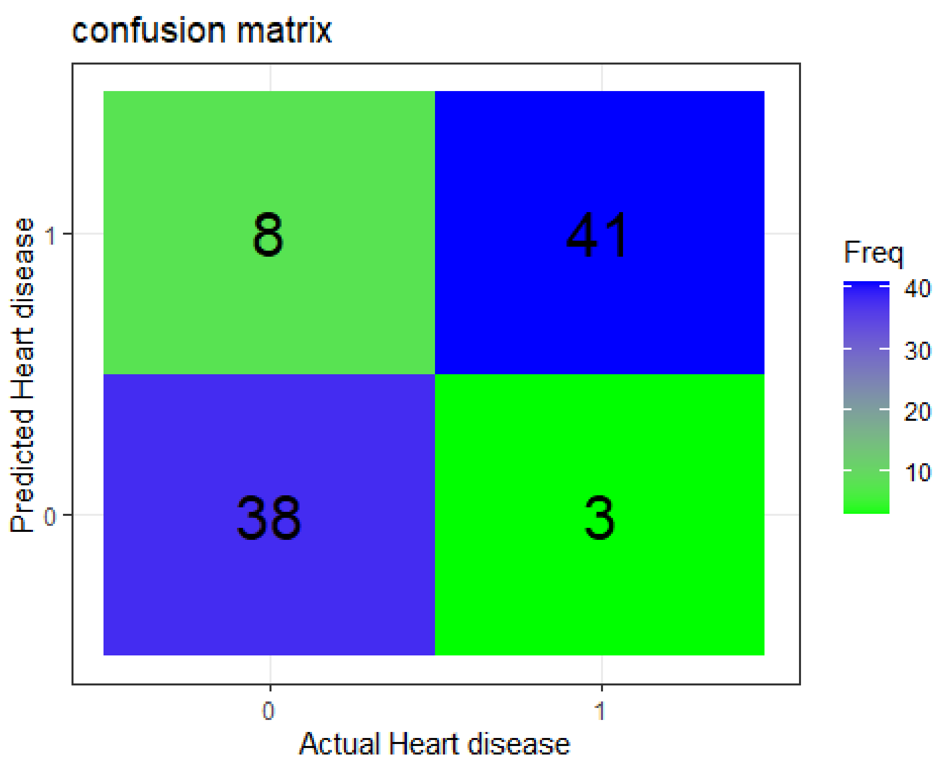
Would you use this classifier if you were diagnosing heart disease? Discuss with reference to factors that you identified as important and the probability of no heart disease.

```
# model on best factors
set.seed(10)
inTrainRows <- createDataPartition(human_heart$target,p=0.7,list=FALSE)
trainData <- human_heart[inTrainRows,]
testData <-  human_heart[-inTrainRows,]
trainData$target=as.factor(trainData$target)

best_model <- randomForest(target~thal+cp+ca,
                      data=trainData)

model_prediction = predict(best_model,testData,type="prob")
model_prediction=model_prediction[,2]

#plot roc curve
library(caTools)
colAUC(model_prediction, testData$target, plotROC = TRUE)
```



```
##              [,1]
## 0 vs. 1 0.9182312

#confusion matrix summary
pred_cf = predict(best_model,testData,type="response")
confusionMatrix(pred_cf, as.factor(testData$target))

## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 38  3
##          1  8 41
##
##                Accuracy : 0.8778
##                  95% CI : (0.7918, 0.9374)
##     No Information Rate : 0.5111
##     P-Value [Acc > NIR] : 1.735e-13
##
##                   Kappa : 0.756
##
##  Mcnemar's Test P-Value : 0.2278
##
##             Sensitivity : 0.8261
##             Specificity : 0.9318
##          Pos Pred Value : 0.9268
##          Neg Pred Value : 0.8367
##              Prevalence : 0.5111
##          Detection Rate : 0.4222
##    Detection Prevalence : 0.4556
##       Balanced Accuracy : 0.8790
##
##        'Positive' Class : 0
##
```

```
#plot confusion matrix graphically
library(ggthemes)
t<-table(pred_cf, testData$target)
df<-as.data.frame(t)
plot_rf =ggplot(data = df, aes(x = Var2, y = pred_cf, label=Freq)) +
  geom_tile(aes(fill = Freq)) +
  scale_fill_gradient(low="green", high="blue") +
  theme_economist()+
  xlab("Actual Heart disease") +
  ylab("Predicted Heart disease") +
  geom_text(size=8) +
  ggtitle("confusion matrix")
plot_rf + theme_bw()
```



- from ROC curve it is clear that model performs good without much discrimination of classes with AUC score of 0.91%.
- From confusion matrix we can see that specificity is 93% in which our model correctly identify patients without the disease.
- And Sensitivity is 82% in which model correctly identify patients with a disease.