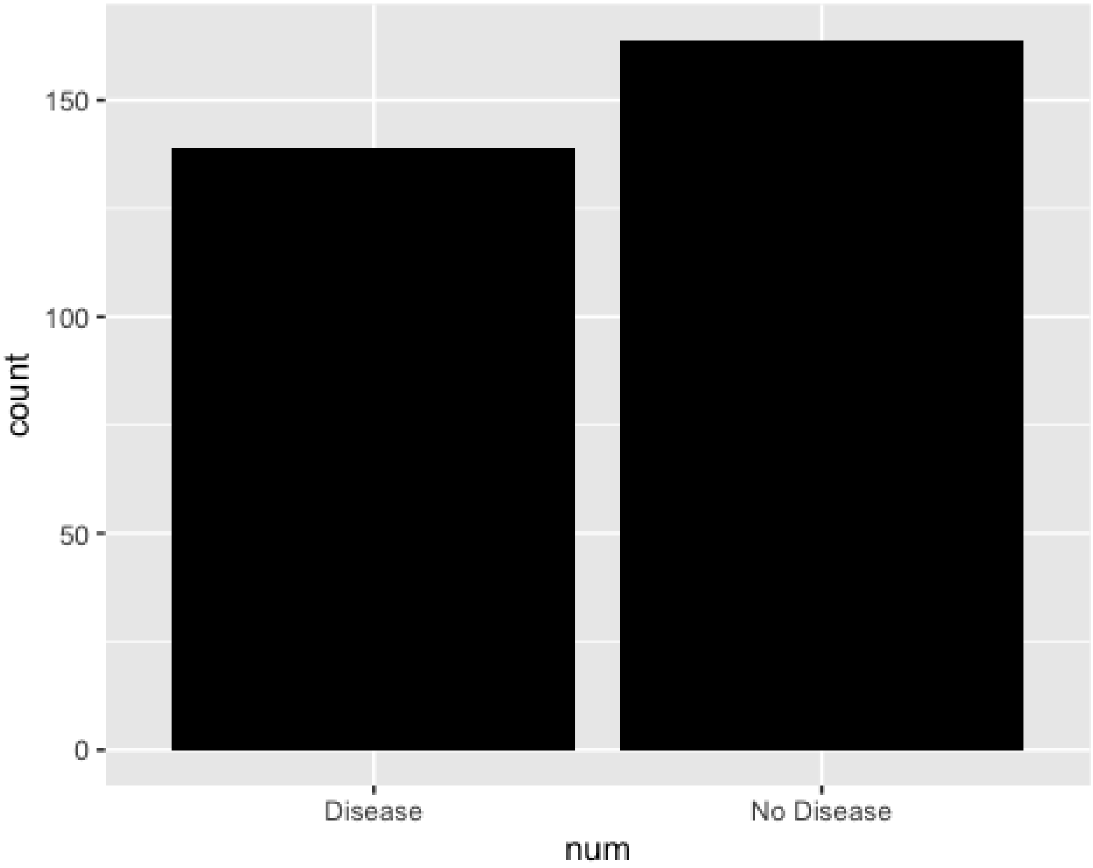
Project Report

Paulette Rodriguez

5/9/2020

```
require(ggplot2)

## Loading required package: ggplot2

require(pROC)

## Loading required package: pROC

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

##
## Attaching package: 'pROC'

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

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

dim(heart)

## [1] 303  14

heart$num = ifelse(heart$num > 0, "Disease", "No Disease")
table(heart$num)

##
##    Disease No Disease
##        139        164
```

```r
ggplot(heart, aes(x = num))+
  geom_bar(fill = "black")
```



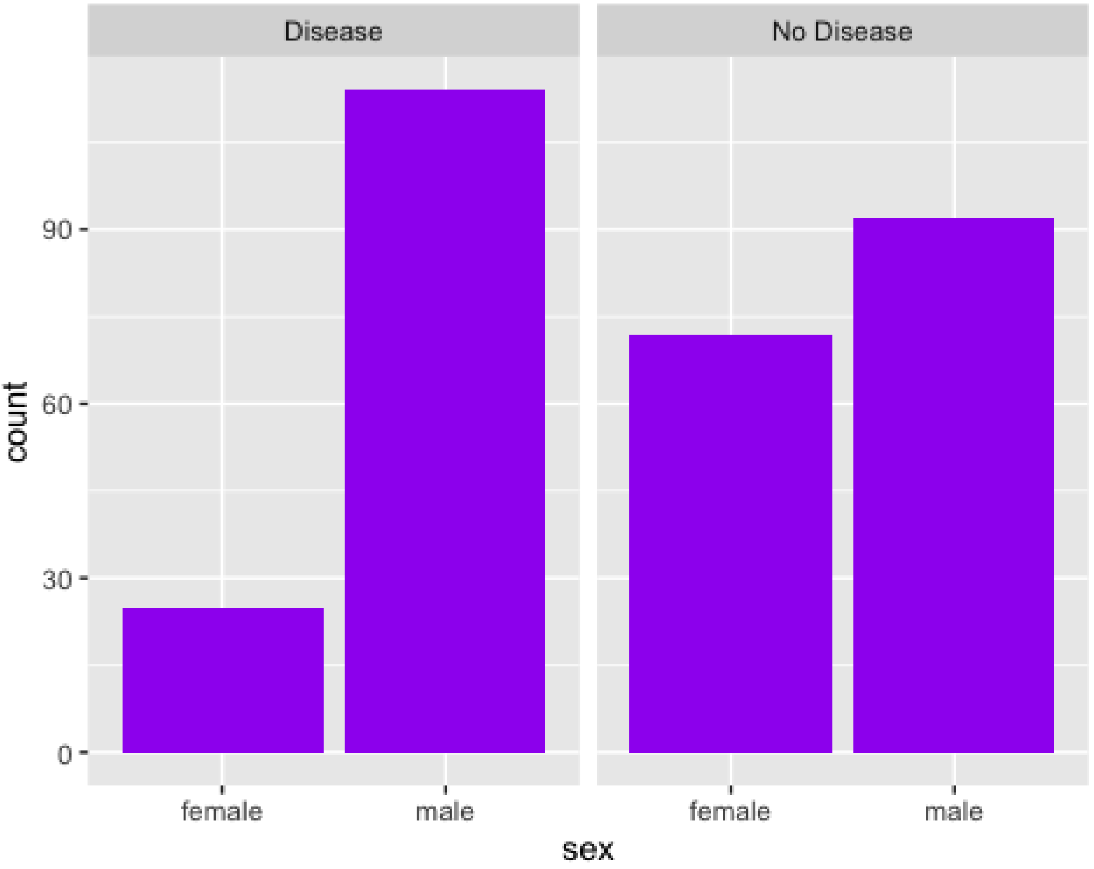
```r
#heart$sex = ifelse()

heart$sex = ifelse(heart$sex == 0, "female", "male")

table(heart$sex)
```

```
## 
## female   male 
##     97    206
```

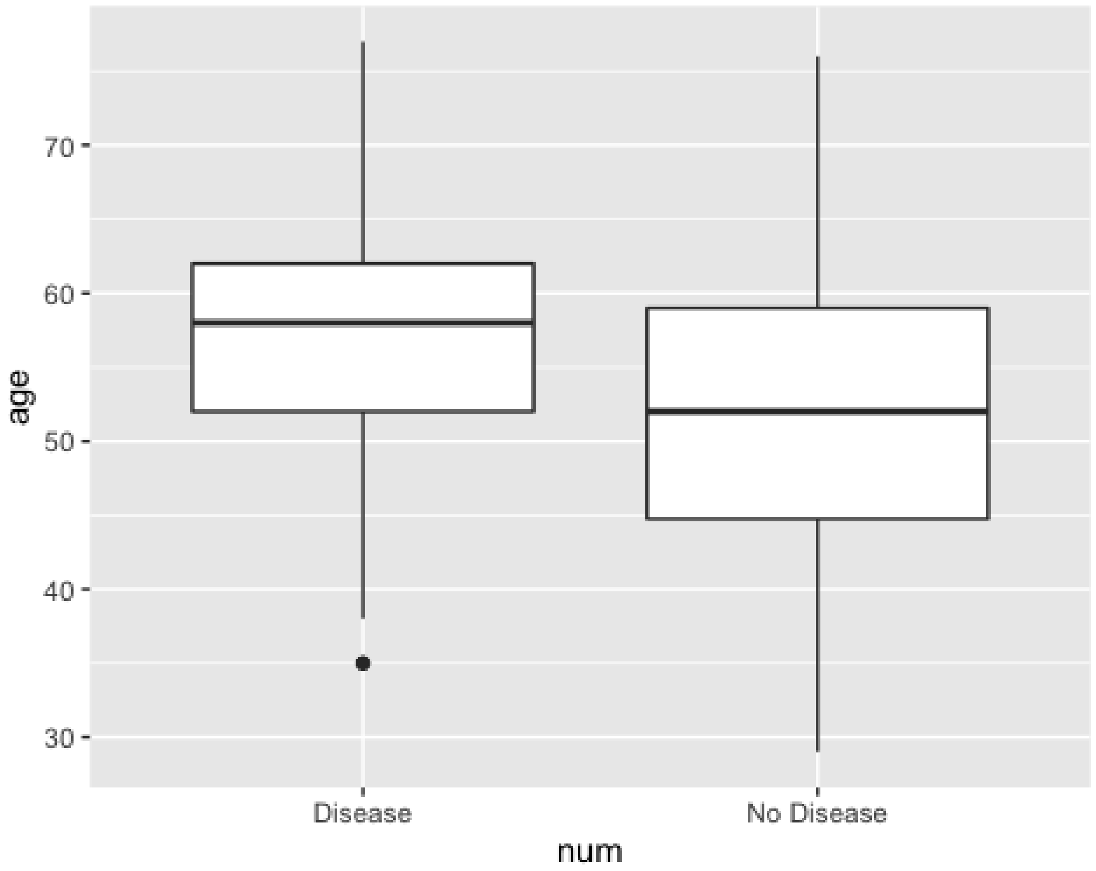
```r
table(sex = heart$sex, disease = heart$num)
```

```
##         disease
## sex      Disease No Disease
##   female      25         72
##   male       114         92
```

```r
ggplot(heart, aes(x = sex))+
  geom_bar(fill = "purple")+
  facet_wrap(~num)
```



```
by(heart$age, heart$num, summary)

## heart$num: Disease
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   35.00   52.00   58.00   56.63   62.00   77.00
## --------------------------------------------------------
## heart$num: No Disease
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   29.00   44.75   52.00   52.59   59.00   76.00

ggplot(heart, aes(x = num, y = age))+
  geom_boxplot()
```



```
cor.test(age, chol)
```

```
##
##  Pearson's product-moment correlation
##
## data:  age and chol
## t = 3.707, df = 301, p-value = 0.0002496
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.09859353 0.31423005
## sample estimates:
##       cor
## 0.2089503
```

```
table(cp, num)
```

```
##    num
## cp   0  1  2  3  4
##   1 16  5  1  0  1
##   2 41  6  1  2  0
##   3 68  9  4  4  1
##   4 39 35 30 29 11
```

```
table(exang, num)
```

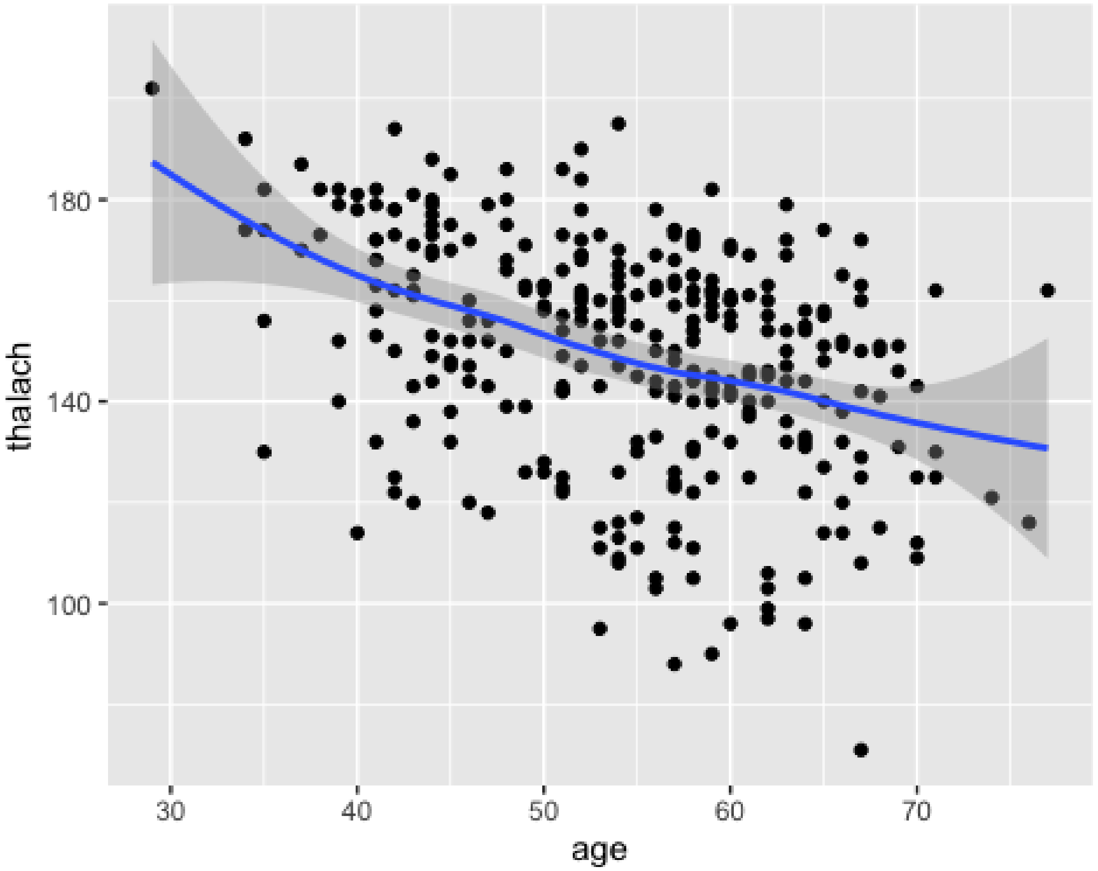
```
##       num
## exang   0   1   2   3   4
##     0 141  30  14  12   7
##     1  23  25  22  23   6
```

```
cor.test(age, thalach)
```

```
##
##  Pearson's product-moment correlation
##
## data:  age and thalach
## t = -7.4329, df = 301, p-value = 1.109e-12
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.4849644 -0.2941816
## sample estimates:
##        cor
## -0.3938058
```

```
ggplot(heart, aes(x = age, y = thalach))+
  geom_point()+
  geom_smooth()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



```r
library(caret)
```

```
## Loading required package: lattice
```

```r
set.seed(20)
Train = createDataPartition(heart$num, p = 0.7, list = FALSE)
train = heart[Train,]
test = heart[-Train,]
nrow(train)/(nrow(test) + nrow(train))
```

```
## [1] 0.7029703
```

```r
feature.names = names(heart)

for (f in feature.names) {
  if (class(heart[[f]]) == "factor"){
    levels = unique(c(heart[[f]]))
    heart[[f]] = factor(heart[[f]], labels = make.names(levels))
  }
}

heart$num = as.factor(heart$num)
levels(heart$num) = c("No Disease", "Disease")
table(heart$num)
```

```
## 
## No Disease    Disease 
##        139        164
```

```r
set.seed(10)
Train = createDataPartition(heart$num, p = 0.7, list = FALSE)
train2 = heart[Train,]
test2 = heart[-Train,]
fitControl = trainControl(method = "repeatedcv", number = 10, repeats = 10,
classProbs = TRUE, summaryFunction = twoClassSummary)
svmModel = train(num ~ ., data = na.omit(train2), scale = FALSE, kernel =
"radial", cost = 8)
svmModel
```

```
## Random Forest 
## 
## 208 samples
##  13 predictor
##   2 classes: 'No Disease', 'Disease' 
## 
## No pre-processing
## Resampling: Bootstrapped (25 reps) 
## Summary of sample sizes: 208, 208, 208, 208, 208, 208, ... 
## Resampling results across tuning parameters:
## 
##   mtry  Accuracy   Kappa
##    2    0.8284155  0.6487799
##    7    0.8030677  0.5973287
##   13    0.7842023  0.5603422
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 2.
```

```
svmPrediction = predict(svmModel, test2)
svmPredictProb = predict(svmModel, test2, type = 'prob')[2]
ConfMatrix = confusionMatrix(svmPrediction, na.omit(test2)$num)
ConfMatrix
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   No Disease Disease
##   No Disease         31       5
##   Disease            10      43
##
##                Accuracy : 0.8315
##                  95% CI : (0.7373, 0.9025)
##     No Information Rate : 0.5393
##     P-Value [Acc > NIR] : 6.345e-09
##
##                   Kappa : 0.6578
##
##  Mcnemar's Test P-Value : 0.3017
##
##             Sensitivity : 0.7561
##             Specificity : 0.8958
##          Pos Pred Value : 0.8611
##          Neg Pred Value : 0.8113
##              Prevalence : 0.4607
##          Detection Rate : 0.3483
##    Detection Prevalence : 0.4045
##       Balanced Accuracy : 0.8260
##
##        'Positive' Class : No Disease
##
```

```
AUC = roc(na.omit(test2)$num, as.numeric(as.matrix((svmPredictProb))))$auc
```

```
## Setting levels: control = No Disease, case = Disease
```

```
## Setting direction: controls < cases
```

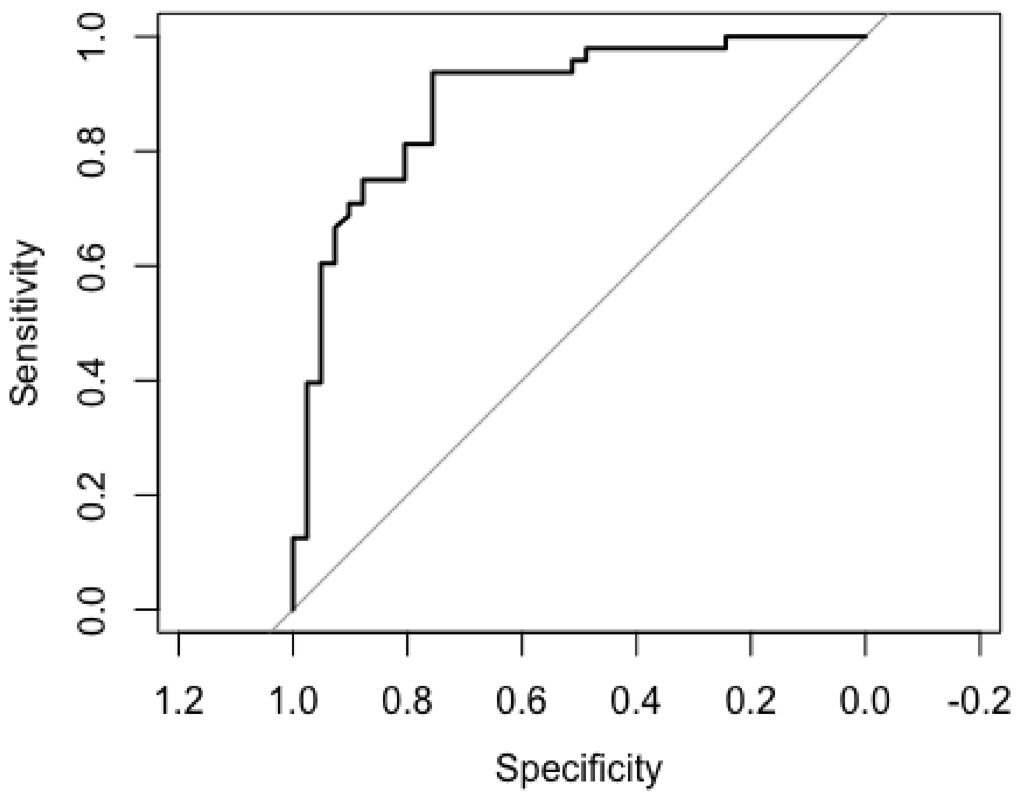
```
Accuracy = ConfMatrix$overall['Accuracy']
svmPerf = cbind(AUC, Accuracy)
svmPerf
```

```
##                AUC  Accuracy
## Accuracy 0.890498 0.8314607
```

```
aucroc = roc(na.omit(test2)$num, as.numeric(as.matrix((svmPredictProb))))
```

```
## Setting levels: control = No Disease, case = Disease
## Setting direction: controls < cases
```

```
plot(aucroc)
```



```
library(tidyverse)
```

```
## ── Attaching packages
────────────────────────────────────────────────────────────── tidyverse 1.2.1 ──

## ✔ tibble  2.1.3     ✔ purrr   0.3.2
## ✔ tidyr   1.0.0     ✔ dplyr   0.8.3
## ✔ readr   1.3.1     ✔ stringr 1.4.0
## ✔ tibble  2.1.3     ✔ forcats 0.4.0

## ── Conflicts ────────────────────────────────────────────────────────────────────
tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ✖ purrr::lift()   masks caret::lift()
```

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

```
## '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 ...
```

```r
head(heart)
```

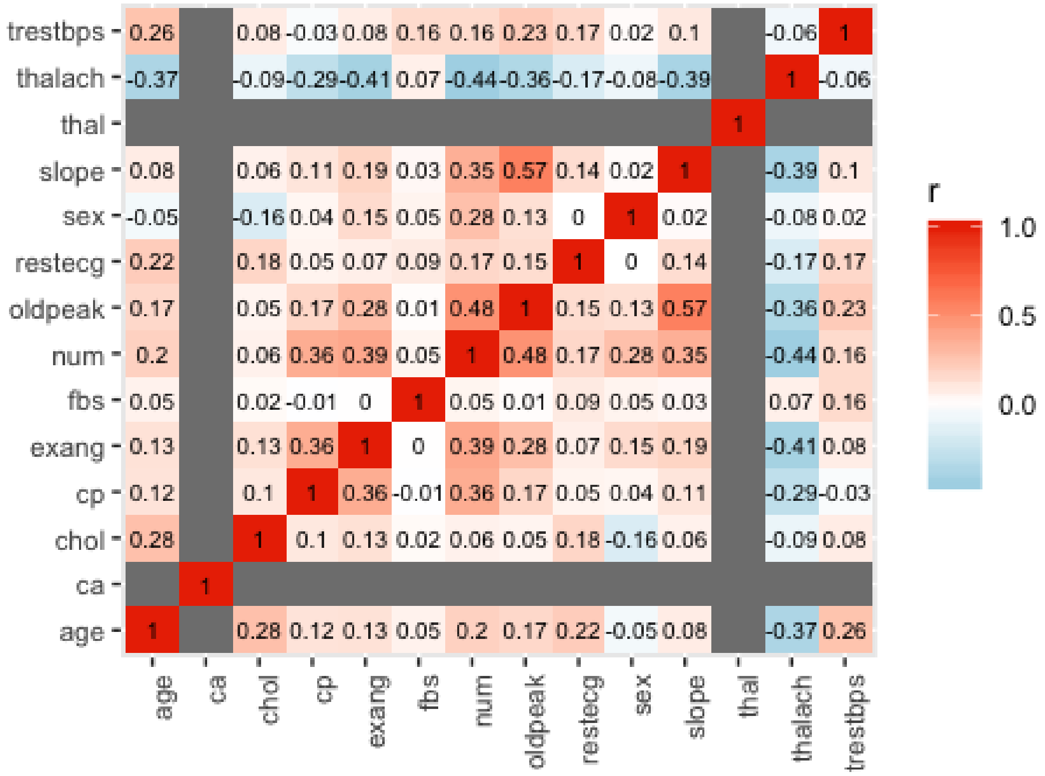
```
##   age sex cp trestbps chol fbs restecg thalach exang oldpeak slope ca thal
## 1  63   1  1      145  233   1       2     150     0     2.3     3  0    6
## 2  67   1  4      160  286   0       2     108     1     1.5     2  3    3
## 3  67   1  4      120  229   0       2     129     1     2.6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

```r
library(caTools)
set.seed(7)
split = sample.split(heart$num, SplitRatio = 0.7)
train = heart[split, ]
test = heart[!split, ]
nrow(train)
```

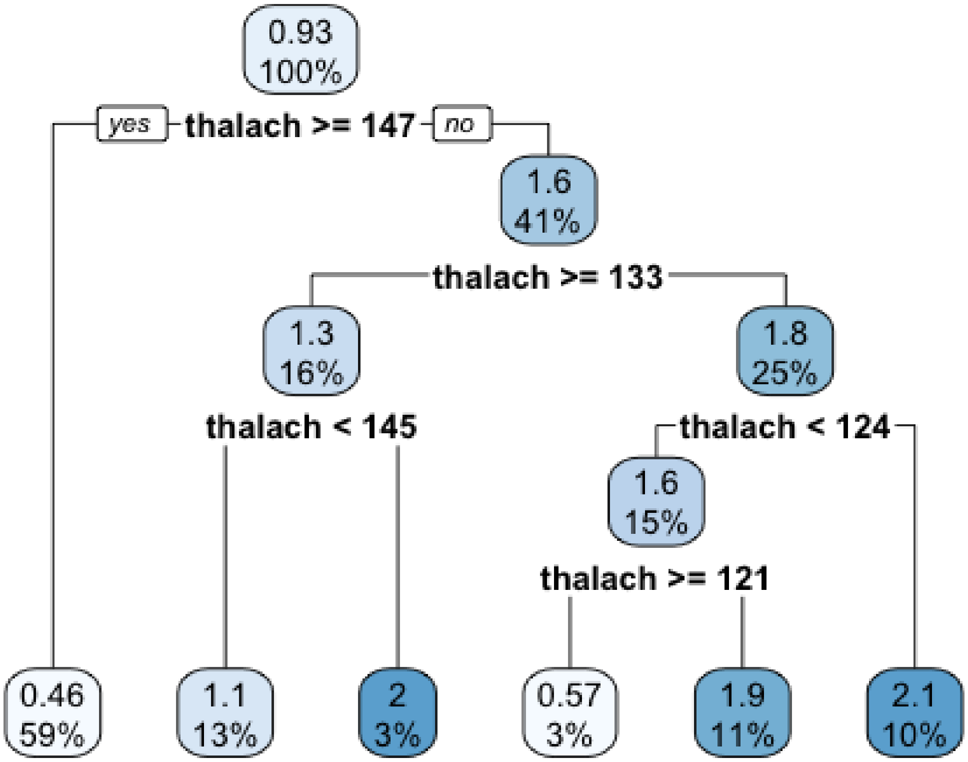
```
## [1] 211
```

```r
nrow(test)
```

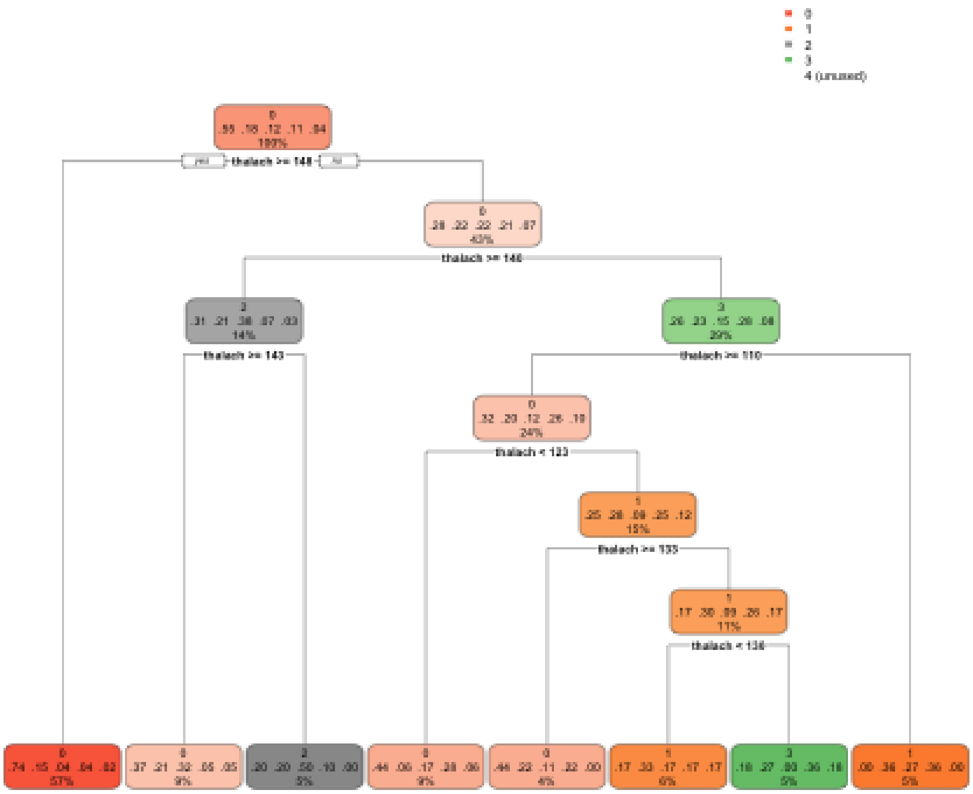
```
## [1] 92

nrow(heart)

## [1] 303

corMatrix = as.data.frame(cor(train))
corMatrix$var1 = rownames(corMatrix)
corMatrix %>%
  gather(key = var2, value = r, 1:14) %>%
  ggplot(aes(x = var1, y = var2, fill = r))+
      geom_tile()+
      geom_text(aes(label = round(r,2)), size = 2.6)+
      scale_fill_gradient2(low = "#00a6c8", high = "#eb3300", mid = "white")+
      labs(title = "", x = "", y = "")+
      theme(axis.text.x = element_text(angle = 90, hjust = 1))

## Warning: Removed 50 rows containing missing values (geom_text).
```



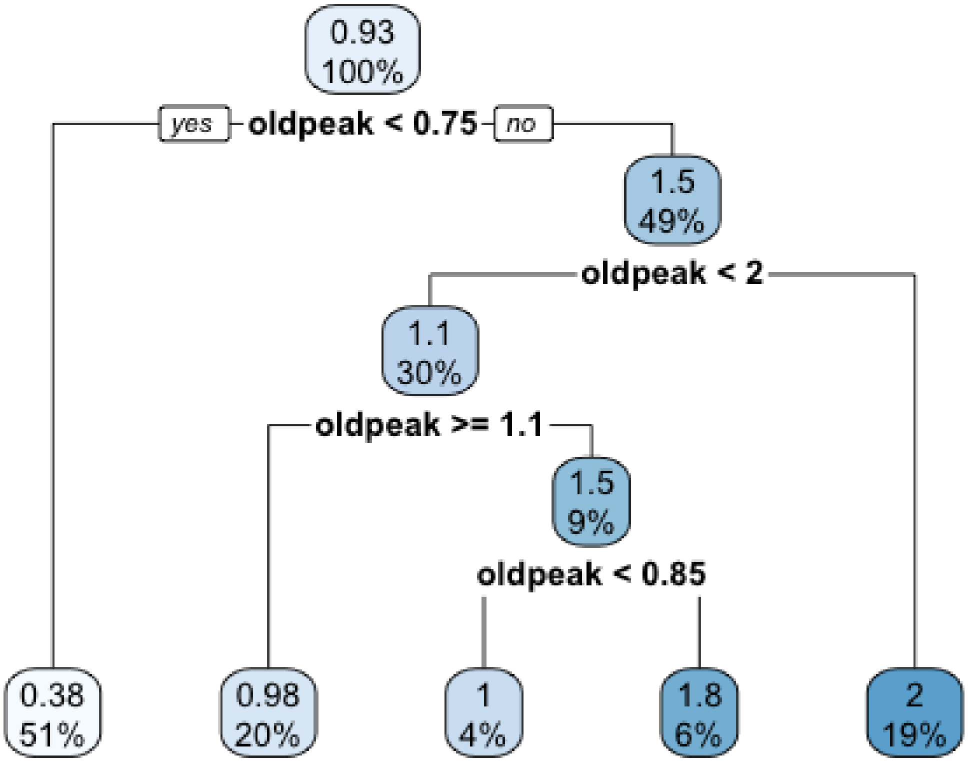
```
library(rpart)
library(rpart.plot)
regressionTree1 <- rpart(num ~ thalach, data = train, method = "anova")
rpart.plot(regressionTree1)
```



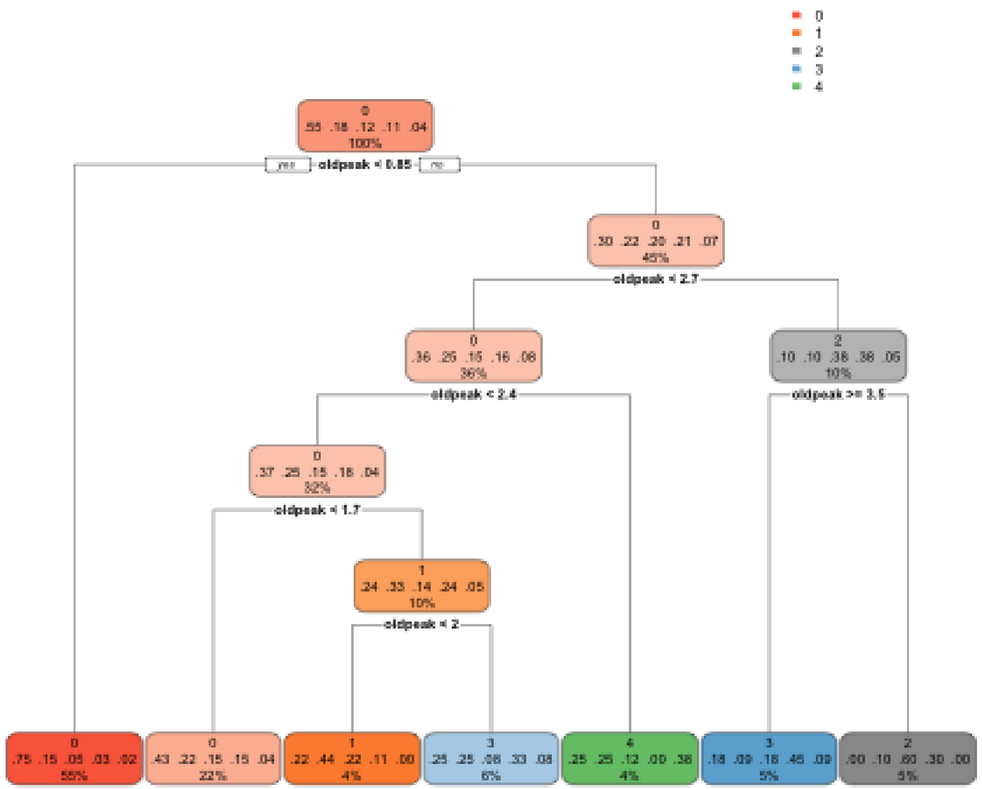
```
classificationTree1 <- rpart(num ~ thalach, data = train, method = "class")
rpart.plot(classificationTree1)
```



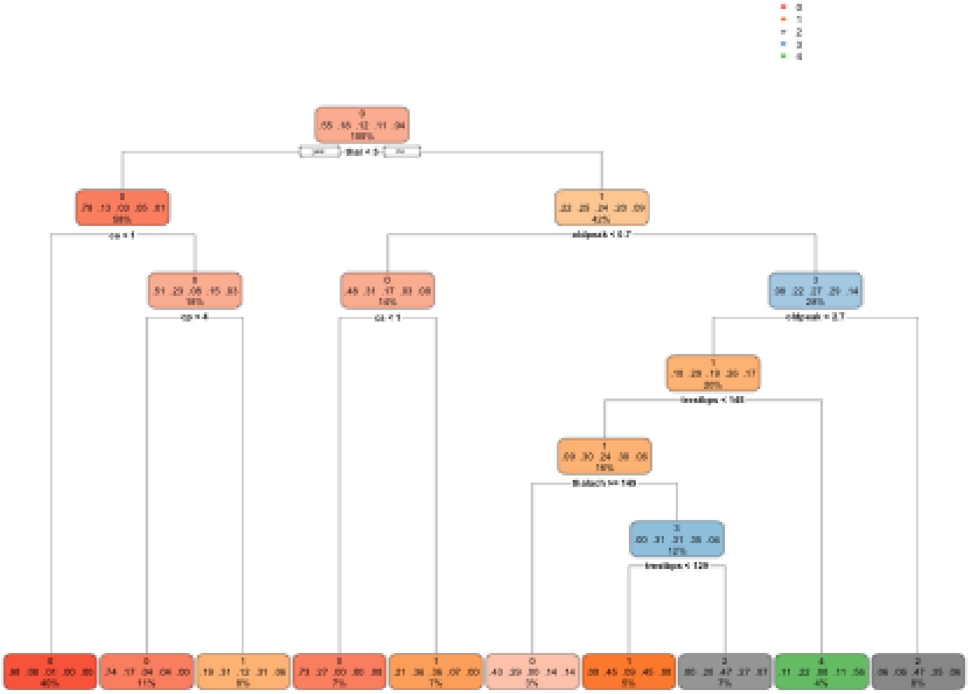
```
#summary(classificationTree1)

regressionTree2 <- rpart(num ~ oldpeak, data = train, method = "anova")
rpart.plot(regressionTree2)
```



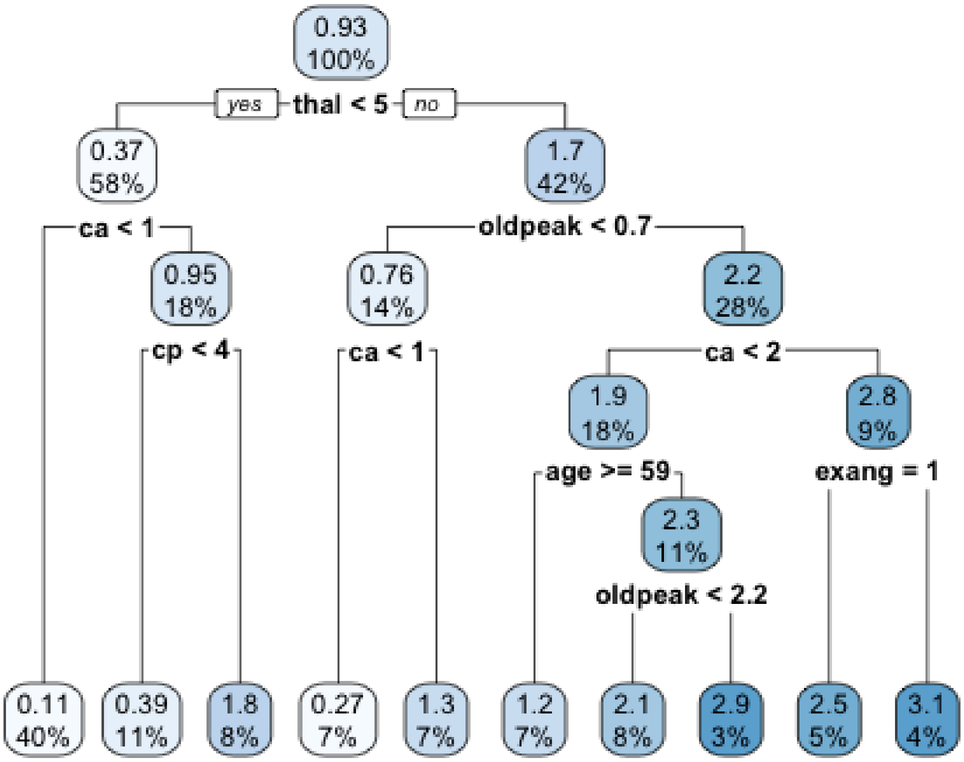
```r
#summary(regressionTree2)

classificationTree2 <- rpart(num ~ oldpeak, data = train, method = "class")
rpart.plot(classificationTree2)
```



```
#summary(classificationTree2)

classificationtree = rpart(num ~ ., data = train, method = "class")
rpart.plot(classificationtree)
```



```r
#summary(classificationtree)

regressiontree = rpart(num ~ ., data = train, method = "anova", cp = 0.005)
rpart.plot(regressiontree)
```



```
#summary(regressiontree)
```