code.R

#Download the data  
data <- read.csv("C:/Users/Acer\_302/Downloads//heart.csv")  
  
#See the provided data  
str(data)

## 'data.frame': 303 obs. of 14 variables:  
## $ п.їage : int 63 37 41 56 57 57 56 44 52 57 ...  
## $ sex : int 1 1 0 1 0 1 0 1 1 1 ...  
## $ cp : int 3 2 1 1 0 0 1 1 2 2 ...  
## $ trestbps: int 145 130 130 120 120 140 140 120 172 150 ...  
## $ chol : int 233 250 204 236 354 192 294 263 199 168 ...  
## $ fbs : int 1 0 0 0 0 0 0 0 1 0 ...  
## $ restecg : int 0 1 0 1 1 1 0 1 1 1 ...  
## $ thalach : int 150 187 172 178 163 148 153 173 162 174 ...  
## $ exang : int 0 0 0 0 1 0 0 0 0 0 ...  
## $ oldpeak : num 2.3 3.5 1.4 0.8 0.6 0.4 1.3 0 0.5 1.6 ...  
## $ slope : int 0 0 2 2 2 1 1 2 2 2 ...  
## $ ca : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ thal : int 1 2 2 2 2 1 2 3 3 2 ...  
## $ target : int 1 1 1 1 1 1 1 1 1 1 ...

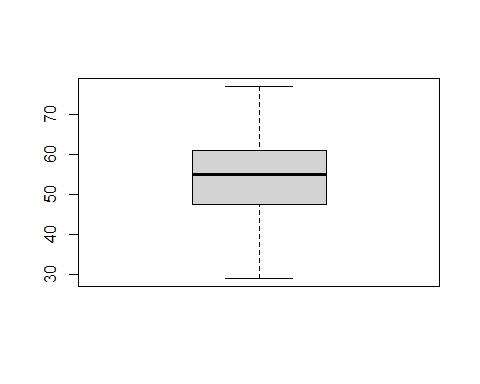
#Change name to more understandable  
names(data)[1] <- "age"  
  
#In my analysis I well use all 14 of provided variables  
#Change the class of target variable to factor   
data$target <- as.factor(data$target)  
  
# Dealing with NAs  
colSums(is.na(data))

## age sex cp trestbps chol fbs restecg thalach   
## 0 0 0 0 0 0 0 0   
## exang oldpeak slope ca thal target   
## 0 0 0 0 0 0

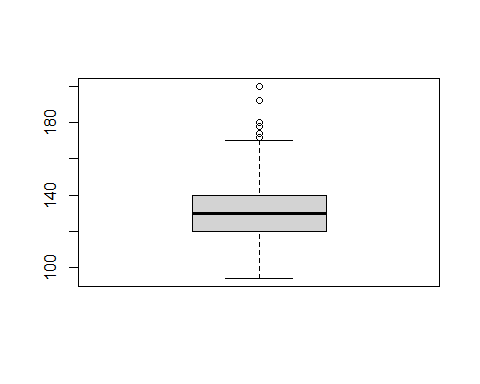
#There are no NAs in the data  
  
#Is the data imbalanced  
table(data$target)

##   
## 0 1   
## 138 165

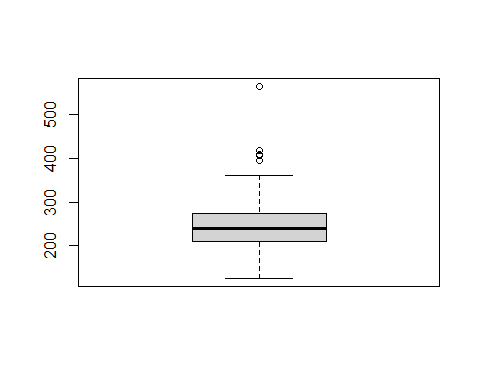
#No, data is well balanced  
  
#Check for outliers:  
boxplot(data$age)



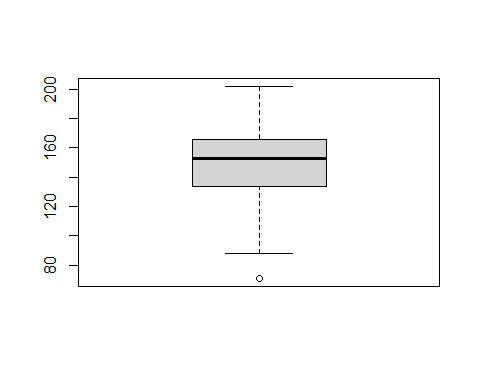
boxplot(data$trestbps)



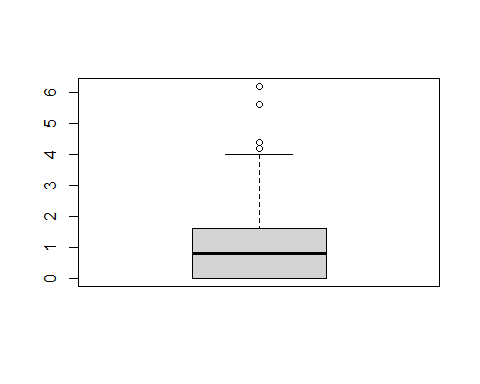
boxplot(data$chol)



boxplot(data$thalach)



boxplot(data$oldpeak)  
  
#There some outliers in these variables: trestbps, chol, thalach, oldpeak  
#Handling the outliers  
library("DescTools")



data$trestbps <- Winsorize(data$trestbps)  
data$chol <- Winsorize(data$chol)  
data$thalach <- Winsorize(data$thalach)  
data$oldpeak <- Winsorize(data$oldpeak)  
  
#Splitting the data into training and tasting sets  
set.seed(111)  
index\_train = sample(1:nrow(data), 2 / 3 \* nrow(data))  
training\_set = data[index\_train, ]  
test\_set = data[-index\_train, ]  
  
#Estimating the logistic model  
logmodel <- glm(target ~ ., family = binomial(link = "logit"), data = training\_set)  
summary(logmodel)

##   
## Call:  
## glm(formula = target ~ ., family = binomial(link = "logit"),   
## data = training\_set)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.7578 -0.2917 0.1533 0.5644 2.6138   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 2.293861 3.586045 0.640 0.522391   
## age 0.006564 0.029331 0.224 0.822907   
## sex -1.484051 0.589986 -2.515 0.011890 \*   
## cp 1.033245 0.263891 3.915 9.02e-05 \*\*\*  
## trestbps -0.022559 0.015832 -1.425 0.154189   
## chol -0.002585 0.005918 -0.437 0.662291   
## fbs -0.454689 0.666443 -0.682 0.495072   
## restecg 0.239588 0.456064 0.525 0.599348   
## thalach 0.028315 0.014613 1.938 0.052653 .   
## exang -0.857012 0.527883 -1.623 0.104485   
## oldpeak -0.835458 0.288897 -2.892 0.003829 \*\*   
## slope 0.544995 0.448186 1.216 0.223984   
## ca -0.906069 0.237254 -3.819 0.000134 \*\*\*  
## thal -0.986862 0.379433 -2.601 0.009298 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 278.05 on 201 degrees of freedom  
## Residual deviance: 131.67 on 188 degrees of freedom  
## AIC: 159.67  
##   
## Number of Fisher Scoring iterations: 6

#Prediction with the cutoff point equals 0.3  
test\_set$log\_pred = predict(logmodel, newdata = test\_set, type = 'response')  
test\_set$log\_pred = ifelse(test\_set$log\_pred < 0.3, 0, 1)  
  
#Evaluating the quality of the model (confusion matrix)  
library('caret')

## Loading required package: lattice

## Loading required package: ggplot2

##   
## Attaching package: 'caret'

## The following objects are masked from 'package:DescTools':  
##   
## MAE, RMSE

test\_set$target = factor(test\_set$target)  
test\_set$log\_pred = factor(test\_set$log\_pred)  
mat1 <- confusionMatrix(test\_set$log\_pred, test\_set$target)  
mat1

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 31 5  
## 1 16 49  
##   
## Accuracy : 0.7921   
## 95% CI : (0.6999, 0.8664)  
## No Information Rate : 0.5347   
## P-Value [Acc > NIR] : 6.755e-08   
##   
## Kappa : 0.5757   
##   
## Mcnemar's Test P-Value : 0.0291   
##   
## Sensitivity : 0.6596   
## Specificity : 0.9074   
## Pos Pred Value : 0.8611   
## Neg Pred Value : 0.7538   
## Prevalence : 0.4653   
## Detection Rate : 0.3069   
## Detection Prevalence : 0.3564   
## Balanced Accuracy : 0.7835   
##   
## 'Positive' Class : 0   
##

#Evaluating the quality of the model (AUC)  
library(pROC)

## Type 'citation("pROC")' for a citation.

##   
## Attaching package: 'pROC'

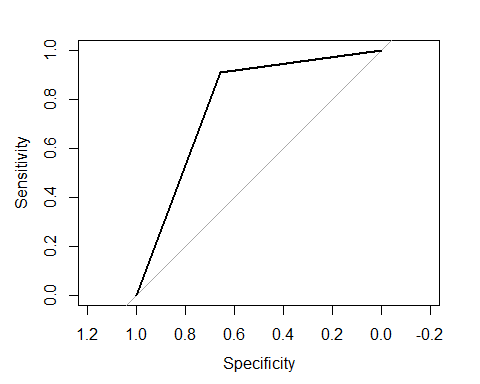
## The following objects are masked from 'package:stats':  
##   
## cov, smooth, var

ROC\_log1 = roc(as.numeric(test\_set$target), as.numeric(test\_set$log\_pred))

## Setting levels: control = 1, case = 2

## Setting direction: controls < cases

plot(ROC\_log1)



auc(ROC\_log1)

## Area under the curve: 0.7835

#choice of the cutoff  
library(e1071)  
library(tidyr)  
cutoffs\_vec = c()  
accur\_vec = c()  
sens\_vec = c()  
spec\_vec = c()  
  
test\_set$log\_pred = predict(logmodel, newdata = test\_set, type = 'response')  
for (i in seq(0, 1, by = 0.05)) {  
 print(i)  
 cutoffs\_vec = c(cutoffs\_vec, i)  
 test\_set$cutoff2 = ifelse(test\_set$log\_pred < i, 0, 1)  
 test\_set$cutoff2 = factor(test\_set$cutoff2)  
 mat1 = confusionMatrix(test\_set$cutoff2, test\_set$target)  
 accur\_vec = c(accur\_vec, mat1$overall[3])  
 sens\_vec = c(sens\_vec, mat1$byClass[1])  
 spec\_vec = c(spec\_vec, mat1$byClass[2])  
   
}

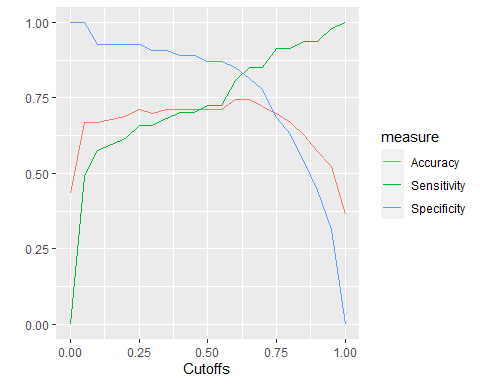
## [1] 0

## Warning in confusionMatrix.default(test\_set$cutoff2, test\_set$target): Levels  
## are not in the same order for reference and data. Refactoring data to match.

## [1] 0.05  
## [1] 0.1  
## [1] 0.15  
## [1] 0.2  
## [1] 0.25  
## [1] 0.3  
## [1] 0.35  
## [1] 0.4  
## [1] 0.45  
## [1] 0.5  
## [1] 0.55  
## [1] 0.6  
## [1] 0.65  
## [1] 0.7  
## [1] 0.75  
## [1] 0.8  
## [1] 0.85  
## [1] 0.9  
## [1] 0.95  
## [1] 1

## Warning in confusionMatrix.default(test\_set$cutoff2, test\_set$target): Levels  
## are not in the same order for reference and data. Refactoring data to match.

d\_loop = data.frame(cutoffs\_vec, accur\_vec, sens\_vec, spec\_vec)  
d\_loop\_long = gather(d\_loop, key = measure, value = value, -cutoffs\_vec)  
d\_loop\_long$measure = factor(d\_loop\_long$measure, levels=c("accur\_vec", "sens\_vec", "spec\_vec"),   
   
 labels=c("Accuracy", "Sensitivity", "Specificity"))  
  
ggplot(data = d\_loop\_long, aes(x = cutoffs\_vec, y = value, col = measure)) +  
 geom\_line() +  
 xlab("Cutoffs") +  
 ylab("")



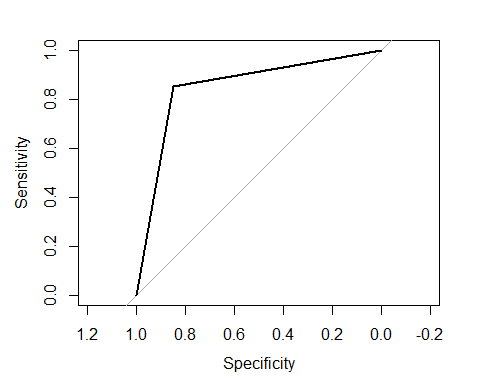
#According to the graph, the best cutoff point is equal 0.625  
test\_set$log\_pred2 = predict(logmodel, newdata = test\_set, type = 'response')  
test\_set$log\_pred2 = ifelse(test\_set$log\_pred2 < 0.625, 0, 1)  
  
  
#Evaluating the quality of the model with new cutoff point (confusion matrix)  
test\_set$target = factor(test\_set$target)  
test\_set$log\_pred2 = factor(test\_set$log\_pred2)  
mat2 <- confusionMatrix(test\_set$log\_pred2, test\_set$target)  
mat2

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 40 8  
## 1 7 46  
##   
## Accuracy : 0.8515   
## 95% CI : (0.7669, 0.9144)  
## No Information Rate : 0.5347   
## P-Value [Acc > NIR] : 1.579e-11   
##   
## Kappa : 0.7019   
##   
## Mcnemar's Test P-Value : 1   
##   
## Sensitivity : 0.8511   
## Specificity : 0.8519   
## Pos Pred Value : 0.8333   
## Neg Pred Value : 0.8679   
## Prevalence : 0.4653   
## Detection Rate : 0.3960   
## Detection Prevalence : 0.4752   
## Balanced Accuracy : 0.8515   
##   
## 'Positive' Class : 0   
##

#Evaluating the quality of the model with new cutoff point (AUC)  
ROC\_log2 = roc(as.numeric(test\_set$target), as.numeric(test\_set$log\_pred2))

## Setting levels: control = 1, case = 2  
## Setting direction: controls < cases

plot(ROC\_log2)



auc(ROC\_log2)

## Area under the curve: 0.8515

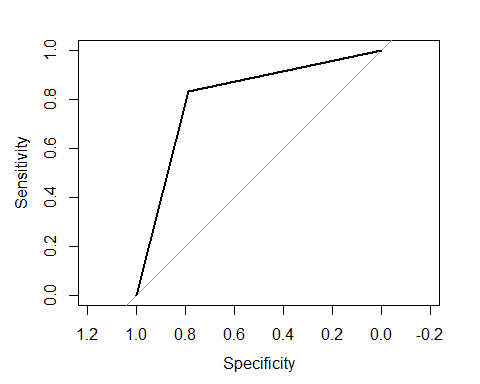
#Random forest without cross validation  
rf\_tt = train(target ~ ., method= "rf", ntree = 100, data = training\_set)  
test\_set$rf\_tt <- predict(rf\_tt, test\_set)  
  
#Evaluating the quality of the model (confusion matrix)  
test\_set$target = factor(test\_set$target)  
test\_set$rf\_tt = factor(test\_set$rf\_tt)  
mat3 <- confusionMatrix(test\_set$rf\_tt, test\_set$target)  
mat3

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 37 9  
## 1 10 45  
##   
## Accuracy : 0.8119   
## 95% CI : (0.7219, 0.8828)  
## No Information Rate : 0.5347   
## P-Value [Acc > NIR] : 5.384e-09   
##   
## Kappa : 0.6214   
##   
## Mcnemar's Test P-Value : 1   
##   
## Sensitivity : 0.7872   
## Specificity : 0.8333   
## Pos Pred Value : 0.8043   
## Neg Pred Value : 0.8182   
## Prevalence : 0.4653   
## Detection Rate : 0.3663   
## Detection Prevalence : 0.4554   
## Balanced Accuracy : 0.8103   
##   
## 'Positive' Class : 0   
##

#Evaluating the quality of the model (AUC)  
ROC\_rf\_tt = roc(test\_set$target, as.numeric(test\_set$rf\_tt))

## Setting levels: control = 0, case = 1  
## Setting direction: controls < cases

plot(ROC\_rf\_tt)



auc(ROC\_rf\_tt)

## Area under the curve: 0.8103

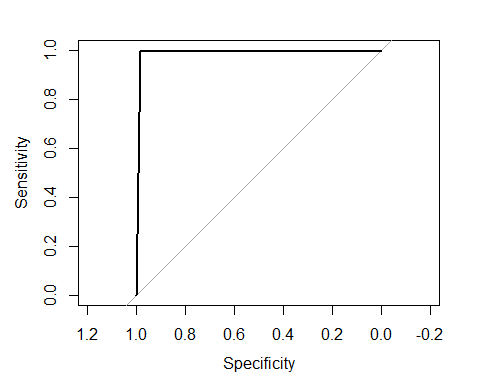
#Random forest with cross validation  
train.control = trainControl(method = "cv", number = 5)  
rf\_cv = train(target ~ ., method= "rf", ntree = 50, data = data, trControl = train.control)  
data$rf\_cv\_pred <- predict(rf\_cv, data)  
  
#Evaluating the quality of the model (confusion matrix)  
data$target = factor(data$target)  
data$rf\_cv\_pred = factor(data$rf\_cv\_pred)  
mat4 <- confusionMatrix(data$rf\_cv\_pred, data$target)  
mat4

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 136 1  
## 1 2 164  
##   
## Accuracy : 0.9901   
## 95% CI : (0.9713, 0.998)  
## No Information Rate : 0.5446   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.98   
##   
## Mcnemar's Test P-Value : 1   
##   
## Sensitivity : 0.9855   
## Specificity : 0.9939   
## Pos Pred Value : 0.9927   
## Neg Pred Value : 0.9880   
## Prevalence : 0.4554   
## Detection Rate : 0.4488   
## Detection Prevalence : 0.4521   
## Balanced Accuracy : 0.9897   
##   
## 'Positive' Class : 0   
##

#Evaluating the quality of the model (AUC)  
ROC\_rf\_cv\_pred = roc(data$target, as.numeric(data$rf\_cv\_pred))

## Setting levels: control = 0, case = 1  
## Setting direction: controls < cases

plot(ROC\_rf\_cv\_pred)



auc(ROC\_rf\_cv\_pred)

## Area under the curve: 0.9897