INFX 573 Final Exam

Pierre Augustamar

Due: Tueday, November 29, 2016

Setup:

```
#load  required libraries
library(tidyverse)
library(AER)
library(pROC)
library(randomForest)
library(bestglm)
library(ggfortify)
library(ISLR)
library(MASS)
library(rpart)
library(DAAG)
library(data.table)
```

Problem 1:

In this problem we will use data about infidelitys, known as the Fair's Affairs dataset. The `Affairs` dataset is available as part of the AER package in R. This data comes from a survey conducted by Psychology Today in 1969, see Greene (2003) and Fair (1978) for more information.

The dataset contains various self-reported characteristics of 601 participants, including how often the respondent engaged in extramarital sexual intercourse during the past year, as well as their gender, age, year married, whether they had children, their religiousness (on a 5-point scale, from 1=anti to 5=very), education, occupation (Hillinghead 7-point classification with reverse numbering), and a numeric self-rating of their marriage (from 1=very unhappy to 5=very happy).

Explore the data

```
data("Affairs") # Get the Fair's affairs dataset
head(Affairs) # View the first 5 records from this data set
```

```
##    affairs gender age yearsmarried children religiousness education
## 4        0   male  37        10.00       no             3        18
## 5        0 female  27         4.00       no             4        14
## 11       0 female  32        15.00      yes             1        12
## 16       0   male  57        15.00      yes             5        18
## 23       0   male  22         0.75       no             2        17
## 29       0 female  32         1.50       no             2        17
##    occupation rating
## 4           7      4
## 5           6      4
## 11          1      4
## 16          6      5
## 23          6      3
## 29          5      5
```

Check Data scruture and data types

```r
str(Affairs) # Show object's data type
```

```
## 'data.frame':    601 obs. of  9 variables:
##  $ affairs      : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ gender       : Factor w/ 2 levels "female","male": 2 1 1 2 2 1 1 2 1 2 ...
##  $ age          : num  37 27 32 57 22 32 22 57 32 22 ...
##  $ yearsmarried : num  10 4 15 15 0.75 1.5 0.75 15 15 1.5 ...
##  $ children     : Factor w/ 2 levels "no","yes": 1 1 2 2 1 1 1 2 2 1 ...
##  $ religiousness: int  3 4 1 5 2 2 2 2 4 4 ...
##  $ education    : num  18 14 12 18 17 17 12 14 16 14 ...
##  $ occupation   : int  7 6 1 6 6 5 1 4 1 4 ...
##  $ rating       : int  4 4 4 5 3 5 3 4 2 5 ...
```

```r
summary(Affairs) # Generate a summary of the dataset
```

```
##     affairs          gender         age         yearsmarried    children
##  Min.   : 0.000   female:315   Min.   :17.50   Min.   : 0.125   no :171
##  1st Qu.: 0.000   male  :286   1st Qu.:27.00   1st Qu.: 4.000   yes:430
##  Median : 0.000                Median :32.00   Median : 7.000
##  Mean   : 1.456                Mean   :32.49   Mean   : 8.178
##  3rd Qu.: 0.000                3rd Qu.:37.00   3rd Qu.:15.000
##  Max.   :12.000                Max.   :57.00   Max.   :15.000
##  religiousness     education       occupation        rating
##  Min.   :1.000   Min.   : 9.00   Min.   :1.000   Min.   :1.000
##  1st Qu.:2.000   1st Qu.:14.00   1st Qu.:3.000   1st Qu.:3.000
##  Median :3.000   Median :16.00   Median :5.000   Median :4.000
##  Mean   :3.116   Mean   :16.17   Mean   :4.195   Mean   :3.932
##  3rd Qu.:4.000   3rd Qu.:18.00   3rd Qu.:6.000   3rd Qu.:5.000
##  Max.   :5.000   Max.   :20.00   Max.   :7.000   Max.   :5.000
```

```r
# Using the data frame locally
affairs = as_data_frame(Affairs)
```

Variable	Description
affairs	How often engaged in extramarital sexual intercourse during the past year?
gender	factor indicating gender.
age	numeric variable coding age in years: 17.5 = under 20, 22 = 20-24, 27 = 25-29, 32 = 30-34, 37 = 35-39, 42 = 40-44, 47 = 45-49, 52 = 50-54, 57 = 55 or over.
yearsmarried	numeric variable coding number of years married: 0.125 = 3 months or less, 0.417 = 4-6 months, 0.75 = 6 months-1 year, 1.5 = 1-2 years, 4 = 3-5 years, 7 = 6-8 years, 10 = 9-11 years, 15 = 12 or more years.
children	factor. Are there children in the marriage?
religiousness	numeric variable coding religiousness: 1 = anti, 2 = not at all, 3 = slightly, 4 = somewhat, 5 = very.
education	numeric variable coding level of education: 9 = grade school, 12 = high school graduate, 14 = some college, 16 = college graduate, 17 = some graduate work, 18 = master's degree, 20 = Ph.D., M.D., or other advanced degree.
occupation	numeric variable coding occupation according to Hollingshead classification (reverse numbering).
rating	numeric variable coding self rating of marriage: 1 = very unhappy, 2 = somewhat unhappy, 3 = average, 4 = happier than average, 5 = very happy.

Table 1: Description of variables in the Fair's Extramarital Affairs Data

(a) Describe the participants. Use descriptive, summarization, and exploratory techniques to describe the participants in the study. For example, what proportion of respondents are female? What is the average age of respondents?

Solution: The data set contains 601 observations with 9 variables. Among those participants the summary analysis shows that 38.60 percent have a very happy marriage, 2.66 percent were very unhappy in their marriage, rougly 1 percent were either at or under 20 years old, 3.66 percent where either at or above 55 years old. Also, 28.46 percent of the participants did not have children, while 71.60 percent did have children. 11.65 percent considered themselves very religious while roughly 8% considered themselves atheist. In the education forum, 31.95 percent either had a Masters or higher education while 7.32% only had a high-school diploma. In the overall category, 75% of all the participants did not engage in any extramarital affairs, 2.16 percent very religious participants had an affair while 3.32 percent of atheist had an affair, 8.82 percent of participants with an advanced degree had an affair while 2.16 percent of people with a high school diploma had an affair. Meanwhile, roughly 0.50 percent of people under 20 had an affair, and similarly 0.50% of people 55 and older had an affair. In the gender specific, 52.41 percent considered themselves a female, 27.3 percent of male participants had an affair while 22.9 percent of women had an affair.

```r
# Summarize participants' marriage rating and added a description label for the marriage rating
group_rating = affairs %>%
                group_by(rating) %>%
                summarise(frequency = n()) %>%
                arrange(desc(frequency))

# Map marriage's rating with actual description
marriage_rating_coding = group_rating %>%
                  mutate(marriage_rating=ifelse(rating==5, "Very happy",
                                         ifelse(rating==4, "happier than average",
```

```r
                                         ifelse(rating==3, "average",
                                         ifelse(rating==2, "somewhat unhappy",
                                         ifelse(rating==1, "very unhappy",NA)
                                         )))))

# Show frequency of marriage rating
print(marriage_rating_coding)
```

```
## # A tibble: 5 × 3
##   rating frequency      marriage_rating
##    <int>     <int>                <chr>
## 1      5       232           Very happy
## 2      4       194 happier than average
## 3      3        93              average
## 4      2        66     somewhat unhappy
## 5      1        16         very unhappy
```

```r
# Generate a summary of the marriage rating
summary(marriage_rating_coding)
```

```
##      rating    frequency     marriage_rating
##  Min.   :1   Min.   : 16.0   Length:5
##  1st Qu.:2   1st Qu.: 66.0   Class :character
##  Median :3   Median : 93.0   Mode  :character
##  Mean   :3   Mean   :120.2
##  3rd Qu.:4   3rd Qu.:194.0
##  Max.   :5   Max.   :232.0
```

```r
# Calculate percentage of participants that have a very happy marriage
percentage.happy.marriage = (nrow(filter(affairs, rating==5))/nrow(affairs))*100
print(percentage.happy.marriage)
```

```
## [1] 38.60233
```

```r
# Calculate percentage of particants that have a very unhappy marriage
percentage.unhappy.marriage = (nrow(filter(affairs, rating==1))/nrow(affairs))*100
print(percentage.unhappy.marriage)
```

```
## [1] 2.66223
```

```r
# Summarize participant's age and added a description label for the marriage coding
group_age  = affairs %>%
             group_by(age) %>%
             summarise(frequency=n()) %>%
             arrange(desc(frequency))

marriage_age_coding  = group_age %>%
                  mutate(age_level=ifelse(age <=20, "under 20",
                                ifelse(age >20 & age <=24, "20-24",
                                ifelse(age >24 & age<=29, "25-29",
                                ifelse(age >29 & age<=34, "30-34",
```

```r
                                  ifelse(age >34 & age <=39, "35-39",
                                  ifelse(age > 40 & age <=44, "40-44",
                                  ifelse(age >44 & age <=49, "45-49",
                                  ifelse(age > 49 & age <=54, "50-54",
                                  ifelse( age >=55, "55 or over", NA)
                                  )))))))))


# Show frequency of marriage age
print(marriage_age_coding)
```

```
## # A tibble: 9 × 3
##      age frequency  age_level
##    <dbl>     <int>      <chr>
## 1  27.0       153      25-29
## 2  22.0       117      20-24
## 3  32.0       115      30-34
## 4  37.0        88      35-39
## 5  42.0        56      40-44
## 6  47.0        23      45-49
## 7  57.0        22 55 or over
## 8  52.0        21      50-54
## 9  17.5         6   under 20
```

```r
# Generate a summary of the participants' age
summary(group_age)
```

```
##       age          frequency
##  Min.   :17.50   Min.   :  6.00
##  1st Qu.:27.00   1st Qu.: 22.00
##  Median :37.00   Median : 56.00
##  Mean   :37.06   Mean   : 66.78
##  3rd Qu.:47.00   3rd Qu.:115.00
##  Max.   :57.00   Max.   :153.00
```

```r
# summarize length of time participants have been married
 group_years_married = affairs %>%
                        group_by(yearsmarried) %>%
                        summarise(frequency=n()) %>%
                        arrange(desc(frequency))

 years_married_coding = group_years_married %>%
                        mutate(Year_married=ifelse(yearsmarried<=0.125, "3 months orless",
                            ifelse(yearsmarried>0.125 & yearsmarried<=0.417, "4-6 months",
                            ifelse(yearsmarried>0.417 & yearsmarried <=0.75, "6 months-1 year",
                            ifelse(yearsmarried>0.75 & yearsmarried<=1.5, "1-2 years",
                            ifelse(yearsmarried>1.5 & yearsmarried<=4, "3-5 years",
                            ifelse(yearsmarried>4 & yearsmarried <=7, "6-8 years",
                            ifelse(yearsmarried>7 & yearsmarried<=10, "9-11 years",
                            ifelse(yearsmarried>10 & yearsmarried <=15, "12 or more years", NA)
                             ))))))))
```

```r
# Show frequency of years married
print(years_married_coding)
```

```
## # A tibble: 8 × 3
##   yearsmarried frequency     Year_married
##          <dbl>     <int>            <chr>
## 1       15.000       204 12 or more years
## 2        4.000       105        3-5 years
## 3        1.500        88        1-2 years
## 4        7.000        82        6-8 years
## 5       10.000        70       9-11 years
## 6        0.750        31  6 months-1 year
## 7        0.125        11  3 months orless
## 8        0.417        10       4-6 months
```

```r
table(group_years_married)
```

```
##             frequency
## yearsmarried 10 11 31 70 82 88 105 204
##        0.125  0  1  0  0  0  0   0   0
##        0.417  1  0  0  0  0  0   0   0
##        0.75   0  0  1  0  0  0   0   0
##        1.5    0  0  0  0  0  1   0   0
##        4      0  0  0  0  0  0   1   0
##        7      0  0  0  0  1  0   0   0
##        10     0  0  0  1  0  0   0   0
##        15     0  0  0  0  0  0   0   1
```

```r
# Generate length of time partipants were married
summary(group_years_married)
```

```
##   yearsmarried       frequency     
##  Min.   : 0.1250   Min.   : 10.00  
##  1st Qu.: 0.6667   1st Qu.: 26.00  
##  Median : 2.7500   Median : 76.00  
##  Mean   : 4.8490   Mean   : 75.12  
##  3rd Qu.: 7.7500   3rd Qu.: 92.25  
##  Max.   :15.0000   Max.   :204.00
```

```r
# Calculate percentage of participants that are under 20 years old
percentage.under.twenty = (nrow(filter(affairs, age <=20))/nrow(affairs))*100
print(percentage.under.twenty)
```

```
## [1] 0.9983361
```

```r
# Calculate percentage of particants that are 55 and above
percentage.above.fiftyfive = (nrow(filter(affairs, age >=55))/nrow(affairs))*100
print(percentage.above.fiftyfive)
```

```
## [1] 3.660566
```

```r
# Summarize participants with or without children
group.children = affairs %>%
                  group_by(children) %>%
                  summarise(frequency=n()) %>%
                  arrange(desc(frequency))

# Explore frequency of with/out children
table(group.children)
```

```
##          frequency
## children 171 430
##      no    1   0
##      yes   0   1
```

```r
# Calculate percentage of parents that have children
percentage.have.children = (nrow(filter(affairs, children=="yes"))/nrow(affairs))*100
percentage.have.children
```

```
## [1] 71.54742
```

```r
# Calculate percentage of parents that do not have children
percentage.no.children = 100 - percentage.have.children
print(percentage.no.children)
```

```
## [1] 28.45258
```

```r
# Summarize participants' religous background
group.religiousness = affairs %>%
                      group_by(religiousness) %>%
                      summarize(frequency=n()) %>%
                      arrange(desc(frequency))

group.religiousness.coding = group.religiousness %>%
                          mutate(religion_background=ifelse(religiousness==1, "anti",
                              ifelse(religiousness==2, "not at all",
                              ifelse(religiousness==3, "slightly",
                              ifelse(religiousness==4, "somewhat",
                              ifelse(religiousness==5, "very", NA)
                          )))))

# Show frequency of participant's religious background
print(group.religiousness.coding)
```

```
## # A tibble: 5 × 3
##   religiousness frequency religion_background
##           <int>     <int>               <chr>
## 1             4       190            somewhat
## 2             2       164          not at all
## 3             3       129            slightly
## 4             5        70                very
## 5             1        48                anti
```

```r
# Generate a summary of the participants' religious background
summary(group.religiousness)
```

```
##  religiousness   frequency
##  Min.   :1     Min.   : 48.0
##  1st Qu.:2     1st Qu.: 70.0
##  Median :3     Median :129.0
##  Mean   :3     Mean   :120.2
##  3rd Qu.:4     3rd Qu.:164.0
##  Max.   :5     Max.   :190.0
```

```r
# Percentage of the participants that are very religious
percentage.very.religious = (nrow(filter(affairs, religiousness==5))/nrow(affairs))*100
print(percentage.very.religious)
```

```
## [1] 11.64725
```

```r
# Percentage of the participants that are atheist
percentage.anti.religious = (nrow(filter(affairs, religiousness==1))/nrow(affairs))*100
print(percentage.anti.religious)
```

```
## [1] 7.986689
```

```r
# Summarize participants's education level
group.education = affairs %>%
                  group_by(education) %>%
                  summarize(frequency=n()) %>%
                  arrange(desc(frequency))

group.education.coding = group.education %>%
                         mutate(education_level=(ifelse(education==9, "grade school",
                            ifelse(education==12, "high school graduate",
                            ifelse(education==14, "some college",
                            ifelse(education==16, "college graduate",
                            ifelse(education==17, "some graduate work",
                            ifelse(education==18, "master's degree",
                            ifelse(education==20, "Ph.D, M.D, or other advanced degree", NA)
                           ))))))))

# Show frequency of participants education level
print(group.education.coding)
```

```
## # A tibble: 7 × 3
##   education frequency                       education_level
##       <dbl>     <int>                                 <chr>
## 1        14       154                          some college
## 2        16       115                      college graduate
## 3        18       112                       master's degree
## 4        17        89                    some graduate work
## 5        20        80 Ph.D, M.D, or other advanced degree
## 6        12        44                  high school graduate
## 7         9         7                          grade school
```

```r
# Generate a summary of the participants' graduate work
summary(group.education)
```

```
##    education       frequency
##  Min.   : 9.00   Min.   :  7.00
##  1st Qu.:13.00   1st Qu.: 62.00
##  Median :16.00   Median : 89.00
##  Mean   :15.14   Mean   : 85.86
##  3rd Qu.:17.50   3rd Qu.:113.50
##  Max.   :20.00   Max.   :154.00
```

```r
# Percentage of participants that have a masters or above
percentage.higher.education = (nrow(filter(affairs, education %in% c(18,20)))/nrow(affairs))*100
print(percentage.higher.education)
```

```
## [1] 31.94676
```

```r
# Percentage of participants that only have a high school diploma
percentage.highschool.diploma = (nrow(filter(affairs, education==12))/nrow(affairs))*100
print(percentage.highschool.diploma)
```

```
## [1] 7.321131
```

```r
# Summarize partitcipants' occupation
group.occupation = affairs %>%
                   group_by(occupation) %>%
                   summarize(frequency=n()) %>%
                   arrange(desc(frequency))

group.occupation.coding = group.occupation %>%
                     mutate(occupation_coding= (ifelse(occupation==1, "student",
                     ifelse(occupation==2, "farming, agriculture,unskilled worker",
                     ifelse(occupation==3, "white-collar(sales,clerical,secretarial",
                     ifelse(occupation==4, "teacher,counselor,social-worker,nurse,artist,writer",
                     ifelse(occupation==5, "managerial,administrative,business",
                     ifelse(occupation==6, "professional with advanced degree", NA
                      ))))))))

# Show frequency of participants based on occupation
print(group.occupation.coding)
```

```
## # A tibble: 7 × 3
##   occupation frequency                                   occupation_coding
##        <int>     <int>                                               <chr>
## 1          5       204                  managerial,administrative,business
## 2          6       143                   professional with advanced degree
## 3          1       113                                             student
## 4          4        68 teacher,counselor,social-worker,nurse,artist,writer
## 5          3        47             white-collar(sales,clerical,secretarial
## 6          2        13               farming, agriculture,unskilled worker
## 7          7        13                                                <NA>
```

```r
# Generate a summary of the participants' occupation
summary(group.occupation)
```

```
##    occupation    frequency
##  Min.   :1.0   Min.   : 13.00
##  1st Qu.:2.5   1st Qu.: 30.00
##  Median :4.0   Median : 68.00
##  Mean   :4.0   Mean   : 85.86
##  3rd Qu.:5.5   3rd Qu.:128.00
##  Max.   :7.0   Max.   :204.00
```

```r
# Summarize husband's occupation
group.husband = filter(affairs, gender %in% c("male")) %>%
                group_by(occupation) %>%
                summarize(frequency=n()) %>%
                arrange(desc(frequency))

group.husband.occupation = group.husband %>%
                     mutate(occupation_coding= (ifelse(occupation==1, "student",
                     ifelse(occupation==2, "farming, agriculture,unskilled worker",
                     ifelse(occupation==3, "white-collar(sales,clerical,secretarial",
                     ifelse(occupation==4, "teacher,counselor,social-worker,nurse,artist,writer",
                     ifelse(occupation==5, "managerial,administrative,business",
                     ifelse(occupation==6, "professional with advanced degree", NA
                            ))))))))

# Show frequency of male's occupation
print(group.husband.occupation)
```

```
## # A tibble: 7 × 3
##   occupation frequency                                   occupation_coding
##        <int>     <int>                                               <chr>
## 1          6       116                   professional with advanced degree
## 2          5        89                  managerial,administrative,business
## 3          4        39 teacher,counselor,social-worker,nurse,artist,writer
## 4          3        20             white-collar(sales,clerical,secretarial
## 5          7        11                                                <NA>
## 6          2        10               farming, agriculture,unskilled worker
## 7          1         1                                             student
```

```r
# Generate a summary of male's occupation
summary(group.husband)
```

```
##    occupation    frequency
##  Min.   :1.0   Min.   :  1.00
##  1st Qu.:2.5   1st Qu.: 10.50
##  Median :4.0   Median : 20.00
##  Mean   :4.0   Mean   : 40.86
##  3rd Qu.:5.5   3rd Qu.: 64.00
##  Max.   :7.0   Max.   :116.00
```

```r
# summarize female's occupation
group.wife = filter(affairs, gender %in% c("female")) %>%
                group_by(occupation) %>%
                summarize(frequency=n()) %>%
                arrange(desc(frequency))

group.wife.occupation = group.wife %>%
                    mutate(occupation_coding= (ifelse(occupation==1, "student",
                    ifelse(occupation==2, "farming, agriculture,unskilled worker",
                    ifelse(occupation==3, "white-collar(sales,clerical,secretarial",
                    ifelse(occupation==4, "teacher,counselor,social-worker,nurse,artist,writer",
                    ifelse(occupation==5, "managerial,administrative,business",
                    ifelse(occupation==6, "professional with advanced degree", NA
                           ))))))))

# Show frequency of female's occupation
print(group.wife.occupation)
```

```
## # A tibble: 7 × 3
##   occupation frequency                                    occupation_coding
##        <int>     <int>                                                <chr>
## 1          5       115                   managerial,administrative,business
## 2          1       112                                              student
## 3          4        29 teacher,counselor,social-worker,nurse,artist,writer
## 4          3        27              white-collar(sales,clerical,secretarial
## 5          6        27                    professional with advanced degree
## 6          2         3                farming, agriculture,unskilled worker
## 7          7         2                                                 <NA>
```

```r
# Generate a summary of female's occupation
summary(group.wife)
```

```
##    occupation    frequency
##  Min.   :1.0   Min.   :  2.0
##  1st Qu.:2.5   1st Qu.: 15.0
##  Median :4.0   Median : 27.0
##  Mean   :4.0   Mean   : 45.0
##  3rd Qu.:5.5   3rd Qu.: 70.5
##  Max.   :7.0   Max.   :115.0
```

```r
# Percentage of participants that are female
female.participants =  filter(affairs, gender %in% c("female"))
percentage.female = (nrow(female.participants) / nrow(affairs))*100

# Show percentage of female's occupation
print(percentage.female)
```

```
## [1] 52.41265
```

```r
# Percentage of participants that didn't have an affair
participants.have.noaffair =  filter(affairs, affairs==0)
percentage.have.noaffair= (nrow(participants.have.noaffair) / nrow(affairs))*100
```

```r
# Show percentage partcipants who did not have an affair
print(percentage.have.noaffair)
```

```
## [1] 75.0416
```

```r
# Show Percentage of particpants who had an affair
percentage.have.anaffair = 100 - percentage.have.noaffair
print(percentage.have.anaffair)
```

```
## [1] 24.9584
```

```r
# Percentage of very religious participants who had an affair
participants.religious.have.affair =  filter(affairs, affairs >=1 & religiousness==5)
percentage.religious.have.affair= (nrow(participants.religious.have.affair) / nrow(affairs))*100

# Show percentage of very religious participants that had an affair
print(percentage.religious.have.affair)
```

```
## [1] 2.163062
```

```r
# Percentage of atheist participants who had an affair
participants.atheist.have.affair =  filter(affairs, affairs >=1 & religiousness==1)
percentage.atheist.have.affair= (nrow(participants.atheist.have.affair) / nrow(affairs))*100
print(percentage.atheist.have.affair)
```

```
## [1] 3.327787
```

```r
# Percentage of participants who have a masters or an advanced degree and had an affair
participants.advancedDegree.have.affair =  filter(affairs, affairs >=1 & (education==18 | education
percentage.advancedDegree.have.affair= (nrow(participants.advancedDegree.have.affair) / nrow(affai

# Show percentage of very religious participants that had an affair
print(percentage.advancedDegree.have.affair)
```

```
## [1] 8.818636
```

```r
# Percentage of participants with a high-school diploma and had an affair
participants.highschool.have.affair =  filter(affairs, affairs >=1 & education==12)
percentage.highschool.have.affair= (nrow(participants.highschool.have.affair) / nrow(affairs))*100
print(percentage.highschool.have.affair)
```

```
## [1] 2.163062
```

```r
# Percentage of participants under the age of 20 who had an affair
participants.underTwenty.have.affair =  filter(affairs, affairs >=1 & age <=20)
percentage.underTwenty.have.affair= (nrow(participants.underTwenty.have.affair) / nrow(affairs))*1

# Show percentage of pariticipants that are under 20 and had an affair
print(percentage.underTwenty.have.affair)
```

```
## [1] 0.4991681
```

```r
# Percentage of participants over the age of 55 who had an affair
participants.over55.have.affair =  filter(affairs, affairs >=1 & age >=55)
percentage.over55.have.affair= (nrow(participants.over55.have.affair) / nrow(affairs))*100
print(percentage.over55.have.affair)
```

```
## [1] 0.4991681
```

```r
# Percentage of male participants who had an affair
male.participants.have.affair =  filter(affairs, gender=="male" & affairs >=1)
male.population = nrow(filter(affairs, gender=="male"))
percentage.male.have.affair= nrow(male.participants.have.affair) / (male.population)*100

# Show percentage of male who had an affair
print(percentage.male.have.affair)
```

```
## [1] 27.27273
```

```r
# Percentage of female participants who had an affair
female.participants.have.affair =  filter(affairs, gender =="female" & affairs >=1)
female.population = nrow(filter(affairs, gender=="female"))
percentage.female.have.affair= nrow(female.participants.have.affair) / (female.population)*100
print(percentage.female.have.affair)
```

```
## [1] 22.85714
```

References:

- http://www.doviak.net/courses/statistics/Fair-JPE-1978_data-descriptions.pdf

(b) Suppose we want to explore the characteristics of participants who engage in extramarital sexual intercourse (i.e. affairs). Instead of modeling the number of affairs, we will consider the binary outcome - had an affair versus didn't have an affair. Create a new variable to capture this response variable of interest.

Solution: We added a new label, "had_an_affair" that will set to 0 if the participant did not have extramarital sexual intercourse. Otherwise, it will set any other values to yes which will signify that the user had an affair.

```r
# Adding a new variable to hold binary (Yes or No) the particpants had an affair
Affairs = Affairs %>%
          mutate(had_an_affair= ifelse(affairs==0, "NO", "YES"))


# Investigate whether the result is consistent and expected
head(Affairs)
```

```
##   affairs gender age yearsmarried children religiousness education
## 1       0   male  37        10.00       no             3        18
## 2       0 female  27         4.00       no             4        14
## 3       0 female  32        15.00      yes             1        12
## 4       0   male  57        15.00      yes             5        18
## 5       0   male  22         0.75       no             2        17
## 6       0 female  32         1.50       no             2        17
```

```
##   occupation rating had_an_affair
## 1          7      4            NO
## 2          6      4            NO
## 3          1      4            NO
## 4          6      5            NO
## 5          6      3            NO
## 6          5      5            NO
```

```r
tail(Affairs)
```

```
##     affairs gender age yearsmarried children religiousness education
## 596       7   male  47         15.0      yes             3        16
## 597       1   male  22          1.5      yes             1        12
## 598       7 female  32         10.0      yes             2        18
## 599       2   male  32         10.0      yes             2        17
## 600       2   male  22          7.0      yes             3        18
## 601       1 female  32         15.0      yes             3        14
##     occupation rating had_an_affair
## 596          4      2           YES
## 597          2      5           YES
## 598          5      4           YES
## 599          6      5           YES
## 600          6      2           YES
## 601          1      5           YES
```

```r
# Calculate frequency of participants who either had an affair or did not
 participants.have.affair = Affairs %>%
                        group_by(had_an_affair) %>%
                        summarise(frequency=n()) %>%
                        arrange(desc(frequency))

# Show a table representing participants had/not affair based on the binary
table(participants.have.affair)
```

```
##              frequency
## had_an_affair 150 451
##           NO    0   1
##           YES   1   0
```

(c) Use an appropriate regression model to explore the relationship between having an affair and other personal characteristics. Comment on which covariates seem to be predictive of having an affair and which do not.

Solution: In order to generate a glm with a binomial family we needed to have a column with a binary value of 0 and 1. Similarly, to what was done in the previous question, we added a new column that have 0 for affairs = 0 or no affair, and then 1 for affairs. The data generated can be explained as follows:

Partipants in general have 1.37 increase chance of having an affair. This will be our baseline. A male participant has 0.28 increase chance of having an affair. A participant's Age is 0.044 less of an indicator to cause someone to have an affair. The length of marriage is more than 0.094 chance to be the reason for someone to have an affair. Participants with children have 0.39 chance to have an affair. Participants religious beliefs or no beliefs is 0.32 less of an indicator to cause someone to have an affair. An educated participant has 0.02 more of a chance to have an affair. A participant with an occupation has 0.030 more of chance to have an affair. Participant's self rating of marriage has 0.47 less chance to be an indicator to cause someone to have an affair.

The model shows that age, yearsmarried, religiousness, and rating are factors that are under a significant value of 0.05. Thus, these values can be used to either reject or fail to reject the null hypothesis on whether a participant eith had or did not have an affair.

```r
set.seed(1233)
# Add a new label to hold affair has either 0 or 1
Affairs$hadaffair = ifelse(Affairs$affairs==0, 0, 1)
Affairs$hadaffair = as.integer(Affairs$hadaffair)

attach(Affairs) # Make these objects global
```

```
## The following object is masked _by_ .GlobalEnv:
##
##     affairs
```

```r
# Fit a logistic regression model for Fair's extreamarital affairs data
model = glm(hadaffair~gender+age+yearsmarried+children+religiousness+education+
              occupation+rating, data=Affairs, family = binomial)

summary(model) # display  results
```

```
##
## Call:
## glm(formula = hadaffair ~ gender + age + yearsmarried + children +
##     religiousness + education + occupation + rating, family = binomial,
##     data = Affairs)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5713  -0.7499  -0.5690  -0.2539   2.5191
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)    1.37726    0.88776   1.551 0.120807
## gendermale     0.28029    0.23909   1.172 0.241083
## age           -0.04426    0.01825  -2.425 0.015301 *
## yearsmarried   0.09477    0.03221   2.942 0.003262 **
## childrenyes    0.39767    0.29151   1.364 0.172508
## religiousness -0.32472    0.08975  -3.618 0.000297 ***
## education      0.02105    0.05051   0.417 0.676851
## occupation     0.03092    0.07178   0.431 0.666630
## rating        -0.46845    0.09091  -5.153 2.56e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 675.38  on 600  degrees of freedom
## Residual deviance: 609.51  on 592  degrees of freedom
## AIC: 627.51
##
## Number of Fisher Scoring iterations: 4
```

(d) Use an all subsets model selection procedure to obtain a "best" fit model. Is the model different from the full model you fit in part (c)? Which variables are included in the "best" fit model? You might found the bestglm() function available in the bestglm package helpful.

Solution: The "best" fit model provides multiple options to judge the quality of this model. We could have used either BIC or AIC as both are maximum likelihood estimate driven. However, we decided to use the Akaike information criterion(AIC) because it is better suited for prediction. Ideally, the model with the smallest AIC is preferred. Based on the AIC's subset result, line 5 which has true for religiouness, yearsmarried, age, gender, and rating has the lowest AIC. Thus, it will be the best fit.

```r
set.seed(1245)
# Generate the argument Xy needed for bestglm
Xy = cbind(gender,age,yearsmarried,children,religiousness,education,occupation,rating,hadaffair)
Xy = as.data.frame(Xy) # Set Xy as a dataframe
```

```r
# Calculate best subset using AIC
bestAIC = bestglm(Xy, IC="AIC")
print(bestAIC)
```

```
## AIC
## BICq equivalent for q in (0.821508156450582, 0.932574998701229)
## Best Model:
##                  Estimate  Std. Error   t value     Pr(>|t|)
## (Intercept)    0.75797526 0.108552706  6.982555 7.772812e-12
## gender         0.06360652 0.034902126  1.822425 6.889227e-02
## age           -0.00739692 0.002988183 -2.475390 1.358642e-02
## yearsmarried   0.01859607 0.004970389  3.741371 2.007405e-04
## religiousness -0.05442460 0.014815335 -3.673531 2.607919e-04
## rating        -0.08759874 0.015763910 -5.556917 4.146818e-08
```

```r
print(bestAIC$Subsets)
```

```
##    (Intercept) gender   age yearsmarried children religiousness education
## 0         TRUE  FALSE FALSE        FALSE    FALSE         FALSE     FALSE
## 1         TRUE  FALSE FALSE        FALSE    FALSE         FALSE     FALSE
## 2         TRUE  FALSE FALSE        FALSE    FALSE          TRUE     FALSE
## 3         TRUE  FALSE FALSE         TRUE    FALSE          TRUE     FALSE
## 4         TRUE  FALSE  TRUE         TRUE    FALSE          TRUE     FALSE
## 5*        TRUE   TRUE  TRUE         TRUE    FALSE          TRUE     FALSE
## 6         TRUE   TRUE  TRUE         TRUE     TRUE          TRUE     FALSE
## 7         TRUE   TRUE  TRUE         TRUE     TRUE          TRUE     FALSE
## 8         TRUE   TRUE  TRUE         TRUE     TRUE          TRUE      TRUE
##    occupation rating logLikelihood       AIC
## 0       FALSE  FALSE      503.3637 -1006.727
## 1       FALSE   TRUE      523.3742 -1044.748
## 2       FALSE   TRUE      528.3506 -1052.701
## 3       FALSE   TRUE      532.5297 -1059.059
## 4       FALSE   TRUE      534.6637 -1061.327
## 5*      FALSE   TRUE      536.3364 -1062.673
## 6       FALSE   TRUE      536.9088 -1061.818
## 7        TRUE   TRUE      537.1692 -1060.338
## 8        TRUE   TRUE      537.2351 -1058.470
```

```r
summary(bestAIC$BestModel)  #Overall best models
```

```
##
## Call:
## lm(formula = y ~ ., data = data.frame(Xy[, c(bestset[-1], FALSE),
##     drop = FALSE], y = y))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.6304 -0.2663 -0.1586  0.1077  1.0250
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    0.757975   0.108553   6.983 7.77e-12 ***
## gender         0.063607   0.034902   1.822 0.068892 .
## age           -0.007397   0.002988  -2.475 0.013586 *
## yearsmarried   0.018596   0.004970   3.741 0.000201 ***
## religiousness -0.054425   0.014815  -3.674 0.000261 ***
## rating        -0.087599   0.015764  -5.557 4.15e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4117 on 595 degrees of freedom
## Multiple R-squared:  0.1039, Adjusted R-squared:  0.09639
## F-statistic:  13.8 on 5 and 595 DF,  p-value: 9.04e-13
```

  - http://www2.uaem.mx/r-mirror/web/packages/bestglm/vignettes/bestglm.pdf
  - http://stats.stackexchange.com/questions/577/is-there-any-reason-to-prefer-the-aic-or-bic-over-the-other

(e) Interpret the model parameters using the model from part (d). Solution: The "best" fit model is in this format bestglm(Xy, IC=". Xy represents the arguments containing the variables to be analyzed in the model with the last column containing the response. The result of this model can be expalined as follows: Partipants in general have 0.75 increase chance of having an affair. This will be our baseline. A male or a female participant has 0.06 increase chance of having an affair. A participant's Age is 0.007 less of an indicator to cause someone to have an affair.A participant's age has 0.0073 less chance to be an indicator to cause someone to have an affair. The length of marriage is more than 0.018 chance to be the reason for someone to have an affair. A participant's religious beliefs or no beliefs are 0.05 less of and indicator to cause them to have an affair. Participant's self rating of marriage has 0.08 less chance to be an indicator to cause someone to have an affair.

    The model shows that age, yearsmarried, religiousness, and rating are factors that are under a significant value of 0.05. Thus, these values can be used to either reject or fail to reject the null hypothesis on whether a participant eith had or did not have an affair.

    Also, it's worth nothing that even though both the glm and the bestglm tend to come up with similar best fit model, the bestfit coefficients value seem to be a little stronger becuse it shows more than 3 stars for yearsmarried while glm only give the same variable only a 2 star.

(f) Create an artificial test dataset where martial rating varies from 1 to 5 and all other variables are set to their means. Use this test dataset and the predict function to obtain predicted probabilities of having an affair for case in the test data. Interpret your results and use a visualization to support your interpretation.

    Solution: When generating the mean for all variables, we decided to skip gender and children. We did this because for this analysis getting the mean of being a male or female does not apply and also calculating the mean of either having children or not does not apply for this analysis.

The residual graphs show multiple scattered plots to detect non-linearity, unequal error variances, and outlisers. The residual vs fitted suggest that there is a decreasing linear relationship when ratings are involved.

We generated multiple graph analysis. Figrue 6 shows that the probability to have an affair decreases when the marriage rating goes from very unhappy to very happy. Thus, people who have a very happy marriage tends not to have an affair.

```r
set.seed(23456)
# create a new model to be used for the prediction
model2 = glm(hadaffair~as.factor(rating)+age+yearsmarried+religiousness+education+occupation, data=
summary(model2)
```

```
##
## Call:
## glm(formula = hadaffair ~ as.factor(rating) + age + yearsmarried +
##     religiousness + education + occupation, family = binomial,
##     data = Affairs)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6403  -0.7478  -0.5728  -0.2756   2.4455
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)          0.57967    0.92974   0.623 0.532975
## as.factor(rating)2  0.05500    0.57887   0.095 0.924312
## as.factor(rating)3 -0.78191    0.57422  -1.362 0.173295
## as.factor(rating)4 -1.06789    0.55459  -1.926 0.054158 .
## as.factor(rating)5 -1.58940    0.56565  -2.810 0.004956 **
## age                -0.04151    0.01800  -2.306 0.021105 *
## yearsmarried        0.10673    0.02965   3.600 0.000318 ***
## religiousness      -0.31788    0.08984  -3.538 0.000403 ***
## education           0.03177    0.05018   0.633 0.526664
## occupation          0.04921    0.06682   0.736 0.461466
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 675.38  on 600  degrees of freedom
## Residual deviance: 611.63  on 591  degrees of freedom
## AIC: 631.63
##
## Number of Fisher Scoring iterations: 4
```

```r
confint(model2)
```

```
## Waiting for profiling to be done...
```

```
##                          2.5 %      97.5 %
## (Intercept)        -1.23948298  2.419709484
## as.factor(rating)2 -1.09579877  1.201078540
## as.factor(rating)3 -1.92700047  0.352450841
```

```
## as.factor(rating)4 -2.17582138  0.029759915
## as.factor(rating)5 -2.71932626 -0.471827599
## age                -0.07766852 -0.006912926
## yearsmarried        0.04914152  0.165538706
## religiousness      -0.49592309 -0.143157461
## education          -0.06609203  0.130944652
## occupation         -0.08046741  0.182035231
```

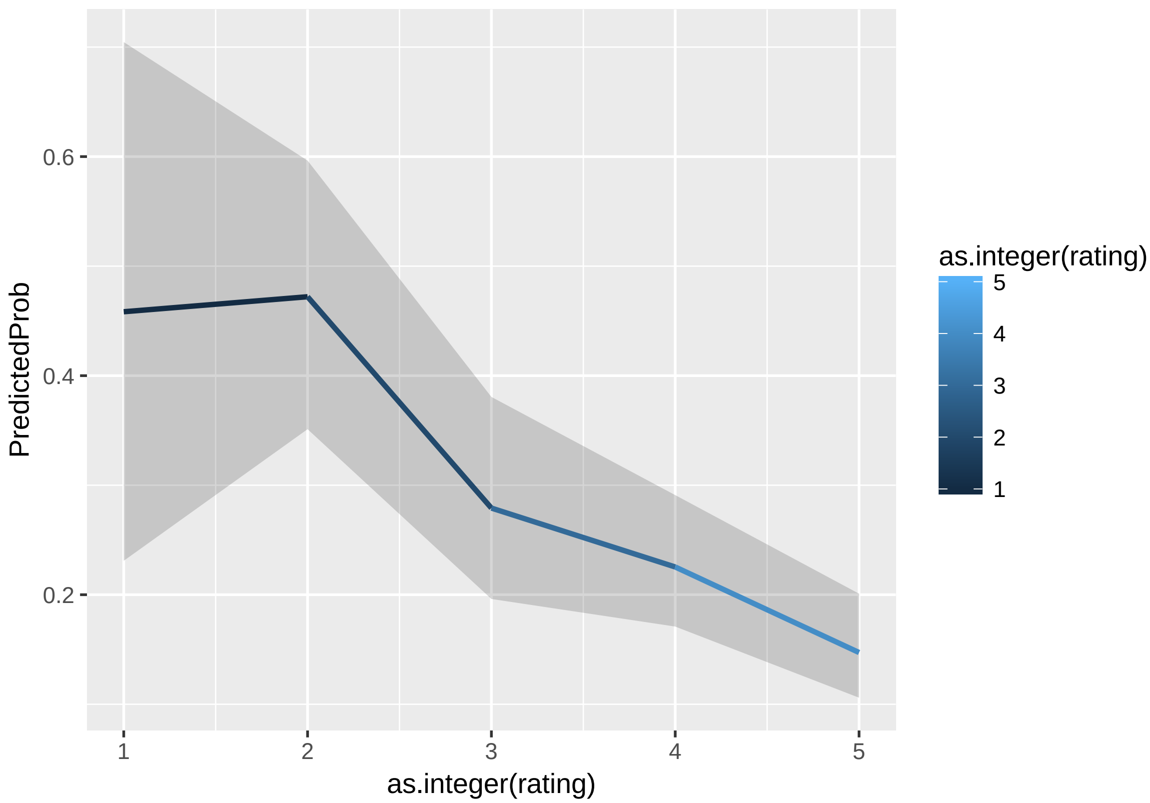
```
# Generate a new data that has affairs and the mean of all the other variables.
newdata2 = with(Affairs, data.frame(age=mean(age),yearsmarried = mean(yearsmarried), religiousness

# Generate predicted probabilities
newdata3 <- cbind(newdata2, predict(model2, newdata = newdata2, type="link", se=TRUE))
newdata3 <- within(newdata3, {
  PredictedProb <- plogis(fit)
  LL <- plogis(fit - (1.96 * se.fit)) # Add lower bound standard error
  UL <- plogis(fit + (1.96 * se.fit)) # Add upper bound standard error
})

head(newdata3) # View top 5 rows from the recordset
```

```
##        age yearsmarried religiousness education occupation rating
## 1 32.48752     8.177696      3.116473  16.16639   4.194676      1
## 2 32.48752     8.177696      3.116473  16.16639   4.194676      2
## 3 32.48752     8.177696      3.116473  16.16639   4.194676      3
## 4 32.48752     8.177696      3.116473  16.16639   4.194676      4
## 5 32.48752     8.177696      3.116473  16.16639   4.194676      5
##          fit    se.fit residual.scale        UL        LL PredictedProb
## 1 -0.1669146 0.5285132              1 0.7045316 0.2309774     0.4583680
## 2 -0.1119193 0.2562330              1 0.5963543 0.3511163     0.4720493
## 3 -0.9488242 0.2356007              1 0.3805895 0.1961392     0.2791214
## 4 -1.2348094 0.1755614              1 0.2909635 0.1709499     0.2253408
## 5 -1.7563174 0.1918499              1 0.2009622 0.1059927     0.1472522
```

```
# Plotting diagnostics for glm
autoplot(model2, data = Affairs,
         colour = 'rating', label.size = 3, "Figure 6")
```

```
## NULL
```

```
ggplot(newdata3, aes(x = as.integer(rating), y = PredictedProb)) +
  geom_ribbon(aes(ymin = LL, ymax = UL), alpha = .2) +
  geom_line(aes(colour = as.integer(rating)), size=1)
```



References:

- https://rstudio-pubs-static.s3.amazonaws.com/119859_a290e183ff2f46b2858db66c3bc9ed3a.html
- http://www.ats.ucla.edu/stat/r/dae/logit.htm

Problem 2:

In this problem we will revisit the state dataset. This data, available as part of the base R package, contains various data related to the 50 states of the United States of America.

Suppose you want to explore the relationship between a state's Murder rate and other characteristics of the state, for example population, illiteracy rate, and more. Follow the questions below to perform this analysis.

Solution: The dataset contains 50 observations with 8 variables.

```
# convert the matrix data as a data frame
stateInfo <- as.data.frame(state.x77)

# Check the internal structure of the state data
str(stateInfo)
```

```
## 'data.frame':    50 obs. of  8 variables:
##  $ Population: num  3615 365 2212 2110 21198 ...
##  $ Income    : num  3624 6315 4530 3378 5114 ...
##  $ Illiteracy: num  2.1 1.5 1.8 1.9 1.1 0.7 1.1 0.9 1.3 2 ...
##  $ Life Exp  : num  69 69.3 70.5 70.7 71.7 ...
##  $ Murder    : num  15.1 11.3 7.8 10.1 10.3 6.8 3.1 6.2 10.7 13.9 ...
```

```
##  $ HS Grad   : num  41.3 66.7 58.1 39.9 62.6 63.9 56 54.6 52.6 40.6 ...
##  $ Frost     : num  20 152 15 65 20 166 139 103 11 60 ...
##  $ Area      : num  50708 566432 113417 51945 156361 ...
```

```
dim(stateInfo)
```

```
## [1] 50  8
```

```
attach(stateInfo) # Make the objects global
```

(a) Examine the bivariate relationships present in the data. Briefly discuss notable results. You might find the scatterplotMatrix() function available in the car package helpful.

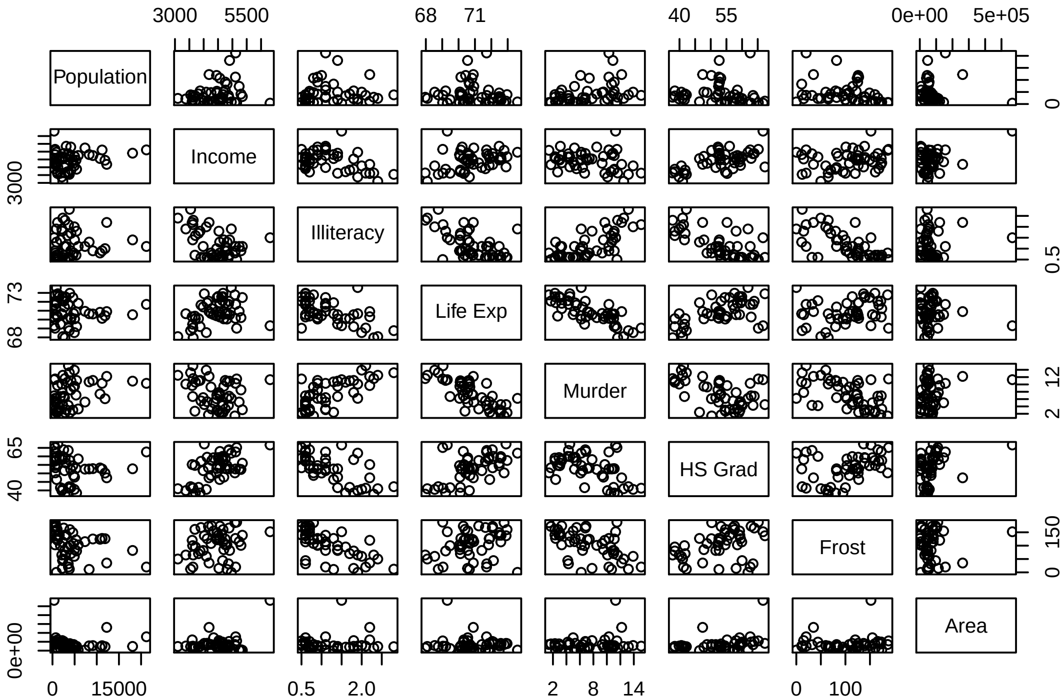
Solution: The pairs function gives a general idea of all the possible relationships between all the variables.

The scattter plot depicts multiple possible relationships. We decided to investigate the muder and life expectancy variables in relation to the other variables. We have identified the followings: Muder is negatively related to life expectancy Murder is negatively related to Frost Murder is negatively related to illiteracy murder is negatively related to income
life expectancy is positively related to income life expectancy is negatively related to illeteracy life expectancy is positively related to High school graduate life expectancy is postiveely related to frost

Also, we calculated the Pearson's r value to confirm the observations that we noticed from the scatterplot. We have found the followings for murder rate: When comparing murder to illiteracy, the coefficient value = 0.70.This means that there is a strong positive linear correlation. When comparing murder to life expectancy, the coefficient value = -0.78. This means that there is a strong negative correlation between murder rate and life expectancy rate. when comparing murder to frost, the coefficient value = -0.538. This means that there is a moderate negative correlation between murder rate and frost rate. When comparing murder to income, the coefficient value = -0.230. This means that there is a small negative correlation between murder rate and income rate.

Also, we have foudn the followings for life expectancy rate: when comparing life expectancy to income, the coefficient value = 0.340. This means there is a small positive correlatoin between life expectancy and income. When comparing life expectancy to illiteracy, the coefficient value = - 0.59. This means there is a strong negative correlation between life expectancy and illiteracy. when comparing life expectancy to High School, the coefficient value = 0.58. This means there is a strong postive correlation between life expectancy and high school graduation rate. when comparing life expectancy to frost, the coefficient value = 0.26. This means there is a moderate positve correlation between life expectancy and frost.

```
# Generate a matrix of scattered plots for all of the variables.
pairs(stateInfo)
```




```r
# Calculate correlation coefficient for further analysis of the the model
cor(stateInfo)
```

```
##             Population     Income  Illiteracy    Life Exp     Murder
## Population  1.00000000  0.2082276  0.10762237 -0.06805195  0.3436428
## Income      0.20822756  1.0000000 -0.43707519  0.34025534 -0.2300776
## Illiteracy  0.10762237 -0.4370752  1.00000000 -0.58847793  0.7029752
## Life Exp   -0.06805195  0.3402553 -0.58847793  1.00000000 -0.7808458
## Murder      0.34364275 -0.2300776  0.70297520 -0.78084575  1.0000000
## HS Grad    -0.09848975  0.6199323 -0.65718861  0.58221620 -0.4879710
## Frost      -0.33215245  0.2262822 -0.67194697  0.26206801 -0.5388834
## Area        0.02254384  0.3633154  0.07726113 -0.10733194  0.2283902
##                HS Grad      Frost        Area
## Population -0.09848975 -0.3321525  0.02254384
## Income      0.61993232  0.2262822  0.36331544
## Illiteracy -0.65718861 -0.6719470  0.07726113
## Life Exp    0.58221620  0.2620680 -0.10733194
## Murder     -0.48797102 -0.5388834  0.22839021
## HS Grad     1.00000000  0.3667797  0.33354187
## Frost       0.36677970  1.0000000  0.05922910
## Area        0.33354187  0.0592291  1.00000000
```

(b) Fit a multiple linear regression model. How much variance in the murder rate across states do the predictor variables explain?

Solution: The equation for the regression model will be as followed: Y= 1.22e+02 + 1.88e-04*population + (-1.59e-04)*Income + (1.37e+00)*Illiteracy+ (-1.65e+00)*(`Life Exp`)+ (3.23e-02)*(`HS Grad`)+(-1.29e-02)*Frost + (5.97e-06)*Area. Y represent the murder rate for a specific state,thus, replacing the variables for a specific state then we can estimate whether murder rate will increase or decrease. Also, to estimate the variance in the murder rate, we analyze the residual standard error and the R-squared values.

Based on the residual standard error, the actual murder rate increase or decrease in a specific state can deviate from the actual regression line by approximately 1.75 percentage rate. In other words, given that the mean possibiliy for the murder rate to increase for a state is around 1.22e+02. Note that the Residual Standard Error was calculated with 42 degrees of freedom.

In this analysis, the R-squared we get is 0.8008. In other words, 80% of the variance found in the response variable (murder rate) can be explained by the predictor variables. This strong R-squared is saying that the predictors for this model are good predictors to estimated possible increase or decrese for a murder rate.

```r
# Linear model to show relationship between murder rate and all the other variables
state.model = lm(stateInfo$Murder ~ ., data=stateInfo)

# Generate a summary of the linear model
summary(state.model)
```

```
##
## Call:
## lm(formula = stateInfo$Murder ~ ., data = stateInfo)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4452 -1.1016 -0.0598  1.1758  3.2355
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.222e+02  1.789e+01   6.831 2.54e-08 ***
## Population   1.880e-04  6.474e-05   2.905  0.00584 **
## Income      -1.592e-04  5.725e-04  -0.278  0.78232
## Illiteracy   1.373e+00  8.322e-01   1.650  0.10641
## `Life Exp`  -1.655e+00  2.562e-01  -6.459 8.68e-08 ***
## `HS Grad`    3.234e-02  5.725e-02   0.565  0.57519
## Frost       -1.288e-02  7.392e-03  -1.743  0.08867 .
## Area         5.967e-06  3.801e-06   1.570  0.12391
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.746 on 42 degrees of freedom
## Multiple R-squared:  0.8083, Adjusted R-squared:  0.7763
## F-statistic: 25.29 on 7 and 42 DF,  p-value: 3.872e-13
```

References:

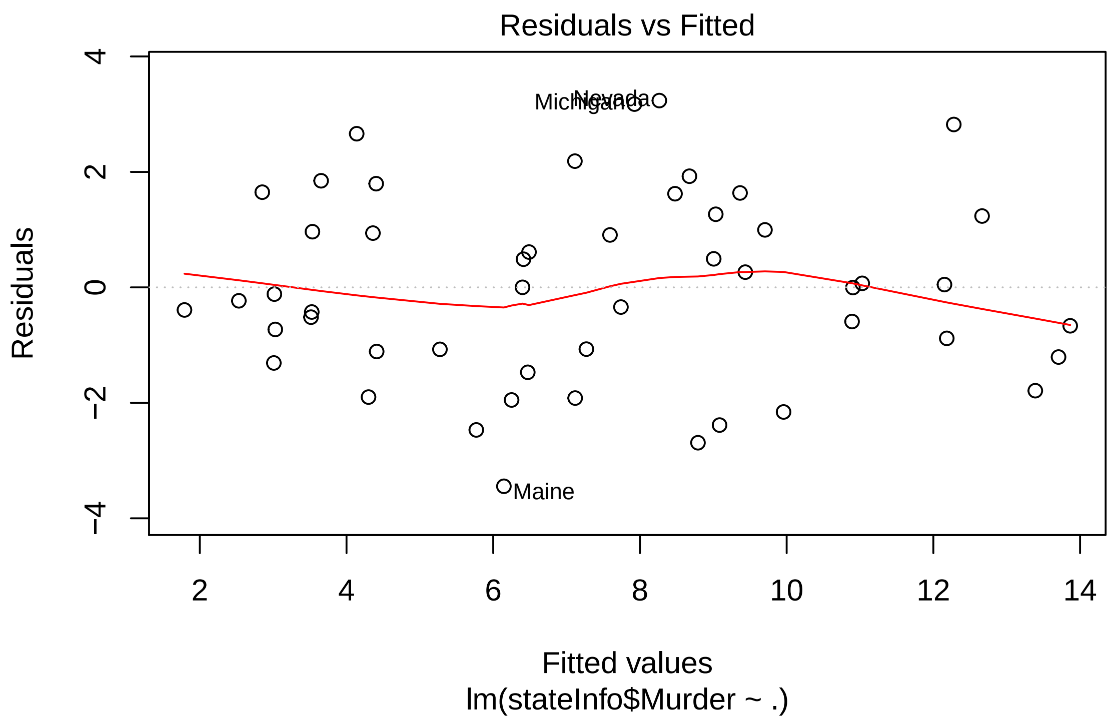
– https://rstudio-pubs-static.s3.amazonaws.com/119859_a290e183ff2f46b2858db66c3bc9ed3a.html

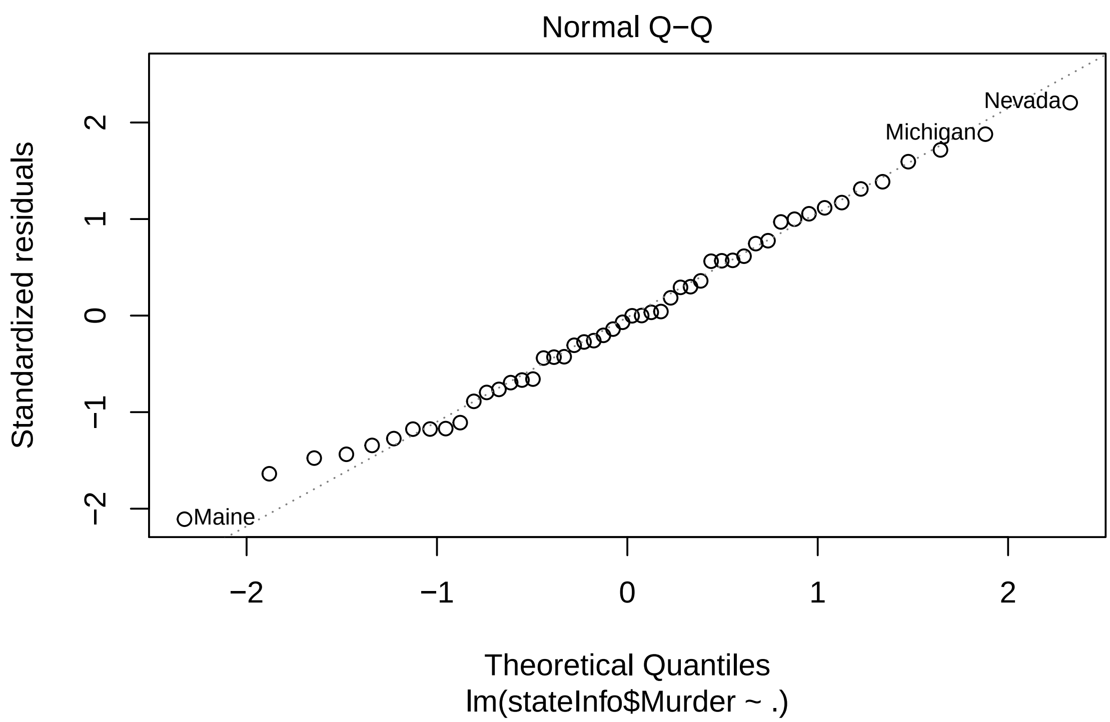
(c) Evaluate the statistical assumptions in your regression analysis from part (b) by performing a basic analysis of model residuals and any unusual observations. Discuss any concerns you have about your model.

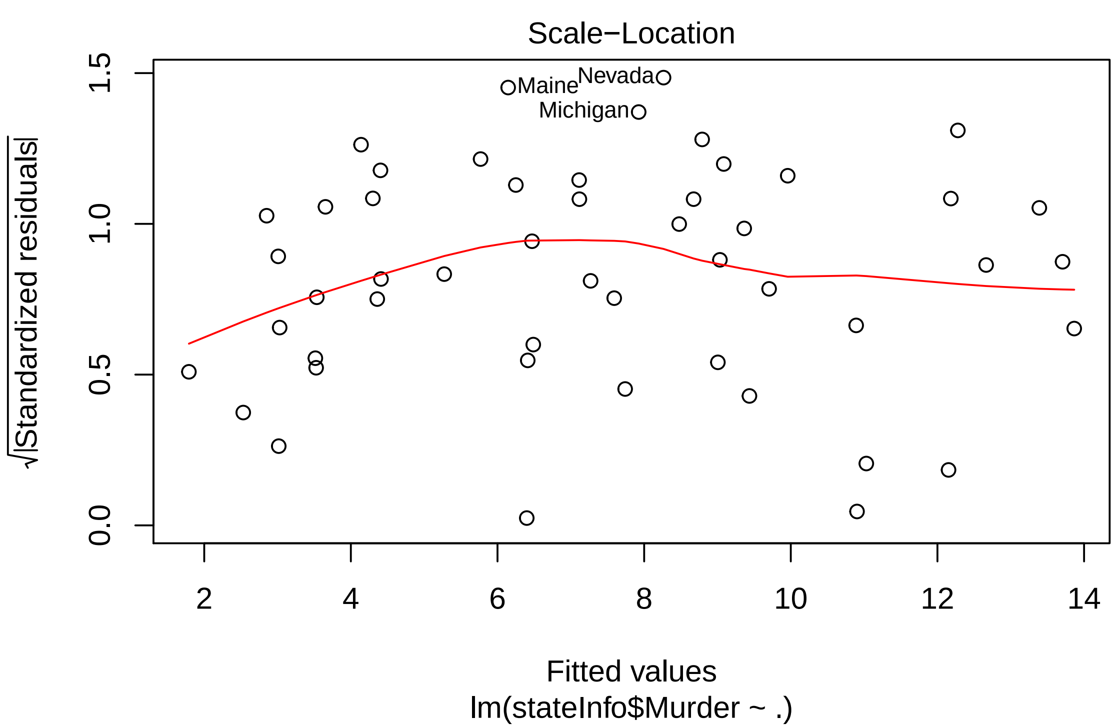
Solution:We have analyzed the residual standard error in the previous quetion. Next, we are analyzing the residuals from part b. It shows five summary points. The symmetrical distribution of these points toward the mean is an indicator on how the model fit the data. Here we see that the distribution of the residuals is not too far off from the mean.

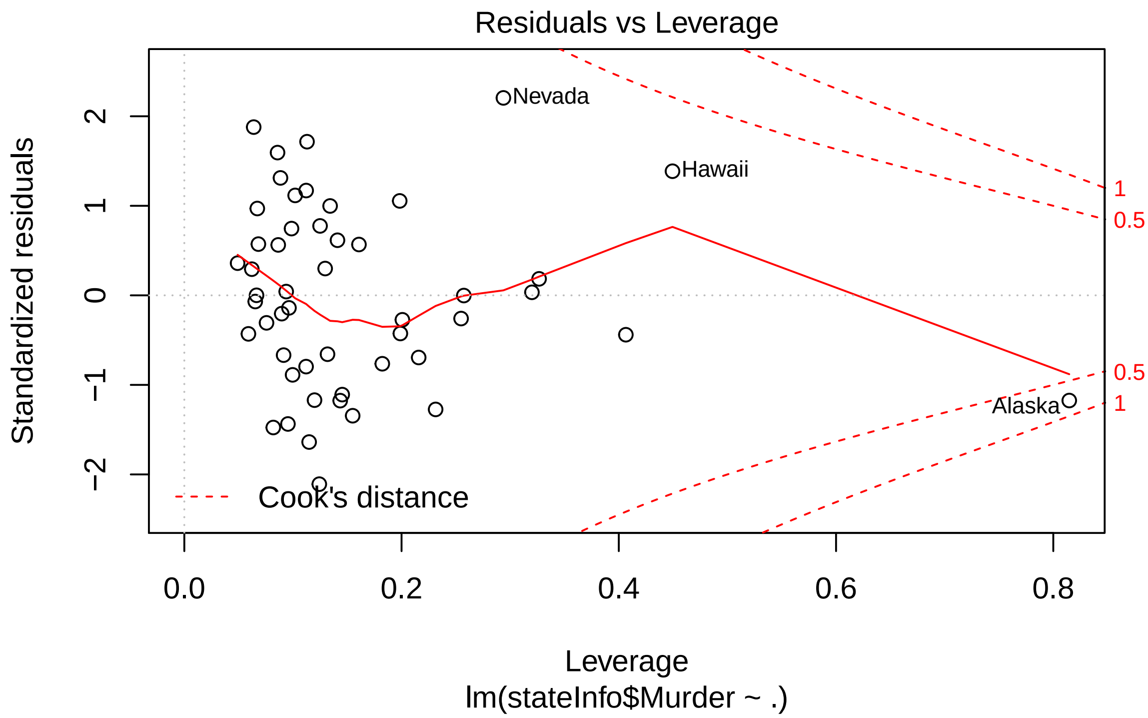
Also, we plotted out the residuals for this model. The graphs showed somewhat a well-behaved residual vs fit. We can note the followings: The residuals somewhat bounce randomly around the 0 line. This will suggest that there is a possible relationship and it is linear. The residuals somewhat form an horizontal line around the 0 line. This suggests that the variances of the error terms are equal. Also, No one residuals are that far away to be considered as outliers.

```
# Get the residual model for this model
plot(state.model)
```




Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(stateInfo$Murder ~ .)
Nevada
Michigan
Maine
−2
−1
0
1
2


Scale−Location
√|Standardized residuals|
Fitted values
lm(stateInfo$Murder ~ .)
Maine
Nevada
Michigan
0.0
0.5
1.0
1.5
2
4
6
8
10
12
14



References:

- http://www.statmethods.net/stats/regression.html
- https://onlinecourses.science.psu.edu/stat501/node/36

(d) Use a stepwise model selection procedure of your choice to obtain a “best” fit model. Is the model different from the full model you fit in part (b)? If yes, how so?

Solution: For this analysis, we decided to use a stepAIC function from the MASS package. We used the “both” direction option. It gives a best fit model based on the aIC value. Ideally, the best model will be the one with the lowest AIC value. Based on this. It shows that our model will have the best fit when population, illiteracy, life expectancy, frost,and area are selected as predictors for estimating possible increase or decrease of murder rate.

Yes, the model presented the stepAIC is different from the full model in part b. The reason for this is because in part be we considered the variables that were significant at 0.05. On the other hand, the stepAIC ran a more detailed analysis on the impact of adding or removing a variable from the model.

```
# Stepwise Regression

step <- stepAIC(state.model, direction="both")
```

```
## Start:  AIC=63.01
## stateInfo$Murder ~ Population + Income + Illiteracy + `Life Exp` +
##     `HS Grad` + Frost + Area
##
##                Df Sum of Sq    RSS    AIC
## - Income        1     0.236 128.27 61.105
## - `HS Grad`     1     0.973 129.01 61.392
```

```
## <none>                       128.03 63.013
## - Area         1     7.514 135.55 63.865
## - Illiteracy  1     8.299 136.33 64.154
## - Frost        1     9.260 137.29 64.505
## - Population  1    25.719 153.75 70.166
## - `Life Exp`  1   127.175 255.21 95.503
##
## Step:  AIC=61.11
## stateInfo$Murder ~ Population + Illiteracy + `Life Exp` + `HS Grad` +
##     Frost + Area
##
##              Df Sum of Sq    RSS    AIC
## - `HS Grad`   1     0.763 129.03 59.402
## <none>                    128.27 61.105
## - Area        1     7.310 135.58 61.877
## - Illiteracy  1     8.715 136.98 62.392
## - Frost       1     9.345 137.61 62.621
## + Income      1     0.236 128.03 63.013
## - Population  1    27.142 155.41 68.702
## - `Life Exp`  1   127.500 255.77 93.613
##
## Step:  AIC=59.4
## stateInfo$Murder ~ Population + Illiteracy + `Life Exp` + Frost +
##     Area
##
##              Df Sum of Sq    RSS    AIC
## <none>                    129.03 59.402
## - Illiteracy  1     8.723 137.75 60.672
## + `HS Grad`   1     0.763 128.27 61.105
## + Income      1     0.026 129.01 61.392
## - Frost       1    11.030 140.06 61.503
## - Area        1    15.937 144.97 63.225
## - Population  1    26.415 155.45 66.714
## - `Life Exp`  1   140.391 269.42 94.213
```

```r
step$anova # display results
```

```
## Stepwise Model Path
## Analysis of Deviance Table
##
## Initial Model:
## stateInfo$Murder ~ Population + Income + Illiteracy + `Life Exp` +
##     `HS Grad` + Frost + Area
##
## Final Model:
## stateInfo$Murder ~ Population + Illiteracy + `Life Exp` + Frost +
##     Area
##
##
##          Step Df  Deviance Resid. Df Resid. Dev      AIC
## 1                                42   128.0331 63.01329
## 2    - Income  1 0.2357225        43   128.2688 61.10526
## 3 - `HS Grad`  1 0.7627900        44   129.0316 59.40172
```

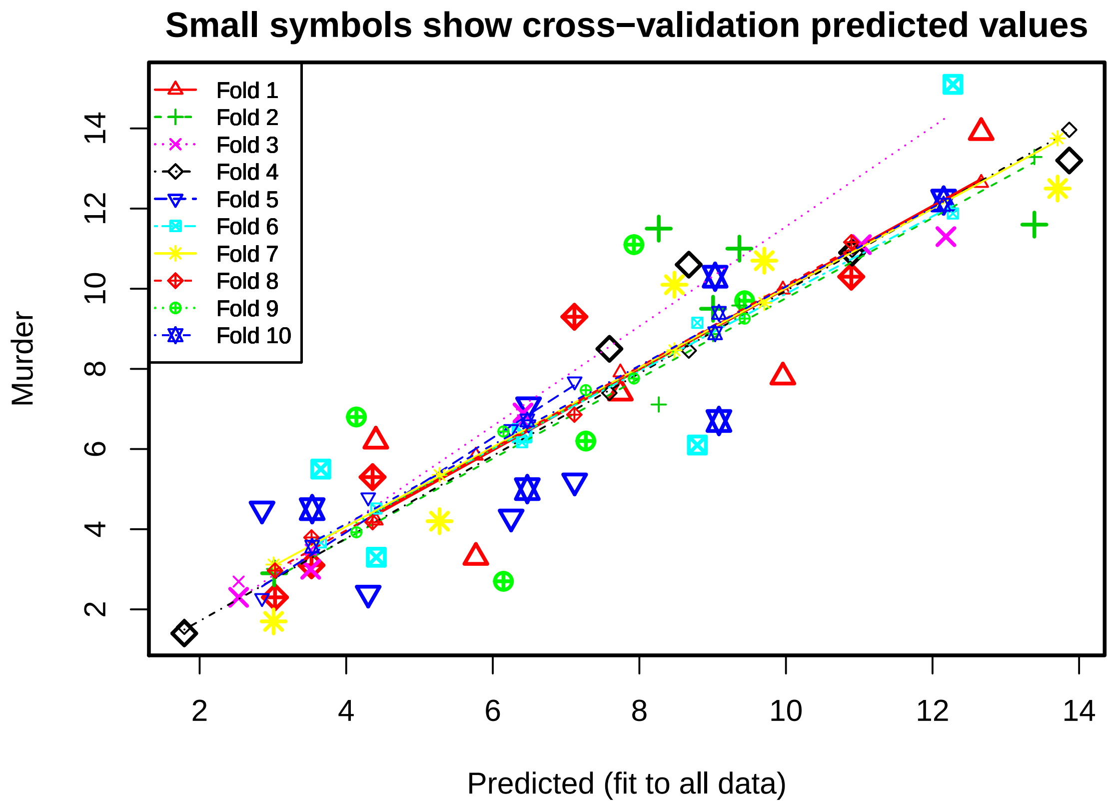
(e) Assess the model (from part (d)) generalizability. Perform a 10-fold cross validation to esti-mate model performance. Report the results.

Solution: The chart shows 10 different colors nad shapes since we are performing a 10-fold cross-validation. The dotted lines represent the best fit per fold, and they appear to be parralel to each other.

```
# Calculate a Stepwise Regression
cv =  CVlm(data = stateInfo, m=10, form.lm = formula(Murder ~ Population + Income +
                                  Illiteracy + `Life Exp` + `HS Grad` + Frost + Area))
```

```
## Analysis of Variance Table
##
## Response: Murder
##            Df Sum Sq Mean Sq F value  Pr(>F)
## Population  1   78.9    78.9   25.87 8.0e-06 ***
## Income      1   63.5    63.5   20.83 4.3e-05 ***
## Illiteracy  1  236.2   236.2   77.48 4.4e-11 ***
## `Life Exp`  1  139.5   139.5   45.75 3.2e-08 ***
## `HS Grad`   1    8.1     8.1    2.65    0.11
## Frost       1    6.1     6.1    2.00    0.16
## Area        1    7.5     7.5    2.46    0.12
## Residuals  42  128.0     3.0
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## Warning in CVlm(data = stateInfo, m = 10, form.lm = formula(Murder ~ Population + :
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate
```




```
##
## fold 1
## Observations in test set: 5
##               Arizona Georgia Hawaii Massachusetts   Ohio
## Predicted        9.96   12.66   4.40          5.77  7.741
## cvpred           9.98   12.64   4.22          5.83  7.911
## Murder           7.80   13.90   6.20          3.30  7.400
## CV residual     -2.18    1.26   1.98         -2.53 -0.511
##
## Sum of squares = 17     Mean square = 3.39     n = 5
##
## fold 2
## Observations in test set: 5
##               Nebraska Nevada South Carolina Tennessee Virginia
## Predicted        3.017   8.26          13.39      9.36    9.006
## cvpred           3.009   7.11          13.29      9.58    8.869
## Murder           2.900  11.50          11.60     11.00    9.500
## CV residual     -0.109   4.39          -1.69      1.42    0.631
##
## Sum of squares = 24.5     Mean square = 4.91     n = 5
##
## fold 3
## Observations in test set: 5
##               Alaska Minnesota North Carolina Wisconsin Wyoming
## Predicted      12.18     2.532        11.0301     3.516   6.412
## cvpred         15.98     2.693        11.0612     3.484   6.718
## Murder         11.30     2.300        11.1000     3.000   6.900
```

```
## CV residual  -4.68    -0.393         0.0388    -0.484   0.182
##
## Sum of squares = 22.3    Mean square = 4.46    n = 5
##
## fold 4
## Observations in test set: 5
##             Kentucky Louisiana Maryland New York North Dakota
## Predicted       8.67    13.866     7.59  10.9032        1.791
## cvpred          8.46    13.966     7.40  10.9682        1.564
## Murder         10.60    13.200     8.50  10.9000        1.400
## CV residual     2.14    -0.766     1.10  -0.0682       -0.164
##
## Sum of squares = 6.43    Mean square = 1.29    n = 5
##
## fold 5
## Observations in test set: 5
##             Indiana New Jersey Rhode Island Utah Washington
## Predicted     6.488       7.12         4.30 2.85       6.25
## cvpred        6.617       7.69         4.79 2.28       6.50
## Murder        7.100       5.20         2.40 4.50       4.30
## CV residual   0.483      -2.49        -2.39 2.22      -2.20
##
## Sum of squares = 21.9    Mean square = 4.38    n = 5
##
## fold 6
## Observations in test set: 5
##             Alabama New Hampshire Oklahoma Pennsylvania Vermont
## Predicted     12.28          4.41     6.40         8.79    3.65
## cvpred        11.88          4.52     6.17         9.15    3.67
## Murder        15.10          3.30     6.40         6.10    5.50
## CV residual    3.22         -1.22     0.23        -3.05    1.83
##
## Sum of squares = 24.6    Mean square = 4.91    n = 5
##
## fold 7
## Observations in test set: 5
##             Arkansas Florida Mississippi Oregon South Dakota
## Predicted       8.48    9.70       13.71   5.27         3.01
## cvpred          8.46    9.65       13.75   5.35         3.12
## Murder         10.10   10.70       12.50   4.20         1.70
## CV residual     1.64    1.05       -1.25  -1.15        -1.42
##
## Sum of squares = 8.68    Mean square = 1.74    n = 5
##
## fold 8
## Observations in test set: 5
##             California Connecticut Idaho   Iowa Missouri
## Predicted       10.892       3.526  4.36  3.029     7.11
## cvpred          11.161       3.793  4.17  2.969     6.86
## Murder          10.300       3.100  5.30  2.300     9.30
## CV residual     -0.861      -0.693  1.13 -0.669     2.44
##
## Sum of squares = 8.9    Mean square = 1.78    n = 5
##
```

```
## fold 9
## Observations in test set: 5
##              Colorado Delaware Maine Michigan New Mexico
## Predicted       4.14     7.27  6.15     7.93      9.436
## cvpred          3.92     7.47  6.44     7.76      9.255
## Murder          6.80     6.20  2.70    11.10      9.700
## CV residual     2.88    -1.27 -3.74     3.34      0.445
##
## Sum of squares = 35.2    Mean square = 7.04    n = 5
##
## fold 10
## Observations in test set: 5
##              Illinois Kansas Montana  Texas West Virginia
## Predicted       9.03  3.536    6.47 12.151          9.09
## cvpred          8.89  3.564    6.71 12.097          9.40
## Murder         10.30  4.500    5.00 12.200          6.70
## CV residual     1.41  0.936   -1.71  0.103         -2.70
##
## Sum of squares = 13.1    Mean square = 2.62    n = 5
##
## Overall (Sum over all 5 folds)
##   ms
## 3.65
```

Reference:

- http://math.furman.edu/~dcs/courses/math47/R/library/DAAG/html/CVlm.html

(f) Fit a regression tree using the same covariates in your "best" fit model from part (d). Use cross validation to select the "best" tree.

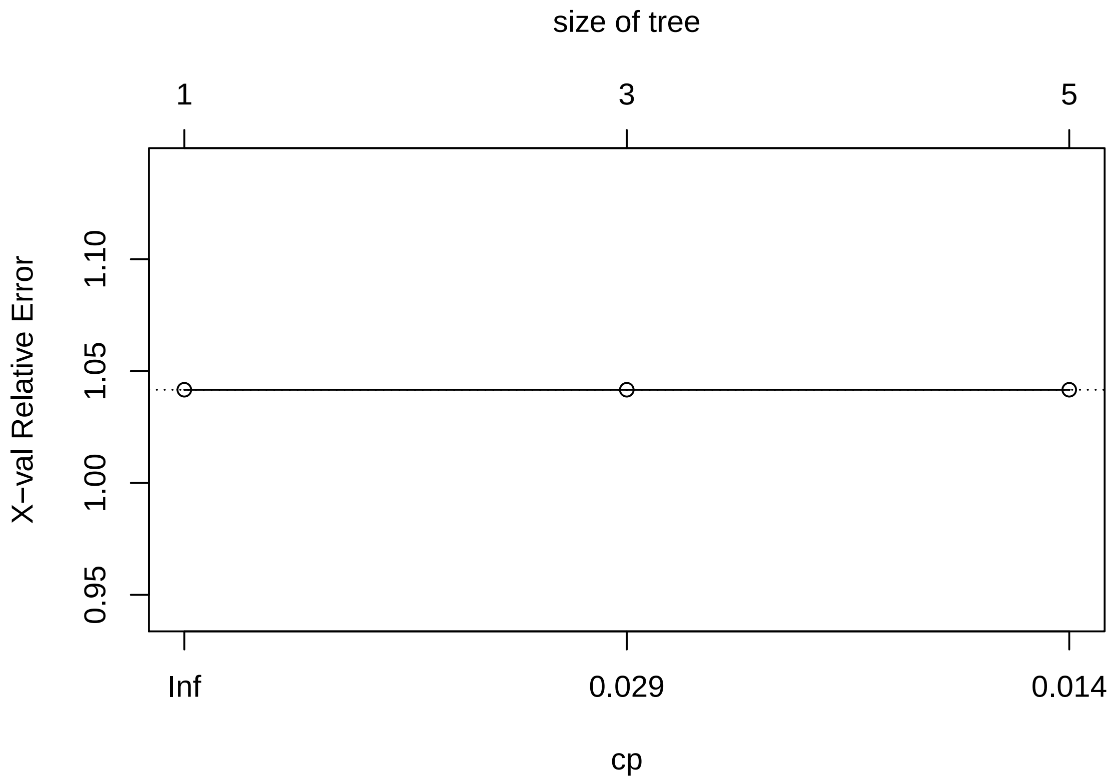
Solution: the decsion tree is divided into multiple decision makers. We notice that the life expectation above 72.31, population greater than 5377, life expectancy above 70.67 and Area under 9147 were used to split the tree.

```
# Grow tree using best fit from part d
fit <- rpart(Murder~Population+Illiteracy+`Life Exp`+Frost+Area,
             method="class", data=stateInfo)

printcp(fit) # display the results
```

```
##
## Classification tree:
## rpart(formula = Murder ~ Population + Illiteracy + `Life Exp` +
##     Frost + Area, data = stateInfo, method = "class")
##
## Variables actually used in tree construction:
## [1] Area       Life Exp   Population
##
## Root node error: 48/50 = 1
##
## n= 50
##
##     CP nsplit rel error xerror xstd
## 1 0.04      0       1.0      1    0
```

```
## 2 0.02      2       0.9      1    0
## 3 0.01      4       0.9      1    0
```

```r
plotcp(fit) # visualize cross-validation results
```



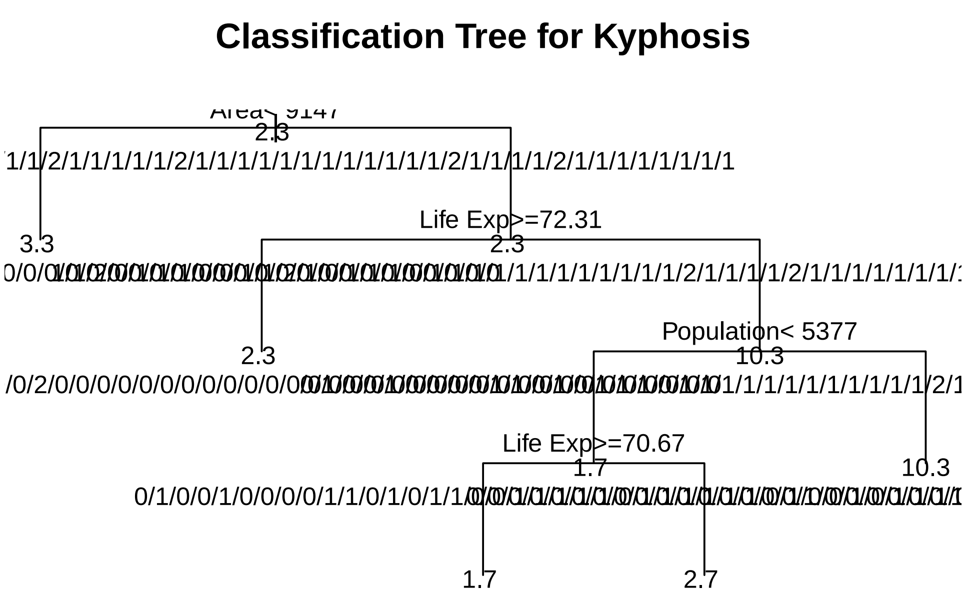

```r
summary(fit) # detailed summary of splits
```

```
## Call:
## rpart(formula = Murder ~ Population + Illiteracy + `Life Exp` +
##     Frost + Area, data = stateInfo, method = "class")
##   n= 50
##
##         CP nsplit rel error xerror xstd
## 1 0.0417      0     1.000   1.04    0
## 2 0.0208      2     0.917   1.04    0
## 3 0.0100      4     0.875   1.04    0
##
## Variable importance
##   Life Exp       Area Population Illiteracy      Frost
##         41         29         22          7          2
##
## Node number 1: 50 observations,    complexity param=0.0417
##   predicted class=2.3   expected loss=0.96  P(node) =1
##     class counts:     1     1     2     1     1     1     1     1     2     1     1     2     1
##    probabilities: 0.020 0.020 0.040 0.020 0.020 0.020 0.020 0.020 0.040 0.020 0.020 0.040 0.020
##   left son=2 (7 obs) right son=3 (43 obs)
##   Primary splits:
```

```
##       Area       < 9150   to the left,  improve=1.52, (0 missing)
##       Life Exp   < 72.5   to the right, improve=1.47, (0 missing)
##       Illiteracy < 0.65   to the left,  improve=1.28, (0 missing)
##       Population < 5380   to the left,  improve=1.28, (0 missing)
##       Frost      < 140    to the right, improve=1.13, (0 missing)
##
## Node number 2: 7 observations
##   predicted class=3.3   expected loss=0.714  P(node) =0.14
##     class counts:     0     0     0     1     0     0     0     1     2     0     0     0     0
##    probabilities: 0.000 0.000 0.000 0.143 0.000 0.000 0.000 0.143 0.286 0.000 0.000 0.000 0.000
##
## Node number 3: 43 observations,    complexity param=0.0417
##   predicted class=2.3   expected loss=0.953  P(node) =0.86
##     class counts:     1     1     2     0     1     1     1     0     0     1     1     2     1
##    probabilities: 0.023 0.023 0.047 0.000 0.023 0.023 0.023 0.000 0.000 0.023 0.023 0.047 0.023
##   left son=6 (7 obs) right son=7 (36 obs)
##   Primary splits:
##       Life Exp   < 72.3   to the right, improve=1.50, (0 missing)
##       Population < 5380   to the left,  improve=1.38, (0 missing)
##       Illiteracy < 0.65   to the left,  improve=1.25, (0 missing)
##       Frost      < 140    to the right, improve=1.17, (0 missing)
##       Area       < 80500  to the right, improve=1.10, (0 missing)
##
## Node number 6: 7 observations
##   predicted class=2.3   expected loss=0.714  P(node) =0.14
##     class counts:     1     0     2     0     0     1     1     0     0     0     0     2     0
##    probabilities: 0.143 0.000 0.286 0.000 0.000 0.143 0.143 0.000 0.000 0.000 0.000 0.286 0.000
##
## Node number 7: 36 observations,    complexity param=0.0208
##   predicted class=10.3  expected loss=0.944  P(node) =0.72
##     class counts:     0     1     0     0     1     0     0     0     0     1     1     0     1
##    probabilities: 0.000 0.028 0.000 0.000 0.028 0.000 0.000 0.000 0.000 0.028 0.028 0.000 0.028
##   left son=14 (27 obs) right son=15 (9 obs)
##   Primary splits:
##       Population < 5380   to the left,  improve=1.33, (0 missing)
##       Area       < 48300  to the right, improve=1.07, (0 missing)
##       Illiteracy < 0.85   to the left,  improve=1.06, (0 missing)
##       Frost      < 140    to the right, improve=1.03, (0 missing)
##       Life Exp   < 70.6   to the right, improve=0.98, (0 missing)
##   Surrogate splits:
##       Area < 151000 to the left,  agree=0.778, adj=0.111, (0 split)
##
## Node number 14: 27 observations,    complexity param=0.0208
##   predicted class=1.7   expected loss=0.963  P(node) =0.54
##     class counts:     0     1     0     0     1     0     0     0     0     1     1     0     1
##    probabilities: 0.000 0.037 0.000 0.000 0.037 0.000 0.000 0.000 0.000 0.037 0.037 0.000 0.037
##   left son=28 (9 obs) right son=29 (18 obs)
##   Primary splits:
##       Life Exp   < 70.7   to the right, improve=1, (0 missing)
##       Population < 1470   to the left,  improve=1, (0 missing)
##       Illiteracy < 1.85   to the left,  improve=1, (0 missing)
##       Frost      < 124    to the right, improve=1, (0 missing)
##       Area       < 43100  to the left,  improve=1, (0 missing)
##   Surrogate splits:
```

```
##       Illiteracy < 0.85   to the left,  agree=0.815, adj=0.444, (0 split)
##       Frost      < 164    to the right, agree=0.704, adj=0.111, (0 split)
##       Area       < 62300  to the right, agree=0.704, adj=0.111, (0 split)
##
## Node number 15: 9 observations
##   predicted class=10.3  expected loss=0.778  P(node) =0.18
##     class counts:     0     0     0     0     0     0     0     0     0     0     0     0     0
##    probabilities: 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
##
## Node number 28: 9 observations
##   predicted class=1.7   expected loss=0.889  P(node) =0.18
##     class counts:     0     1     0     0     0     0     0     0     0     1     1     0     0
##    probabilities: 0.000 0.111 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.111 0.111 0.000 0.000
##
## Node number 29: 18 observations
##   predicted class=2.7   expected loss=0.944  P(node) =0.36
##     class counts:     0     0     0     0     1     0     0     0     0     0     0     0     1
##    probabilities: 0.000 0.000 0.000 0.000 0.056 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.056
```

```r
# plot tree
plot(fit, uniform=TRUE,
    main="Classification Tree for Kyphosis")
text(fit, use.n=TRUE, all=TRUE, cex=.8)
```



References:

- http://www.statmethods.net/advstats/cart.html
- http://web.stanford.edu/class/stats315b/minitech.pdf

(g) Compare the models from part (d) and (f) based on their performance. Which do you prefer? Be sure to justify your preference.

Solution: the tree shows a better performance based on the split and the number of selections. It has a specific ways to identify the best fits. And it shows life expectancy as the best predictor to fit a model.

Problem 3:

The Wisconsin Breast Cancer dataset is available as a comma-delimited text file on the UCI Machine Learning Repository http://archive.ics.uci.edu/ml. Our goal in this problem will be to predict whether observations (i.e. tumors) are malignant or benign.

(a) Obtain the data, and load it into R by pulling it directly from the web. (Do not download it and import it from a CSV le.) Give a brief description of the data.

Solution: The UCI repository contains multitple data sets, however, based on the documentations provided and the question asked, we concluded that the Wisconsin Diagnostic Breast Cancer (WDBC) contains data related to predicting whether the obvervations were diagnosed as either having a malignant or benign tumor. Thus, for this analysis we will be downloading the data captured for the Wisconsin Diagnostic Breast Cancer or WDBC.

The data set containst 569 observations and 32 variables.

```
# Load data from UCI repository
mydat <- fread(
  'https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/wdbc.data'
  ,header = FALSE
  )

# Explore data sctructure
str(mydat)
```

```
## Classes 'data.table' and 'data.frame':   569 obs. of  32 variables:
##  $ V1 : int  842302 842517 84300903 84348301 84358402 843786 844359 84458202 844981 84501001 ...
##  $ V2 : chr  "M" "M" "M" "M" ...
##  $ V3 : num  18 20.6 19.7 11.4 20.3 ...
##  $ V4 : num  10.4 17.8 21.2 20.4 14.3 ...
##  $ V5 : num  122.8 132.9 130 77.6 135.1 ...
##  $ V6 : num  1001 1326 1203 386 1297 ...
##  $ V7 : num  0.1184 0.0847 0.1096 0.1425 0.1003 ...
##  $ V8 : num  0.2776 0.0786 0.1599 0.2839 0.1328 ...
##  $ V9 : num  0.3001 0.0869 0.1974 0.2414 0.198 ...
##  $ V10: num  0.1471 0.0702 0.1279 0.1052 0.1043 ...
##  $ V11: num  0.242 0.181 0.207 0.26 0.181 ...
##  $ V12: num  0.0787 0.0567 0.06 0.0974 0.0588 ...
##  $ V13: num  1.095 0.543 0.746 0.496 0.757 ...
##  $ V14: num  0.905 0.734 0.787 1.156 0.781 ...
##  $ V15: num  8.59 3.4 4.58 3.44 5.44 ...
##  $ V16: num  153.4 74.1 94 27.2 94.4 ...
##  $ V17: num  0.0064 0.00522 0.00615 0.00911 0.01149 ...
##  $ V18: num  0.049 0.0131 0.0401 0.0746 0.0246 ...
##  $ V19: num  0.0537 0.0186 0.0383 0.0566 0.0569 ...
##  $ V20: num  0.0159 0.0134 0.0206 0.0187 0.0188 ...
##  $ V21: num  0.03 0.0139 0.0225 0.0596 0.0176 ...
##  $ V22: num  0.00619 0.00353 0.00457 0.00921 0.00511 ...
```

```
##  $ V23: num  25.4 25 23.6 14.9 22.5 ...
##  $ V24: num  17.3 23.4 25.5 26.5 16.7 ...
##  $ V25: num  184.6 158.8 152.5 98.9 152.2 ...
##  $ V26: num  2019 1956 1709 568 1575 ...
##  $ V27: num  0.162 0.124 0.144 0.21 0.137 ...
##  $ V28: num  0.666 0.187 0.424 0.866 0.205 ...
##  $ V29: num  0.712 0.242 0.45 0.687 0.4 ...
##  $ V30: num  0.265 0.186 0.243 0.258 0.163 ...
##  $ V31: num  0.46 0.275 0.361 0.664 0.236 ...
##  $ V32: num  0.1189 0.089 0.0876 0.173 0.0768 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```r
head(mydat)
```

```
##          V1 V2   V3   V4    V5   V6     V7     V8     V9    V10   V11
## 1:   842302  M 18.0 10.4 122.8 1001 0.1184 0.2776 0.3001 0.1471 0.242
## 2:   842517  M 20.6 17.8 132.9 1326 0.0847 0.0786 0.0869 0.0702 0.181
## 3: 84300903  M 19.7 21.2 130.0 1203 0.1096 0.1599 0.1974 0.1279 0.207
## 4: 84348301  M 11.4 20.4  77.6  386 0.1425 0.2839 0.2414 0.1052 0.260
## 5: 84358402  M 20.3 14.3 135.1 1297 0.1003 0.1328 0.1980 0.1043 0.181
## 6:   843786  M 12.4 15.7  82.6  477 0.1278 0.1700 0.1578 0.0809 0.209
##        V12   V13   V14  V15   V16     V17    V18    V19    V20    V21
## 1: 0.0787 1.095 0.905 8.59 153.4 0.00640 0.0490 0.0537 0.0159 0.0300
## 2: 0.0567 0.543 0.734 3.40  74.1 0.00522 0.0131 0.0186 0.0134 0.0139
## 3: 0.0600 0.746 0.787 4.58  94.0 0.00615 0.0401 0.0383 0.0206 0.0225
## 4: 0.0974 0.496 1.156 3.44  27.2 0.00911 0.0746 0.0566 0.0187 0.0596
## 5: 0.0588 0.757 0.781 5.44  94.4 0.01149 0.0246 0.0569 0.0188 0.0176
## 6: 0.0761 0.335 0.890 2.22  27.2 0.00751 0.0335 0.0367 0.0114 0.0216
##        V22  V23  V24   V25  V26   V27   V28   V29   V30   V31    V32
## 1: 0.00619 25.4 17.3 184.6 2019 0.162 0.666 0.712 0.265 0.460 0.1189
## 2: 0.00353 25.0 23.4 158.8 1956 0.124 0.187 0.242 0.186 0.275 0.0890
## 3: 0.00457 23.6 25.5 152.5 1709 0.144 0.424 0.450 0.243 0.361 0.0876
## 4: 0.00921 14.9 26.5  98.9  568 0.210 0.866 0.687 0.258 0.664 0.1730
## 5: 0.00511 22.5 16.7 152.2 1575 0.137 0.205 0.400 0.163 0.236 0.0768
## 6: 0.00508 15.5 23.8 103.4  742 0.179 0.525 0.535 0.174 0.399 0.1244
```

(b) Tidy the data, ensuring that each variable is properly named and cast as the correct data type. Discuss any missing data.

Solution: The data set does not appear to have any missing values or "NA" values. Thus,there are no changes or cleanups needed.

```r
# Look for any missing values
grepl("NA", mydat)
```

```
##  [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [12] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [23] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

We have renamed the variables based on the attribute information provided by the names documentation at: https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/wdbc.names. As stated, the columns are listed as: Id, diagnosis, and the following ten-real valued features:

a) radius (mean of distances from center to points on the perimeter)

b) texture (standard deviation of gray-scale values)
c) perimeter
d) area
e) smoothness (local variation in radius lengths)
f) compactness (perimeter^2 / area - 1.0)
g) concavity (severity of concave portions of the contour)
h) concave points (number of concave portions of the contour)
i) symmetry
j) fractal dimension ("coastline approximation" - 1)

The ten-real valued features are organized based on the mean, standard error, and worst. Thus, there will be 3 set of these ten-real valued, and for readability we will append them by "mean", "standardError", and "worst" respectively.

```r
# Label the variables with descriptive value
colnames(mydat) = c(
                    "id","diagnosis","radius_mean","texture_mean",
                     "perimeter_mean", "area_mean","smoothness_mean",
                     "compactness_mean", "concavity_mean","concave_mean",
                     "symmetry_mean", "fractal_mean",
                     "radius_standardError","texture_standardError",
                     "perimeter_standardError","area_standardError",
                     "smoothness_standardError","compactness_standardError",
                     "concavity_standardError","concave_standardError",
                     "symmetry_standardError","fractal_standardError",
                     "radius_worst", "texture_worst","perimeter_worst",
                     "area_worst","smoothness_worst","compactness_worst",
                     "concavity_worst", "concave_worst", "symmetry_worst",
                      "fractal_worst"
                   )

# Update the diagnosis label to it's full code name
mydat$diagnosis = ifelse(mydat$diagnosis=="B", "benign","malignant")

# Make diagnosis a factor variable
mydat$diagnosis = as.factor(mydat$diagnosis)
```

```r
# Inspect updated variable name
str(mydat)
```

```
## Classes 'data.table' and 'data.frame':   569 obs. of  32 variables:
##  $ id                         : int  842302 842517 84300903 84348301 84358402 843786 844359 84458
##  $ diagnosis                  : Factor w/ 2 levels "benign","malignant": 2 2 2 2 2 2 2 2 2 2 ...
##  $ radius_mean                : num  18 20.6 19.7 11.4 20.3 ...
##  $ texture_mean               : num  10.4 17.8 21.2 20.4 14.3 ...
##  $ perimeter_mean             : num  122.8 132.9 130 77.6 135.1 ...
##  $ area_mean                  : num  1001 1326 1203 386 1297 ...
##  $ smoothness_mean            : num  0.1184 0.0847 0.1096 0.1425 0.1003 ...
##  $ compactness_mean           : num  0.2776 0.0786 0.1599 0.2839 0.1328 ...
##  $ concavity_mean             : num  0.3001 0.0869 0.1974 0.2414 0.198 ...
##  $ concave_mean               : num  0.1471 0.0702 0.1279 0.1052 0.1043 ...
```

```
##  $ symmetry_mean           : num  0.242 0.181 0.207 0.26 0.181 ...
##  $ fractal_mean            : num  0.0787 0.0567 0.06 0.0974 0.0588 ...
##  $ radius_standardError    : num  1.095 0.543 0.746 0.496 0.757 ...
##  $ texture_standardError   : num  0.905 0.734 0.787 1.156 0.781 ...
##  $ perimeter_standardError : num  8.59 3.4 4.58 3.44 5.44 ...
##  $ area_standardError      : num  153.4 74.1 94 27.2 94.4 ...
##  $ smoothness_standardError : num  0.0064 0.00522 0.00615 0.00911 0.01149 ...
##  $ compactness_standardError: num  0.049 0.0131 0.0401 0.0746 0.0246 ...
##  $ concavity_standardError  : num  0.0537 0.0186 0.0383 0.0566 0.0569 ...
##  $ concave_standardError    : num  0.0159 0.0134 0.0206 0.0187 0.0188 ...
##  $ symmetry_standardError   : num  0.03 0.0139 0.0225 0.0596 0.0176 ...
##  $ fractal_standardError    : num  0.00619 0.00353 0.00457 0.00921 0.00511 ...
##  $ radius_worst             : num  25.4 25 23.6 14.9 22.5 ...
##  $ texture_worst            : num  17.3 23.4 25.5 26.5 16.7 ...
##  $ perimeter_worst          : num  184.6 158.8 152.5 98.9 152.2 ...
##  $ area_worst               : num  2019 1956 1709 568 1575 ...
##  $ smoothness_worst         : num  0.162 0.124 0.144 0.21 0.137 ...
##  $ compactness_worst        : num  0.666 0.187 0.424 0.866 0.205 ...
##  $ concavity_worst          : num  0.712 0.242 0.45 0.687 0.4 ...
##  $ concave_worst            : num  0.265 0.186 0.243 0.258 0.163 ...
##  $ symmetry_worst           : num  0.46 0.275 0.361 0.664 0.236 ...
##  $ fractal_worst            : num  0.1189 0.089 0.0876 0.173 0.0768 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
# Generate numerical summaries
summary(mydat)
```

```
##        id                diagnosis     radius_mean      texture_mean 
##  Min.   :8.67e+03   benign   :357   Min.   : 6.98   Min.   : 9.7  
##  1st Qu.:8.69e+05   malignant:212   1st Qu.:11.70   1st Qu.:16.2  
##  Median :9.06e+05                   Median :13.37   Median :18.8  
##  Mean   :3.04e+07                   Mean   :14.13   Mean   :19.3  
##  3rd Qu.:8.81e+06                   3rd Qu.:15.78   3rd Qu.:21.8  
##  Max.   :9.11e+08                   Max.   :28.11   Max.   :39.3  
##  perimeter_mean    area_mean    smoothness_mean  compactness_mean
##  Min.   : 43.8   Min.   : 144   Min.   :0.0526   Min.   :0.019   
##  1st Qu.: 75.2   1st Qu.: 420   1st Qu.:0.0864   1st Qu.:0.065   
##  Median : 86.2   Median : 551   Median :0.0959   Median :0.093   
##  Mean   : 92.0   Mean   : 655   Mean   :0.0964   Mean   :0.104   
##  3rd Qu.:104.1   3rd Qu.: 783   3rd Qu.:0.1053   3rd Qu.:0.130   
##  Max.   :188.5   Max.   :2501   Max.   :0.1634   Max.   :0.345   
##  concavity_mean   concave_mean     symmetry_mean    fractal_mean   
##  Min.   :0.000   Min.   :0.0000   Min.   :0.106   Min.   :0.0500  
##  1st Qu.:0.030   1st Qu.:0.0203   1st Qu.:0.162   1st Qu.:0.0577  
##  Median :0.062   Median :0.0335   Median :0.179   Median :0.0615  
##  Mean   :0.089   Mean   :0.0489   Mean   :0.181   Mean   :0.0628  
##  3rd Qu.:0.131   3rd Qu.:0.0740   3rd Qu.:0.196   3rd Qu.:0.0661  
##  Max.   :0.427   Max.   :0.2012   Max.   :0.304   Max.   :0.0974  
##  radius_standardError texture_standardError perimeter_standardError
##  Min.   :0.112        Min.   :0.36          Min.   : 0.76          
##  1st Qu.:0.232        1st Qu.:0.83          1st Qu.: 1.61          
##  Median :0.324        Median :1.11          Median : 2.29          
##  Mean   :0.405        Mean   :1.22          Mean   : 2.87          
##  3rd Qu.:0.479        3rd Qu.:1.47          3rd Qu.: 3.36          
```

```
##  Max.   :2.873        Max.   :4.88          Max.   :21.98
##  area_standardError smoothness_standardError compactness_standardError
##  Min.   :  7        Min.   :0.00171          Min.   :0.0023
##  1st Qu.: 18        1st Qu.:0.00517          1st Qu.:0.0131
##  Median : 25        Median :0.00638          Median :0.0204
##  Mean   : 40        Mean   :0.00704          Mean   :0.0255
##  3rd Qu.: 45        3rd Qu.:0.00815          3rd Qu.:0.0324
##  Max.   :542        Max.   :0.03113          Max.   :0.1354
##  concavity_standardError concave_standardError symmetry_standardError
##  Min.   :0.000           Min.   :0.0000        Min.   :0.0079
##  1st Qu.:0.015           1st Qu.:0.0076        1st Qu.:0.0152
##  Median :0.026           Median :0.0109        Median :0.0187
##  Mean   :0.032           Mean   :0.0118        Mean   :0.0205
##  3rd Qu.:0.042           3rd Qu.:0.0147        3rd Qu.:0.0235
##  Max.   :0.396           Max.   :0.0528        Max.   :0.0790
##  fractal_standardError  radius_worst  texture_worst  perimeter_worst
##  Min.   :0.00089       Min.   : 7.9   Min.   :12.0   Min.   : 50.4
##  1st Qu.:0.00225       1st Qu.:13.0   1st Qu.:21.1   1st Qu.: 84.1
##  Median :0.00319       Median :15.0   Median :25.4   Median : 97.7
##  Mean   :0.00379       Mean   :16.3   Mean   :25.7   Mean   :107.3
##  3rd Qu.:0.00456       3rd Qu.:18.8   3rd Qu.:29.7   3rd Qu.:125.4
##  Max.   :0.02984       Max.   :36.0   Max.   :49.5   Max.   :251.2
##    area_worst   smoothness_worst compactness_worst concavity_worst
##  Min.   : 185   Min.   :0.0712   Min.   :0.027     Min.   :0.000
##  1st Qu.: 515   1st Qu.:0.1166   1st Qu.:0.147     1st Qu.:0.114
##  Median : 686   Median :0.1313   Median :0.212     Median :0.227
##  Mean   : 881   Mean   :0.1324   Mean   :0.254     Mean   :0.272
##  3rd Qu.:1084   3rd Qu.:0.1460   3rd Qu.:0.339     3rd Qu.:0.383
##  Max.   :4254   Max.   :0.2226   Max.   :1.058     Max.   :1.252
##  concave_worst    symmetry_worst  fractal_worst
##  Min.   :0.0000   Min.   :0.156   Min.   :0.0550
##  1st Qu.:0.0649   1st Qu.:0.250   1st Qu.:0.0715
##  Median :0.0999   Median :0.282   Median :0.0800
##  Mean   :0.1146   Mean   :0.290   Mean   :0.0839
##  3rd Qu.:0.1614   3rd Qu.:0.318   3rd Qu.:0.0921
##  Max.   :0.2910   Max.   :0.664   Max.   :0.2075
```

(c) Split the data into a training and validation set such that a random 70% of the observations are in the training set.

Solution: the train data contains 398 records which is 70% of the whole data set. Similarly, the test data set contains 171 which is roughly 30% of the overall data set. Note that the original data set contains 569 records.

```r
# Split into a training set and a test set
set.seed(3456)
nrow(mydat)
```

```
## [1] 569
```

```r
num_train = round(.7*nrow(mydat))
trainInds = sample(1:nrow(mydat), num_train)
train = mydat[trainInds,]
test  = mydat[-trainInds,]
```

```r
# Check total records for train and test data set
nrow(train)
```

```
## [1] 398
```

```r
nrow(test)
```

```
## [1] 171
```

Solution: the training data contains 244 patients that were diagnosed as benign and 154 patients that are diagnosed as having a malignant breast cancer. On the other hand, the test data shows 113 patients that were diagnosed as benign and 58 patients that were diagnosed as having a malignant breast cancer.

```r
#frequency train data based on the diagnosis
table(train$diagnosis)
```

```
##
##    benign malignant
##       244       154
```

```r
#frequency test data based on the diagnosis
table(test$diagnosis)
```

```
##
##    benign malignant
##       113        58
```

(d) Fit a regression model to predict whether tissue samples are malignant or benign. Classify cases in the validation set. Compute and discuss the resulting confusion matrix.

Solution: We chose to use all of the predictor variables (except for id which has no impact on the diagnosis) for the first regression model. Then, we will identify the most significantvariables. Also, we decided to use a logistic regression function for this model because we are predicting a binary outcome of whether a breast cancer patient has either a benign tumor or a malignant tumor. Moreover, we decided to use the data sets corresponding to the mean and standard error and ignore the "worst" data set. We don't expect the "worst" data to provide any significant value in evaluating whether a patient is either begnin or malignant.

```r
# Fit a logistic regression model for breast cancer diagnosis
set.seed(359)
wbc.glm = glm(diagnosis~radius_mean+texture_mean+perimeter_mean+area_mean+smoothness_mean+
               compactness_mean+concavity_mean+concave_mean+symmetry_mean+fractal_mean+
               radius_standardError+texture_standardError+perimeter_standardError+
               area_standardError+smoothness_standardError+compactness_standardError+
               concavity_standardError+concave_standardError+
               symmetry_standardError+fractal_standardError
               ,data = train
              ,family = binomial
              )
```

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

```r
summary(wbc.glm) # Model summary
```

```
##
## Call:
## glm(formula = diagnosis ~ radius_mean + texture_mean + perimeter_mean +
##     area_mean + smoothness_mean + compactness_mean + concavity_mean +
##     concave_mean + symmetry_mean + fractal_mean + radius_standardError +
##     texture_standardError + perimeter_standardError + area_standardError +
##     smoothness_standardError + compactness_standardError + concavity_standardError +
##     concave_standardError + symmetry_standardError + fractal_standardError,
##     family = binomial, data = train)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.703  -0.055  -0.012   0.000   3.398
##
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)                -20.9851    25.9288   -0.81    0.418
## radius_mean                 -1.7584     8.8725   -0.20    0.843
## texture_mean                 0.5650     0.1331    4.24  2.2e-05 ***
## perimeter_mean              -0.2542     1.1738   -0.22    0.829
## area_mean                    0.0490     0.0344    1.42    0.154
## smoothness_mean             91.6184    77.6181    1.18    0.238
## compactness_mean           -92.0388    64.6320   -1.42    0.154
## concavity_mean             141.9806    50.5724    2.81    0.005 **
## concave_mean               -20.4291    71.2557   -0.29    0.774
## symmetry_mean               51.5035    27.6205    1.86    0.062 .
## fractal_mean               209.2255   215.0836    0.97    0.331
## radius_standardError       -12.9707    22.2273   -0.58    0.560
## texture_standardError       -2.1386     1.2625   -1.69    0.090 .
## perimeter_standardError     -0.8896     1.5924   -0.56    0.576
## area_standardError           0.2960     0.2051    1.44    0.149
## smoothness_standardError   -90.0832   191.5724   -0.47    0.638
## compactness_standardError  120.8828    87.3968    1.38    0.167
## concavity_standardError   -170.4424    97.2757   -1.75    0.080 .
## concave_standardError      104.1190   194.2242    0.54    0.592
## symmetry_standardError     -48.6699    83.6985   -0.58    0.561
## fractal_standardError     -699.3076   569.0220   -1.23    0.219
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 531.216  on 397  degrees of freedom
## Residual deviance:  61.962  on 377  degrees of freedom
## AIC: 104
##
## Number of Fisher Scoring iterations: 11
```

We found that texture_mean,concativity_mean,area_standardError, and concavity_standardError were the only predictors that were significant at 0.05.

Next, let's consider the performance of this model.

```r
# Get predicted probabilities
yhat = predict(wbc.glm, newdata = test, type = "response")
```

Next, we will Use a threshold of 0.5 to classify predictions.

```r
# Construct confusion matrix
table(test$diagnosis, yhat>.5)
```

```
##
##              FALSE TRUE
##   benign       110    3
##   malignant      2   56
```

The confusion matrix tells us that the classifier is accurate at 95.37% or (108+57)/173. Also, we noticed that the model makes 4 false positives related to begnin diagnosis. In other words, these false positive are cases where we predcited patients would be diagnosed as having a benign breast cancer, but instead they had a malignant breast cancer. These false positive rate is at 3.6% or 4/112. Although the wrong diagnosis could be a life changer, we can however say that at 3.6% false positive rate, this is a very good prediction model.

References:

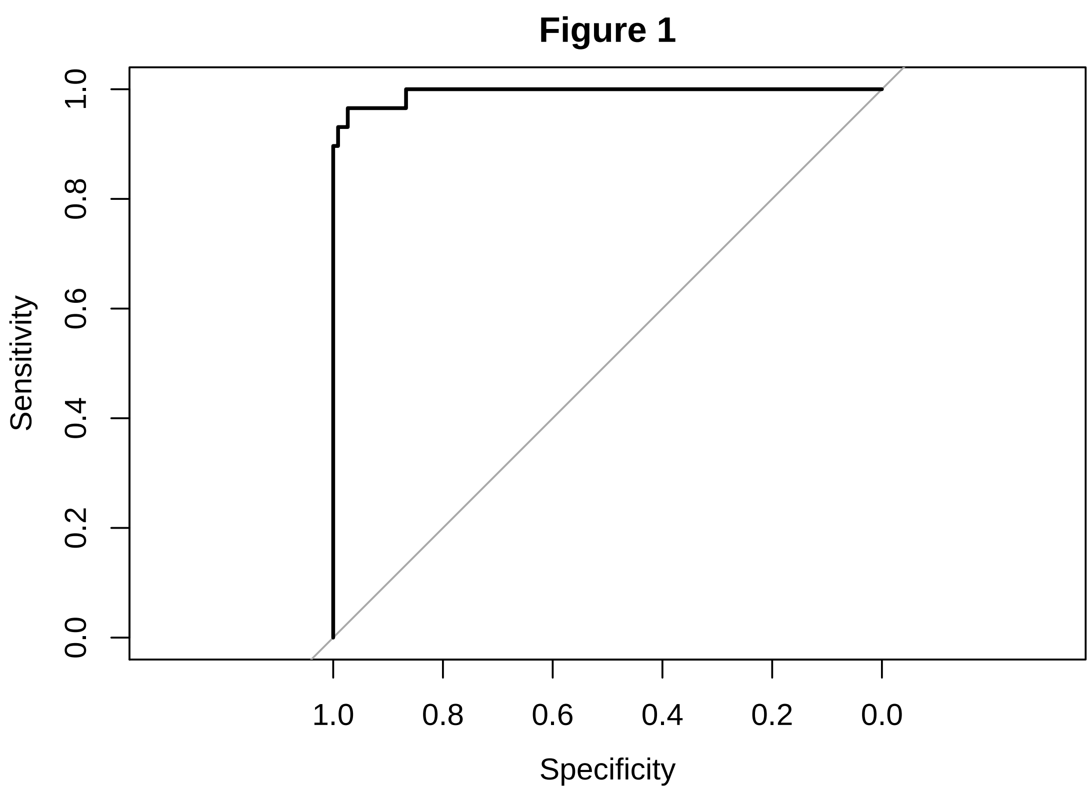
- http://www.dataschool.io/simple-guide-to-confusion-matrix-terminology/

Next, we will plot the ROC curve for this model

```r
# Plot the ROC curve
roc(test$diagnosis, yhat)
```

```
##
## Call:
## roc.default(response = test$diagnosis, predictor = yhat)
##
## Data: yhat in 113 controls (test$diagnosis benign) < 58 cases (test$diagnosis malignant).
## Area under the curve: 0.994
```

```r
plot(roc(test$diagnosis, yhat), main="Figure 1")
```




```
##
## Call:
## roc.default(response = test$diagnosis, predictor = yhat)
##
## Data: yhat in 113 controls (test$diagnosis benign) < 58 cases (test$diagnosis malignant).
## Area under the curve: 0.994
```

We decided to use ROC because it's the most commonly used way to visualize the performance of a binary classifier. Also, we noticed that the AUC or Area under the curve is at 0.99. Ideally, the closer the AUC value is to 1, then the true positive rate will increase quickly. In this case, we will get a model with a higher number of patients being diagnosed accordingly on the type of breast cancer tumor they may have had. Thus, because we have an AUC that's closed to 1 than we can conclude that the classifier is very good.

When analyzing the ROC curve, we noticed that the curve is above the diagonal line. Also, the ROC shows a pretty good curve that's continualy increasing. In fact, it shows that the curve is getting closer to the upper left hand corner. This is a sign of a close fit of the model.

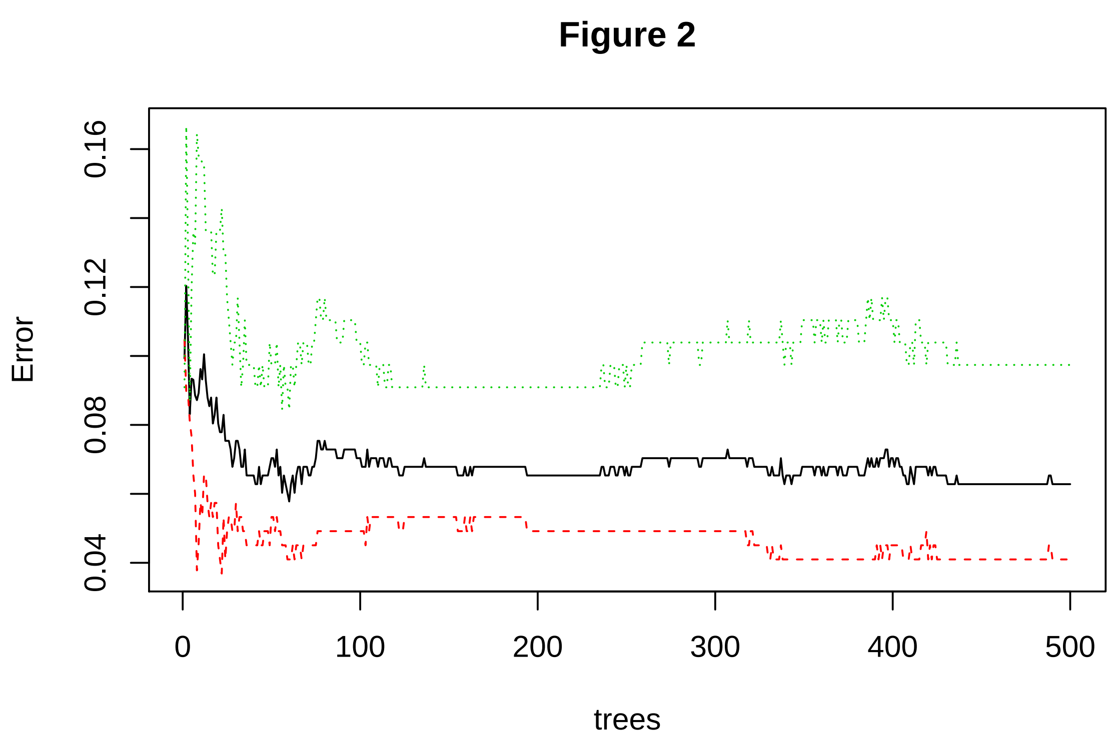
References:

– https://www.kaggle.com/wiki/AreaUnderCurve
– http://blog.yhat.com/posts/roc-curves.html

(e) Fit a random forest model to predict whether tissue samples are malignant or benign. Classify cases in the validation set. Compute and discuss the resulting confusion matrix.

Solution: Below we fit the random forest model. We explore the models based on the visualization.

```r
#Factors needed for RandomForest classifier
train$diagnosis = factor(train$diagnosis)

# Fit the RandomForest model
wbc.rf = randomForest(diagnosis~radius_mean+texture_mean+perimeter_mean+area_mean+smoothness_mean+
                        compactness_mean+concavity_mean+concave_mean+symmetry_mean+fractal_mean+
                        radius_standardError+texture_standardError+perimeter_standardError+
                        area_standardError+smoothness_standardError+compactness_standardError+
                        concavity_standardError+concave_standardError+symmetry_standardError+
                        fractal_standardError
                      ,data = train
                      ,na.action=na.exclude)

# Explore the model
plot(wbc.rf, main="Figure 2")
```




```r
importance(wbc.rf)
```

```
##                          MeanDecreaseGini
## radius_mean                         20.37
## texture_mean                         7.64
## perimeter_mean                      22.71
## area_mean                           24.37
## smoothness_mean                      3.37
## compactness_mean                     7.01
## concavity_mean                      23.58
```

```
## concave_mean                        32.18
## symmetry_mean                        2.41
## fractal_mean                         1.56
## radius_standardError                 6.53
## texture_standardError                1.75
## perimeter_standardError              5.62
## area_standardError                  12.60
## smoothness_standardError             2.41
## compactness_standardError            2.65
## concavity_standardError              3.64
## concave_standardError                2.42
## symmetry_standardError               2.59
## fractal_standardError                2.95
```

Solution: 500 decision trees or a forest has been built using the Random Forest algorithm based learning. The plot seems to indicate that around 350 decision trees, there is not a significant reduction in error rate. Also the mean importance showed that variables: Concave_mean, area_mean, concativity_mean, perimeter_mean, radius_mean, and area_standardError have the highest MeanDecreaseGini score. Thus, these variables have the highest importance for this model.

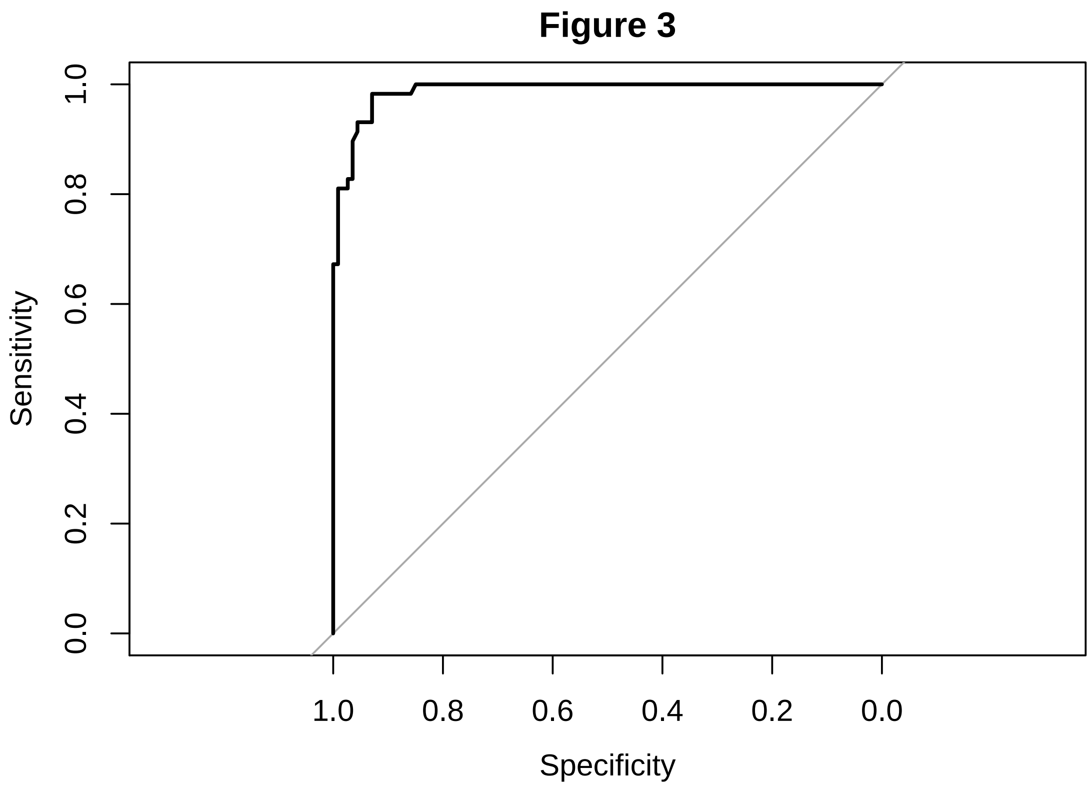
```r
# Save prediction from random forest model
yhat2 = predict(wbc.rf, newdata = test, type="prob")[,1]
```

Next, we will plot the ROC curve for this model

```r
# Plot the ROC curve
roc(test$diagnosis, yhat2)
```

```
##
## Call:
## roc.default(response = test$diagnosis, predictor = yhat2)
##
## Data: yhat2 in 113 controls (test$diagnosis benign) > 58 cases (test$diagnosis malignant).
## Area under the curve: 0.988
```

```r
plot(roc(test$diagnosis, yhat2), main="Figure 3")
```




```
##
## Call:
## roc.default(response = test$diagnosis, predictor = yhat2)
##
## Data: yhat2 in 113 controls (test$diagnosis benign) > 58 cases (test$diagnosis malignant).
## Area under the curve: 0.988
```

When analyzing the ROC curve, we noticed that the curve is above the diagonal line. Also, the ROC shows a pretty good curve that's continualy increasing. In fact, it shows that the curve is getting closer to the upper left hand corner. This is a sign of a close fit of the model. Also, it shows an AUC = 0.988. This AUC is very good as an AUC = 1 is considered to be a good fit.

References:

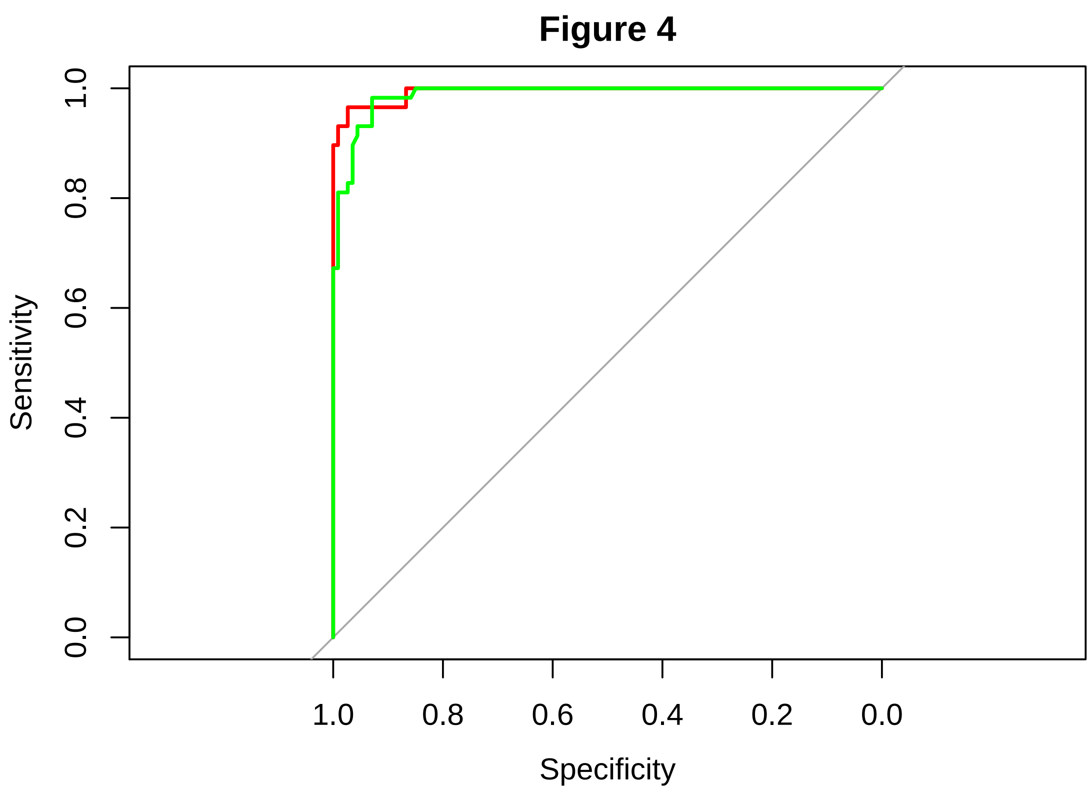
- http://www.listendata.com/2014/11/random-forest-with-r.html

(f) Compare the models from part (d) and (e) using ROC curves. Which do you prefer? Be sure to justify your preference.

```r
plot(roc(test$diagnosis, yhat), col='red', main="Figure 4")
```

```
##
## Call:
## roc.default(response = test$diagnosis, predictor = yhat)
##
## Data: yhat in 113 controls (test$diagnosis benign) < 58 cases (test$diagnosis malignant).
## Area under the curve: 0.994
```

```r
lines(roc(test$diagnosis, yhat2), col='green')
```



Solution: The above code compares the ROC curves for both the regression model (red line) and the randomforest model (green line). We can see that both models are very close to each other. There is a slight improvement of the logistic regression model over the random forest model. Also, when comparing the AUC(Area Under the Curve) for both models, we noticed that the linear regression model fair a little better with a AUC = 0.994 (see figure 1) while the random forest model has a AUC = 0.988 (see figure 3).

Problem 4:

Please answer the questions below by writing a short response.

(a) Describe three real-life applications in which classiffication might be useful. Describe the response, as well as the predictors. Is the goal in each application inference or predictions? Explain your answer.

Solution:

- Real-life application 1 - Should a college footbal team be admitted in the football playoff? Response: Admit/No Admit. Predictors: strength of schedule, championships won, head-to-head competition, Comparative outcomes of common opponents, number of win games. The goal will be a Prediction. Example: University of Washgington college football team.
- Real-life application 2 - Is this new mobile game will be successfull or not? Response: Success/Failure. Predictors: money spent for development, money made from selling ads, easy to learn and play, ease of game progression, goal-oriented gameplay, exciting story-line/plot, target audience. The goal will be a prediction. Example: Angry Birds.

– Real-life application 3 - Should I buy a new stock for my investment? Response: Buy/Not buy. Predictors: 52 week range, volume (number of stocks bought and sold in a single day), price earnings ratio, earnings per share, market cap, stock volatility, dividends, open interests in option chains, insider activity, news/popularity. The goal will be a prediction. Example: EXP (Expedia)

References:

– https://toughnickel.com/personal-finance/10-Factors-to-Consider-When-Selecting-a-Stock
– http://www.gamasutra.com/blogs/IgorMatrofailo/20160107/263164/5_Criteria_of_a_Successful_Mobile_Game.php

(b) Describe three real-life applications in which regression might be useful. Describe the response, as well as the predictors. Is the goal in each application inference or predictions? Explain your answer.

Solution:

– Real-life regression 1 - What is the average salary that new computer engineers should expect over the next five years? Response: graduate computer engineer salary for the first year, second year, etc. Predictors: education level, jobs demand, gender, years of experience, speciality, school attended, cities or states where job is located, professional certifications, total internships completed. The goal will be an inference. Example: Google.
– Real-life regression 2 - Gas milleage that a new jet liner design will result in. Response: variations on the fuel economy by an airline. Predictors: aircraft makers, how fast it flies, how far it flies, number of passengers on the airplane, size of cargo packs, wide or narrow body-jet, structure of the plane, size of fuel on the plane. The goal will be an inference. Example: Alaska Airlines.
– Real-life regression 3 - High-school graduation increase for minority students. Response: what is the graduation rate predicted to be by 2030 for minority students? Predictors: academic preparation, teacher's skills, math and science rating, readability score, peer-mentoring availability, summer remedial courses availability, minority teachers availability, family and community involvement. The goal will be an inference. Example: Minnesota high school.

References:

– http://www.wsj.com/articles/SB10001424052748704901104575423261677748380
– http://www.startribune.com/minnesota-graduation-rates-flat-but-more-minority-students-finishing-school/369661641/

(c) What are the advantages and disadvantages of a very flexible (versus a less flexible) approach for regression or classiffication? Under what circumstances might a more flexible approach be preferred to a less flexible approach? When might a less flexible approach be preferred?

Solution:

The advantages of a very flexible approach for regression or classification are:

– Tends to reduce bias
– Less likely to overfit when using a larget data set
– In most cases, it will perform better than a less flexible approach
– Help in fiding a non-linear option

The disavantages of a very flexible approach are:

– More prone to overfitting

More flexible approach is preffered over a less flexible when:

– The sample size is large and the number of predictors is small.

- The relationship between the predictors and the sample size is small
- Use for prediction model

Less flexible approach is preffered over a more flexible when:

- The number of predictors is large and the sample size is small
- The variance of the errors is large
- Use for inference model

Problem 5

Suppose we have a dataset with ve predictors, X1 = GPA, X2 = IQ, X3 = Gender (1 for Female, and 0 for Male), X4 = Interaction between GPA and IQ, and X5 = Interaction between GPA and Gender. The response is starting salary after graduation (in thousands of dollars). Suppose we use least squares to fit the model and get, β_0=50,β_1=20,β_2=0.07,β_3=35,β_4=0.01, and β_5=-10.

(a) Which answer is correct and why?

i. For a fixed value of IQ and GPA, males earn more on average than females.
ii. For a fixed value of IQ and GPA, females earn more on average than males.
iii. For a fixed value of IQ and GPA, males earn more on average than females provided that the GPA is high enough.
iv. For a fixed value of IQ and GPA, females earn more on average than males provided that the GPA is high enough.

Solution:

Salary = 50 + 20X1 + 0.07X2 + 35X3 + 0.01X4-10X5

X4 = X1*X2 because it represents the interaction between GPA and IQ. X5 = X1*X3 because it represents the interaction between GPA and Gender. Since, the question is about gender, then we will consider terms that contain gender and everything else will be part of the constant value. Thus, our equation will change to:

Salary = 35X3 - 10X5 + constant

replacing X5 by it's correspond values, we got Salary = 35X3 - 10(X1*X3) + constant

*Case 1: calculating a female's salary knowing that a female's gender = 1 then replacing X3 by 1 thus we have salary = 35 -10(1*GPA) + constant Salary = 35 - 10GPA + constant

*Case 2: calculating a male's salary knowing that a male's gender = 0 then replacing X3 by 0 thus we have salary = 35*0 -10(0*GPA) + constant Salary = constant

Assuming both gender's make the salary then we can solve for GPA. 35-10GPA+constant = constant 35 -10GPA = 0 -10GPA = -35 or 10GPA = 35 or GPA = 35/10 or 3.5

We can conclude the followings:

- scenario 1. if men have a GPA greater than 3.5 then women will make less. 35-10*4 will result to a negative value and adding to a constant will always be less than men's constant value.
- scenario 2. If men have a GPA less than 3.5 then women will make more than men. 35-10*2.0 will result to a positive value added to the constant value which wil be greater than men's constant value.
- scenario 3. If men have a GPA = 3.5 then both women and men will make the same salary. 35-10.3.5 = 0 will result to both men and women having the same constant salary.

Based on the scenearios listed, then option 3 is the correct answer.

(b) Predict the salary of a female with IQ of 110 and a GPA of 4.0.

Solution:

Salary = 50 + 20X1 + 0.07X2 + 35X3 + 0.01(X1*X2)-10(X1*X3) salary = 50 + 20*4.0 + 0.07*110 + 35*1 + 0.01(4.0*110) - 10(4.0*1) salary = 50+80.0+7.7+35+4.4-40 = 137.1

A female's with IQ = 110 and a GPA = 4.0 will have a salary = $137,100 per year.

(c) True or false: Since the coefficient for the GPA/IQ interaction term is very small, there is little evidence of an interaction effect. Justify your answer.

Solution: **False.** We will need to calculate the null hypothesis probability of wheher there is little evidence of an interaction effect. if the null hypothesis is less than the selected type 1 error threshold then we reject the null hypothesis.

Problem 6 - Extra Credit

Apply boosting, bagging and random forests to a dataset of your choice that we have used in class. Be sure to fit the models on a training set and evaluate their performance on a test set.

(a) How accurate are the results compared to simple methods like linear or logistic regression?

(b) Which of the approaches yields the best performance?

Problem 7 - Extra Credit

Suppose that X1; : : : Xn form a random sample from a Poisson distribution for which the mean pierre is unknown, (> 0).

(a) Determine the MLE of pierre, assuming that at least one of the observed values is different from 0. Show your work.

(b) Show that the MLE of does not exists if every observed value is 0.

Statement of Compliance

I affirm that I have had no conversation regarding this exam with any persons other than the instructor (Dr. Emma Spiro). Further, I certify that the attached work represents my own thinking. Any information, concepts, or words that originate from other sources are cited in accordance with University of Washington guidelines as published in the Academic Code (available on the course website). I am aware of the serious consequences that result from improper discussions with others or from the improper citation of work that is not my own.

(signature) Pierre Augustamar

(date) 12/13/2016