Homework 3

Vikas Chand Gunnam

11/23/2021

Answer all questions:

Problem 1: Logistic Regression This question should be answered using the ***"Banknote Authentication"*** data set. Description about the data set can be found on the link provided. Objective of this question is to fit an logistic regression model to classify forged banknote from genuine banknotes. (Presumably 0 for genuine and 1 for forged bank notes)

a. Produce some numerical and graphical summaries of the data set. Explain the relationships.

b. Is this a balanced data set?.

c. Use the full data set to perform a logistic regression with *Class* as the response variable. Do any of the predictors appear to be statistically significant? If so, which ones?

d. Compute the confusion matrix and overall fraction of correct predictions. Explain what the confusion matrix id telling you about the types of mistakes made by logistic regression.

e. Create a training set with 80% of the observations, and a testing set containing the remaining 20%.Compute the confusion matrix and the overall fraction of correct prediction for the testing data set.

f. Produce some numerical and graphical summaries of the data set. Explain the relationships.

```
library(ISLR)
library(GGally)
```

```
## Loading required package: ggplot2
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
library(corrplot)
```

```
## corrplot 0.90 loaded
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v tibble  3.1.4     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   2.0.1     v forcats 0.5.1
## v purrr   0.3.4

## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library("reshape2")
```

```
##
## Attaching package: 'reshape2'

## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
hw_3 <- read.csv("/Users/Vikas/BankNote.csv")
names(hw_3)
```

```
## [1] "Variance" "skewness" "curtosis" "entropy"  "class"
```

```
dim(hw_3)
```

```
## [1] 1372    5
```

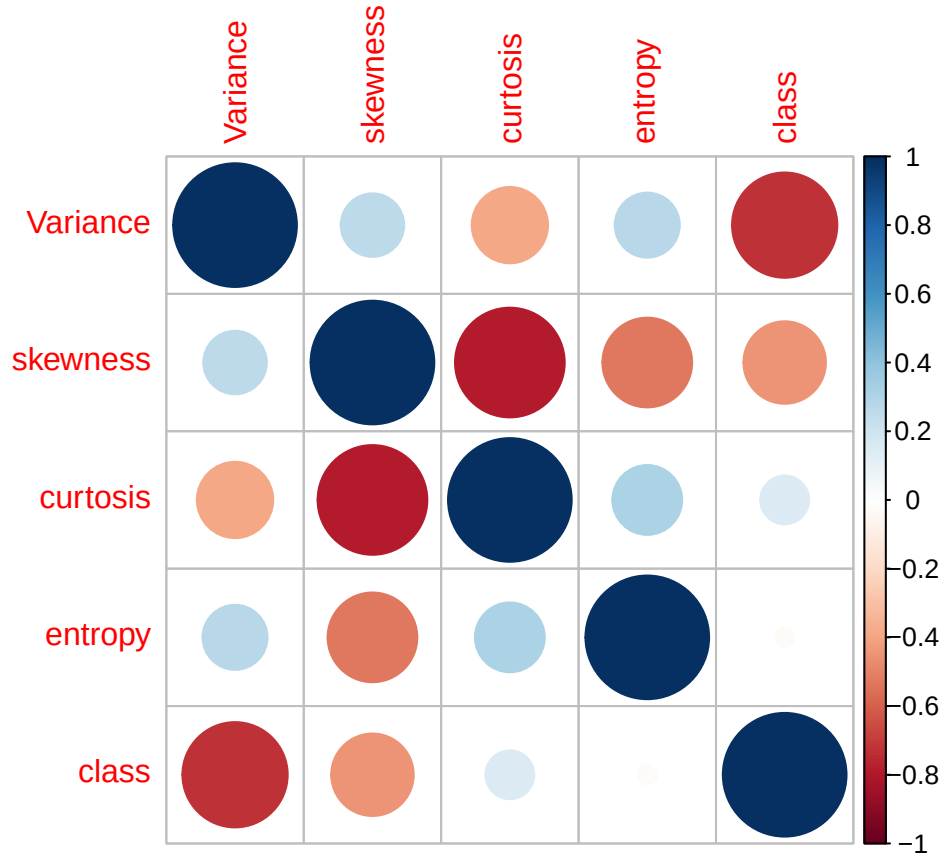
```
str(hw_3)
```

```
## 'data.frame':    1372 obs. of  5 variables:
##  $ Variance: num  3.622 4.546 3.866 3.457 0.329 ...
##  $ skewness: num  8.67 8.17 -2.64 9.52 -4.46 ...
##  $ curtosis: num  -2.81 -2.46 1.92 -4.01 4.57 ...
##  $ entropy : num  -0.447 -1.462 0.106 -3.594 -0.989 ...
##  $ class   : int  0 0 0 0 0 0 0 0 0 0 ...
```

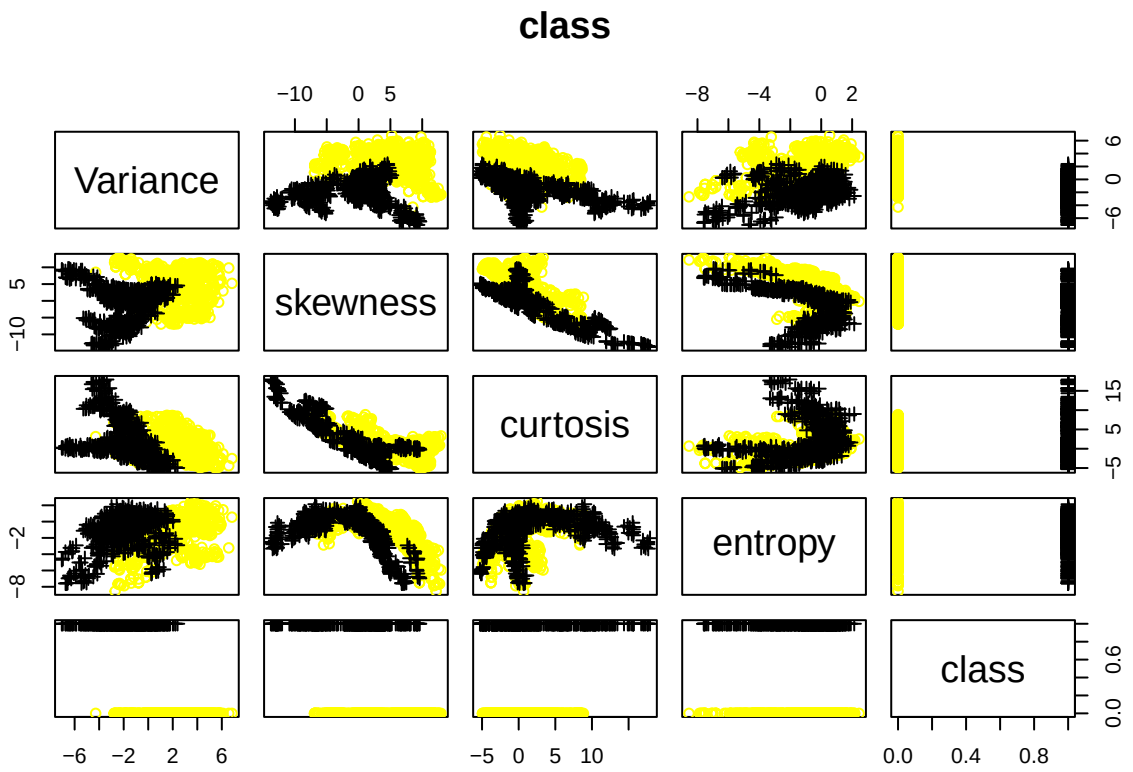
```
summary(hw_3)
```

```
##     Variance          skewness          curtosis          entropy
##  Min.   :-7.0421   Min.   :-13.773   Min.   :-5.2861   Min.   :-8.5482
##  1st Qu.:-1.7730   1st Qu.: -1.708   1st Qu.:-1.5750   1st Qu.:-2.4135
##  Median : 0.4962   Median :  2.320   Median : 0.6166   Median :-0.5867
##  Mean   : 0.4337   Mean   :  1.922   Mean   : 1.3976   Mean   :-1.1917
##  3rd Qu.: 2.8215   3rd Qu.:  6.815   3rd Qu.: 3.1793   3rd Qu.: 0.3948
##  Max.   : 6.8248   Max.   : 12.952   Max.   :17.9274   Max.   : 2.4495
##      class
##  Min.   :0.0000
##  1st Qu.:0.0000
##  Median :0.0000
##  Mean   :0.4446
##  3rd Qu.:1.0000
##  Max.   :1.0000
```

```
corrplot(cor(hw_3[,-6]), method="circle")
```

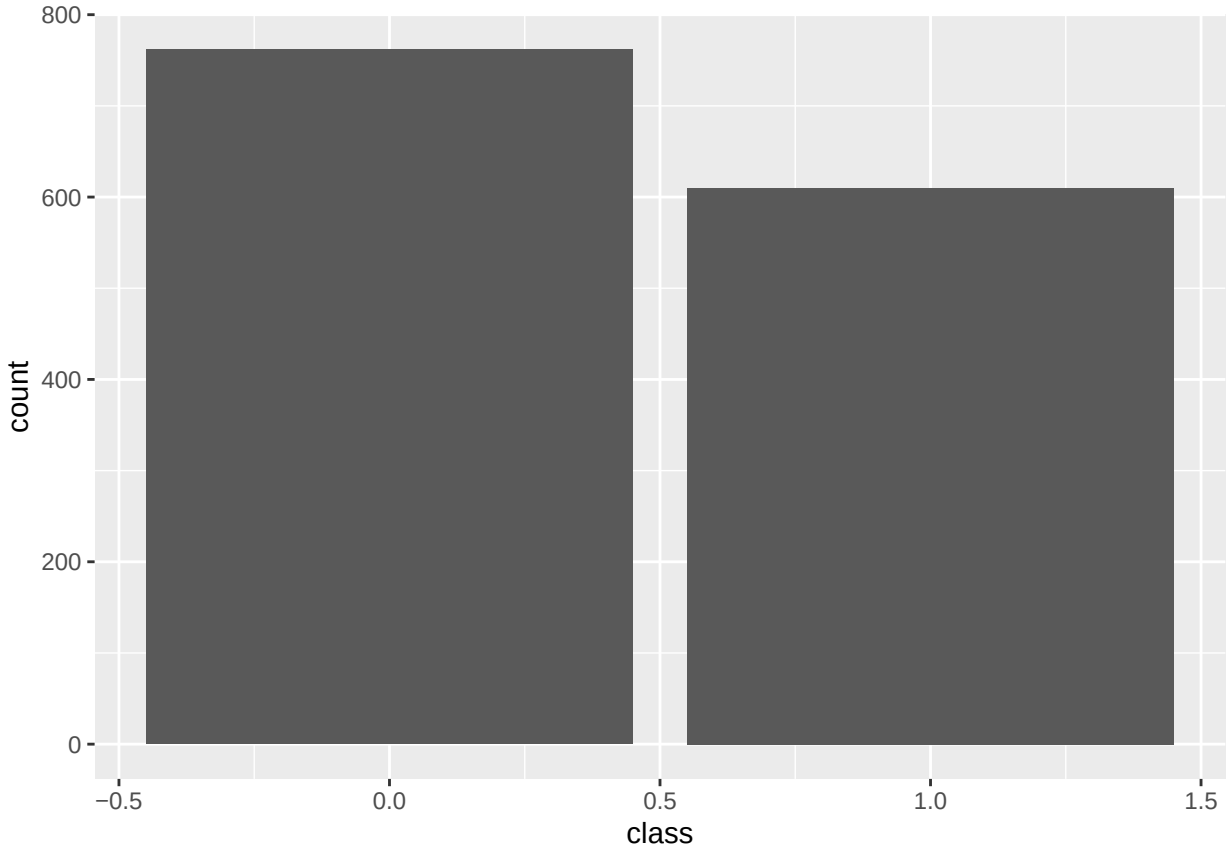



```
pairs(hw_3[], main = "class" , pch = c(1,3)[(unclass(factor(hw_3$class)))],
      col = c("yellow","black")[unclass(factor(hw_3$class))])
```



b. Is this a balanced data set? Yes this dataset is balanced. The number of instances on both the classes are almost equal.

```
ggplot(data=hw_3,aes(x=class))+geom_bar()
```



c. Use the full data set to perform a logistic regression with *Class* as the response variable. Do any of the predictors appear to be statistically significant? If so, which ones?

```
glm.balance = glm(class~.,data=hw_3,family="binomial")
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(glm.balance )
```

```
##
## Call:
## glm(formula = class ~ ., family = "binomial", data = hw_3)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.70001   0.00000   0.00000   0.00029   2.24614
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   7.3218     1.5589   4.697 2.64e-06 ***
## Variance     -7.8593     1.7383  -4.521 6.15e-06 ***
## skewness     -4.1910     0.9041  -4.635 3.56e-06 ***
## curtosis     -5.2874     1.1612  -4.553 5.28e-06 ***
## entropy      -0.6053     0.3307  -1.830   0.0672 .
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1885.122  on 1371  degrees of freedom
## Residual deviance:   49.891  on 1367  degrees of freedom
## AIC: 59.891
##
## Number of Fisher Scoring iterations: 12
```

d. Compute the confusion matrix and overall fraction of correct predictions. Explain what the confusion matrix is telling you about the types of mistakes made by logistic regression.

```
glm.prob = predict(glm.balance,type="response")
glm.prob[1:10]
```

```
##            1            2            3            4            5            6
## 2.220446e-16 2.220446e-16 2.185822e-10 2.220446e-16 4.579103e-01 2.220446e-16
##            7            8            9           10
## 2.220446e-16 1.435064e-11 2.220446e-16 2.220446e-16
```

```
contrasts(factor(hw_3$class))
```

```
##   1
## 0 0
## 1 1
```

```
glm.pred=rep("No",nrow(hw_3))
glm.pred[glm.prob>.5]="Yes"

table(glm.pred,hw_3$class)
```

```
##
## glm.pred   0   1
##      No  757   6
##      Yes   5 604
```

e. Create a training set with 80% of the observations, and a testing set containing the remaining 20%.Compute the confusion matrix and the overall fraction of correct prediction for the testing data set.

```
set.seed(272)
train=sample(c(TRUE,FALSE),size=nrow(hw_3),prob=c(0.8,0.2),replace=TRUE)
hw_3.test=hw_3[!train,]
dim(hw_3.test)
```

```
## [1] 288   5
```

```
model1=glm(class~.,data=hw_3,family=binomial,subset=train)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
glm.probs=predict(glm.balance,hw_3.test,type="response")
glm.probs[1:10]
```

```
##            1            4            7           11           12           17
## 2.220446e-16 2.220446e-16 2.220446e-16 2.078294e-12 5.780712e-12 2.982926e-13
##           20           26           37           45
## 2.220446e-16 2.450794e-08 2.220446e-16 2.220446e-16
```

```
glm.pred=rep("No",nrow(hw_3.test))
glm.pred[glm.probs>.5]="Yes"

table(glm.pred,hw_3.test$class)
```

```
##
## glm.pred   0   1
##      No  159   1
##      Yes   1 127
```

Problem 2: Tree based models This question should be answered using the ***"Wine Quality"*** data set. Description about the data set can be found on the link provided. Objective of this question is to fit an regression tree model to predict quality of wine.

```
wine_quality <- read.csv("/Users/Vikas/winequality.csv")
```

a. Produce some numerical and graphical summaries of the data set. Explain the relationships.

```
library(ISLR)
library(GGally)
library(corrplot)
library(tidyverse)
library("reshape2")
names(wine_quality)
```

```
##  [1] "fixed.acidity"        "volatile.acidity"     "citric.acid"
##  [4] "residual.sugar"       "chlorides"            "free.sulfur.dioxide"
##  [7] "total.sulfur.dioxide" "density"              "pH"
## [10] "sulphates"            "alcohol"              "quality"
```

```
dim(wine_quality)
```

```
## [1] 4898   12
```

```
str(wine_quality)
```

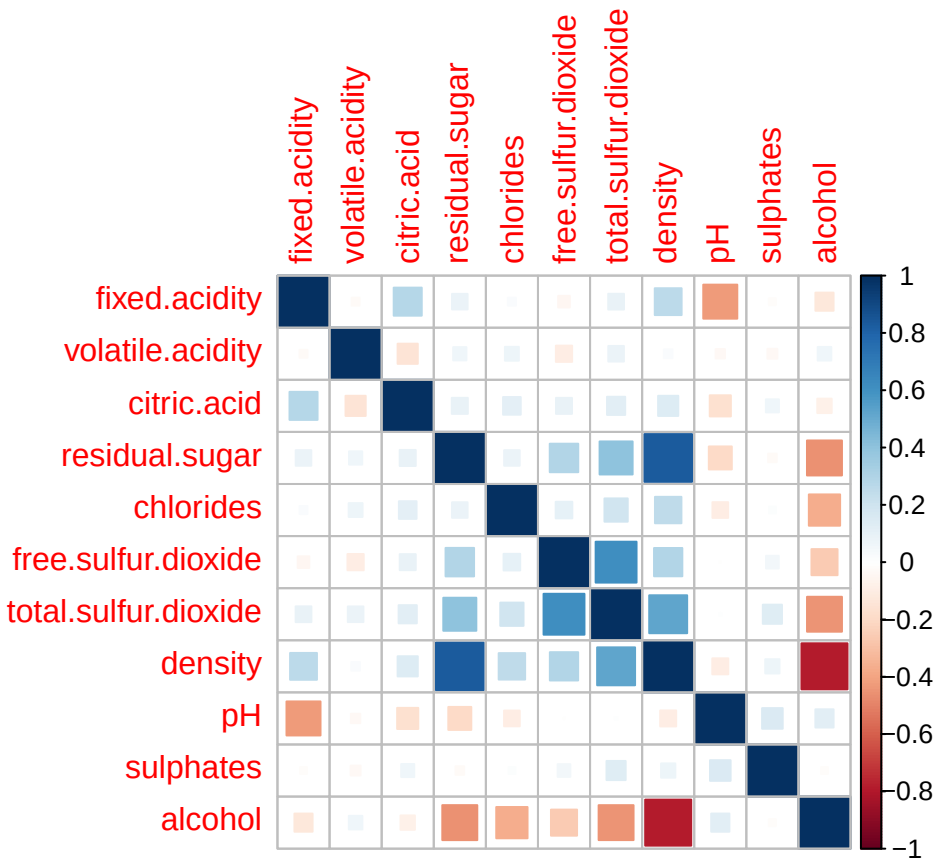
```
## 'data.frame':    4898 obs. of  12 variables:
##  $ fixed.acidity       : num  7 6.3 8.1 7.2 7.2 8.1 6.2 7 6.3 8.1 ...
##  $ volatile.acidity    : num  0.27 0.3 0.28 0.23 0.23 0.28 0.32 0.27 0.3 0.22 ...
##  $ citric.acid         : num  0.36 0.34 0.4 0.32 0.32 0.4 0.16 0.36 0.34 0.43 ...
##  $ residual.sugar      : num  20.7 1.6 6.9 8.5 8.5 6.9 7 20.7 1.6 1.5 ...
```

```
##  $ chlorides           : num  0.045 0.049 0.05 0.058 0.058 0.05 0.045 0.045 0.049 0.044 ...
##  $ free.sulfur.dioxide : num  45 14 30 47 47 30 30 45 14 28 ...
##  $ total.sulfur.dioxide: num  170 132 97 186 186 97 136 170 132 129 ...
##  $ density             : num  1.001 0.994 0.995 0.996 0.996 ...
##  $ pH                  : num  3 3.3 3.26 3.19 3.19 3.26 3.18 3 3.3 3.22 ...
##  $ sulphates           : num  0.45 0.49 0.44 0.4 0.4 0.44 0.47 0.45 0.49 0.45 ...
##  $ alcohol             : num  8.8 9.5 10.1 9.9 9.9 10.1 9.6 8.8 9.5 11 ...
##  $ quality             : int  6 6 6 6 6 6 6 6 6 6 ...
```

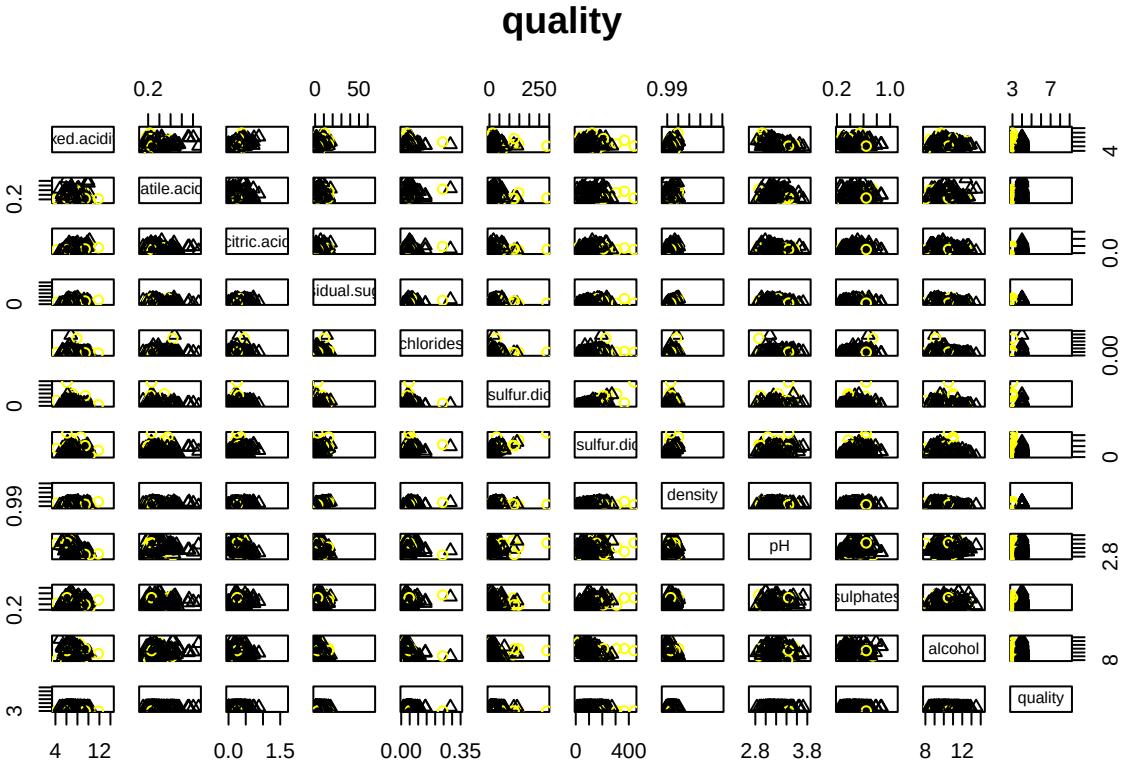
```
summary(wine_quality)
```

```
##  fixed.acidity    volatile.acidity  citric.acid     residual.sugar
##  Min.   : 3.800   Min.   :0.0800   Min.   :0.0000   Min.   : 0.600
##  1st Qu.: 6.300   1st Qu.:0.2100   1st Qu.:0.2700   1st Qu.: 1.700
##  Median : 6.800   Median :0.2600   Median :0.3200   Median : 5.200
##  Mean   : 6.855   Mean   :0.2782   Mean   :0.3342   Mean   : 6.391
##  3rd Qu.: 7.300   3rd Qu.:0.3200   3rd Qu.:0.3900   3rd Qu.: 9.900
##  Max.   :14.200   Max.   :1.1000   Max.   :1.6600   Max.   :65.800
##    chlorides       free.sulfur.dioxide total.sulfur.dioxide    density
##  Min.   :0.00900   Min.   :  2.00      Min.   :  9.0        Min.   :0.9871
##  1st Qu.:0.03600   1st Qu.: 23.00      1st Qu.:108.0        1st Qu.:0.9917
##  Median :0.04300   Median : 34.00      Median :134.0        Median :0.9937
##  Mean   :0.04577   Mean   : 35.31      Mean   :138.4        Mean   :0.9940
##  3rd Qu.:0.05000   3rd Qu.: 46.00      3rd Qu.:167.0        3rd Qu.:0.9961
##  Max.   :0.34600   Max.   :289.00      Max.   :440.0        Max.   :1.0390
##        pH          sulphates         alcohol         quality
##  Min.   :2.720   Min.   :0.2200   Min.   : 8.00   Min.   :3.000
##  1st Qu.:3.090   1st Qu.:0.4100   1st Qu.: 9.50   1st Qu.:5.000
##  Median :3.180   Median :0.4700   Median :10.40   Median :6.000
##  Mean   :3.188   Mean   :0.4898   Mean   :10.51   Mean   :5.878
##  3rd Qu.:3.280   3rd Qu.:0.5500   3rd Qu.:11.40   3rd Qu.:6.000
##  Max.   :3.820   Max.   :1.0800   Max.   :14.20   Max.   :9.000
```

```
corrplot(cor(wine_quality[,-12]), method="square")
```




```
pairs(wine_quality[], main = "quality" , pch = c(1,2,3,4,5,6,7,8,9,10,11,12)[(unclass(factor(wine_quali
      col = c("yellow","black")[unclass(factor(wine_quality$quality))])
```



b. Create a training set with 80% of the observations, and a testing set containing the remaining 20%.

```
set.seed(272)
train=sample(c(TRUE,FALSE),size=nrow(wine_quality),prob=c(0.8,0.2),replace=TRUE)
wine_quality.test=wine_quality[!train,]
dim(wine_quality.test)
```

```
## [1] 998  12
```

```
model2=glm(quality~.,data=wine_quality,subset=train)
```

c. Fit a regression tree with *quality* as the response variable using the training set. Plot the tree and interpret the results. What test MSE do you obtain?

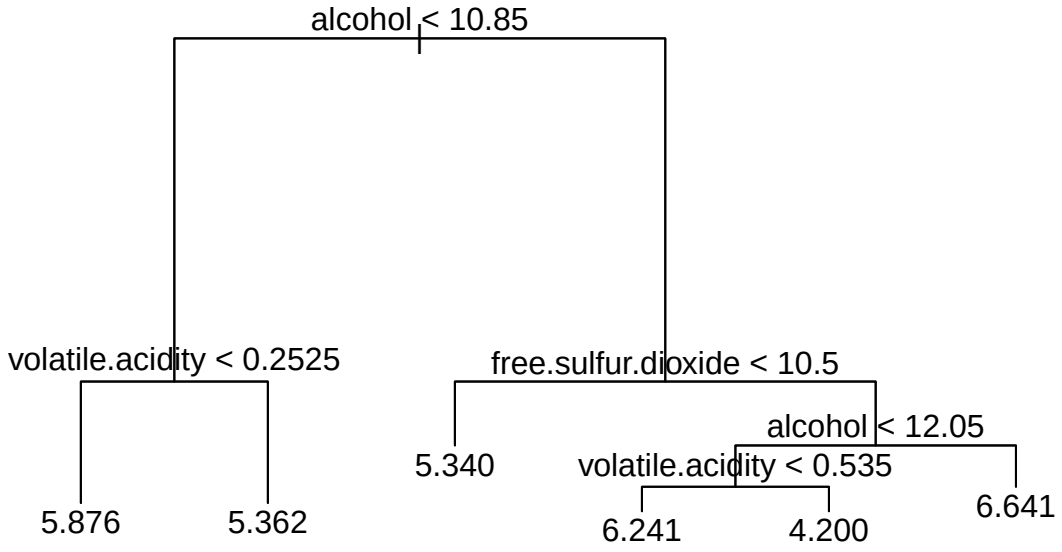
```
library(tree)
```

```
## Registered S3 method overwritten by 'tree':
##   method     from
##   print.tree cli
```

```
set.seed(1)
training = sample(1:nrow(wine_quality), nrow(wine_quality)/2)
e=tree(quality~.,wine_quality,subset=training)
summary(e)
```

```
##
## Regression tree:
## tree(formula = quality ~ ., data = wine_quality, subset = training)
## Variables actually used in tree construction:
## [1] "alcohol"             "volatile.acidity"    "free.sulfur.dioxide"
## Number of terminal nodes:  6
## Residual mean deviance:  0.5862 = 1432 / 2443
## Distribution of residuals:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -2.8760 -0.3618 -0.2409  0.0000  0.6382  2.6600
```

```
plot(e)
text(e,pretty=0)
```




```
X=predict(e,newdata=wine_quality[-training,])
Y=wine_quality[-training,"quality"]
mean((X-Y)^2)
```

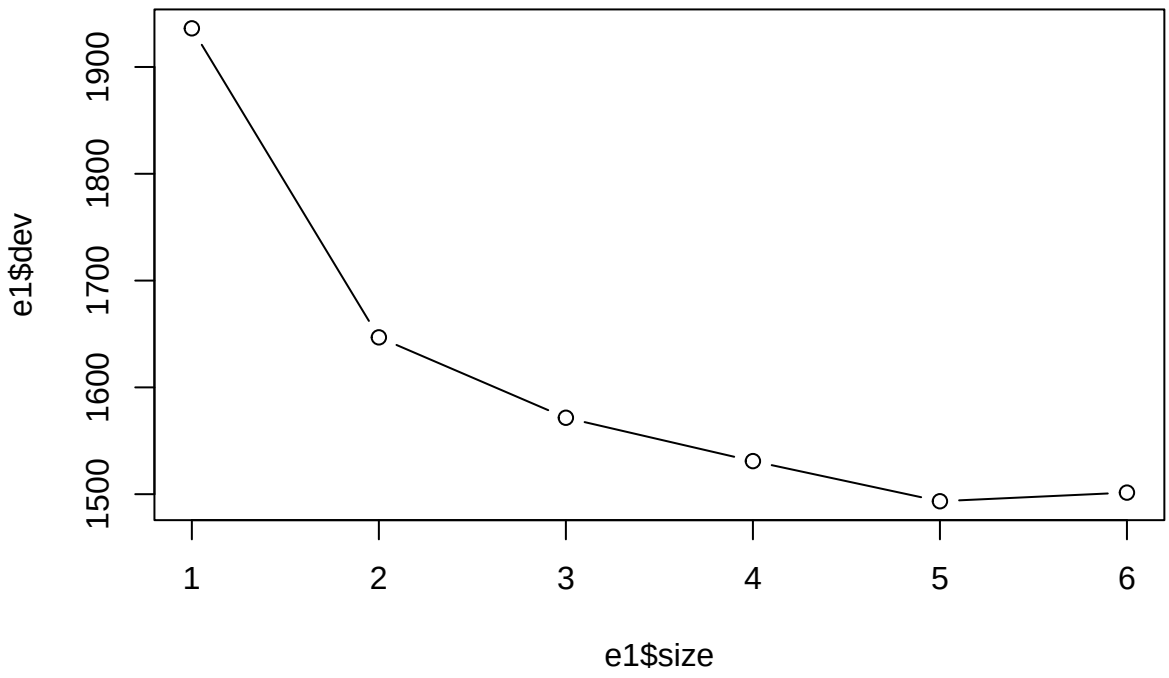
```
## [1] 0.5703777
```

d. Use cross-validation in order to determine the optimal level of tree complexity. Does pruning the tree improve the test MSE?

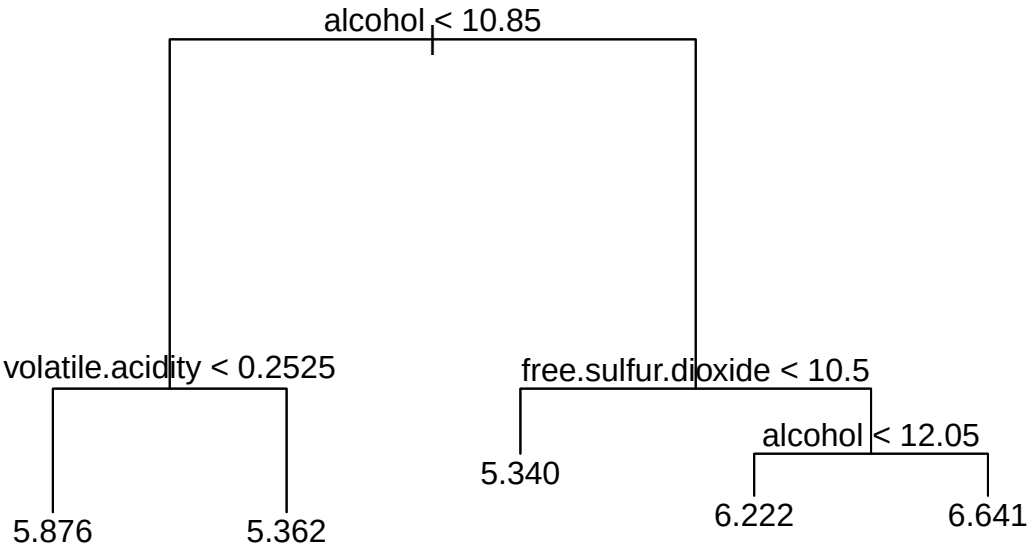
```
head(wine_quality)
```

```
##   fixed.acidity volatile.acidity citric.acid residual.sugar chlorides
## 1           7.0             0.27        0.36           20.7     0.045
## 2           6.3             0.30        0.34            1.6     0.049
## 3           8.1             0.28        0.40            6.9     0.050
## 4           7.2             0.23        0.32            8.5     0.058
## 5           7.2             0.23        0.32            8.5     0.058
## 6           8.1             0.28        0.40            6.9     0.050
##   free.sulfur.dioxide total.sulfur.dioxide density   pH sulphates alcohol
## 1                  45                  170  1.0010 3.00      0.45     8.8
## 2                  14                  132  0.9940 3.30      0.49     9.5
## 3                  30                   97  0.9951 3.26      0.44    10.1
## 4                  47                  186  0.9956 3.19      0.40     9.9
## 5                  47                  186  0.9956 3.19      0.40     9.9
## 6                  30                   97  0.9951 3.26      0.44    10.1
##   quality
## 1       6
## 2       6
## 3       6
## 4       6
## 5       6
## 6       6
```

```
e1 = cv.tree(e)
plot (e1$size,e1$dev,type='b')
```




```
prune.model=prune.tree(e,best=5)
plot(prune.model)
text(prune.model,pretty=0)
```



e. Use random forests to analyze this data. What test MSE do you obtain?

```
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:dplyr':
##
##     combine

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

```
set.seed(1)
rf.boston=randomForest(quality~.,data=wine_quality,subset=train,mtry=6,importance=TRUE)
yhat.rf = predict(rf.boston,wine_quality.test)
mean((yhat.rf-wine_quality.test$quality)^2)
```

```
## [1] 0.3480268
```

f. Use the *importance()* function to determine which variables are most important.

```
importance(rf.boston)
```

```
##                        %IncMSE IncNodePurity
## fixed.acidity         45.16204      182.5464
## volatile.acidity     108.81668      331.1563
## citric.acid           46.68103      184.1549
## residual.sugar        49.27096      219.7038
## chlorides             48.08844      225.3104
## free.sulfur.dioxide   80.72993      323.2359
## total.sulfur.dioxide  52.09243      214.1285
## density               35.06721      301.6101
## pH                    60.02288      208.1337
## sulphates             50.26623      173.9504
## alcohol               95.45544      609.5946
```