Heart Disease Prediction

Introduction:

This database contains 76 attributes, but all published experiments refer to using a subset of 14 of them.The "goal" field refers to the presence of heart disease in the patient. It is integer valued from 0 (no presence) to 4.

Attribute:

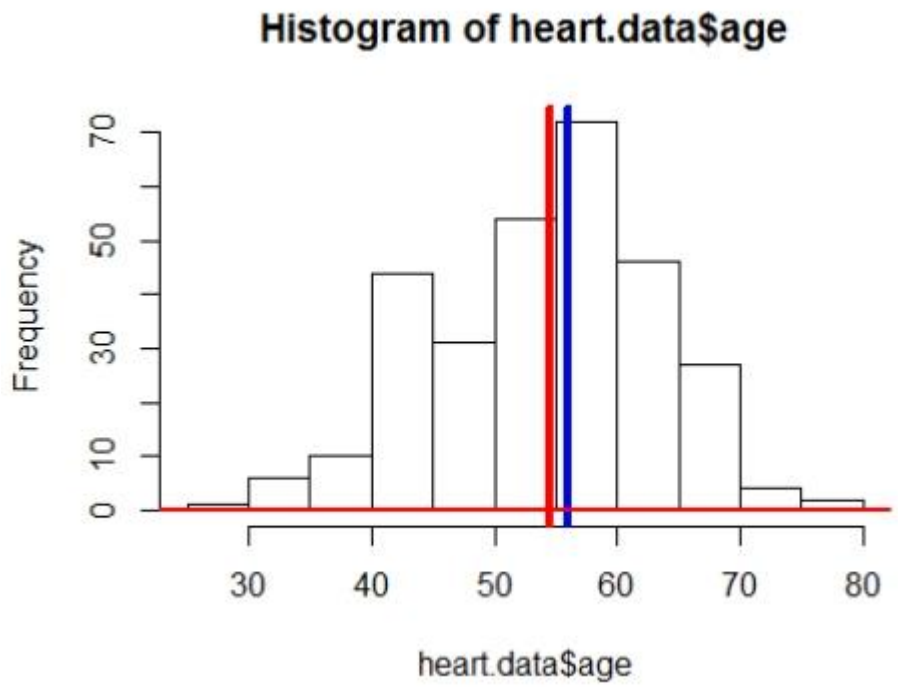
- age: age in years
- sex: (1 = male; 0 = female)
- cp: chest pain type
  - Value 1: typical angina
  - Value 2: atypical angina
  - Value 3: non-anginal pain
  - Value 4: asymptomatic
- trestbps: resting blood pressure (in mm Hg on admission to the hospital)
- chol: serum cholestoral in mg/dl
- fbs: (fasting blood sugar > 120 mg/dl)
  - 1 = true
  - 0 = false
- restecg: (resting electrocardiographic results)
  - Value 0: normal
  - Value 1: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV)
  - Value 2: showing probable or definite left ventricular hypertrophy by Estes' criteria
- thalach: maximum heart rate achieved
- exang: exercise induced angina
  - 1 = yes
  - 0 = no
- oldpeak: ST depression induced by exercise relative to rest
- slope: slope of the peak exercise ST segment

        - Value 1: upsloping
        - Value 2: flat
        - Value 3: downsloping
    - ca: number of major vessels (0-3) colored by flourosopy
    - thal: thalium heart scan
        - 3 = normal (no cold spots)
        - 6 = fixed defect (cold spots during rest and exercise)
        - 7 = reversible defect (when cold spots only appear during exercise)
    - pred_attribute: (the predicted attribute) diagnosis of heart disease (angiographic disease status)
        - Value 0: < 50% diameter narrowing
        - Value 1: > 50% diameter narrowing (in any major vessel: attributes 59 through 68 are vessels)

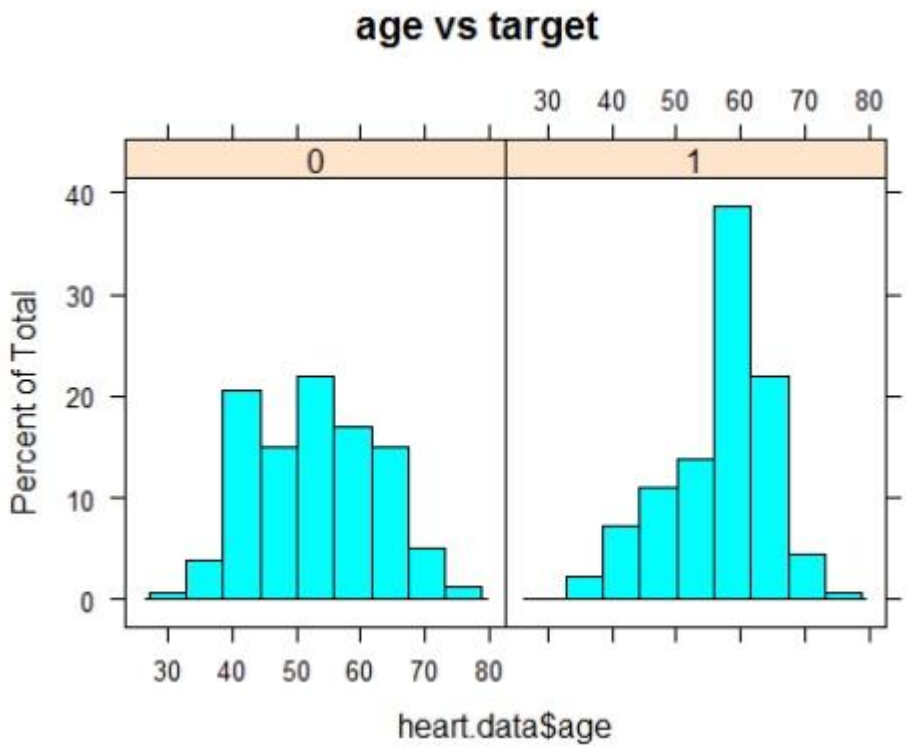
```
heart.data <- read.csv("https://archive.ics.uci.edu/ml/machine-learning-
databases/heartdisease/processed.cleveland.data",header=FALSE,sep=",",na.s
trings = '?')
head(heart.data,3)

##   V1 V2 V3  V4  V5 V6 V7  V8 V9 V10 V11 V12 V13 V14
## 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

names(heart.data) <- c( "age", "sex", "cp", "trestbps", "chol","fbs",
"restecg","thalach","exang", "oldpeak","slope", "ca", "thal", "num")
head(heart.data,3)

##   age sex cp trestbps chol fbs restecg thalach exang oldpeak slope ca
thal
## 1  63   1  1      145  233   1       2     150     0     2.3     3  0
6
## 2  67   1  4      160  286   0       2     108     1     1.5     2  3
3
## 3  67   1  4      120  229   0       2     129     1     2.6     2  2
7
##   num
## 1   0
## 2   2
## 3   1

str(heart.data)
```

```
## 'data.frame':    303 obs. of  14 variables:
##  $ age     : num  63 67 67 37 41 56 62 57 63 53 ...
##  $ sex     : num  1 1 1 1 0 1 0 0 1 1 ...
##  $ cp      : num  1 4 4 3 2 2 4 4 4 4 ...
##  $ trestbps: num  145 160 120 130 130 120 140 120 130 140 ...
##  $ chol    : num  233 286 229 250 204 236 268 354 254 203 ...
##  $ fbs     : num  1 0 0 0 0 0 0 0 0 1 ...
##  $ restecg : num  2 2 2 0 2 0 2 0 2 2 ...
##  $ thalach : num  150 108 129 187 172 178 160 163 147 155 ...
##  $ exang   : num  0 1 1 0 0 0 0 1 0 1 ...
##  $ oldpeak : num  2.3 1.5 2.6 3.5 1.4 0.8 3.6 0.6 1.4 3.1 ...
##  $ slope   : num  3 2 2 3 1 1 3 1 2 3 ...
##  $ ca      : num  0 3 2 0 0 0 2 0 1 0 ...
##  $ thal    : num  6 3 7 3 3 3 3 3 7 7 ...
##  $ num     : int  0 2 1 0 0 0 3 0 2 1 ...

heart.data$num[heart.data$num > 0] <- 1
head(heart.data,3)

##   age sex cp trestbps chol fbs restecg thalach exang oldpeak slope ca
thal
## 1  63   1  1      145  233   1       2     150     0     2.3     3  0
6
## 2  67   1  4      160  286   0       2     108     1     1.5     2  3
3
## 3  67   1  4      120  229   0       2     129     1     2.6     2  2
7
##   num
## 1   0
## 2   1
## 3   1

heart.data <- na.omit(heart.data)
sum(is.na(heart.data))

## [1] 0

anyNA(heart.data)

## [1] FALSE

set.seed(7)
library(mlbench)

## Warning: package 'mlbench' was built under R version 3.4.4

library(caret)

## Warning: package 'caret' was built under R version 3.4.4

## Loading required package: lattice

## Loading required package: ggplot2
```

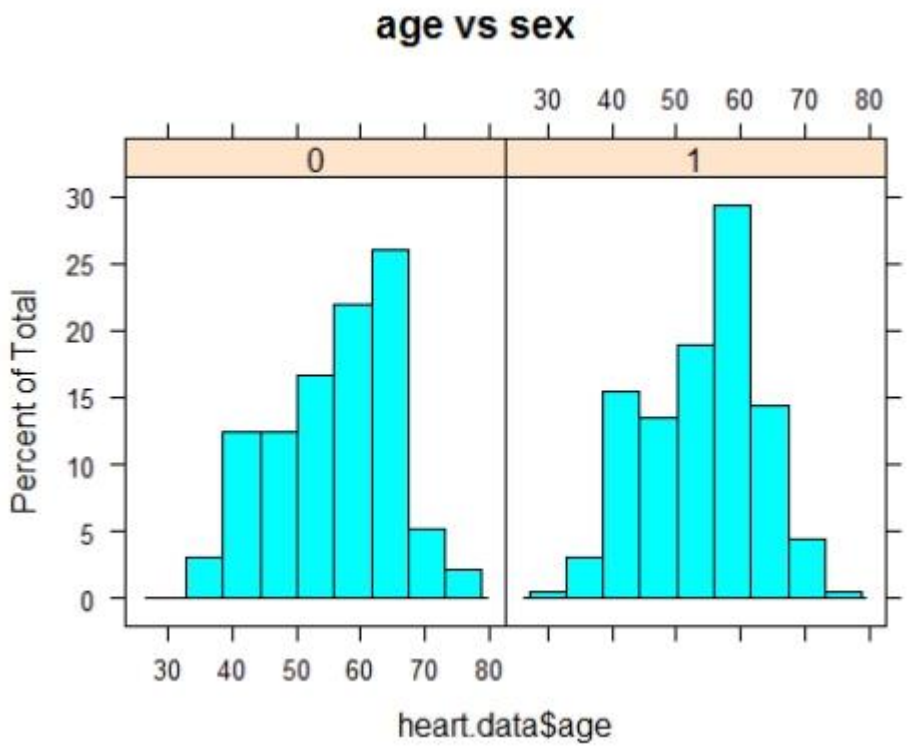
```
## Warning: package 'ggplot2' was built under R version 3.4.4

heart.data$age <- as.numeric(heart.data$age)
heart.data$sex <- as.factor(heart.data$sex)
heart.data$cp <- as.factor(heart.data$cp)
heart.data$trestbps <- as.numeric(heart.data$trestbps)
heart.data$chol <- as.numeric(heart.data$chol)
heart.data$fbs <- as.factor(heart.data$fbs)
heart.data$restecg <- as.factor(heart.data$restecg)
heart.data$thalach <- as.numeric(heart.data$thalach)
heart.data$exang <- as.factor(heart.data$exang)
heart.data$oldpeak <- as.numeric(heart.data$oldpeak)
heart.data$slope <- as.factor(heart.data$slope)
heart.data$ca <- as.numeric(heart.data$ca)
heart.data$thal <- as.factor(heart.data$thal)
heart.data$num <- as.factor(heart.data$num)

hist(heart.data$age)
abline(v=mean(heart.data$age),col="red",lwd=4)
abline(v=median(heart.data$age),col="blue",lwd=4)
lines(density(heart.data$age),col = 2 , lwd = 2)
```



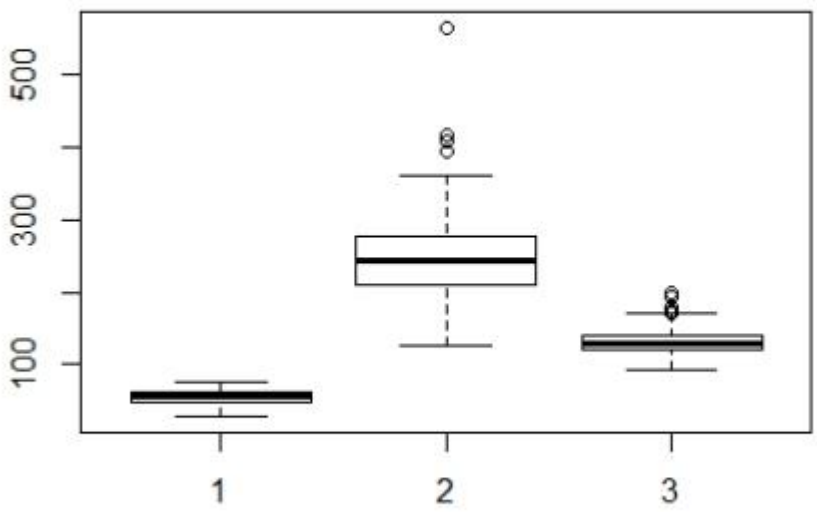

```
library(lattice)
histogram(~heart.data$age|factor(heart.data$num),data = heart.data,main =
"age vs target")
```



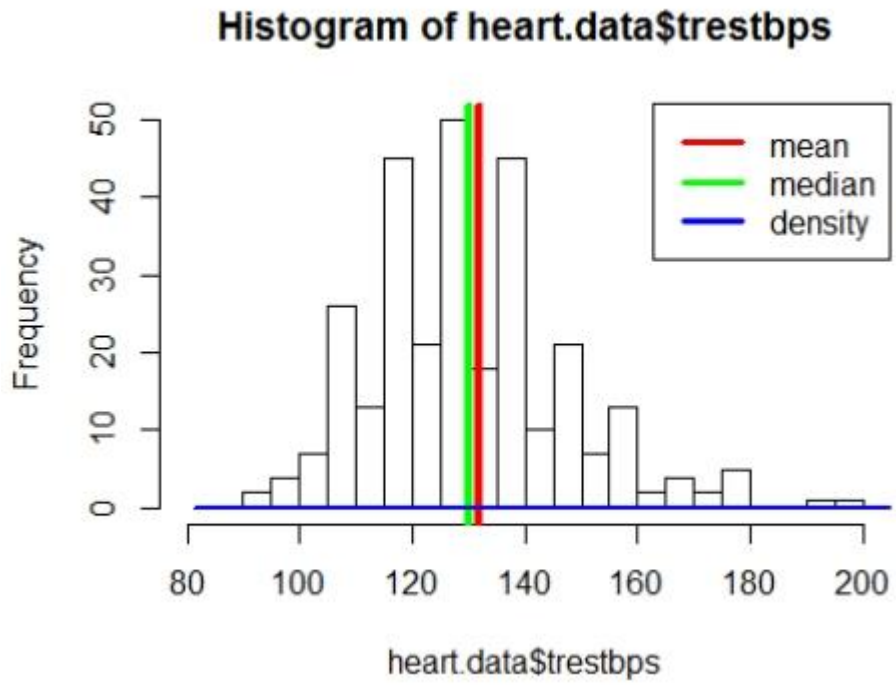

```
histogram(~heart.data$age|factor(heart.data$sex),data = heart.data,main =
"age vs sex")
```



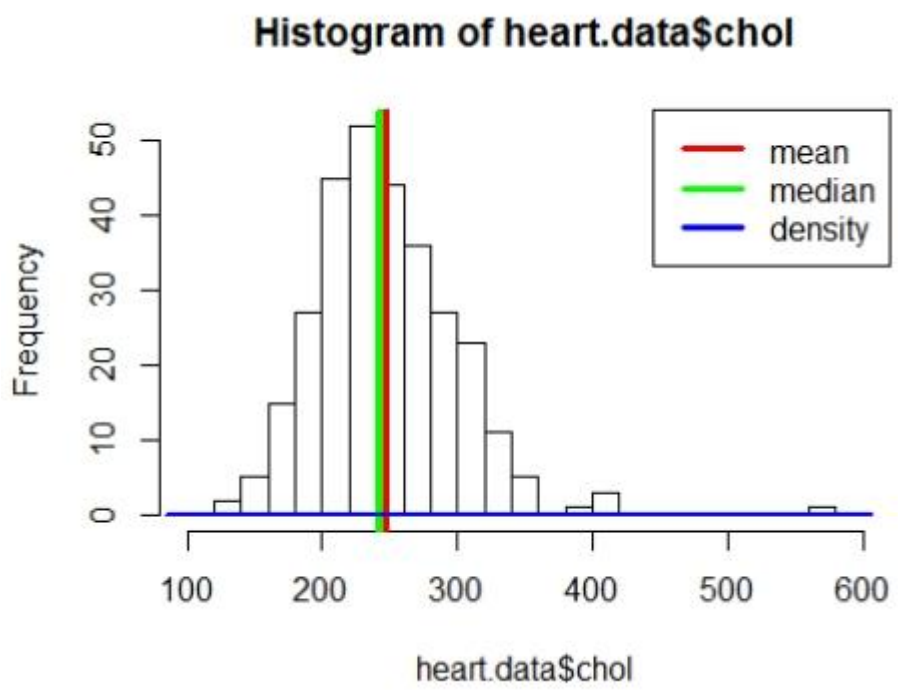

```
boxplot(heart.data$age,heart.data$chol,heart.data$trestbps)
```



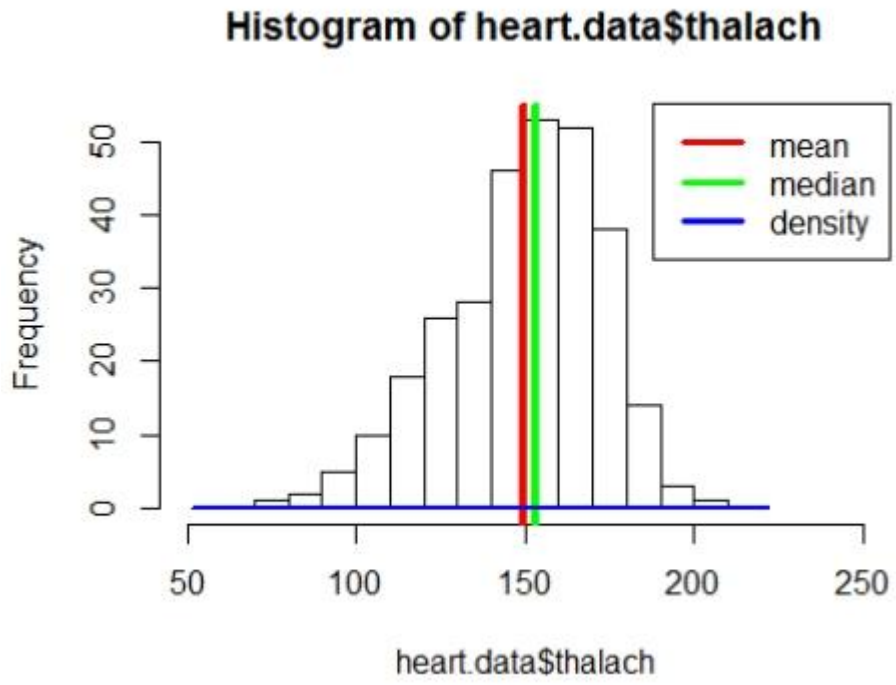

```
hist(heart.data$trestbps,breaks = 30, xlim = c(80,200))
abline(v=mean(heart.data$trestbps),col="RED",lwd=4)
abline(v=median(heart.data$trestbps),col="green",lwd=4)
lines(density(heart.data$trestbps),col = "blue" , lwd = 2)
legend(x="topright",c("mean","median","density"),col=c("red","green","blue
"),lwd=c(3,3))
```



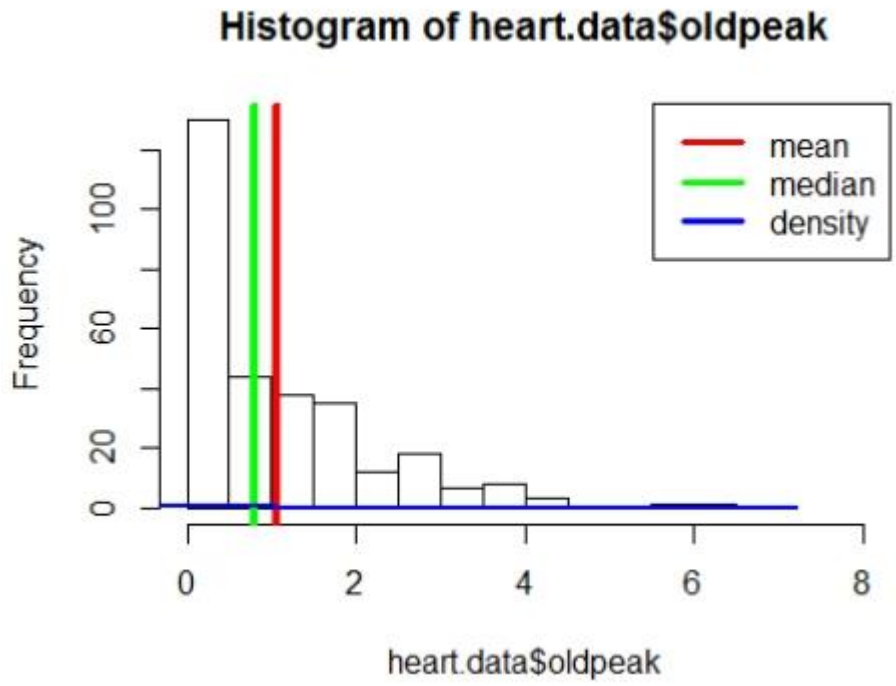
```
hist(heart.data$chol, breaks = 20, xlim = c(100,600))
abline(v=mean(heart.data$chol),col="RED",lwd=4)
abline(v=median(heart.data$chol),col="green",lwd=4)
lines(density(heart.data$chol),col = "blue" , lwd = 2)
legend(x="topright",c("mean","median","density"),col=c("red","green","blue
"),lwd=c(3,3))
```



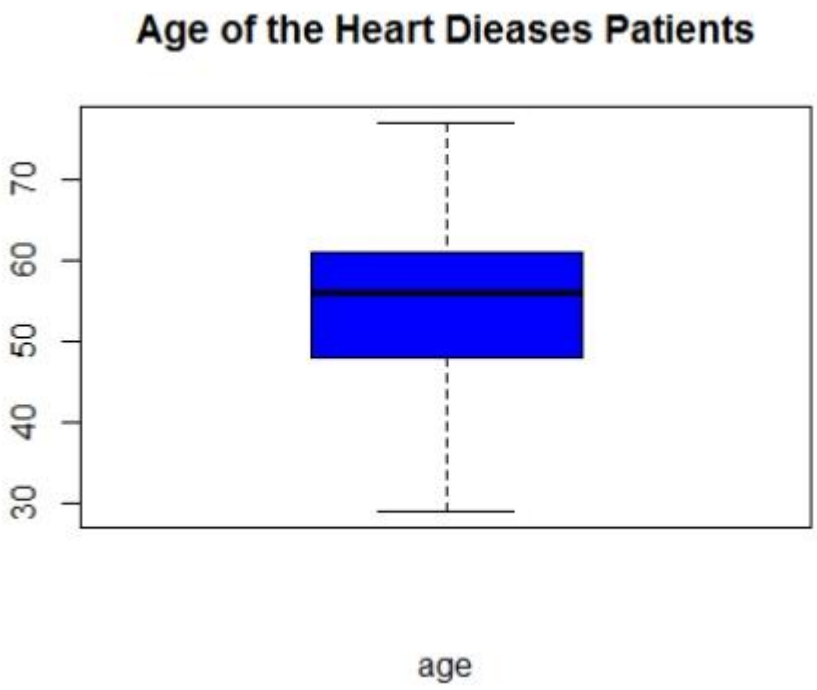

```
hist(heart.data$thalach,breaks=10,xlim= c(50,250))
abline(v=mean(heart.data$thalach),col="Red",lwd=4)
abline(v=median(heart.data$thalach),col="green",lwd=4)
lines(density(heart.data$thalach),col = "blue" , lwd = 2)
legend(x="topright",c("mean","median","density"),col=c("red","green","blue
"),lwd=c(3,3))
```



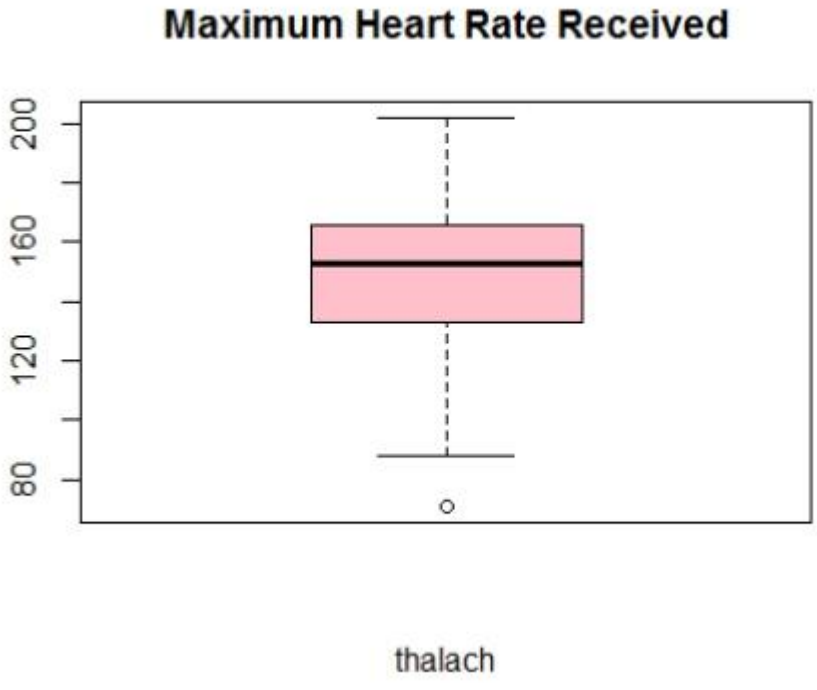

```
hist(heart.data$oldpeak, breaks = 10, xlim = c(0,8))
abline(v=mean(heart.data$oldpeak),col="Red",lwd=4)
abline(v=median(heart.data$oldpeak),col="green",lwd=4)
lines(density(heart.data$oldpeak),col = "blue" , lwd = 2)
legend(x="topright",c("mean","median","density"),col=c("red","green","blue
"),lwd=c(3,3))
```



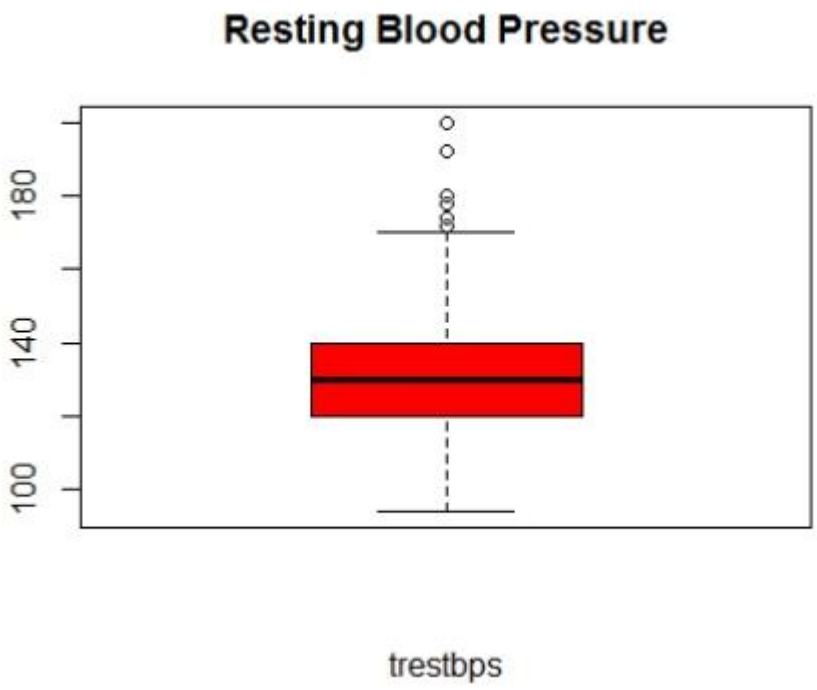
```r
boxplot(heart.data$age,data=heart.data,main="Age of the Heart Dieases Patients",xlab="age",col="blue")
```



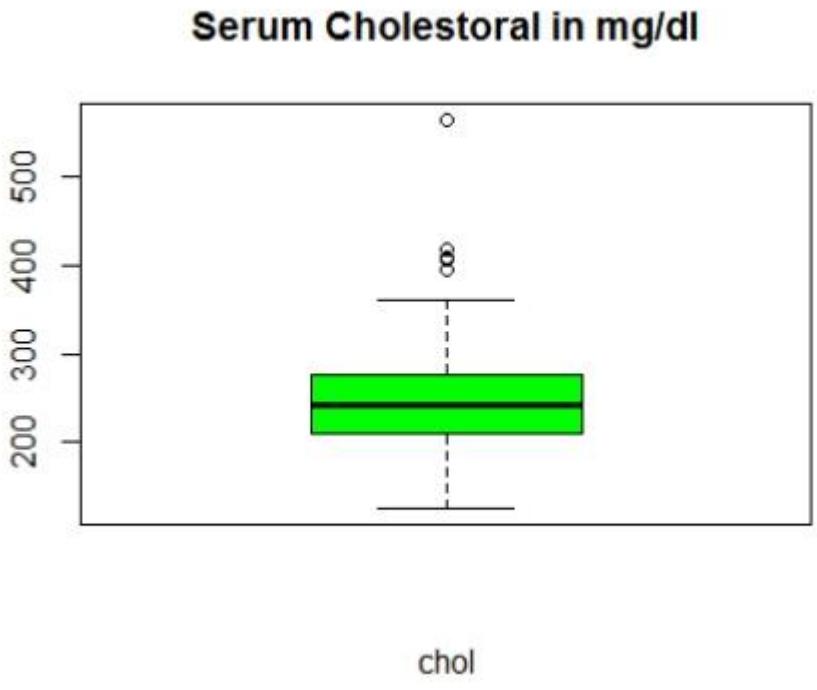

```r
boxplot(heart.data$thalach,data=heart.data,main="Maximum Heart Rate Received",xlab="thalach",col="pink")
```



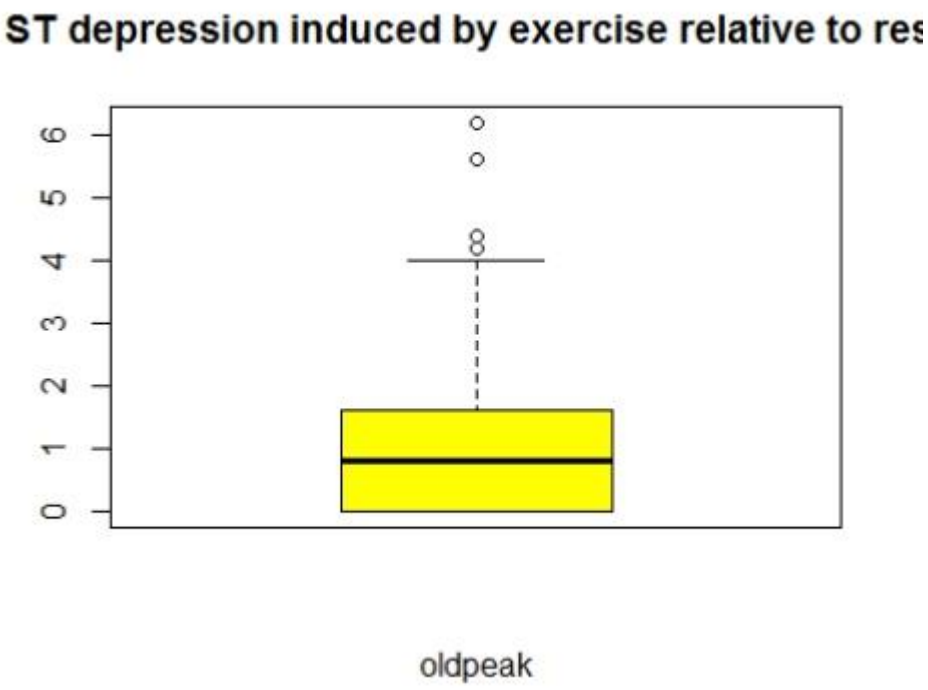
```r
boxplot(heart.data$trestbps,data=heart.data,main="Resting Blood
Pressure",xlab="trestbps",col="red")
```



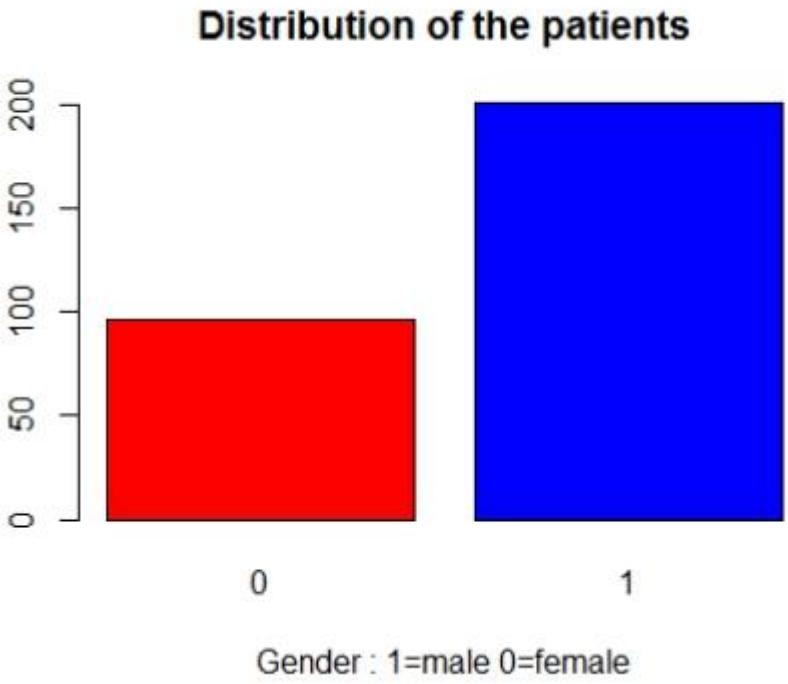

```r
boxplot(heart.data$chol,data=heart.data,main="Serum Cholestoral in
mg/dl",xlab="chol",col="green")
```



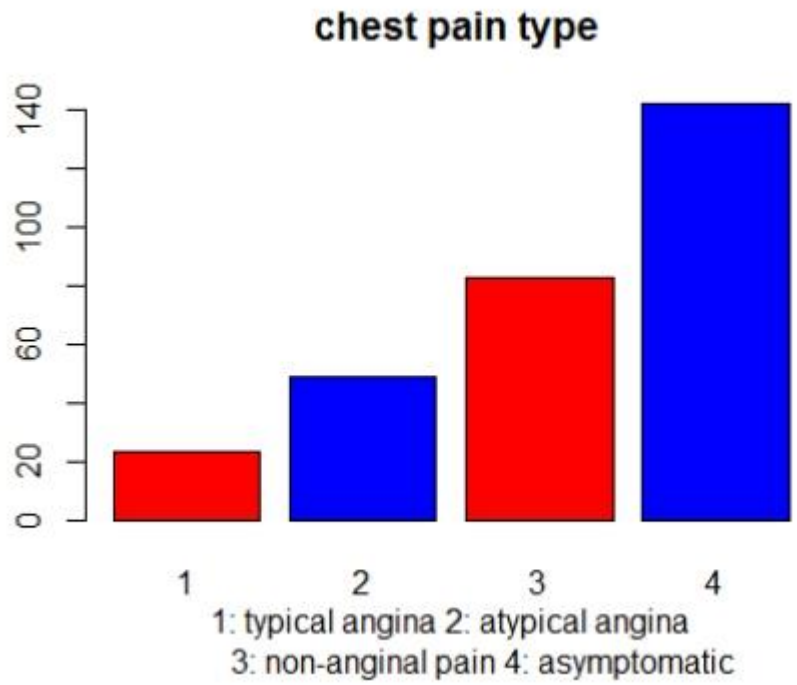
```r
boxplot(heart.data$oldpeak,data=heart.data,main="ST depression induced by
exercise relative to rest",xlab="oldpeak",col="yellow")
```



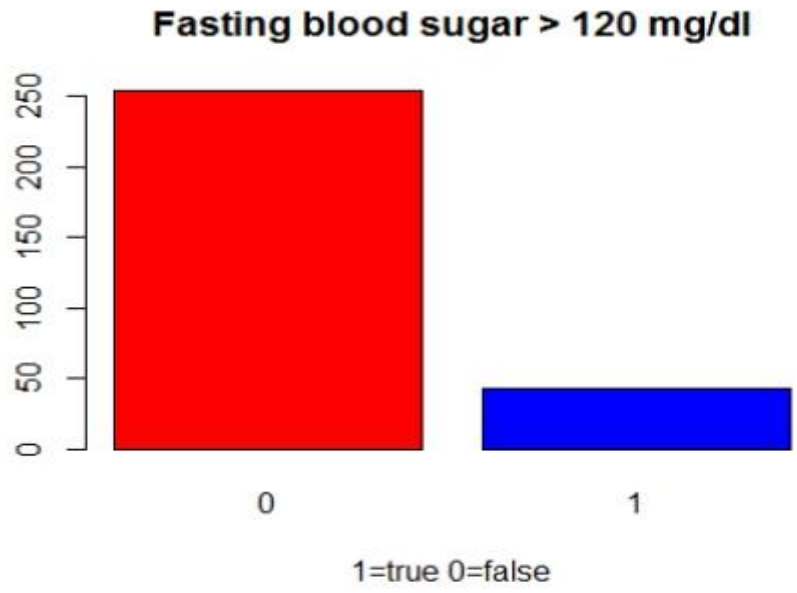

```r
counts <- table(heart.data$sex)
barplot(counts, main="Distribution of the patients",xlab = "Gender :
1=male 0=female", col = c("red","blue"))
```



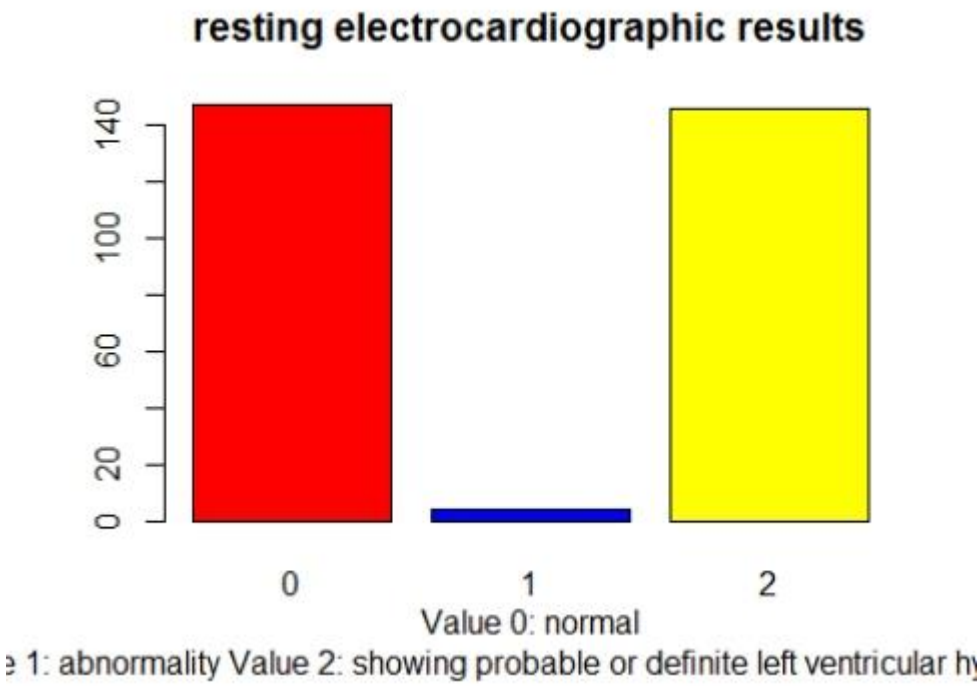
```
counts <- table(heart.data$cp)
barplot(counts, main=" chest pain type",xlab = "1: typical angina 2:
atypical angina
        3: non-anginal pain 4: asymptomatic", col = c("red","blue"))
```



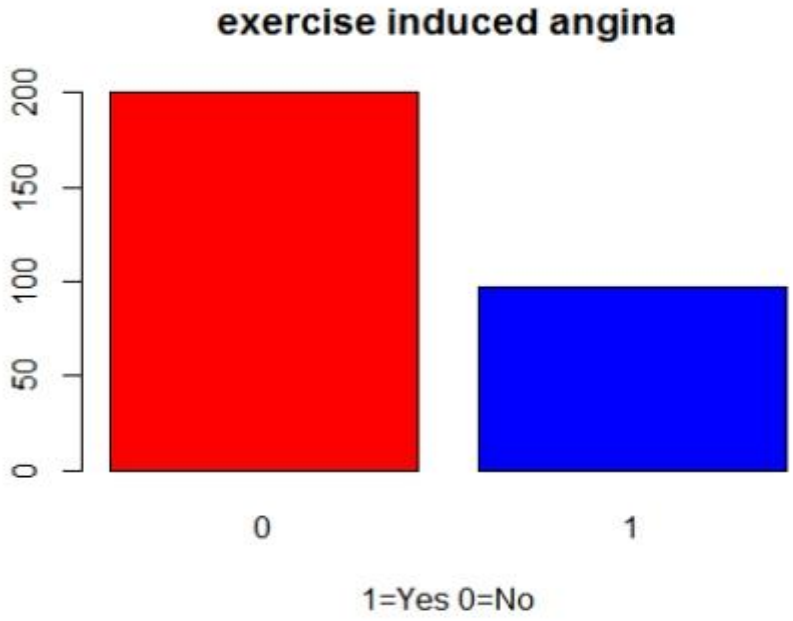

```
counts <- table(heart.data$fbs)
barplot(counts, main="Fasting blood sugar > 120 mg/dl",xlab = "1=true
0=false", col = c("red","blue"))
```



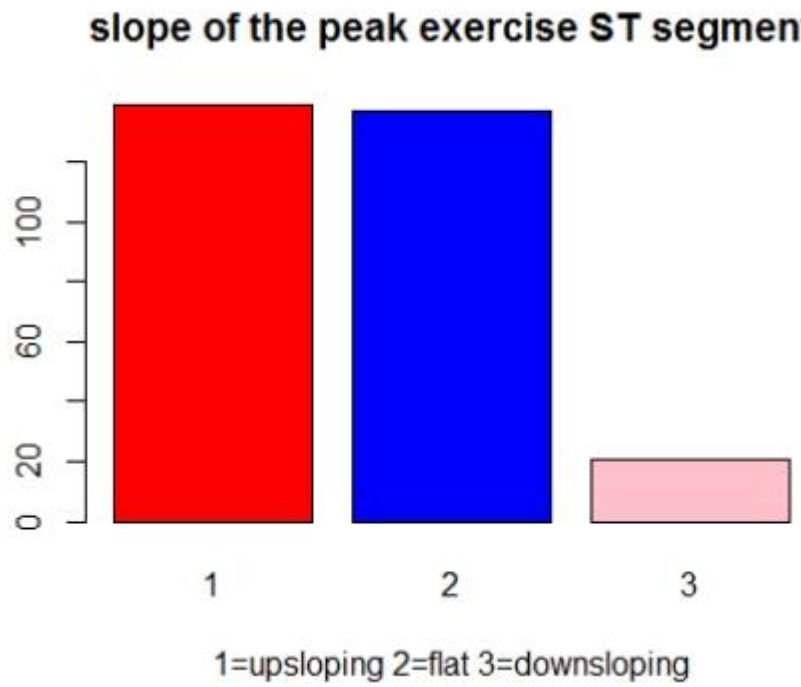
```
counts <- table(heart.data$restecg)
barplot(counts, main="resting electrocardiographic results",xlab = "Value
0: normal
Value 1: abnormality Value 2: showing probable or definite left
ventricular hypertrophy ", col = c("red","blue","yellow"))
```



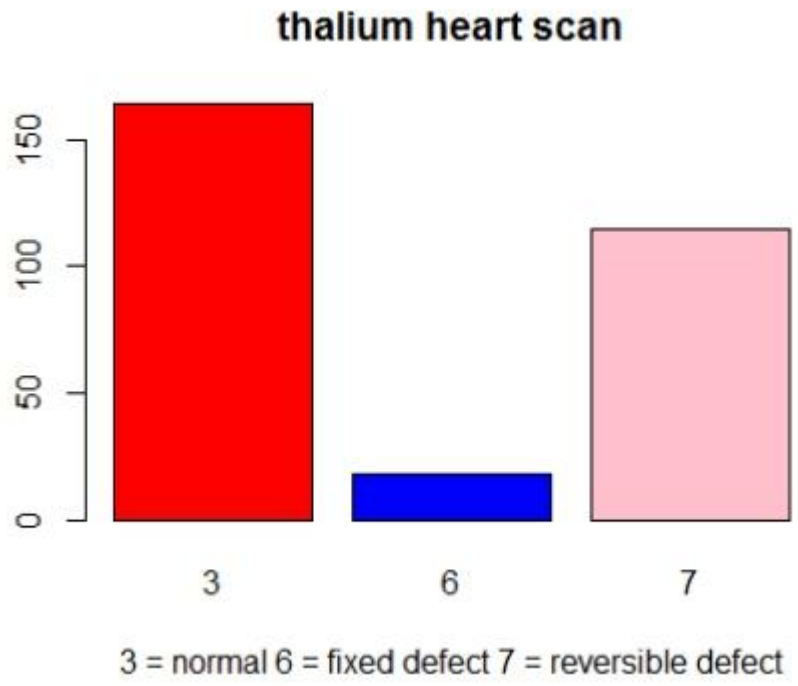

```
counts <- table(heart.data$exang)
barplot(counts, main="exercise induced angina",xlab = "1=Yes 0=No", col =
c("red","blue"))
```



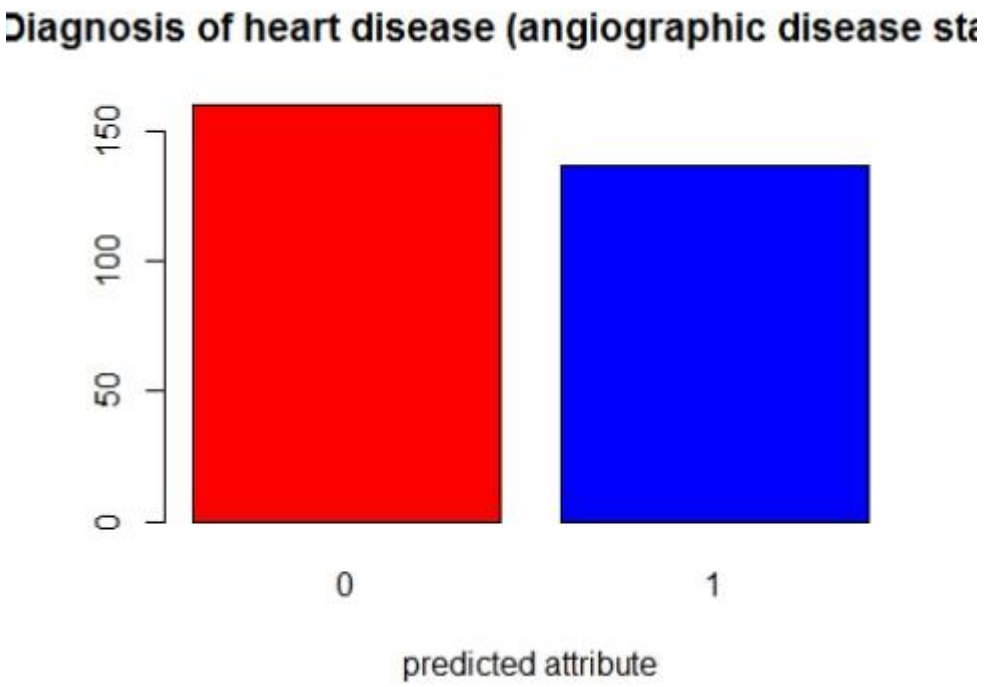
```
counts <- table(heart.data$slope)
barplot(counts, main="slope of the peak exercise ST segment",xlab =
"1=upsloping 2=flat 3=downsloping", col = c("red","blue","pink"))
```



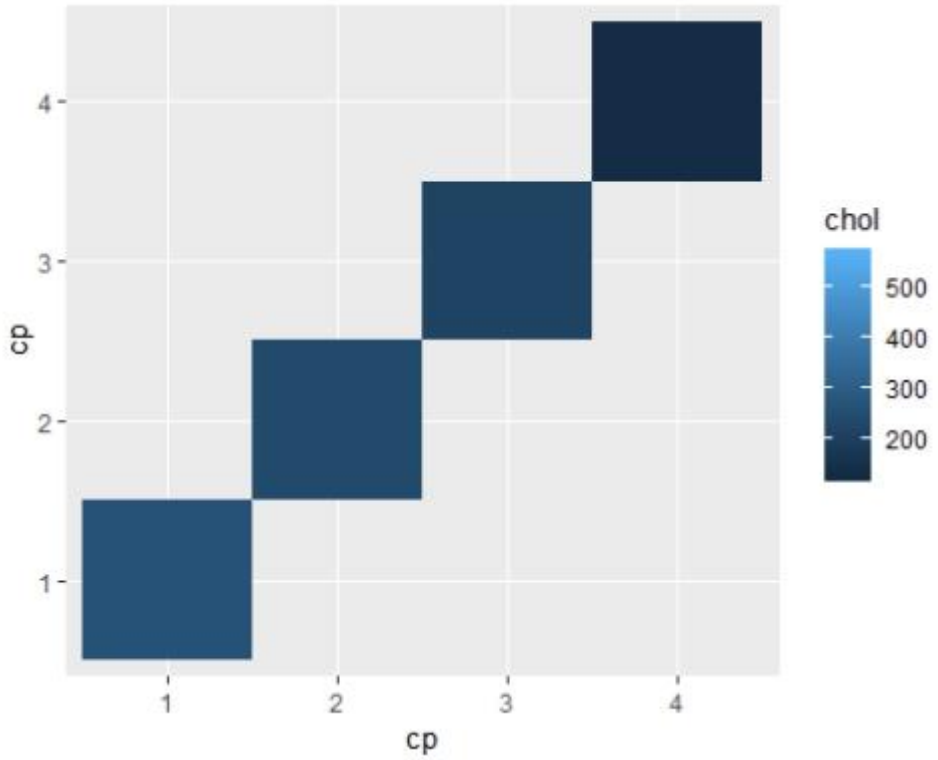

```
counts <- table(heart.data$thal)
barplot(counts, main="thalium heart scan",xlab = "3 = normal 6 = fixed
defect 7 = reversible defect", col = c("red","blue","pink"))
```



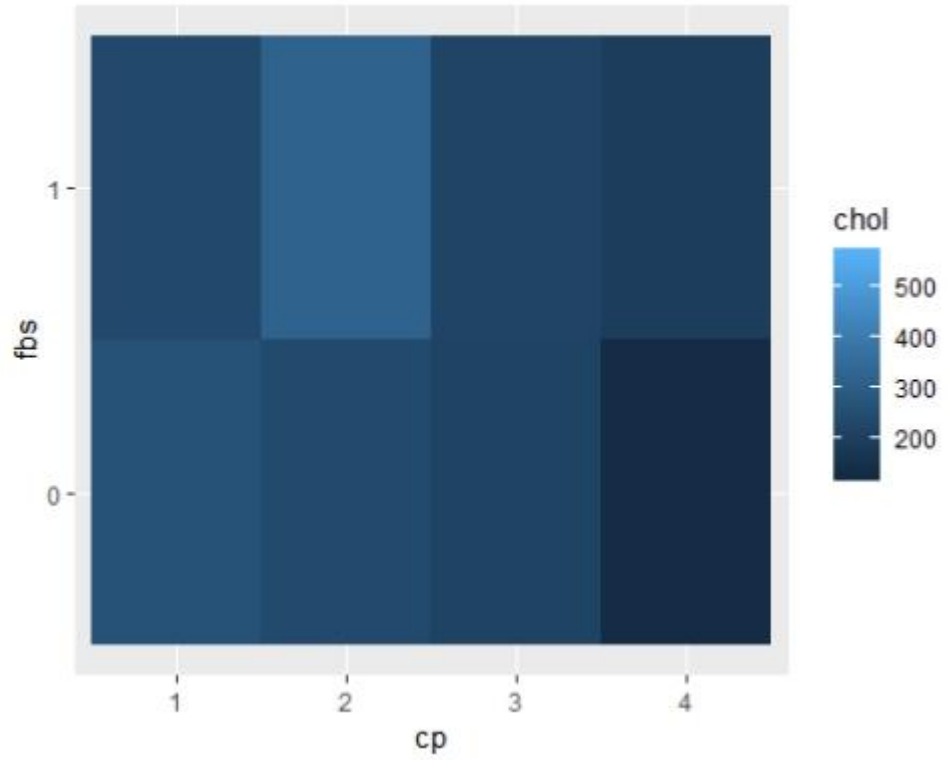
```r
counts <- table(heart.data$num)
barplot(counts, main="Diagnosis of heart disease (angiographic disease
status)",xlab = "predicted attribute", col = c("red","blue"))
```



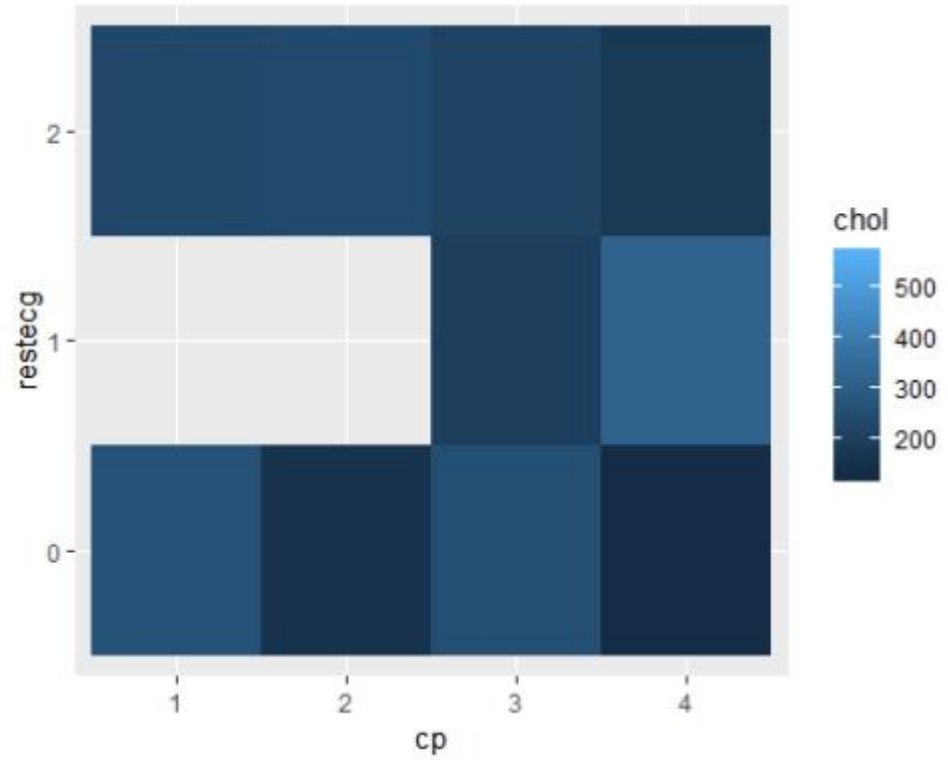

```r
library(ggplot2)
ggplot(data = heart.data, aes(x=cp, y=cp, fill=chol)) + geom_tile()
```



```
ggplot(data = heart.data, aes(x=cp, y=fbs, fill=chol)) + geom_tile()
```




```
ggplot(data = heart.data, aes(x=cp, y=restecg, fill=chol)) + geom_tile()
```



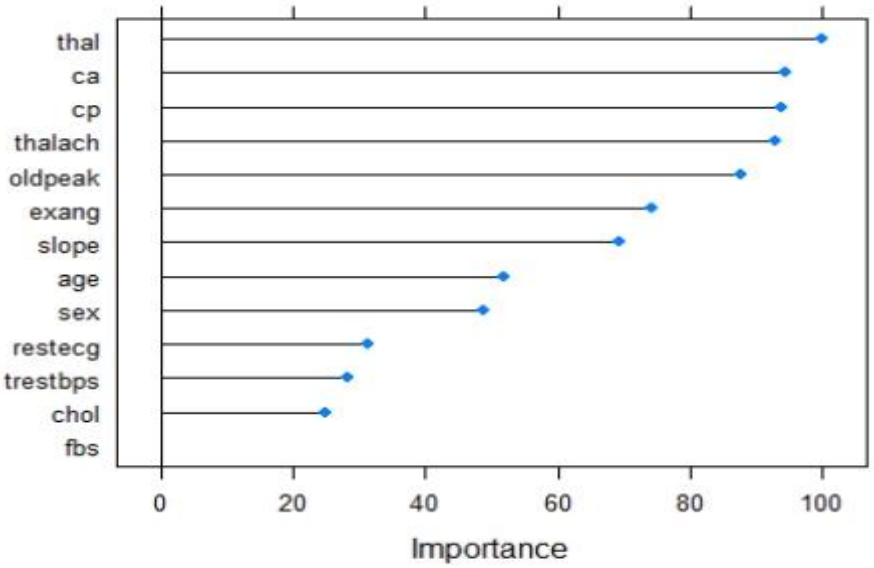

```
heart = heart.data #add labels only for plot
levels(heart$num) = c("No disease","Disease")
levels(heart$sex) = c("female","male","")
```

```r
mosaicplot(heart$sex ~ heart$num,
           main="Fate by Gender", shade=FALSE,color=TRUE,
           xlab="Gender", ylab="Heart disease")
```


```r
set.seed(7)
library(mlbench)
library(caret)
# rank features by importance
#prepare training scheme
control <- trainControl(method="repeatedcv", number=10, repeats=3)
#train the model
model <- train(num~., data=heart.data, method="lvq", preProcess="scale",
trControl=control)
# estimate variable importance
importance <- varImp(model, scale=T)
# summarize importance
print(importance)
```

```
## ROC curve variable importance
##
##          Importance
## thal         100.00
## ca            94.71
## cp            93.86
## thalach       93.08
## oldpeak       87.71
## exang         74.27
## slope         69.43
## age           51.90
## sex           48.81
```

```
## restecg       31.23
## trestbps      28.24
## chol          24.82
## fbs            0.00

# plot importance
plot(importance)
```



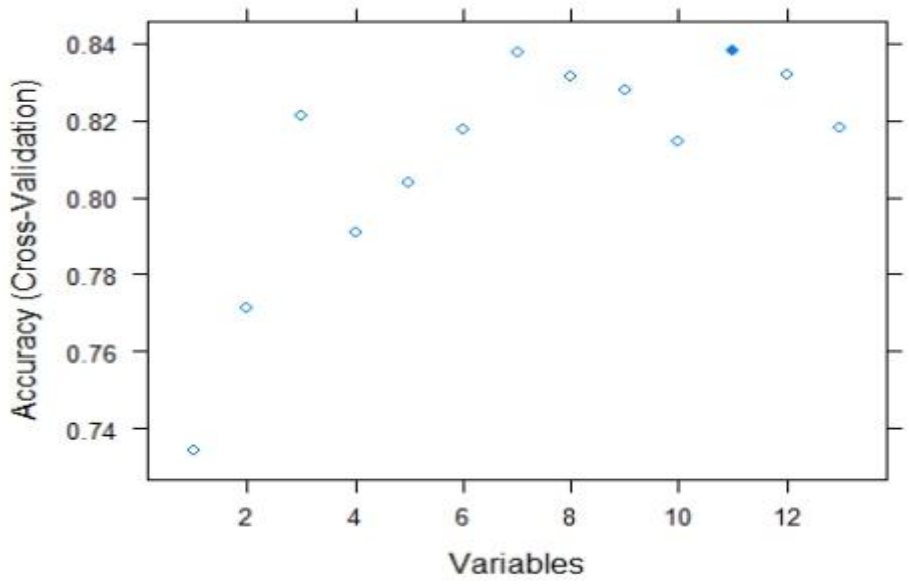

```
# define the control using a random forest selection function
control <- rfeControl(functions=rfFuncs, method="cv", number=10)
# run the RFE algorithm
results <- rfe(heart.data[,1:13], heart.data[,14], sizes=c(1:14),
rfeControl=control)
# summarize the results
print(results)

##
## Recursive feature selection
##
## Outer resampling method: Cross-Validated (10 fold)
##
## Resampling performance over subset size:
##
##  Variables Accuracy  Kappa AccuracySD KappaSD Selected
##          1   0.7340 0.4607    0.07606  0.1535
##          2   0.7713 0.5354    0.08397  0.1760
##          3   0.8213 0.6378    0.07055  0.1438
##          4   0.7907 0.5745    0.09341  0.1909
##          5   0.8040 0.6029    0.08880  0.1786
##          6   0.8177 0.6298    0.07652  0.1546
##          7   0.8379 0.6720    0.05164  0.1034
##          8   0.8314 0.6592    0.06032  0.1201
##          9   0.8279 0.6526    0.06728  0.1349
##         10   0.8147 0.6268    0.07196  0.1431
```

```
##         11   0.8384 0.6744    0.08428  0.1680        *
##         12   0.8317 0.6609    0.07278  0.1447
##         13   0.8182 0.6333    0.08303  0.1653
##
## The top 5 variables (out of 11):
##    ca, thal, cp, oldpeak, sex
```

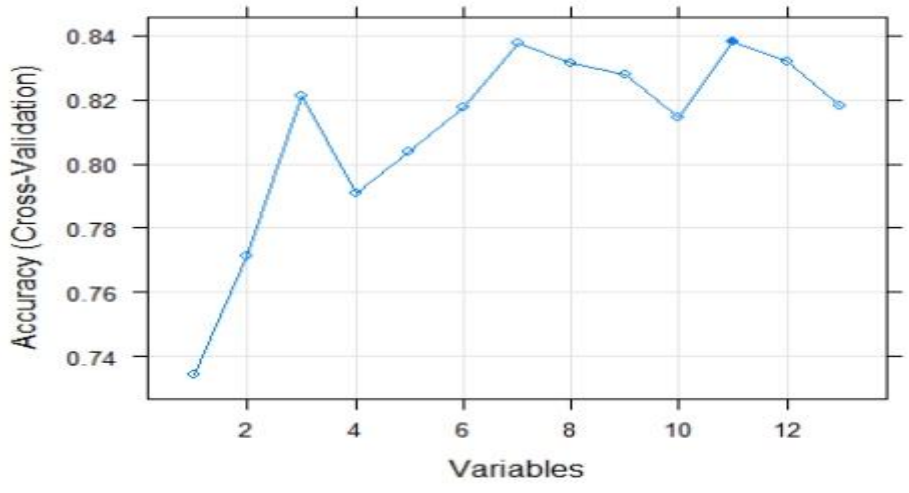
```r
# list the chosen features
predictors(results)
```

```
##  [1] "ca"       "thal"     "cp"       "oldpeak"  "sex"      "thalach"
##  [7] "exang"    "slope"    "age"      "trestbps" "restecg"
```

```r
# plot the results
plot(results)
```



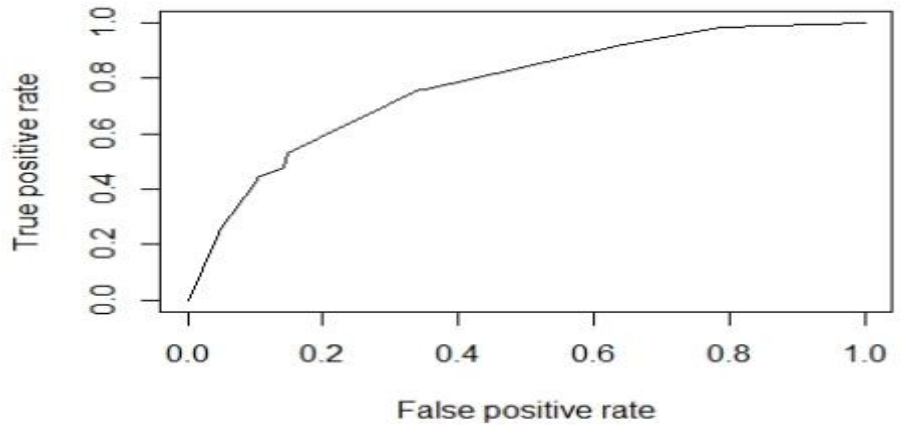

```r
plot(results, type=c("g", "o"))
```



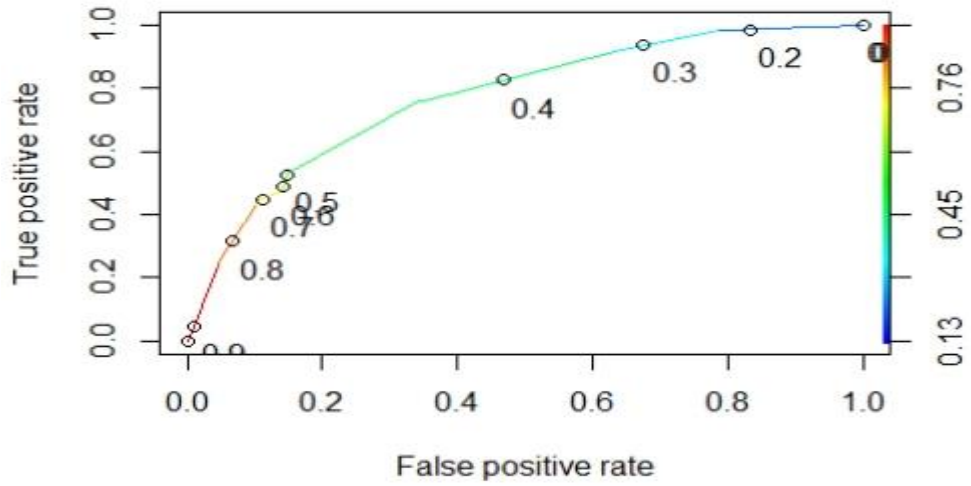
LOGISTIC REGRESSION

```
set.seed(100)  # setting seed to reproduce results of random sampling
trainingRowIndex <- sample(1:nrow(heart.data), 0.8*nrow(heart.data))  #
row indices for training data
trainingData <- heart.data[trainingRowIndex, ]  # model training data
testData  <- heart.data[-trainingRowIndex, ]   # test data

trainingData <- as.data.frame(trainingData)
testData <- as.data.frame(testData)

logmod <- glm(num ~ sex+exang+restecg, data=trainingData, family =
binomial)
summary(logmod)

##
## Call:
## glm(formula = num ~ sex + exang + restecg, family = binomial,
##     data = trainingData)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9232  -0.8968  -0.5353   0.9695   2.0068
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.8704     0.3545  -5.276 1.32e-07 ***
## sex1          1.1672     0.3371   3.463 0.000535 ***
## exang1        1.7943     0.3260   5.504 3.72e-08 ***
## restecg1      1.6602     1.3286   1.250 0.211433
## restecg2      0.5870     0.2981   1.969 0.048937 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 327.03  on 236  degrees of freedom
## Residual deviance: 270.86  on 232  degrees of freedom
## AIC: 280.86
##
## Number of Fisher Scoring iterations: 4

predictTrain <- predict(logmod, type = "response")
summary(predictTrain)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.1335  0.3311  0.4710  0.4599  0.7486  0.8427

library(ROCR)

## Warning: package 'ROCR' was built under R version 3.4.4
```

```
## Loading required package: gplots

## Warning: package 'gplots' was built under R version 3.4.4

##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess

ROCRpred = prediction(predictTrain, trainingData$num)
ROCRperf = performance(ROCRpred, "tpr", "fpr")
plot(ROCRperf)
```




```
plot(ROCRperf, colorize=TRUE)
plot(ROCRperf, colorize=TRUE, print.cutoffs.at=seq(0,1,by=0.1),
text.adj=c(-0.2,1.7))
```




```
#table(testData$num,predictTest >= 0.2)
#table(testData$num,predictTest >= 0.3)
#(15+26)/(15+17+2+26)
result <- predict(logmod,testData)
Final <- cbind(Actual = testData$num , Predicted = result)
final_df <- as.data.frame(Final)
final_df
```

```
##     Actual   Predicted
## 1        1 -0.11609085
## 2        2  1.67823584
## 8        1 -0.07603541
## 10       2  1.67823584
## 11       1 -0.70314033
## 16       1 -0.70314033
## 27       1 -1.87036210
## 28       1 -1.87036210
## 29       1 -0.70314033
error <- (final_df$Actual - final_df$Predicted)
final_df <- cbind(final_df,error)
final_df
```

```
##     Actual   Predicted       error
## 1        1 -0.11609085  1.11609085
## 2        2  1.67823584  0.32176416
## 8        1 -0.07603541  1.07603541
## 10       2  1.67823584  0.32176416
## 11       1 -0.70314033  1.70314033
## 16       1 -0.70314033  1.70314033
## 27       1 -1.87036210  2.87036210
## 28       1 -1.87036210  2.87036210
## 29       1 -0.70314033  1.70314033
## 32       2  1.09118637  0.90881363
## 35       1  1.09118637 -0.09118637
```

```
## 41       2 -1.28331263  3.28331263
## 42       1  1.09118637 -0.09118637
## 43       1 -1.87036210  2.87036210
## 45       2 -1.28331263  3.28331263
## 54       1 -0.11609085  1.11609085
## 59       1 -0.11609085  1.11609085
## 64       1 -1.87036210  2.87036210
## 66       2  1.67823584  0.32176416
## 67       2 -0.11609085  2.11609085
```

```
rmse <- sqrt(mean(final_df$error^2))
rmse
```

```
## [1] 1.948164
```

```
result <- predict(logmod,testData)
```

DECISION TREE

```
library(caret)
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 3.4.4
```

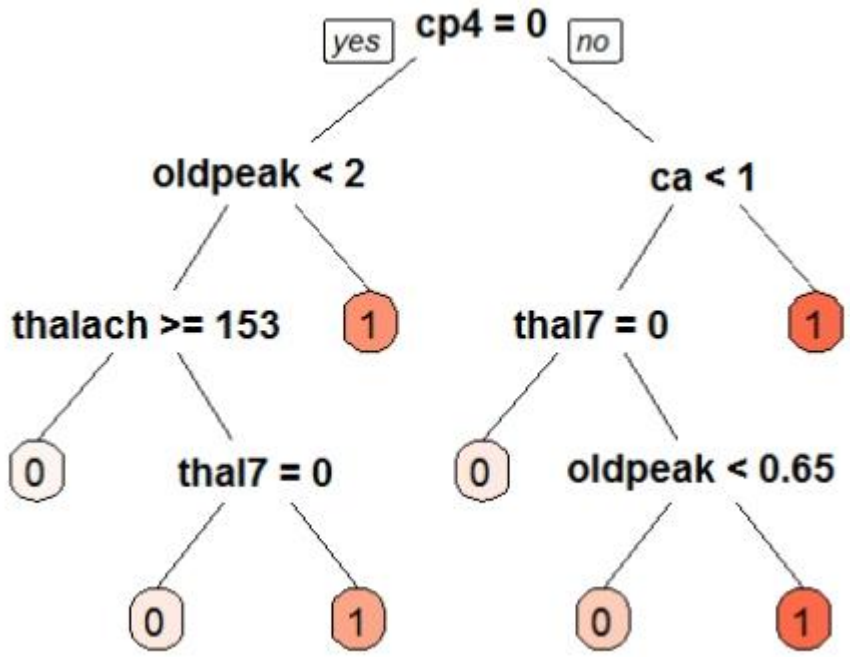
```
## Loading required package: rpart
```

```
trctrl <- trainControl(method = "repeatedcv", number = 10,repeats = 3)
dtree_fit <- train(num ~ca+cp+thalach+oldpeak+exang+slope+thal,data =
trainingData, method = "rpart",
              parms = list(split = "information"),
              trControl=trctrl,
            tuneLength = 10)
dtree_fit
```

```
## CART
##
## 237 samples
##   7 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 214, 213, 213, 213, 213, 213, ...
## Resampling results across tuning parameters:
##
##   cp          Accuracy   Kappa
##   0.00000000  0.7969807  0.5889481
##   0.04994903  0.7830918  0.5590915
##   0.09989806  0.6902174  0.3748759
```

```
##   0.14984709  0.6902174  0.3748759
##   0.19979613  0.6902174  0.3748759
##   0.24974516  0.6902174  0.3748759
##   0.29969419  0.6902174  0.3748759
##   0.34964322  0.6902174  0.3748759
##   0.39959225  0.6902174  0.3748759
##   0.44954128  0.5877415  0.1389443
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.
```

```
prp(dtree_fit$finalModel, box.palette = "Reds", tweak = 1.2)
```




```
predict(dtree_fit, newdata = testData[1,])
```

```
## [1] 1
## Levels: 0 1
```

```
test_pred <- predict(dtree_fit, newdata = testData)
confusionMatrix(test_pred, testData$num)
```

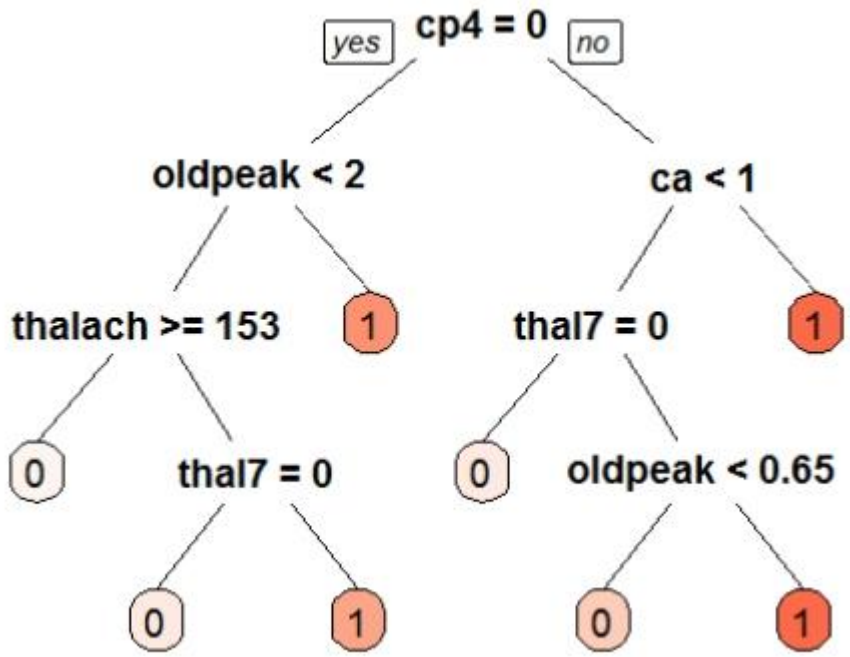
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 28  6
##          1  4 22
##
##                Accuracy : 0.8333
##                  95% CI : (0.7148, 0.9171)
```

```
##     No Information Rate : 0.5333
##     P-Value [Acc > NIR] : 1.056e-06
##
##                   Kappa : 0.6637
##  Mcnemar's Test P-Value : 0.7518
##
##             Sensitivity : 0.8750
##             Specificity : 0.7857
##          Pos Pred Value : 0.8235
##          Neg Pred Value : 0.8462
##              Prevalence : 0.5333
##          Detection Rate : 0.4667
##    Detection Prevalence : 0.5667
##       Balanced Accuracy : 0.8304
##
##        'Positive' Class : 0
##
```

```
# dtree with crieria as gini index

dtree_fit_gini <- train(num ~ ca+cp+thalach+oldpeak+exang+slope+thal ,data
= trainingData, method = "rpart",
                   parms = list(split = "gini"),
                   trControl=trctrl,
                   tuneLength = 10)
dtree_fit_gini
```

```
## CART
##
## 237 samples
##   7 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 214, 213, 213, 213, 213, 213, ...
## Resampling results across tuning parameters:
##
##   cp          Accuracy   Kappa
##   0.00000000  0.8088768  0.6121038
##   0.04994903  0.7760266  0.5445837
##   0.09989806  0.6899155  0.3752086
##   0.14984709  0.6899155  0.3752086
##   0.19979613  0.6899155  0.3752086
##   0.24974516  0.6899155  0.3752086
##   0.29969419  0.6899155  0.3752086
##   0.34964322  0.6899155  0.3752086
##   0.39959225  0.6899155  0.3752086
##   0.44954128  0.6076691  0.1830909
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.
```

```
prp(dtree_fit_gini$finalModel, box.palette = "Reds", tweak = 1.2)
```




```
predict(dtree_fit_gini, newdata = testData[1,])
```

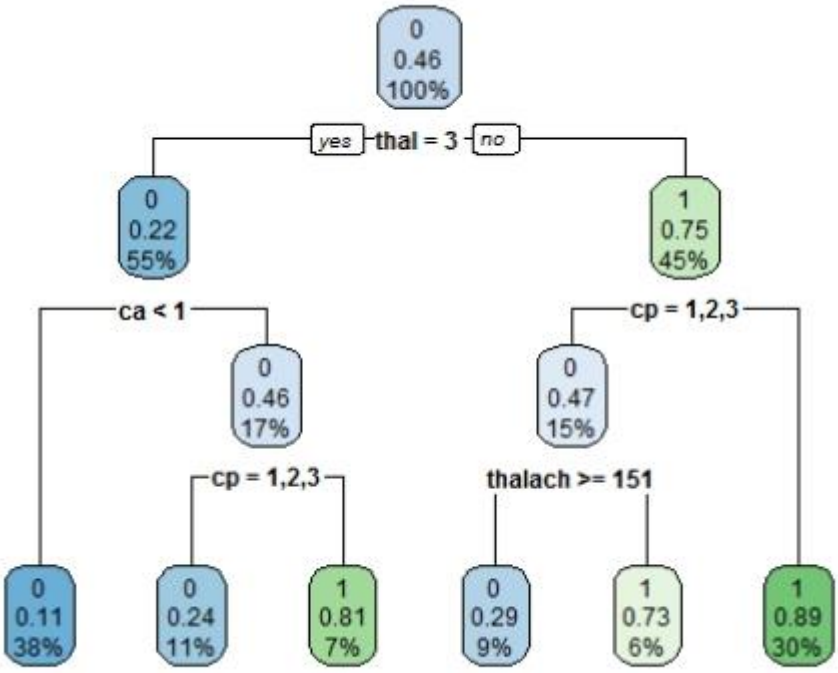
```
## [1] 1
## Levels: 0 1
```

```
test_pred_gini <- predict(dtree_fit_gini, newdata = testData)
confusionMatrix(test_pred_gini, testData$num)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 28  6
##          1  4 22
##
##                Accuracy : 0.8333
##                  95% CI : (0.7148, 0.9171)
##     No Information Rate : 0.5333
##     P-Value [Acc > NIR] : 1.056e-06
##
##                   Kappa : 0.6637
##  Mcnemar's Test P-Value : 0.7518
##
##             Sensitivity : 0.8750
##             Specificity : 0.7857
##          Pos Pred Value : 0.8235
##          Neg Pred Value : 0.8462
##              Prevalence : 0.5333
```

```
##          Detection Rate : 0.4667
##    Detection Prevalence : 0.5667
##       Balanced Accuracy : 0.8304
##
##        'Positive' Class : 0
##
```

```
rtree_fit <- rpart(num ~., trainingData)
rpart.plot(rtree_fit)
```




```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.4.4
```

```
## randomForest 4.6-14
```

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

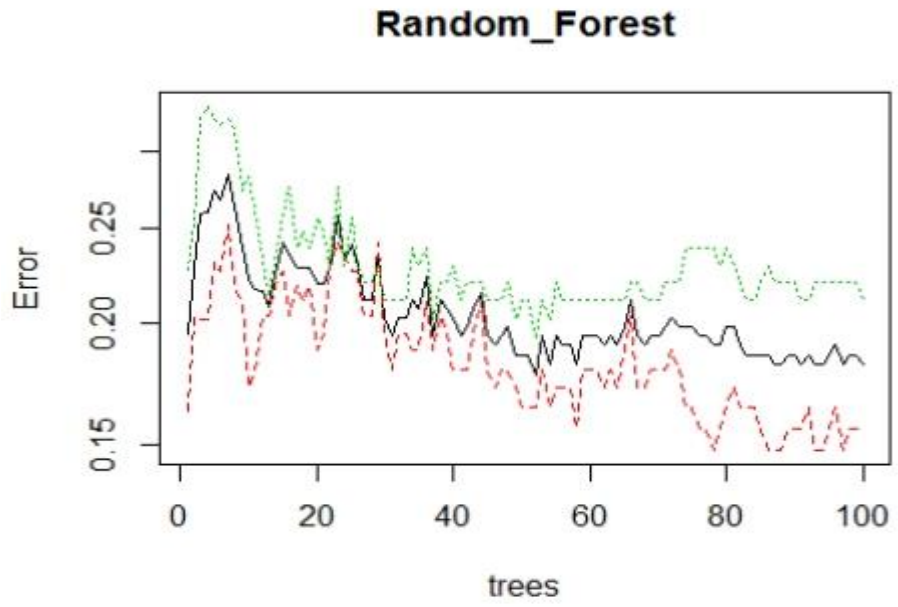
```
##
## Attaching package: 'randomForest'
```

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

```
rf <- randomForest(num~., data = trainingData,importance=TRUE)
rf
```

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

```
##  randomForest(formula = num ~ ., data = trainingData, importance =
TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 19.41%
## Confusion matrix:
##     0  1 class.error
## 0 106 22   0.1718750
## 1  24 85   0.2201835

test_pred <- predict(rf, newdata = testData)
confusionMatrix(test_pred, testData$num)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 31  5
##          1  1 23
##
##                Accuracy : 0.9
##                  95% CI : (0.7949, 0.9624)
##     No Information Rate : 0.5333
##     P-Value [Acc > NIR] : 1.066e-09
##
##                   Kappa : 0.7973
##  Mcnemar's Test P-Value : 0.2207
##
##             Sensitivity : 0.9688
##             Specificity : 0.8214
##          Pos Pred Value : 0.8611
##          Neg Pred Value : 0.9583
##              Prevalence : 0.5333
##          Detection Rate : 0.5167
##    Detection Prevalence : 0.6000
##       Balanced Accuracy : 0.8951
##
##        'Positive' Class : 0
##

Random_Forest <- randomForest(num ~ ., trainingData,
keep.forest=FALSE,main = "RandomForest", ntree=100)
plot(Random_Forest, log="y")
```



SUPPORT VECTOR MACHINE

```
library("e1071")

## Warning: package 'e1071' was built under R version 3.4.4

library("caret")
rctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 3)
set.seed(3233)
svm_Linear <- train(num ~., data = trainingData, method =
"svmLinear",trControl=rctrl,preProcess = c("center", "scale"),tuneLength =
10)
svm_Linear

## Support Vector Machines with Linear Kernel
##
## 237 samples
##  13 predictor
##   2 classes: '0', '1'
##
## Pre-processing: centered (18), scaled (18)
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 213, 213, 214, 214, 214, 213, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.8096618  0.6126685
##
## Tuning parameter 'C' was held constant at a value of 1

test_pred_svm <- predict(svm_Linear, newdata = testData)
test_pred_svm
```

```
##  [1] 0 1 0 1 1 0 0 0 0 1 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 1 1 1 0 1 1 0 1
0 1
## [36] 1 1 1 1 1 0 0 1 0 0 1 1 0 0 1 1 1 0 0 0 0 0 0 1 0
## Levels: 0 1
```

```
confusionMatrix(test_pred_svm, testData$num)
```

```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction  0  1
##          0 30  5
##          1  2 23
##
##                Accuracy : 0.8833
##                  95% CI : (0.7743, 0.9518)
##     No Information Rate : 0.5333
##     P-Value [Acc > NIR] : 7.387e-09
##
##                   Kappa : 0.764
##  Mcnemar's Test P-Value : 0.4497
##
##             Sensitivity : 0.9375
##             Specificity : 0.8214
##          Pos Pred Value : 0.8571
##          Neg Pred Value : 0.9200
##              Prevalence : 0.5333
##          Detection Rate : 0.5000
##    Detection Prevalence : 0.5833
##       Balanced Accuracy : 0.8795
##
##        'Positive' Class : 0
##
```

```
svm_radial <- train(num ~., data = trainingData, method = "svmRadial",
trControl=rctrl,preProcess = c("center", "scale"),tuneLength = 10)
svm_radial
```

```
## Support Vector Machines with Radial Basis Function Kernel
##
## 237 samples
##  13 predictor
##   2 classes: '0', '1'
##
## Pre-processing: centered (18), scaled (18)
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 213, 213, 214, 213, 214, 213, ...
## Resampling results across tuning parameters:
##
##   C       Accuracy   Kappa
##     0.25  0.8211957  0.6388142
##     0.50  0.8111111  0.6177030
##     1.00  0.7955918  0.5854497
```

```
##     2.00  0.7818237  0.5587406
##     4.00  0.7760870  0.5487742
##     8.00  0.7608092  0.5189098
##    16.00  0.7355676  0.4677274
##    32.00  0.7285628  0.4528176
##    64.00  0.7356280  0.4673598
##   128.00  0.7385266  0.4729570
##
## Tuning parameter 'sigma' was held constant at a value of 0.03950965
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were sigma = 0.03950965 and C =
0.25.
```

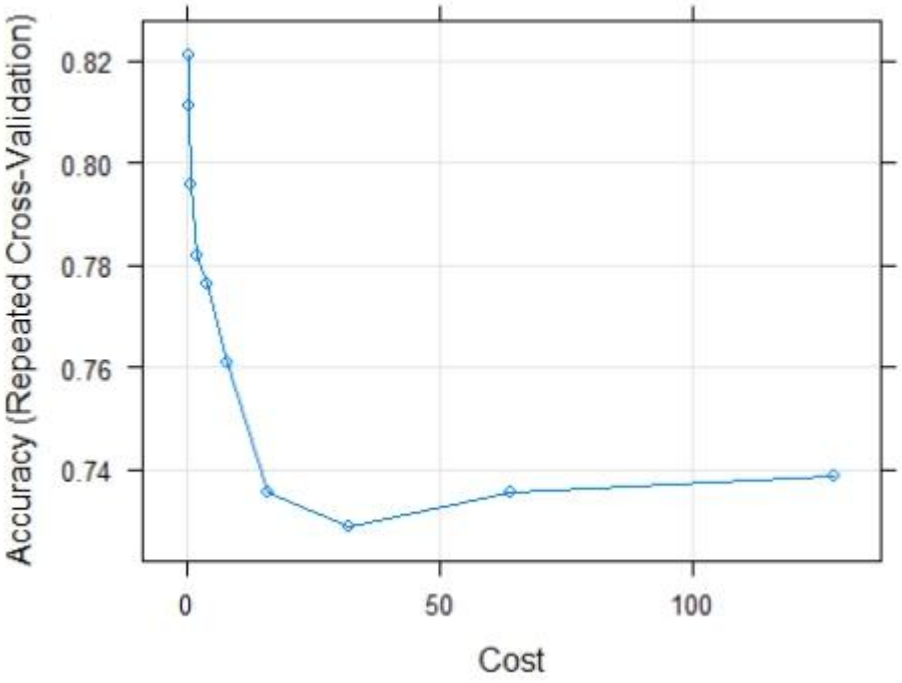
```
test_pred_svm <- predict(svm_radial, newdata = testData)
test_pred_svm
```

```
##  [1] 1 1 0 1 1 0 0 1 0 1 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 1 1 1 0 1 1 0 1
0 1
## [36] 1 1 1 1 0 0 0 1 0 0 1 1 0 0 1 1 1 0 0 0 0 0 1 1 1
## Levels: 0 1
```

```
confusionMatrix(test_pred_svm, testData$num)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 28  4
##          1  4 24
##
##                Accuracy : 0.8667
##                  95% CI : (0.7541, 0.9406)
##     No Information Rate : 0.5333
##     P-Value [Acc > NIR] : 4.403e-08
##
##                   Kappa : 0.7321
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.8750
##             Specificity : 0.8571
##          Pos Pred Value : 0.8750
##          Neg Pred Value : 0.8571
##              Prevalence : 0.5333
##          Detection Rate : 0.4667
##    Detection Prevalence : 0.5333
##       Balanced Accuracy : 0.8661
##
##        'Positive' Class : 0
##
```

```
plot(svm_radial)
```


Accuracy (Repeated Cross-Validation)
0.82
0.80
0.78
0.76
0.74
0
50
100
Cost