R Notebook

Problem 1 (60 Points) 1. (0 pts) Download the data set Census Income Data for Adults along with its explanation. Note that the data file does not contain header names; you may wish to add those. The description of each column can be found in the data set explanation.

```
library(caret)
```

```
## Loading required package: lattice

## Loading required package: ggplot2
```

```
library(dplyr)
```

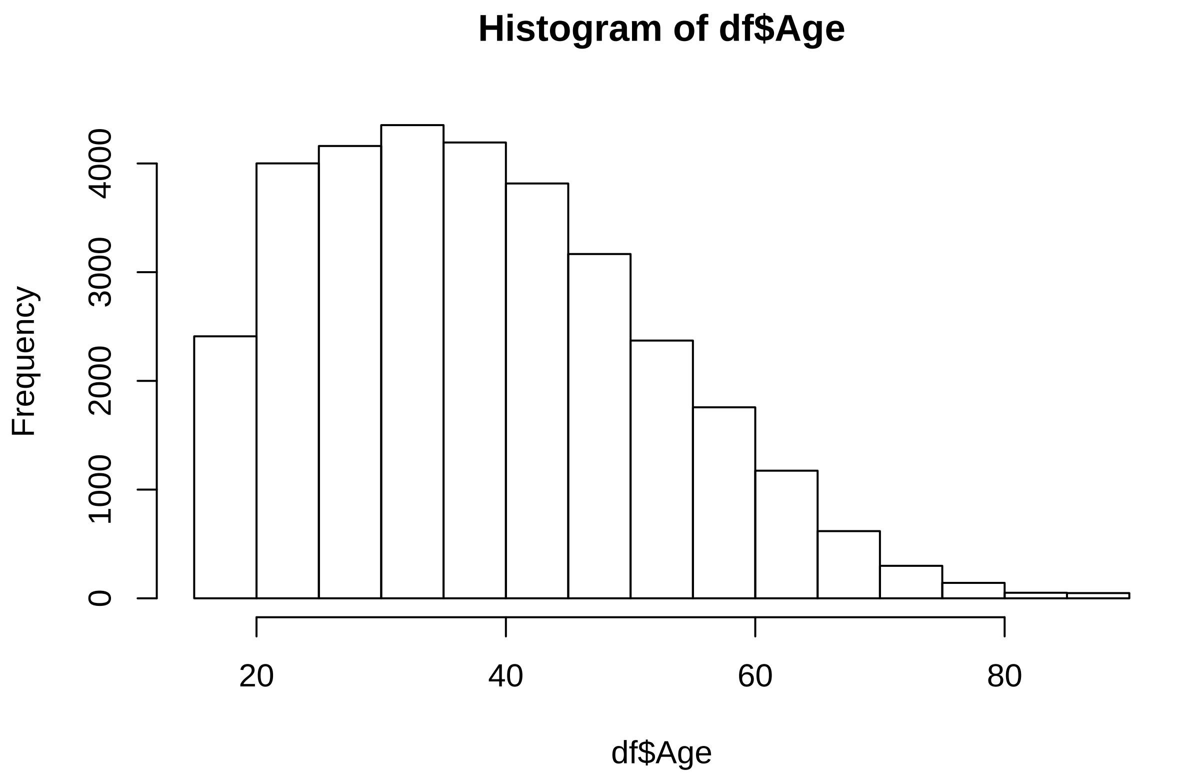
```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
#setwd('~/Documents/ML/')
df <- read.csv("adult.data.txt", header =  F)
colnames(df) <- c("Age","Workclass","fnlwgt","Education","Ed-num","Marital-status","Occupation","Relati
```

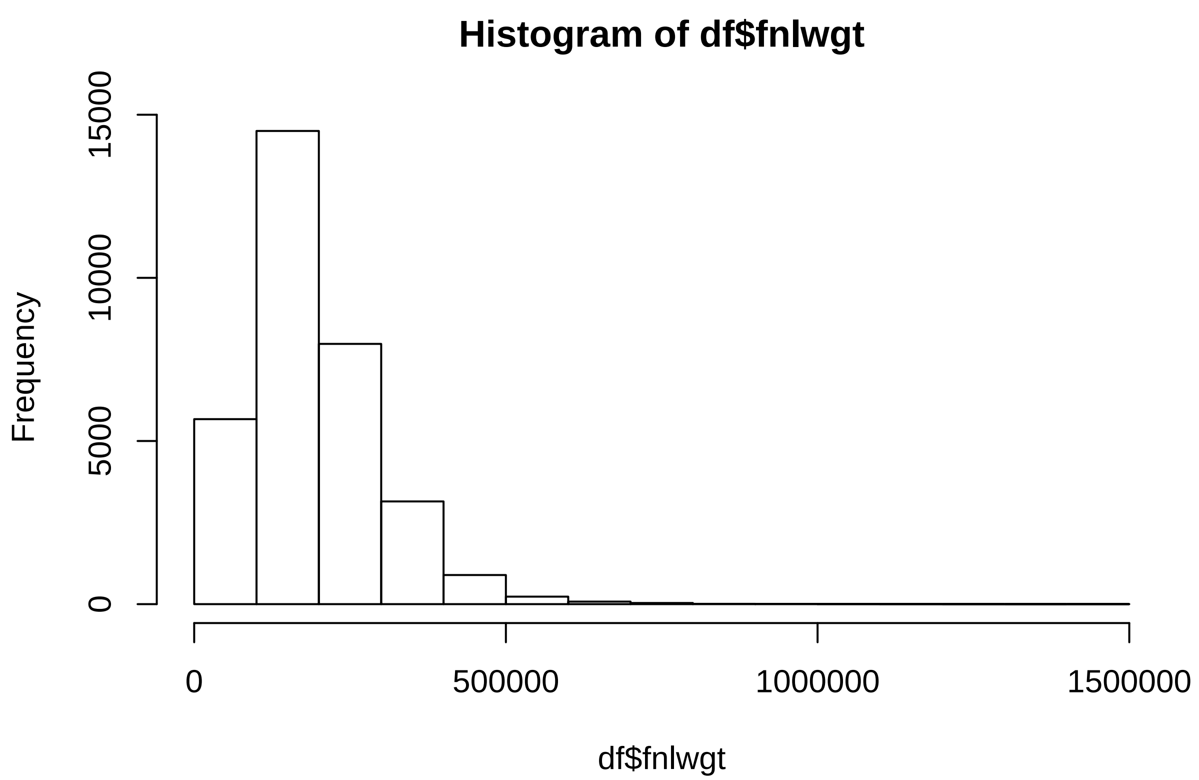
2. (0 pts) Explore the data set as you see fit and that allows you to get a sense of the data and get comfortable with it. Is there distributional skew in any of the features? Is there a need to apply a transform?

Yes, there is a distribution skew in Age, fnlwgt, Education numbers, Capital loss and capital gain. If we were going to use those features, they would have to be normalized and then converted to categorical features by binning. But since we are not going to use them for modeling, there isn't any need to transform them.

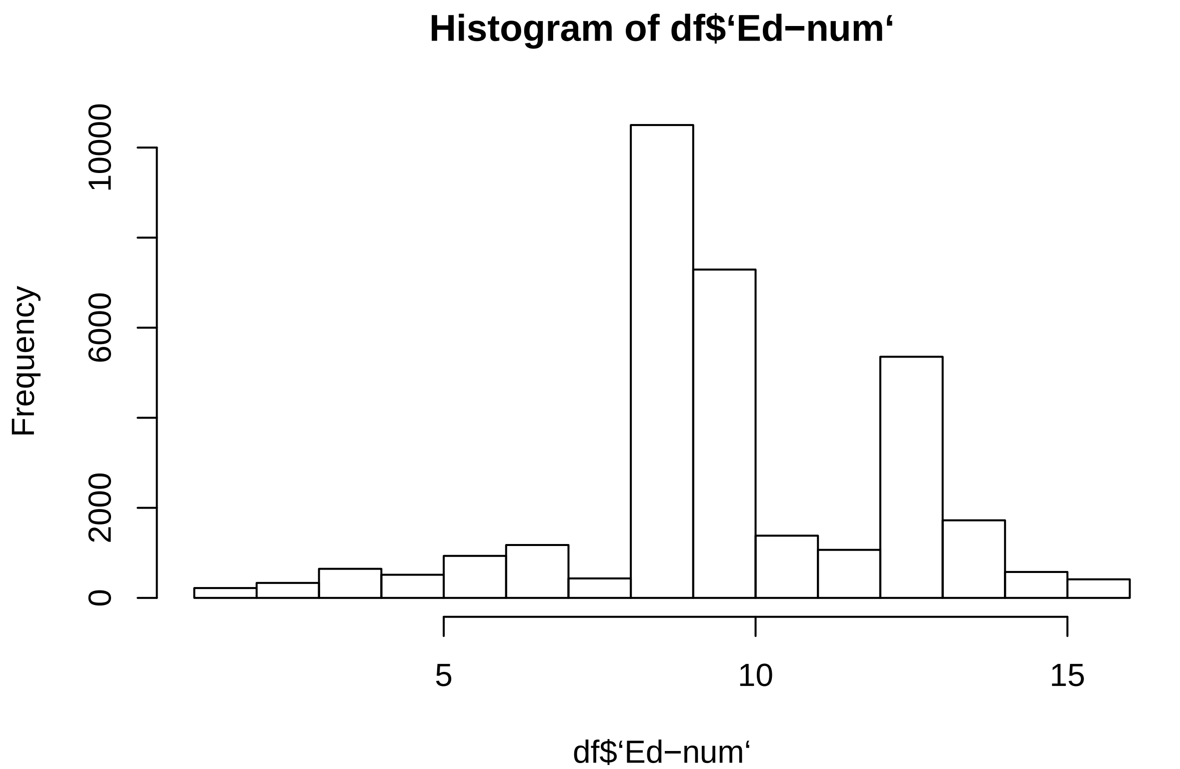
```
hist(df$Age)
```



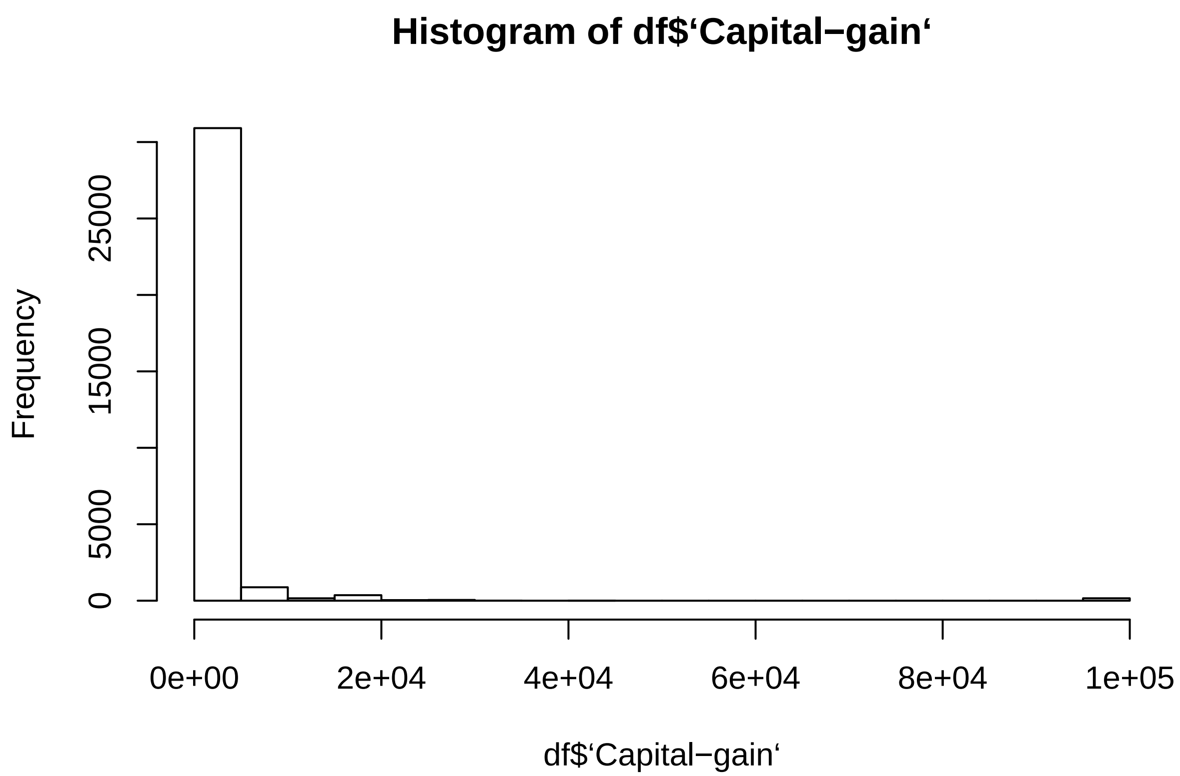

```r
hist(df$fnlwgt)
```



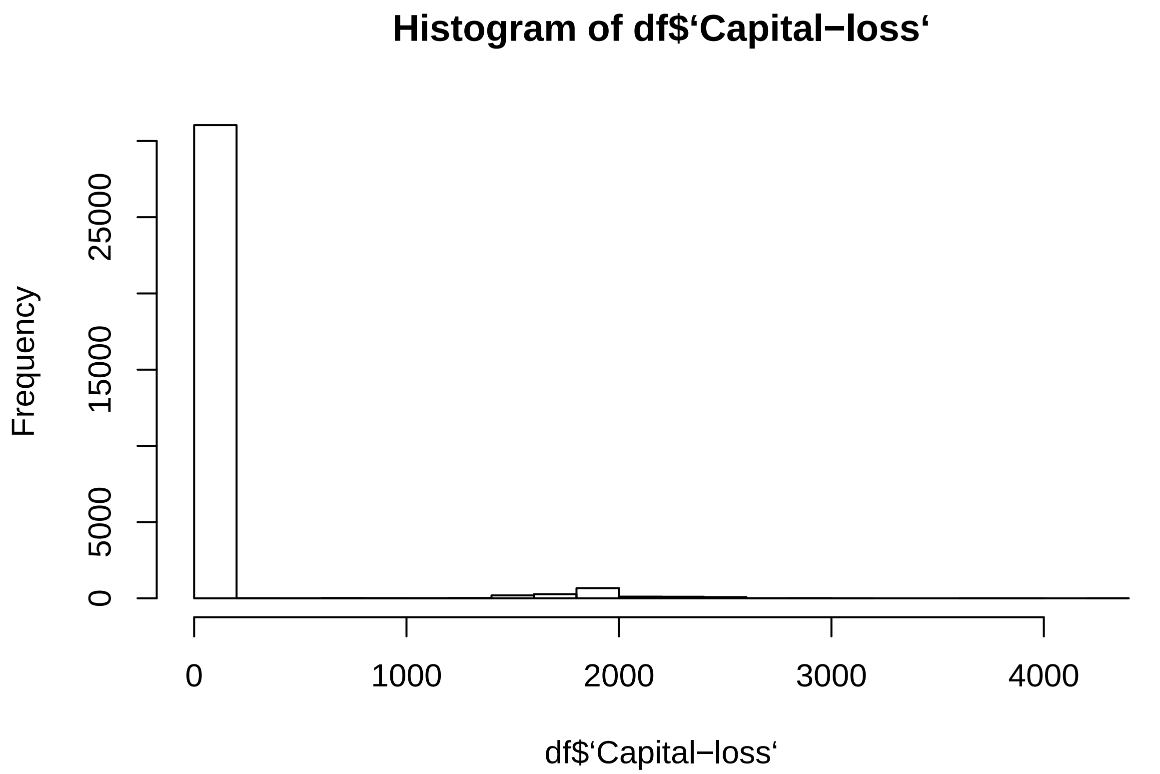

```r
hist(df$`Ed-num`)
```



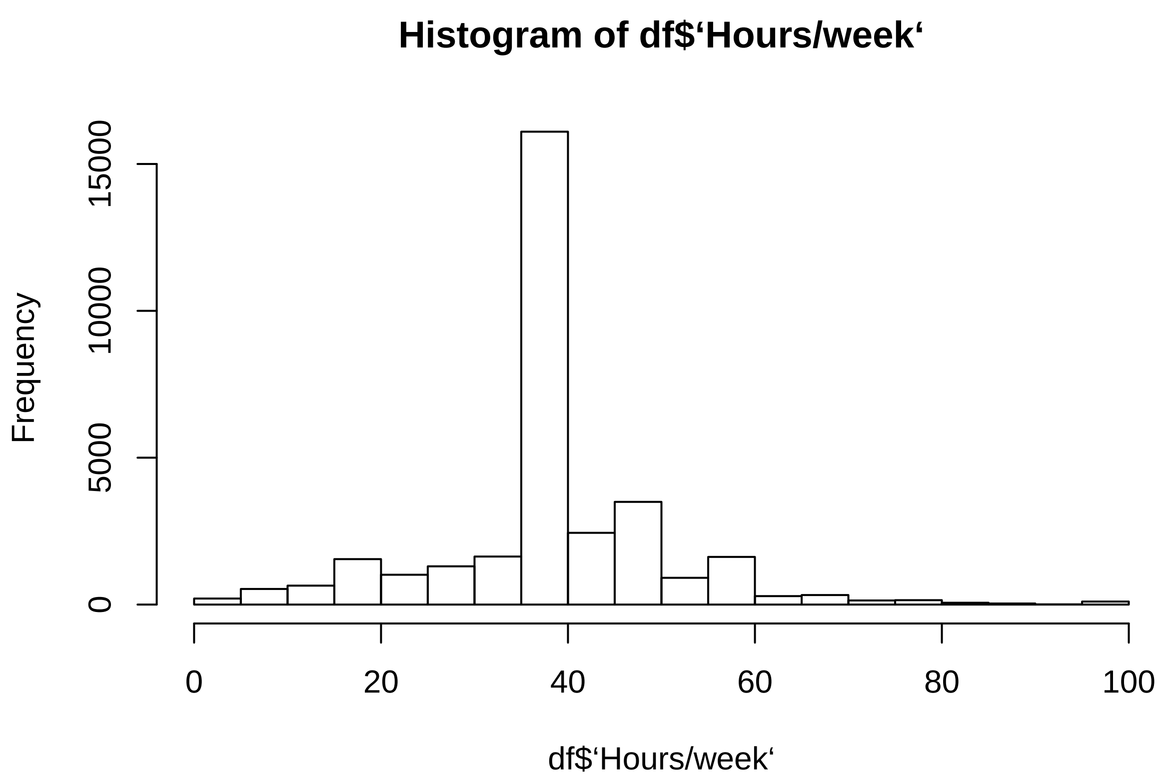

```r
hist(df$`Capital-gain`)
```




```r
hist(df$`Capital-loss`)
```




```r
hist(df$`Hours/week`)
```




```r
table(df$Workclass, df$Class)
```

```
##
##                       <=50K  >50K
```

```
##    ?                  1645   191
##    Federal-gov         589   371
##    Local-gov          1476   617
##    Never-worked          7     0
##    Private           17733  4963
##    Self-emp-inc        494   622
##    Self-emp-not-inc   1817   724
##    State-gov           945   353
##    Without-pay          14     0
```

```r
table(df$Education)
```

```
##
##          10th          11th          12th       1st-4th       5th-6th
##           933          1175           433           168           333
##       7th-8th           9th    Assoc-acdm     Assoc-voc     Bachelors
##           646           514          1067          1382          5355
##     Doctorate       HS-grad       Masters     Preschool   Prof-school
##           413         10501          1723            51           576
##  Some-college
##          7291
```

```r
table(df$`Marital-status`)
```

```
##
##              Divorced     Married-AF-spouse    Married-civ-spouse
##                  4443                    23                 14976
##  Married-spouse-absent         Never-married             Separated
##                   418                 10683                  1025
##               Widowed
##                   993
```

```r
table(df$Occupation)
```

```
##
##                  ?       Adm-clerical       Armed-Forces
##               1843               3770                  9
##       Craft-repair    Exec-managerial    Farming-fishing
##               4099               4066                994
##  Handlers-cleaners  Machine-op-inspct      Other-service
##               1370               2002               3295
##    Priv-house-serv     Prof-specialty    Protective-serv
##                149               4140                649
##              Sales       Tech-support   Transport-moving
##               3650                928               1597
```

```r
table(df$Relationship)
```

```
##
##         Husband   Not-in-family  Other-relative       Own-child
##           13193            8305             981            5068
##       Unmarried            Wife
##            3446            1568
```

```r
table(df$Race)
```

```
##
##  Amer-Indian-Eskimo  Asian-Pac-Islander               Black
```

```
##                  311                 1039                    3124
##                Other                White
##                  271                27816
```

```r
table(df$Sex)
```

```
##
##  Female    Male
##   10771   21790
```

```r
table(df$Native)
```

```
##
##                           ?                    Cambodia
##                         583                          19
##                      Canada                       China
##                         121                          75
##                    Columbia                        Cuba
##                          59                          95
##          Dominican-Republic                     Ecuador
##                          70                          28
##                 El-Salvador                     England
##                         106                          90
##                      France                     Germany
##                          29                         137
##                      Greece                   Guatemala
##                          29                          64
##                       Haiti          Holand-Netherlands
##                          44                           1
##                    Honduras                        Hong
##                          13                          20
##                     Hungary                       India
##                          13                         100
##                        Iran                     Ireland
##                          43                          24
##                       Italy                     Jamaica
##                          73                          81
##                       Japan                        Laos
##                          62                          18
##                      Mexico                   Nicaragua
##                         643                          34
##  Outlying-US(Guam-USVI-etc)                        Peru
##                          14                          31
##                 Philippines                      Poland
##                         198                          60
##                    Portugal                 Puerto-Rico
##                          37                         114
##                    Scotland                       South
##                          12                          80
##                      Taiwan                    Thailand
##                          51                          18
##             Trinadad&Tobago               United-States
##                          19                       29170
##                     Vietnam                  Yugoslavia
##                          67                          16
```

```r
table(df$Class)
```

```
##
##  <=50K   >50K
##  24720   7841
```

```r
# Many of the columns have "?" instead of values. They are missing values and will have to be imputed. 

# Making function of mode
mode <- function(x){
  uniq <- unique(x)
  uniq[which.max(tabulate(match(x,uniq)))]
}

# First convert ? to NA
df[df == " ?"] <- NA

# Now replace with mode
df$Workclass[which(is.na(df$Workclass))] <- mode(df$Workclass)
df$Occupation[which(is.na(df$Occupation))] <- mode(df$Occupation)
df$Native[which(is.na(df$Native))] <- mode(df$Native)
```

3. (10 pts) Create a frequency and then a likelihood table for the categorical features in the data set. Build your own Naive Bayes classifier for those features.

```r
# Creating frequency tables for all the features

WC <- table(df$Workclass, df$Class)
WC <- unclass(WC)
Ed <- unclass(table(df$Education, df$Class))
MS <- unclass(table(df$`Marital-status`, df$Class))
OC <- unclass(table(df$Occupation, df$Class))
Rel <- unclass(table(df$Relationship, df$Class))
Race <- unclass(table(df$Race, df$Class))
Sex <- unclass(table(df$Sex, df$Class))
NT <- unclass(table(df$Native,df$Class))

calc <- function(data, c1, c2){
  df <- data
  for(i in 1:nrow(df)){
    c1[i] <- data[i,2]/sum(data[i,1],data[i,2])
    c2[i] <- data[i,1]/sum(data[i,1],data[i,2])
  }
  df <- cbind(data,c1,c2)
  colnames(df)[3] <- ">50K_ll"
  colnames(df)[4] <- "<=50K_ll"
}
freq <- function(data){
  c1 <- nrow(data)
  c2 <- nrow(data)
  calc(data,c1,c2)
}

# Adding likelihood to the frequency tables
```

```r
WC_l <- freq(WC)
Ed_l <-freq(Ed)
MS_l <- freq(MS)
OC_l <- freq(OC)
Rel_l <- freq(Rel)
Race_l <- freq(Race)
Sex_l <- freq(Sex)
NT_l <- freq(NT)

# Transforming the data to display the levels of features as columns and Class as Rows
WC_l <- t(WC_l)
Ed_l <-t(Ed_l)
MS_l <- t(MS_l)
OC_l <- t(OC_l)
Rel_l <- t(Rel_l)
Race_l <- t(Race_l)
Sex_l <- t(Sex_l)
NT_l <- t(NT_l)

# Converting to dataframes
WC_l <- as.data.frame(WC_l)
Ed_l <- as.data.frame(Ed_l)
MS_l <- as.data.frame(MS_l)
OC_l <- as.data.frame(OC_l)
Rel_l <- as.data.frame(Rel_l)
Race_l <- as.data.frame(Race_l)
Sex_l <- as.data.frame(Sex_l)
NT_l <- as.data.frame(NT_l)

# Probability of class being >50 or <=50K
t50 <- as.data.frame(unclass(table(df$Class)))
pg50 <- t50[2,1]/sum(t50[1,1],t50[2,1])
pl50 <- t50[1,1]/sum(t50[1,1],t50[2,1])

# Build Naive Bayes Classifier for all the categorical features

naivebayes<-function(workclass,column,education,column1,occupation,column2,maritalstatus,
                     column3,relationship,column4,race,column5,sex,column6,nativecountry,column7)
  # the values in function are the likelihood tables of different features and the column which represe
{
  wg<-workclass[3,column]  # probability of workclass of the case having income >50k
  wl<-workclass[4,column]  # probability of workclass of the case having income <=50k

  eg<-education[3,column1] # probability of education of the case having income >50k
  el<-education[4,column1] # probability of education of the case having income <=50k

  og<-occupation[3,column2]  # probability of the occupation of the case having income >50k
  ol<-occupation[4,column2] # probability of the occupation of the case having income <=50k

  mg<-maritalstatus[3,column3]  # probability of the maritalstatus of the case having income >50k
  ml<-maritalstatus[4,column3] # probability of the maritalstatus of the case having income <=50k

  rg<-relationship[3,column4] # probability of the relationship of the case having income >50k
```

```
  rl<-relationship[4,column4] # probability of the relationship of the case having income <=50k

  rcg<-race[3,column5] # probability of the race of the case having income >50k
  rcl<-race[4,column5] # probability of the race of the case having income <=50k

  sg<-sex[3,column6] # probability of the sex of the case having income >50k
  sl<-sex[4,column6] # probability of the sex of the case having income <=50k

  ng<-nativecountry[3,column7] # probability of the native country of the case having income >50k
  nl<-nativecountry[4,column7] # probability of the native country of the case having income <=50k

  lik_g50<-c(wg,eg,og,mg,rg,rcg,sg,ng) # total likelihood of all features having income >50K
  lik_l50<-c(wl,el,ol,ml,rl,rcl,sl,nl) # total likelihood of all features having income <=50K

  p_more50<-prod(lik_g50) # Get the product of likelihood for all the features having income >50K
  p_less50<-prod(lik_l50) # Get the product of likelihood for all the features having income <=50K

  less_50<-(p_less50*pl50) # Multiply the product of likelihood for all the features having income >50K
  more_50<-(p_more50*pg50) # Multiply the product of likelihood for all the features having income <=50

  final_prob_l50<-less_50/(less_50+more_50) #final probability from conditional probability for the giv
  final_prob_l50

}
```

4. (30 pts)Predict the binomial class membership for a white female adult who is a federal government worker with a bachelors degree who immigrated from India. Ignore any other features in your model. You must build your own Naive Bayes Classifier – you may not use a package.

The class membership for a white female adult who is a federal government worker with a bachelors degree who immigrated from India is '<= 50K'

```
naivebayes1<-function(workclass,column1,education,column2,race,column3,sex,column4,nativecountry,column5

  wg<-workclass[3,column1]
  wl<-workclass[4,column1]

  eg<-education[3,column2]
  el<-education[4,column2]


  rcg<-race[3,column3]
  rcl<-race[4,column3]

  sg<-sex[3,column4]
  sl<-sex[4,column4]

  ng<-nativecountry[3,column5]
  nl<-nativecountry[4,column5]

  lik_g50<-c(wg,eg,rcg,sg,ng)
  lik_l50<-c(wl,el,rcl,sl,nl)

  p_more50<-prod(lik_g50)
  p_less50<-prod(lik_l50)
```

```r
  less_50<-(p_less50*pl50)
  more_50<-(p_more50*pg50)

  final_prob_l50<-less_50/(less_50+more_50)
  final_prob_l50
}

naivebayes1(WC_l,' Federal-gov',Ed_l,' Bachelors',Race_l,' White',Sex_l,' Female',
             NT_l,' India')
```

```
##     <=50K
## 0.7591904
```

```r
# Since the probaility of the unkown case having income <=50K is 0.759, we can classify it as having in
```

5. (20 pts) Perform 10-fold cross validation on your algorithm to tune it and report the final accuracy results. The final accuracy is 75%

```r
# This does not give the accuracy. This was my effort to do cross validation. I finally ended up using
# Probability of class being >50 or <=50K
t50 <- as.data.frame(unclass(table(df$Class)))
pg50 <- t50[2,1]/sum(t50[1,1],t50[2,1])
pl50 <- t50[1,1]/sum(t50[1,1],t50[2,1])

# To make 10-fold cross validations, we need 10 subsets of the data
set.seed(999)
index <- createFolds(df$Class, 10, list = T, returnTrain = F)
# Cross Validation
for(i in 1:10){
  train <- df[-index[[i]],]
  test <- df[index[[i]],]
  WC <- table(train$Workclass, train$Class)
  WC <- unclass(WC)
  Ed <- unclass(table(train$Education, train$Class))
  MS <- unclass(table(train$`Marital-status`, train$Class))
  OC <- unclass(table(train$Occupation, train$Class))
  Rel <- unclass(table(train$Relationship, train$Class))
  Race <- unclass(table(train$Race, train$Class))
  Sex <- unclass(table(train$Sex, train$Class))
  NT <- unclass(table(train$Native,train$Class))

calc <- function(data, c1, c2){
  train <- data
  for(i in 1:nrow(train)){
    c1[i] <- data[i,2]/sum(data[i,1],data[i,2])
    c2[i] <- data[i,1]/sum(data[i,1],data[i,2])
  }
  train <- cbind(data,c1,c2)
  colnames(train)[3] <- ">50K_ll"
  colnames(train)[4] <- "<=50K_ll"
}
freq <- function(data){
  c1 <- nrow(data)
  c2 <- nrow(data)
  calc(data,c1,c2)
```

```r
}

# Adding likelihood to the frequency tables
WC_l <- as.data.frame(t(freq(WC)))
Ed_l <-as.data.frame(t(freq(Ed)))
MS_l <- as.data.frame(t(freq(MS)))
OC_l <- as.data.frame(t(freq(OC)))
Rel_l <- as.data.frame(t(freq(Rel)))
Race_l <- as.data.frame(t(freq(Race)))
Sex_l <- as.data.frame(t(freq(Sex)))
NT_l <- as.data.frame(t(freq(NT)))


prob <- naivebayes1(WC_l,' Federal-gov',Ed_l,' Bachelors',Race_l,' White',Sex_l,' Female',
             NT_l,' India')
}
prob
```

```
##     <=50K
## 0.7591904
```

```r
# Since I couldn't figure out how to use test cases in my Naive Bayes function, I used Vaishnavi's code
```

```r
#Download the data set Census Income Data for Adults along with its explanation. Explore the data set a

# getting the data
data <- file('http://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data')
income_con <- read.table(data, fileEncoding="UTF-16", header = FALSE, sep = ',',
                         col.names = c('age', 'workclass', 'fnlwgt', 'education', 'education-num',
                                       'marital-status', 'occupation', 'relationship', 'race', 'sex',
                                       'capital-gain', 'capital-loss', 'hours-per-week',
                                       'native-country', 'income-level'), stringsAsFactors = FALSE)


# extracting the columns with categorical features
income_cat <- income_con[, c(2, 4, 6:10, 14:15)]

# transforming the income column as a factor feature
income_cat$income.level <- factor(income_cat$income.level)


# removing all the rows with missing values, represented as '?' rather than NA in data
fin_income <- income_cat[!(income_cat$workclass == ' ?' | income_cat$occupation == ' ?' |
                             income_cat$native.country == ' ?'), ]

# structure of the transformed dataset
str(fin_income)
```

```
## 'data.frame':    30162 obs. of  9 variables:
##  $ workclass     : chr  " State-gov" " Self-emp-not-inc" " Private" " Private" ...
##  $ education     : chr  " Bachelors" " Bachelors" " HS-grad" " 11th" ...
##  $ marital.status: chr  " Never-married" " Married-civ-spouse" " Divorced" " Married-civ-spouse" ...
##  $ occupation    : chr  " Adm-clerical" " Exec-managerial" " Handlers-cleaners" " Handlers-cleaners"
##  $ relationship  : chr  " Not-in-family" " Husband" " Not-in-family" " Husband" ...
##  $ race          : chr  " White" " White" " White" " Black" ...
```

```
##  $ sex           : chr  " Male" " Male" " Male" " Male" ...
##  $ native.country: chr  " United-States" " United-States" " United-States" " United-States" ...
##  $ income.level  : Factor w/ 2 levels " <=50K"," >50K": 1 1 1 1 1 1 1 2 2 2 ...
```

```r
#Create a frequency and then a likelihood table for the categorical features in the data set. Build you

freq_tbl <- sapply(fin_income[-9], table, fin_income$income.level)

lap_est <- sapply(freq_tbl, function(x) {
  apply(x, 1, function(x) {
    x + 1})})

# creating a likelihood table by dividing the count with the total sum of that column
lik_tbl <- sapply(lap_est, function(x) {
  apply(x, 1, function(x) {
    x / sum(x)})})

# taking a transform of the table to get in the naive bayes classifier form
lik_tbl <- lapply(lik_tbl, t)
head(lik_tbl)
```

```
## $workclass
##
##          Federal-gov  Local-gov   Private  Self-emp-inc  Self-emp-not-inc
##    <=50K  0.02555051 0.06438374 0.7683244    0.02096112        0.07881382
##    >50K   0.04870259 0.08117099 0.6489687    0.07997339        0.09514305
##
##          State-gov  Without-pay
##    <=50K 0.04130444 0.0006619302
##    >50K  0.04590818 0.0001330672
##
## $education
##
##                10th        11th        12th       1st-4th      5th-6th
##    <=50K 0.033612704 0.043670049 0.015394795 0.0064402294 0.012218791
##    >50K  0.007974482 0.007974482 0.003987241 0.0009303562 0.001727804
##
##             7th-8th         9th  Assoc-acdm  Assoc-voc  Bachelors
##    <=50K 0.023070137 0.019011910  0.03321570 0.04252316  0.1287605
##    >50K  0.004784689 0.003455609  0.03415736 0.04585327  0.2826954
##
##           Doctorate   HS-grad    Masters   Preschool  Prof-school
##    <=50K 0.004234671 0.3627702 0.03131892 0.002029113  0.006043229
##    >50K  0.037347156 0.2150452 0.12214248 0.000132908  0.054093567
##
##          Some-college
##    <=50K    0.2356859
##    >50K     0.1776980
##
## $marital.status
##
##           Divorced  Married-AF-spouse  Married-civ-spouse
##    <=50K 0.16605622       0.0005295442           0.3383346
##    >50K  0.06027944       0.0014637392           0.8516301
##
```

```
##           Married-spouse-absent  Never-married   Separated    Widowed
##    <=50K            0.01500375     0.40849918 0.038568466 0.03300825
##    >50K             0.00425815     0.06267465 0.008915502 0.01077844
##
## $occupation
##
##           Adm-clerical  Armed-Forces  Craft-repair  Exec-managerial
##    <=50K    0.14222693  0.0003970355     0.1377713       0.09070055
##    >50K     0.06633874  0.0002658867     0.1208455       0.25764424
##
##           Farming-fishing  Handlers-cleaners  Machine-op-inspct
##    <=50K       0.03860067         0.05593789         0.07596612
##    >50K        0.01542143         0.01116724         0.03270407
##
##           Other-service  Priv-house-serv  Prof-specialty  Protective-serv
##    <=50K     0.13591848     0.0063084524      0.09828834       0.01919005
##    >50K      0.01768147     0.0002658867      0.24089338       0.02805105
##
##              Sales  Tech-support  Transport-moving
##    <=50K 0.1153609    0.02801306        0.05532028
##    >50K  0.1290880    0.03709120        0.04254188
##
## $relationship
##
##            Husband  Not-in-family  Other-relative   Own-child  Unmarried
##    <=50K 0.2994263      0.3046778     0.037731686 0.194307149 0.13239188
##    >50K  0.7559223      0.1096620     0.004791057 0.008650519 0.02848017
##
##                Wife
##    <=50K 0.03146514
##    >50K  0.09249401
##
## $race
##
##           Amer-Indian-Eskimo  Asian-Pac-Islander      Black       Other
##    <=50K         0.011165541          0.02859791 0.10821307 0.009311973
##    >50K          0.004658592          0.03314255 0.04884866 0.002928258
##
##              White
##    <=50K 0.8427115
##    >50K  0.9104219
```

```r
# building a naive bayes classifier

# this classifier calculates the probabilites of a person's income being
# less than >50k
nb <- function(x) {

  # initializing all the required variables
  t1 <- 0
  t2 <- 0
  li.grt50 <- 0
  li.lss50 <- 0
  pr.lss50 <- 0
```

```r
  z1 <- list()
  z2 <- list()
  y <- list()


  for (j in 1:nrow(x)) {
    y[[j]] <- colnames(x[j, ] %>%  select_if(~ !any(is.na(.))))
  }


  for (n in 1:nrow(x)) {
    for (k in 1:length(y[[n]])) {
      t1[k] <- lik_tbl[[y[[n]][k]]][1, x[n, y[[n]][k]]]
    }
    z1[[n]] <- t1
  }

  # similarly, getting the likelihood values for income >50k, again by feeding the
  # column names
  for (n in 1:nrow(x)) {
    for (k in 1:length(y[[n]])) {
      t2[k] <- lik_tbl[[y[[n]][k]]][2, x[n, y[[n]][k]]]
    }
    z2[[n]] <- t2
  }


  for (m in 1:length(z1)) {
    li.lss50[m] <- prod(z1[[m]])
  }


  for (l in 1:length(z2)) {
    li.grt50[l] <- prod(z2[[l]])
  }


  for (q in 1:nrow(x)) {
    pr.lss50[q] <- li.lss50[q]/(li.grt50[q] + li.lss50[q])
  }
  return(pr.lss50)
}

#Predict the binomial class membership for a white female adult who is a federal government worker with

# the test case
test <- fin_income[0,-9]
test[1, ] <- c(' Federal-gov', ' Bachelors', NA, NA, NA, ' White', ' Female', ' India')

# prediciting the binomial class membership for the given case
nb(test)
```

```
## [1] 0.2203777
```

```r
fin_income2 <- fin_income
fin_income2$income.level <- if_else(fin_income$income.level == ' >50K', 0, 1)

# predicting the probability of people earning <=50k
#nb.pred <- nb(fin_income2[-9])

# person with probability >0.5 is determined to be earning >50k
#nb.pred_class <- ifelse(nb.pred > 0.50, 1, 0)

# checking the accuracy of algorithm
#confusionMatrix(nb.pred_class, fin_income2$income.level)


# cross validation for the predictions

# naive bayes classifier function for cross calidation

nb.cv <- function(x) {

  # initializing all the required variables
  t1 <- 0
  t2 <- 0
  li.grt50 <- 0
  li.lss50 <- 0
  pr.lss50 <- 0
  z1 <- list()
  z2 <- list()
  y <- list()


  for (j in 1:nrow(x)) {
    y[[j]] <- colnames(x[j, ] %>%  select_if(~ !any(is.na(.))))
  }

  # getting the likelihood values for the case of income <=50k for each row for the
  # value it has

  for (n in 1:nrow(x)) {
    for (k in 1:length(y[[n]])) {
      t1[k] <- lik_tbl_cv[[y[[n]][k]]][1, x[n, y[[n]][k]]]
    }
    z1[[n]] <- t1
  }

  # similarly, getting the likelihood values for income >50k, again by feeding the
  # column names
  for (n in 1:nrow(x)) {
    for (k in 1:length(y[[n]])) {
      t2[k] <- lik_tbl_cv[[y[[n]][k]]][2, x[n, y[[n]][k]]]
    }
    z2[[n]] <- t2
  }
```

```r
  # calculating the overall likelihood value by multiplying the individual likelihoods
  # when income <=50k
  for (m in 1:length(z1)) {
    li.lss50[m] <- prod(z1[[m]])
  }

  # calculating the overall likelihood value by multiplying the individual likelihoods
  # when income >50k
  for (l in 1:length(z2)) {
    li.grt50[l] <- prod(z2[[l]])
  }

  # transforming the likelihood into probability by dividing with total likelihood
  for (q in 1:nrow(x)) {
    pr.lss50[q] <- li.lss50[q]/(li.grt50[q] + li.lss50[q])
  }
  return(pr.lss50)
}

# initialize the accuracy vector
accuracy <- rep(0,6)

for (i in 1:6) {
  # indices indicate the interval of the test set
  indices <- (((i-1) * round((1/10)*nrow(fin_income2))) + 1):((i*round((1/10) * nrow(fin_income2))))

  # training set
  training <- fin_income[-indices,]

  # test set
  testing <- fin_income2[indices,]

  # building a frequency and a likelihood table from training set
  freq_tbl_cv <- sapply(training[-9], table, training$income.level)

  lap_est_cv <- sapply(freq_tbl_cv, function(x) {
    apply(x, 1, function(x) {
      x + 1})})

  lik_tbl_cv <- sapply(lap_est_cv, function(x) {
    apply(x, 1, function(x) {
      x / sum(x)})})

  lik_tbl_cv <- lapply(lik_tbl_cv, t)

  # make predictions on the test set using the nb.cv function that takes likelihood
  # values from training set
  nb.cv_pred <- nb.cv(testing[-9])

  nb.cv_pred_class <- ifelse(nb.cv_pred > 0.50, 1, 0)

  # generate the confusion matrix
  conf_mat <- table(testing$income.level, nb.cv_pred_class)
```

```r
  # assigning the accuracy of this model to the vector
  accuracy[i] <- sum(diag(conf_mat))/sum(conf_mat)
}

accuracy
```

```
## [1] 0.7529841 0.7443634 0.7536472 0.7549735 0.7430371 0.7500000
```

```r
# mean of accuracies
mean(accuracy)
```

```
## [1] 0.7498342
```

Problem 2 (25 Points)

```r
require(rlang)
```

```
## Loading required package: rlang
```

```r
library(readxl)
require(ggplot2)
require(car)
```

```
## Loading required package: car

## Loading required package: carData

##
## Attaching package: 'car'

## The following object is masked from 'package:dplyr':
##
##     recode
```

```r
#install.packages('corrplot')
require(corrplot)
```

```
## Loading required package: corrplot

## corrplot 0.84 loaded
```

```r
require(psych)
```

```
## Loading required package: psych

##
## Attaching package: 'psych'

## The following object is masked from 'package:car':
##
##     logit

## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```r
#install.packages('rms')
require(rms)
```

```
## Loading required package: rms

## Loading required package: Hmisc
```

```
## Loading required package: survival

##
## Attaching package: 'survival'

## The following object is masked from 'package:caret':
##
##     cluster

## Loading required package: Formula

##
## Attaching package: 'Hmisc'

## The following object is masked from 'package:psych':
##
##     describe

## The following objects are masked from 'package:dplyr':
##
##     src, summarize

## The following objects are masked from 'package:base':
##
##     format.pval, units

## Loading required package: SparseM

##
## Attaching package: 'SparseM'

## The following object is masked from 'package:base':
##
##     backsolve

##
## Attaching package: 'rms'

## The following objects are masked from 'package:car':
##
##     Predict, vif
```

```r
#install.packages('sqldf')
require(sqldf)
```

```
## Loading required package: sqldf

## Loading required package: gsubfn

## Loading required package: proto

## Loading required package: RSQLite
```

```r
require(reshape2)
```

```
## Loading required package: reshape2
```

```r
#install.packages('mice')
require(mice)
```

```
## Loading required package: mice

##
## Attaching package: 'mice'
```

```
## The following objects are masked from 'package:base':
##
##     cbind, rbind
```

```r
#install.packages('gmodels')
require(gmodels)
```

```
## Loading required package: gmodels
```

```r
require(e1071)
```

```
## Loading required package: e1071
```

```
##
## Attaching package: 'e1071'
```

```
## The following object is masked from 'package:Hmisc':
##
##     impute
```

```r
uff_raw <- as.data.frame(read_excel('uffidata.xlsx'))

names(uff_raw) <- gsub(x = names(uff_raw), pattern = " ", replacement = "_")
colnames(uff_raw)[6] <- "Yrs45"

uff_raw <- uff_raw[-c(1)]

# Analyze the data

str(uff_raw)
```

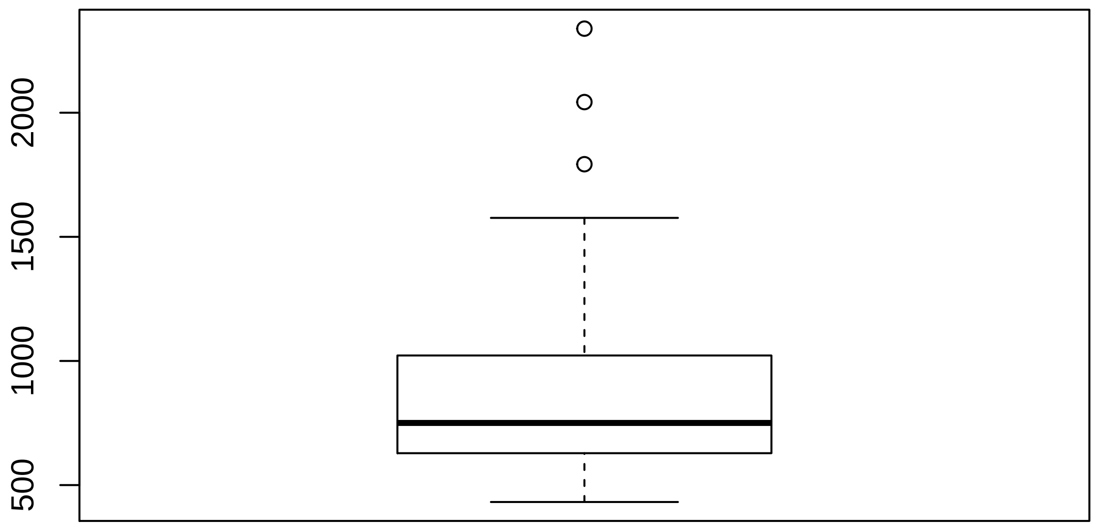
```
## 'data.frame':    99 obs. of  11 variables:
##  $ Year_Sold     : num  2009 2009 2011 2011 2010 ...
##  $ Sale_Price    : num  76900 78000 79000 80000 82000 84000 84000 84000 85000 85000 ...
##  $ UFFI_IN       : num  1 1 0 0 1 1 0 0 0 1 ...
##  $ Brick_Ext     : num  0 0 0 0 0 0 0 0 0 1 ...
##  $ Yrs45         : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ Bsmnt_Fin_SF  : num  0 154 400 0 157 ...
##  $ Lot_Area      : num  2772 4490 5840 5040 5441 ...
##  $ Enc_Pk_Spaces : num  0 0 0 0 0 1 2 0 0 1 ...
##  $ Living_Area_SF: num  1018 536 721 513 672 ...
##  $ Central_Air   : num  0 1 1 0 0 0 0 0 1 0 ...
##  $ Pool          : num  0 0 0 0 0 0 0 0 0 0 ...
```

```r
summary(uff_raw)
```

```
##    Year_Sold      Sale_Price        UFFI_IN         Brick_Ext     
##  Min.   :2009   Min.   : 76900   Min.   :0.0000   Min.   :0.0000  
##  1st Qu.:2011   1st Qu.:102000   1st Qu.:0.0000   1st Qu.:0.0000  
##  Median :2012   Median :115000   Median :0.0000   Median :0.0000  
##  Mean   :2013   Mean   :124450   Mean   :0.2323   Mean   :0.3939  
##  3rd Qu.:2015   3rd Qu.:135000   3rd Qu.:0.0000   3rd Qu.:1.0000  
##  Max.   :2016   Max.   :347000   Max.   :1.0000   Max.   :1.0000  
##      Yrs45         Bsmnt_Fin_SF      Lot_Area     Enc_Pk_Spaces   
##  Min.   :0.0000   Min.   :  0.0   Min.   : 1800   Min.   :0.0000  
##  1st Qu.:1.0000   1st Qu.:  0.0   1st Qu.: 4376   1st Qu.:0.0000  
##  Median :1.0000   Median :248.8   Median : 5205   Median :1.0000  
##  Mean   :0.8182   Mean   :248.0   Mean   : 5709   Mean   :0.8081  
```

```
##  3rd Qu.:1.0000   3rd Qu.:387.2   3rd Qu.: 6509   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :915.1   Max.   :11650   Max.   :2.0000
##  Living_Area_SF    Central_Air          Pool
##  Min.   : 431.9   Min.   :0.0000   Min.   :0.0000
##  1st Qu.: 628.5   1st Qu.:0.0000   1st Qu.:0.0000
##  Median : 750.3   Median :1.0000   Median :0.0000
##  Mean   : 858.4   Mean   :0.5758   Mean   :0.0303
##  3rd Qu.:1022.1   3rd Qu.:1.0000   3rd Qu.:0.0000
##  Max.   :2338.7   Max.   :1.0000   Max.   :1.0000
```

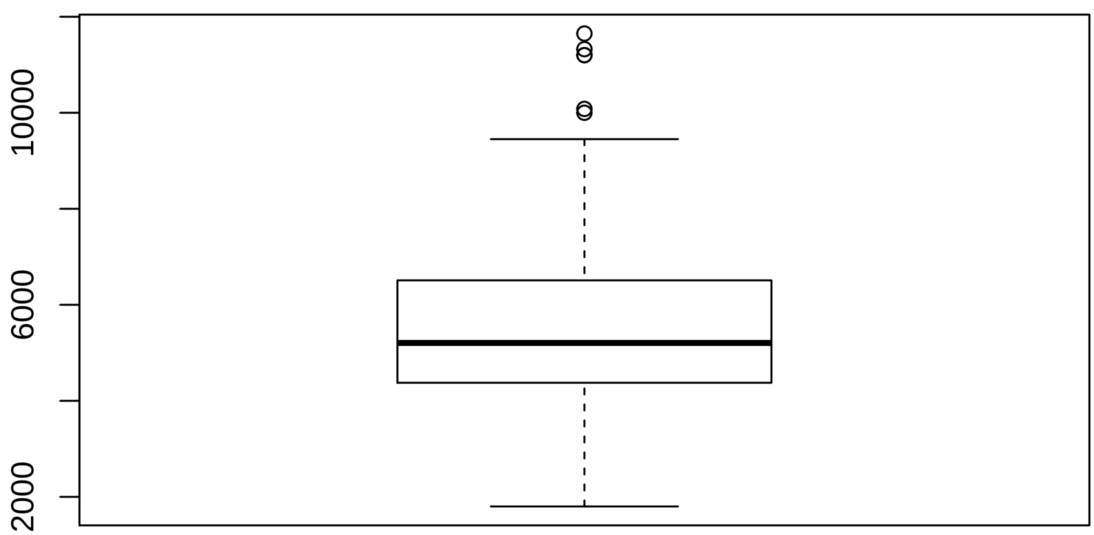
```
OutVals_Liv = boxplot(uff_raw$Living_Area_SF)$out
```




```
which(uff_raw$Living_Area_SF %in% OutVals_Liv)
```

```
## [1] 95 98 99
```

```
OutVals_Lot = boxplot(uff_raw$Lot_Area)$out
```



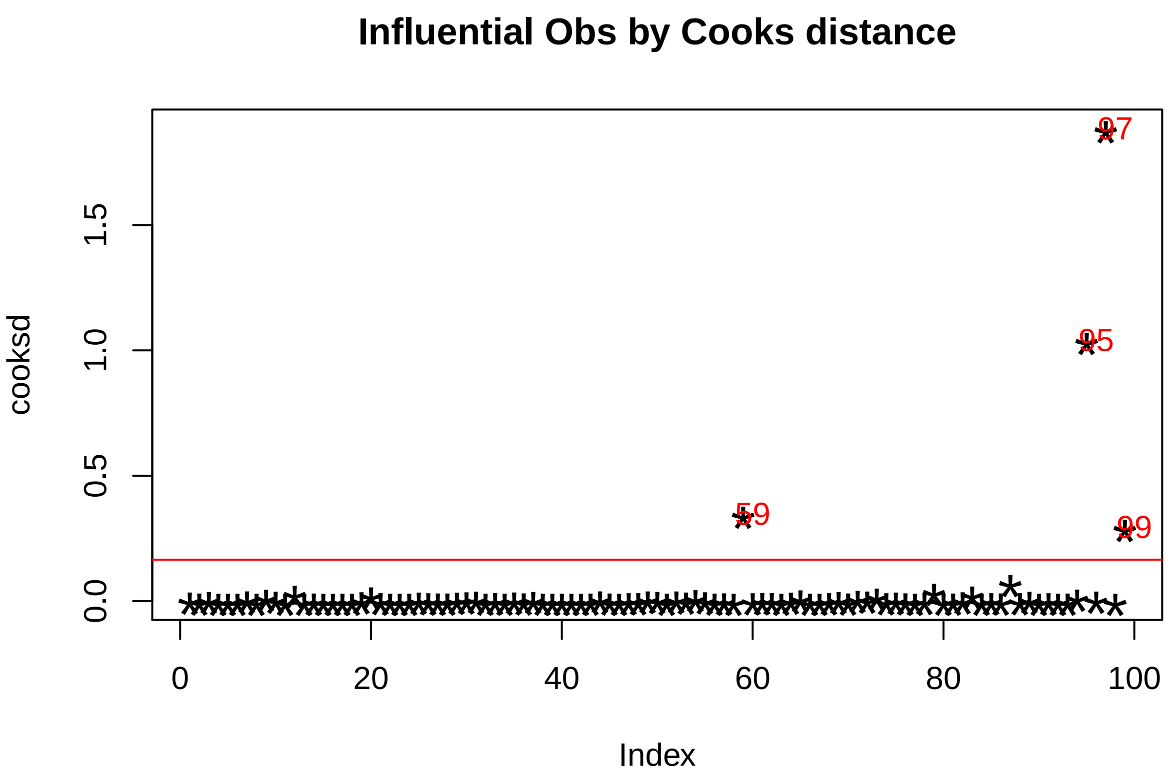

```
which(uff_raw$Lot_Area %in% OutVals_Lot)
```

```
## [1] 52 77 84 93 98 99
```

After reading the case study background information, using the UFFI data set, answer these questions: 1.

(5 pts) Are there outliers in the data set? How do you identify outliers and how do you deal with them? Remove them but create a second data set with outliers removed. Keep the original data set.

```
#As there are multiple feature and imapcting the dependent variable, we shall use the Cook's distance t
#Reference: http://r-statistics.co/Outlier-Treatment-With-R.html

Linear_Model_outlier <- lm(Sale_Price ~ ., data=uff_raw)

cooksd <- cooks.distance(Linear_Model_outlier)

plot(cooksd, pch="*", cex=2, main="Influential Obs by Cooks distance")
abline(h = 4*mean(cooksd, na.rm=T), col="red")
text(x=1:length(cooksd)+1, y=cooksd, labels=ifelse(cooksd>4*mean(cooksd, na.rm=T),names(cooksd),""), co
```




```
influential_outliers <- as.numeric(names(cooksd)[(cooksd > 4*mean(cooksd, na.rm=T))])
head(uff_raw[influential_outliers, ])
```

```
##    Year_Sold Sale_Price UFFI_IN Brick_Ext Yrs45 Bsmnt_Fin_SF Lot_Area
## 59      2010     121500       0         0     0       516.44     6500
## 95      2016     200000       0         0     1       260.40     5544
## 97      2016     347000       0         1     0       441.50     8190
## 99      2010     250000       1         1     1         0.00    11200
##    Enc_Pk_Spaces Living_Area_SF Central_Air Pool
## 59             0        581.930           1    1
## 95             2       1792.630           0    1
## 97             2       1279.690           1    1
## 99             2       2042.947           1    0
```

```
car::outlierTest(Linear_Model_outlier)
```

```
##     rstudent unadjusted p-value Bonferonni p
## 97  7.504580         4.9708e-11   4.9211e-09
```

```
## 95 -4.241616         5.5185e-05   5.4633e-03
```

```r
#From the observations from the box plot, linear model and the outlier test we will eliminate obervatio
uff_wo_outliers <- uff_raw[-c(95,97,99),]
```

2. (2 pts) What are the correlations to the response variable and are there colinearities? Build a full correlation matrix.

```r
## 2.Colinearility

correlation <- cor(uff_wo_outliers)
correlation
```

```
##                  Year_Sold   Sale_Price      UFFI_IN    Brick_Ext
## Year_Sold       1.00000000  0.680124035 -0.202158774  0.212793751
## Sale_Price      0.68012403  1.000000000 -0.206732966  0.156368654
## UFFI_IN        -0.20215877 -0.206732966  1.000000000 -0.024400783
## Brick_Ext       0.21279375  0.156368654 -0.024400783  1.000000000
## Yrs45          -0.12564874 -0.148695245  0.058161185 -0.193313916
## Bsmnt_Fin_SF    0.08392324  0.131863976 -0.039754599 -0.083720690
## Lot_Area        0.30503947  0.417820600  0.126972876 -0.045619814
## Enc_Pk_Spaces   0.25344786  0.424429425 -0.146655065 -0.080654605
## Living_Area_SF  0.38191229  0.740844800  0.002296263  0.121199401
## Central_Air     0.06451258  0.226899880 -0.030271052 -0.008563543
## Pool           -0.11376804  0.004998989 -0.055941445 -0.081248070
##                      Yrs45 Bsmnt_Fin_SF    Lot_Area Enc_Pk_Spaces
## Year_Sold      -0.125648742  0.083923242  0.30503947   0.253447860
## Sale_Price     -0.148695245  0.131863976  0.41782060   0.424429425
## UFFI_IN         0.058161185 -0.039754599  0.12697288  -0.146655065
## Brick_Ext      -0.193313916 -0.083720690 -0.04561981  -0.080654605
## Yrs45           1.000000000 -0.461996984 -0.34571978   0.004249156
## Bsmnt_Fin_SF   -0.461996984  1.000000000  0.25180102  -0.006314933
## Lot_Area       -0.345719776  0.251801015  1.00000000   0.219400345
## Enc_Pk_Spaces   0.004249156 -0.006314933  0.21940035   1.000000000
## Living_Area_SF  0.006167127 -0.058758108  0.33679970   0.303749439
## Central_Air    -0.124694832  0.304762289  0.26932099   0.113460188
## Pool           -0.221170542  0.114792966  0.04536803  -0.118225345
##                Living_Area_SF  Central_Air         Pool
## Year_Sold         0.381912294  0.064512578 -0.113768040
## Sale_Price        0.740844800  0.226899880  0.004998989
## UFFI_IN           0.002296263 -0.030271052 -0.055941445
## Brick_Ext         0.121199401 -0.008563543 -0.081248070
## Yrs45             0.006167127 -0.124694832 -0.221170542
## Bsmnt_Fin_SF     -0.058758108  0.304762289  0.114792966
## Lot_Area          0.336799702  0.269320986  0.045368025
## Enc_Pk_Spaces     0.303749439  0.113460188 -0.118225345
## Living_Area_SF    1.000000000  0.176247299 -0.085221981
## Central_Air       0.176247299  1.000000000  0.088582673
## Pool             -0.085221981  0.088582673  1.000000000
```

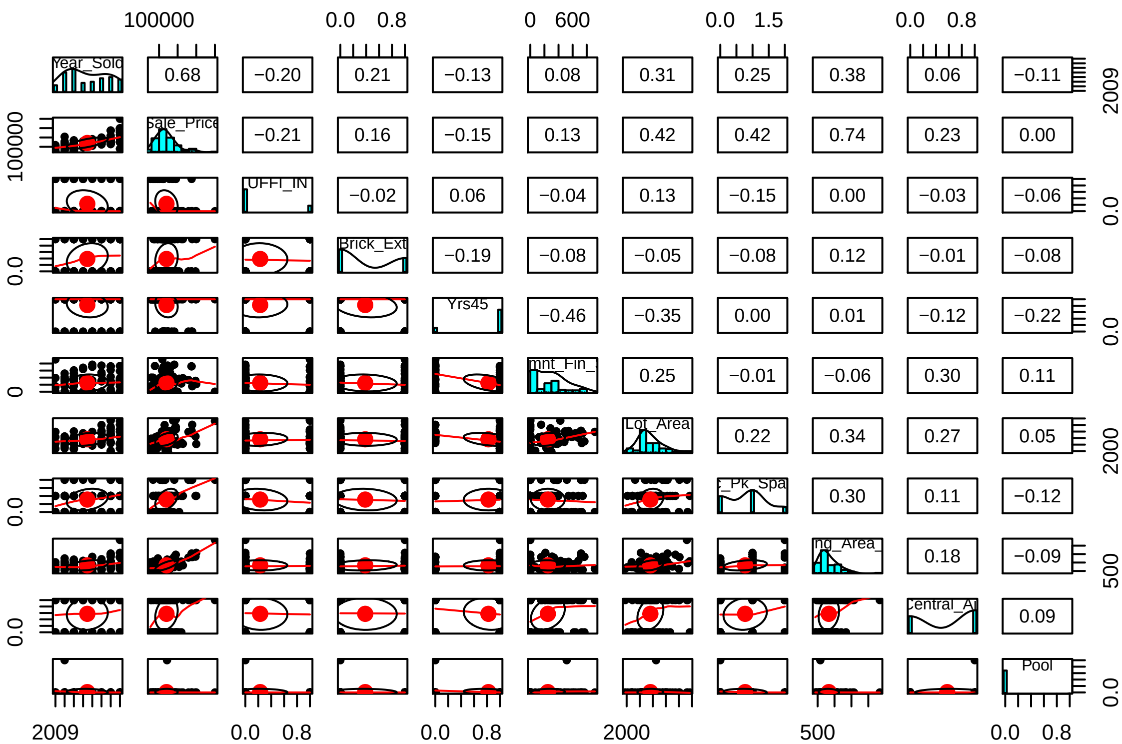
```r
correlation <- data.frame(as.list(correlation[,2]))

correlation <- melt(correlation)
```

```
## No id variables; using all as measure variables
```

```r
correlation$absvalue <- abs(correlation$value)
sqldf("select * from correlation order by absvalue desc")
```

```
##          variable        value    absvalue
## 1      Sale_Price  1.000000000 1.000000000
## 2  Living_Area_SF  0.740844800 0.740844800
## 3       Year_Sold  0.680124035 0.680124035
## 4   Enc_Pk_Spaces  0.424429425 0.424429425
## 5        Lot_Area  0.417820600 0.417820600
## 6     Central_Air  0.226899880 0.226899880
## 7         UFFI_IN -0.206732966 0.206732966
## 8       Brick_Ext  0.156368654 0.156368654
## 9           Yrs45 -0.148695245 0.148695245
## 10   Bsmnt_Fin_SF  0.131863976 0.131863976
## 11           Pool  0.004998989 0.004998989
```

```r
pairs.panels(uff_wo_outliers)
```



3. (10 pts) What is the ideal multiple regression model for predicting home prices in this data set using the data set with outliers removed? Provide a detailed analysis of the model, including Adjusted R-Squared, RMSE, and p-values of principal components. Use backward elimination by p-value to build the model.

```r
## 3.Model

Linear_Model_reg <- lm(Sale_Price ~ ., data=uff_wo_outliers)
summary(Linear_Model_reg)
```

```
##
## Call:
## lm(formula = Sale_Price ~ ., data = uff_wo_outliers)
##
## Residuals:
```

```
##    Min     1Q Median     3Q    Max
## -37226  -8085    340   9009  57418
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -1.066e+07  1.591e+06  -6.703 2.11e-09 ***
## Year_Sold       5.329e+03  7.918e+02   6.730 1.87e-09 ***
## UFFI_IN        -7.497e+03  3.751e+03  -1.999  0.04882 *
## Brick_Ext       2.254e+03  3.341e+03   0.675  0.50179
## Yrs45           1.959e+01  4.896e+03   0.004  0.99682
## Bsmnt_Fin_SF    1.148e+01  7.467e+00   1.537  0.12796
## Lot_Area        1.018e+00  9.348e-01   1.089  0.27902
## Enc_Pk_Spaces   6.514e+03  2.428e+03   2.683  0.00876 **
## Living_Area_SF  5.257e+01  5.777e+00   9.100 3.39e-14 ***
## Central_Air     2.143e+03  3.272e+03   0.655  0.51438
## Pool            2.730e+04  1.532e+04   1.782  0.07836 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 14440 on 85 degrees of freedom
## Multiple R-squared:  0.7953, Adjusted R-squared:  0.7712
## F-statistic: 33.02 on 10 and 85 DF,  p-value: < 2.2e-16
```

```
#Removing Yrs45
Linear_Model_reg_1 <- lm(Sale_Price ~ Year_Sold + UFFI_IN + Brick_Ext + Bsmnt_Fin_SF + Lot_Area + Enc_Pk
summary(Linear_Model_reg_1)
```

```
##
## Call:
## lm(formula = Sale_Price ~ Year_Sold + UFFI_IN + Brick_Ext + Bsmnt_Fin_SF +
##     Lot_Area + Enc_Pk_Spaces + Living_Area_SF + Central_Air +
##     Pool, data = uff_wo_outliers)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -37219  -8087    338   9010  57421
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -1.066e+07  1.581e+06  -6.743 1.69e-09 ***
## Year_Sold       5.329e+03  7.871e+02   6.771 1.49e-09 ***
## UFFI_IN        -7.496e+03  3.715e+03  -2.017  0.04677 *
## Brick_Ext       2.250e+03  3.161e+03   0.712  0.47857
## Bsmnt_Fin_SF    1.147e+01  6.729e+00   1.704  0.09201 .
## Lot_Area        1.017e+00  8.814e-01   1.154  0.25165
## Enc_Pk_Spaces   6.514e+03  2.414e+03   2.699  0.00837 **
## Living_Area_SF  5.257e+01  5.720e+00   9.191 2.01e-14 ***
## Central_Air     2.144e+03  3.239e+03   0.662  0.50974
## Pool            2.729e+04  1.489e+04   1.833  0.07026 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 14360 on 86 degrees of freedom
## Multiple R-squared:  0.7953, Adjusted R-squared:  0.7738
## F-statistic: 37.12 on 9 and 86 DF,  p-value: < 2.2e-16
```

```r
#Removing Central_Air
Linear_Model_reg_2 <- lm(Sale_Price ~ Year_Sold + UFFI_IN + Brick_Ext + Bsmnt_Fin_SF + Lot_Area + Enc_Pk
summary(Linear_Model_reg_2)
```

```
##
## Call:
## lm(formula = Sale_Price ~ Year_Sold + UFFI_IN + Brick_Ext + Bsmnt_Fin_SF +
##     Lot_Area + Enc_Pk_Spaces + Living_Area_SF + Pool, data = uff_wo_outliers)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -38574  -8228    165   8323  56184
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -1.057e+07  1.570e+06  -6.733 1.70e-09 ***
## Year_Sold       5.285e+03  7.816e+02   6.761 1.50e-09 ***
## UFFI_IN        -7.616e+03  3.699e+03  -2.059  0.04250 *
## Brick_Ext       2.318e+03  3.149e+03   0.736  0.46367
## Bsmnt_Fin_SF    1.268e+01  6.455e+00   1.964  0.05274 .
## Lot_Area        1.108e+00  8.678e-01   1.277  0.20507
## Enc_Pk_Spaces   6.605e+03  2.402e+03   2.750  0.00725 **
## Living_Area_SF  5.312e+01  5.642e+00   9.415 6.34e-15 ***
## Pool            2.790e+04  1.481e+04   1.884  0.06290 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 14310 on 87 degrees of freedom
## Multiple R-squared:  0.7942, Adjusted R-squared:  0.7753
## F-statistic: 41.97 on 8 and 87 DF,  p-value: < 2.2e-16
```

```r
#Removing Brick_Ext
Linear_Model_reg_3 <- lm(Sale_Price ~ Year_Sold + UFFI_IN + Bsmnt_Fin_SF + Lot_Area + Enc_Pk_Spaces + Li
summary(Linear_Model_reg_3)
```

```
##
## Call:
## lm(formula = Sale_Price ~ Year_Sold + UFFI_IN + Bsmnt_Fin_SF +
##     Lot_Area + Enc_Pk_Spaces + Living_Area_SF + Pool, data = uff_wo_outliers)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -39949  -8070     75   7681  55390
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -1.083e+07  1.529e+06  -7.082 3.33e-10 ***
## Year_Sold       5.410e+03  7.607e+02   7.112 2.89e-10 ***
## UFFI_IN        -7.589e+03  3.689e+03  -2.057  0.04264 *
## Bsmnt_Fin_SF    1.236e+01  6.423e+00   1.924  0.05760 .
## Lot_Area        1.049e+00  8.619e-01   1.218  0.22667
## Enc_Pk_Spaces   6.341e+03  2.369e+03   2.677  0.00886 **
## Living_Area_SF  5.350e+01  5.604e+00   9.546 3.08e-15 ***
## Pool            2.735e+04  1.475e+04   1.854  0.06713 .
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 14280 on 88 degrees of freedom
## Multiple R-squared:  0.7929, Adjusted R-squared:  0.7765
## F-statistic: 48.14 on 7 and 88 DF,  p-value: < 2.2e-16
```

```r
#Removing Lot_Area
Linear_Model_reg_4 <- lm(Sale_Price ~ Year_Sold + UFFI_IN + Bsmnt_Fin_SF + Enc_Pk_Spaces + Living_Area_S
summary(Linear_Model_reg_4)
```

```
##
## Call:
## lm(formula = Sale_Price ~ Year_Sold + UFFI_IN + Bsmnt_Fin_SF +
##     Enc_Pk_Spaces + Living_Area_SF + Pool, data = uff_wo_outliers)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -41956  -9421   1138   8192  54924
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -1.122e+07  1.498e+06  -7.488 4.83e-11 ***
## Year_Sold       5.607e+03  7.453e+02   7.523 4.09e-11 ***
## UFFI_IN        -6.585e+03  3.606e+03  -1.826  0.07115 .
## Bsmnt_Fin_SF    1.449e+01  6.197e+00   2.339  0.02160 *
## Enc_Pk_Spaces   6.762e+03  2.350e+03   2.878  0.00501 **
## Living_Area_SF  5.512e+01  5.457e+00  10.101  < 2e-16 ***
## Pool            2.915e+04  1.472e+04   1.981  0.05068 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 14310 on 89 degrees of freedom
## Multiple R-squared:  0.7894, Adjusted R-squared:  0.7753
## F-statistic: 55.62 on 6 and 89 DF,  p-value: < 2.2e-16
```

```r
Linear_Model_reg_final <- lm(formula=Sale_Price ~ Year_Sold + UFFI_IN + Bsmnt_Fin_SF + Enc_Pk_Spaces + L
summary(Linear_Model_reg_final)
```

```
##
## Call:
## lm(formula = Sale_Price ~ Year_Sold + UFFI_IN + Bsmnt_Fin_SF +
##     Enc_Pk_Spaces + Living_Area_SF + Pool, data = uff_wo_outliers)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -41956  -9421   1138   8192  54924
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -1.122e+07  1.498e+06  -7.488 4.83e-11 ***
## Year_Sold       5.607e+03  7.453e+02   7.523 4.09e-11 ***
## UFFI_IN        -6.585e+03  3.606e+03  -1.826  0.07115 .
## Bsmnt_Fin_SF    1.449e+01  6.197e+00   2.339  0.02160 *
## Enc_Pk_Spaces   6.762e+03  2.350e+03   2.878  0.00501 **
## Living_Area_SF  5.512e+01  5.457e+00  10.101  < 2e-16 ***
```

```
## Pool            2.915e+04  1.472e+04   1.981  0.05068 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 14310 on 89 degrees of freedom
## Multiple R-squared:  0.7894, Adjusted R-squared:  0.7753
## F-statistic: 55.62 on 6 and 89 DF,  p-value: < 2.2e-16
```

```r
RMSE <- sqrt(mean((Linear_Model_reg_final$residuals)^2))
RMSE
```

```
## [1] 13782.9
```

4. (3 pts) On average, by how much do we expect UFFI to change the value of a property?

```r
# 4.Due to the  significance of UFFI index, for every unit of UFFI index, the sales price will be affec
```

5. (5 pts) If the home in question is older than 45 years old, doesn't have a finished basement, has a lot area of 4000 square feet, has a brick exterior, 1 enclosed parking space, 1480 square feet of living space, central air, and no pool, what is its predicted value and what are the 95% confidence intervals of this home with UFFI and without UFFI?

With UFFI, 95% CI is 148623.9 - 204737.3 Without UFFI, 95% CI is 155208.9 - 211322.3

```r
# 5.
#Yrs45 = 1
#Bsmnt_Fin_SF = 0
#Lot_Area = 4000
#Brick_Ext = 1
#Enc_Pk_Spaces = 1
#Living_Area_SF = 1480
#Central_Air = 1
#Pool = 0

#Considering Year_Sold as 2018

#Equation form with UFFI
withuffi <- -1.122e+07 + 5.607e+03*(2018) + -6.585e+03*(1) + 1.449e+01*(0) + 6.762e+03*(1) + 5.512e+01*
withuffi
```

```
## [1] 176680.6
```

```r
#Upper Bound
withuffi + 1.96*(sqrt(deviance(Linear_Model_reg_final)/df.residual(Linear_Model_reg_final)))
```

```
## [1] 204737.3
```

```r
#Lower Bound
withuffi - 1.96*(sqrt(deviance(Linear_Model_reg_final)/df.residual(Linear_Model_reg_final)))
```

```
## [1] 148623.9
```

```r
#Equation form with UFFI
woithuffi <- -1.122e+07 + 5.607e+03*(2018) + -6.585e+03*(0) + 1.449e+01*(0) + 6.762e+03*(1) + 5.512e+01
woithuffi
```

```
## [1] 183265.6
```

```r
#Upper Bound
woithuffi + 1.96*(sqrt(deviance(Linear_Model_reg_final)/df.residual(Linear_Model_reg_final)))
```

```
## [1] 211322.3
```

```r
#Lower Bound
woithuffi - 1.96*(sqrt(deviance(Linear_Model_reg_final)/df.residual(Linear_Model_reg_final)))
```

```
## [1] 155208.9
```

Problem 3 (35 Points)

1. (5 pts) Divide the provided Titanic Survival Data into two subsets: a training data set and a test data set. Use whatever strategy you believe it best. Justify your answer.

```r
## Load CSV Files ##

titanic_raw <- read.csv('titanic_data.csv',header = TRUE)

#titanic_raw <- titanic_raw[-c(1)]



str(titanic_raw)
```

```
## 'data.frame':    891 obs. of  12 variables:
##  $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Survived   : int  0 1 1 1 0 0 0 0 1 1 ...
##  $ Pclass     : int  3 1 3 1 3 3 1 3 3 2 ...
##  $ Name       : Factor w/ 891 levels "Abbing, Mr. Anthony",..: 109 191 358 277 16 559 520 629 416 58
##  $ Sex        : Factor w/ 2 levels "female","male": 2 1 1 1 2 2 2 2 1 1 ...
##  $ Age        : num  22 38 26 35 35 NA 54 2 27 14 ...
##  $ SibSp      : int  1 1 0 1 0 0 0 3 0 1 ...
##  $ Parch      : int  0 0 0 0 0 0 0 1 2 0 ...
##  $ Ticket     : Factor w/ 681 levels "110152","110413",..: 525 596 662 50 473 276 86 396 345 133 ...
##  $ Fare       : num  7.25 71.28 7.92 53.1 8.05 ...
##  $ Cabin      : Factor w/ 148 levels "","A10","A14",..: 1 83 1 57 1 1 131 1 1 1 ...
##  $ Embarked   : Factor w/ 4 levels "","C","Q","S": 4 2 4 4 4 3 4 4 4 2 ...
```

```r
summary(titanic_raw)
```

```
##   PassengerId       Survived          Pclass     
##  Min.   :  1.0   Min.   :0.0000   Min.   :1.000  
##  1st Qu.:223.5   1st Qu.:0.0000   1st Qu.:2.000  
##  Median :446.0   Median :0.0000   Median :3.000  
##  Mean   :446.0   Mean   :0.3838   Mean   :2.309  
##  3rd Qu.:668.5   3rd Qu.:1.0000   3rd Qu.:3.000  
##  Max.   :891.0   Max.   :1.0000   Max.   :3.000  
##                                                  
##                                     Name         Sex           Age       
##  Abbing, Mr. Anthony                  :  1   female:314   Min.   : 0.42  
##  Abbott, Mr. Rossmore Edward          :  1   male  :577   1st Qu.:20.12  
##  Abbott, Mrs. Stanton (Rosa Hunt)     :  1                Median :28.00  
##  Abelson, Mr. Samuel                  :  1                Mean   :29.70  
##  Abelson, Mrs. Samuel (Hannah Wizosky):  1                3rd Qu.:38.00  
##  Adahl, Mr. Mauritz Nils Martin       :  1                Max.   :80.00  
##  (Other)                              :885                NA's   :177    
##      SibSp           Parch             Ticket         Fare       
##  Min.   :0.000   Min.   :0.0000   1601    :  7   Min.   :  0.00  
##  1st Qu.:0.000   1st Qu.:0.0000   347082  :  7   1st Qu.:  7.91  
##  Median :0.000   Median :0.0000   CA. 2343:  7   Median : 14.45  
```

```
##  Mean   :0.523   Mean   :0.3816   3101295 :  6   Mean   : 32.20
##  3rd Qu.:1.000   3rd Qu.:0.0000   347088  :  6   3rd Qu.: 31.00
##  Max.   :8.000   Max.   :6.0000   CA 2144 :  6   Max.   :512.33
##                                   (Other) :852
##          Cabin     Embarked
##             :687    :  2
##  B96 B98    :  4   C:168
##  C23 C25 C27:  4   Q: 77
##  G6         :  4   S:644
##  C22 C26    :  3
##  D          :  3
##  (Other)    :186
```

```
# 1.Data Split
#Splitting with createDataPartition to have consistent partition of data. 80%-20% is taken so that we h

titanic_train <- createDataPartition(y=titanic_raw$Survived ,p=0.8, list=F)
titanic_train_data <- titanic_raw[titanic_train,]
titanic_test_data <- titanic_raw[-titanic_train,]
```

2. (10 pts) Impute any missing values for the age variable using an imputation strategy of your choice. State why you chose that strategy and what others could have been used and why you didn't choose them.

```
# 2.Imputation using mice
#Package: mice -> Predictive Mean Matching method
# THis is used because we need to take into consideration other factors in the data set as well.
titanic_imp <- titanic_raw
titanic_imp_subset <- subset(titanic_imp, select=c(Age,Sex,Fare,Parch))

imputed_Data <- mice(titanic_imp_subset, m=1,seed = 500, method = "pmm",maxit = 50)
```

```
##
##  iter imp variable
##   1   1  Age
##   2   1  Age
##   3   1  Age
##   4   1  Age
##   5   1  Age
##   6   1  Age
##   7   1  Age
##   8   1  Age
##   9   1  Age
##   10   1  Age
##   11   1  Age
##   12   1  Age
##   13   1  Age
##   14   1  Age
##   15   1  Age
##   16   1  Age
##   17   1  Age
##   18   1  Age
##   19   1  Age
##   20   1  Age
##   21   1  Age
##   22   1  Age
```

```
## 23 1 Age
## 24 1 Age
## 25 1 Age
## 26 1 Age
## 27 1 Age
## 28 1 Age
## 29 1 Age
## 30 1 Age
## 31 1 Age
## 32 1 Age
## 33 1 Age
## 34 1 Age
## 35 1 Age
## 36 1 Age
## 37 1 Age
## 38 1 Age
## 39 1 Age
## 40 1 Age
## 41 1 Age
## 42 1 Age
## 43 1 Age
## 44 1 Age
## 45 1 Age
## 46 1 Age
## 47 1 Age
## 48 1 Age
## 49 1 Age
## 50 1 Age
```

```
summary(imputed_Data)
```

```
## Class: mids
## Number of multiple imputations:  1
## Imputation methods:
##   Age   Sex  Fare Parch
## "pmm"    ""    ""    ""
## PredictorMatrix:
##       Age Sex Fare Parch
## Age     0   1    1     1
## Sex     1   0    1     1
## Fare    1   1    0     1
## Parch   1   1    1     0
```

```
titanic_imp_sorted <- sqldf("select * from titanic_imp order by Age")
titanic_imp_sorted_NA <- titanic_imp_sorted[1:177,]
titanic_imp_sorted_notNA <- titanic_imp_sorted[178:891,]


titanic_imp_sorted_NA$Age <- imputed_Data$imp$Age[[1]]

titanic_imp <- rbind(titanic_imp_sorted_NA,titanic_imp_sorted_notNA)

titanic_imp <- sqldf("select * from titanic_imp order by PassengerId")

titanic_imp$Embarked[62] <- "S"
```

```r
titanic_imp$Embarked[830] <- "S"
```

3. (10 pts) Construct a logistic regression model to predict the probability of a passenger surviving the Titanic accident. Test the statistical significance of all parameters and eliminate those that have a p-value > 0.05 using stepwise backward elimination.

```r
# 3. Model Formulation

titanic_train_data <- titanic_train_data[-c(4,9,11)]
titanic_test_data <- titanic_test_data[-c(4,9,11)]
titanic_train_data <- na.omit(titanic_train_data)
titanic_test_data <- na.omit(titanic_test_data)


Linear_Model_reg_titanic <- glm(titanic_train_data$Survived~.,data=titanic_train_data,family = binomial
summary(Linear_Model_reg_titanic)
```

```
##
## Call:
## glm(formula = titanic_train_data$Survived ~ ., family = binomial,
##     data = titanic_train_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6517  -0.6812  -0.3988   0.6648   2.4050
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  1.680e+01  5.354e+02   0.031  0.97497
## PassengerId  3.806e-04  4.143e-04   0.919  0.35823
## Pclass      -1.137e+00  1.786e-01  -6.365 1.96e-10 ***
## Sexmale     -2.553e+00  2.436e-01 -10.478  < 2e-16 ***
## Age         -4.069e-02  9.169e-03  -4.438 9.08e-06 ***
## SibSp       -3.729e-01  1.424e-01  -2.619  0.00883 **
## Parch       -3.657e-02  1.449e-01  -0.252  0.80070
## Fare         1.432e-03  2.590e-03   0.553  0.58026
## EmbarkedC   -1.170e+01  5.354e+02  -0.022  0.98256
## EmbarkedQ   -1.205e+01  5.354e+02  -0.023  0.98204
## EmbarkedS   -1.198e+01  5.354e+02  -0.022  0.98215
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 771.67  on 571  degrees of freedom
## Residual deviance: 524.52  on 561  degrees of freedom
## AIC: 546.52
##
## Number of Fisher Scoring iterations: 12
```

```r
#Removing Embarked
Linear_Model_reg_titanic_1 <- glm(titanic_train_data$Survived ~ PassengerId + Pclass + Sex + Age + SibSp
summary(Linear_Model_reg_titanic_1)
```

```
##
```

```
## Call:
## glm(formula = titanic_train_data$Survived ~ PassengerId + Pclass +
##     Sex + Age + SibSp + Parch + Fare, family = binomial, data = titanic_train_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7039  -0.6819  -0.3975   0.6499   2.4011
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  4.9155936  0.6871286   7.154 8.44e-13 ***
## PassengerId  0.0003951  0.0004130   0.957  0.33873
## Pclass      -1.1605178  0.1769091  -6.560 5.38e-11 ***
## Sexmale     -2.5638828  0.2419191 -10.598  < 2e-16 ***
## Age         -0.0409945  0.0091163  -4.497 6.90e-06 ***
## SibSp       -0.3877187  0.1414662  -2.741  0.00613 **
## Parch       -0.0347173  0.1437894  -0.241  0.80921
## Fare         0.0019517  0.0025437   0.767  0.44292
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 771.67  on 571  degrees of freedom
## Residual deviance: 525.68  on 564  degrees of freedom
## AIC: 541.68
##
## Number of Fisher Scoring iterations: 5
```

```r
#Removing Fare
Linear_Model_reg_titanic_2 <- glm(titanic_train_data$Survived ~ PassengerId + Pclass + Sex + Age + SibSp
summary(Linear_Model_reg_titanic_2)
```

```
##
## Call:
## glm(formula = titanic_train_data$Survived ~ PassengerId + Pclass +
##     Sex + Age + SibSp + Parch, family = binomial, data = titanic_train_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6707  -0.6837  -0.3956   0.6404   2.4132
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  5.1296466  0.6320653   8.116 4.83e-16 ***
## PassengerId  0.0003879  0.0004127   0.940  0.34729
## Pclass      -1.2287669  0.1545023  -7.953 1.82e-15 ***
## Sexmale     -2.5645632  0.2416476 -10.613  < 2e-16 ***
## Age         -0.0414687  0.0090994  -4.557 5.18e-06 ***
## SibSp       -0.3764697  0.1404691  -2.680  0.00736 **
## Parch       -0.0097668  0.1403997  -0.070  0.94454
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
```

```
##
##     Null deviance: 771.67  on 571  degrees of freedom
## Residual deviance: 526.32  on 565  degrees of freedom
## AIC: 540.32
##
## Number of Fisher Scoring iterations: 5
```

```r
#Removing Parch
Linear_Model_reg_titanic_3 <- glm(titanic_train_data$Survived ~ PassengerId + Pclass + Sex + Age + SibSp
summary(Linear_Model_reg_titanic_3)
```

```
##
## Call:
## glm(formula = titanic_train_data$Survived ~ PassengerId + Pclass +
##     Sex + Age + SibSp, family = binomial, data = titanic_train_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6744  -0.6832  -0.3949   0.6405   2.4134
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  5.1236506  0.6261720   8.182 2.78e-16 ***
## PassengerId  0.0003864  0.0004121   0.938  0.34843
## Pclass      -1.2286463  0.1545071  -7.952 1.83e-15 ***
## Sexmale     -2.5611026  0.2363856 -10.834  < 2e-16 ***
## Age         -0.0414213  0.0090730  -4.565 4.99e-06 ***
## SibSp       -0.3794739  0.1337160  -2.838  0.00454 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 771.67  on 571  degrees of freedom
## Residual deviance: 526.32  on 566  degrees of freedom
## AIC: 538.32
##
## Number of Fisher Scoring iterations: 5
```

```r
#Removing PassengerId
Linear_Model_reg_titanic_4 <- glm(titanic_train_data$Survived ~ Pclass + Sex + Age + SibSp,data=titanic_
summary(Linear_Model_reg_titanic_4)
```

```
##
## Call:
## glm(formula = titanic_train_data$Survived ~ Pclass + Sex + Age +
##     SibSp, family = binomial, data = titanic_train_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6952  -0.6790  -0.4026   0.6481   2.3954
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  5.310836   0.596442   8.904  < 2e-16 ***
## Pclass      -1.233083   0.154295  -7.992 1.33e-15 ***
```

```
## Sexmale     -2.549157   0.235421 -10.828  < 2e-16 ***
## Age         -0.041623   0.009071  -4.589 4.46e-06 ***
## SibSp       -0.389311   0.133140  -2.924  0.00345 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 771.67  on 571  degrees of freedom
## Residual deviance: 527.20  on 567  degrees of freedom
## AIC: 537.2
##
## Number of Fisher Scoring iterations: 5
```

```r
Linear_Model_reg_titanic_final <- glm(titanic_train_data$Survived ~ Pclass + Sex + Age + SibSp,data=tita
summary(Linear_Model_reg_titanic_final)
```

```
##
## Call:
## glm(formula = titanic_train_data$Survived ~ Pclass + Sex + Age +
##     SibSp, family = binomial, data = titanic_train_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6952  -0.6790  -0.4026   0.6481   2.3954
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  5.310836   0.596442   8.904  < 2e-16 ***
## Pclass      -1.233083   0.154295  -7.992 1.33e-15 ***
## Sexmale     -2.549157   0.235421 -10.828  < 2e-16 ***
## Age         -0.041623   0.009071  -4.589 4.46e-06 ***
## SibSp       -0.389311   0.133140  -2.924  0.00345 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 771.67  on 571  degrees of freedom
## Residual deviance: 527.20  on 567  degrees of freedom
## AIC: 537.2
##
## Number of Fisher Scoring iterations: 5
```

4. (5 pts) State the model as a regression equation.

```r
#Equation
#y= 5.941125 - 1.375064(Pclass) - -2.664718(Sex) - 0.050312(Age) - 0.452608(SibSp)
```

5. (5 pts) Test the model against the test data set and determine its prediction accuracy (as a percentage correct).

```r
# 4.Accuracy

pred <- predict(Linear_Model_reg_titanic_final,titanic_test_data,type = "response")
pred <- ifelse(pred > 0.5,1,0)
```

```r
Accuracydata <- as.data.frame(cbind(as.integer(pred),titanic_test_data$Survived))
Accuracydata <- as.data.frame(Accuracydata)
colnames(Accuracydata) <- c("Predicted","Actual")
Accuracydata$accuracy <- ifelse(Accuracydata$Predicted == Accuracydata$Actual, 1,0)
mean(Accuracydata$accuracy)*100
```

```
## [1] 83.80282
```

Problem 4 (10 Points) (10 pts) Elaborate on the use of kNN and Naive Bayes for data imputation. Explain in reasonable detail how you would use these algorithms to impute missing data and why it can work.

kNN: kNN is a machine learning algorithm for classification which can also be used for data imputation. It can be used for continuous, discrete, ordinal and categorical data imputation. There is an underlying assumption that a point can be approximated by values of the points which are nearest to it, based on other features. It matches a given point with its closest neighbours in a multidimensional space based on distances. Distances between different data points are calculated based on distance measures such as the Euclidean, Manhattan, Hamming distance etc. Then the data points are arranged by the distances in a multidimensional space and we consider a given number of closest points(neighbors) for the missing data based on the value of 'k' taken. The selection of k is also quite important. If the value of k is very low, it increases influence of noise and if it is high, it doesn't take local effects in account. Also if the classes are binary, k should be an odd value so that ties can be avoided. Then after considering the k nearest neighbors, any of the aggregation methods such as mean, median or mode are used for imputation of the missing data if the data is numeric and mode if it is categorical.

Naive Bayes: Naive Bayes is a classifier which is based on Bayesian methods which determines the empirical probabilities of each outcome based on frequencies of each of feature values. It is used for categorical data and if the data is numerical, it is first converted to categorical by binning it. When the classifier is then applied to unlabelled cases, it uses the empirical probabilities to predict the most likely case for the unknown class. Naive Bayes uses all the features in the data simultaneously. It makes the assumption that the features are independent of each other. However, even if they are not, Bayes classifier still works really well. For classifying missing data using Naive Bayes, frequency tables of all the features are made for all the categories which are present int the dataset which we will be using for imputation. From the frequency, likelihoods of each of the values in all the features are calculated to build a likelihood table. After doing so, the condidtional probabilities are multiplied and then divided by the total likelihood. This transforms each class likelihood into a probability and then based on the probability, imputation of missing data is done by replacing the missing values with the class having highest probability for the same.

Refernces: Lecture Videos and Textbook https://towardsdatascience.com/the-use-of-knn-for-missing-values-cf33d935c637 http://conteudo.icmc.usp.br/pessoas/gbatista/files/his2002.pdf