Practicum_2

Minxin Cheng

```r
#install.packages("dplyr")
#install.packages("DMwR")
#install.packages("psych")
#install.packages("klaR")
#install.packages("caret")
#install.packages("MuMIn")
library(dplyr)
library(DMwR)
library(psych)
library(klaR)
library(caret)
library(MuMIn)
```

Problem 1

Question 1

Download the data set Census Income Data for Adults along with its explanation. There are two data sets (adult.data and adult.test). 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. Combine the two data sets into a single data set.

1.1 Read in data files

```r
# read adult. data
adult.data <- read.csv("adult.data", header = FALSE)
# give column names to adult.data
colnames(adult.data) <- c("age", "workclass", "fnlwgt", "edu",
                          "edunum", "status", "occupation", "relationship",
                          "race", "sex", "cgain", "closs", "hrperweek",
                          "country", "class")
# read adult.test
adult.test <- read.csv("adult.test", header = FALSE)
adult.test <- adult.test[-1, ]
# give column names to adult. test
colnames(adult.test) <- c("age", "workclass", "fnlwgt", "edu", "edunum",
                          "status", "occupation", "relationship", "race",
                          "sex", "cgain", "closs", "hrperweek",
                          "country", "class")
```

1.2 Combine two data sets

```r
adult.data <- rbind(adult.data, adult.test)
```

Question 2

Explore the combined data set as you see fit and that allows you to get a sense of the data and get comfortable with it.

2.1 Check column types

```r
# check each column's type
sapply(adult.data, class)
```

```
##          age    workclass       fnlwgt          edu       edunum       status
##  "character"  "character"    "integer"  "character"    "integer"  "character"
##   occupation relationship         race          sex        cgain        closs
##  "character"  "character"  "character"  "character"    "integer"    "integer"
##    hrperweek      country        class
##    "integer"  "character"  "character"
```

```r
# change age, fnlwgt, edunum, captial-gain, captial-loss, and hours-per-weed to numeric
adult.data[ , c(1, 3, 5, 11:13)] <- lapply(adult.data[ , c(1, 3, 5, 11:13)],
                                           as.numeric)
# change all the rest to factor
adult.data[ , -c(1, 3, 5, 11:13)] <- lapply(adult.data[ , -c(1, 3, 5, 11:13)],
                                            as.factor)
```

2.2 Summarize the data set

```r
# check if there is any missing data
any(is.na(adult.data))
```

```
## [1] FALSE
```

```r
# check the structure of the data
str(adult.data)
```

```
## 'data.frame':    48842 obs. of  15 variables:
##  $ age          : num  39 50 38 53 28 37 49 52 31 42 ...
##  $ workclass    : Factor w/ 9 levels " ?"," Federal-gov",..: 8 7 5 5 5 5 5 7 5 5 ...
##  $ fnlwgt       : num  77516 83311 215646 234721 338409 ...
##  $ edu          : Factor w/ 16 levels " 10th"," 11th",..: 10 10 12 2 10 13 7 12 13 10 ...
##  $ edunum       : num  13 13 9 7 13 14 5 9 14 13 ...
##  $ status       : Factor w/ 7 levels " Divorced"," Married-AF-spouse",..: 5 3 1 3 3 3 4 3 5 3 ...
##  $ occupation   : Factor w/ 15 levels " ?"," Adm-clerical",..: 2 5 7 7 11 5 9 5 11 5 ...
```

```
## $ relationship: Factor w/ 6 levels " Husband"," Not-in-family",..: 2 1 2 1 6 6 2 1 2 1 ...
## $ race        : Factor w/ 5 levels " Amer-Indian-Eskimo",..: 5 5 5 3 3 5 3 5 5 5 ...
## $ sex         : Factor w/ 2 levels " Female"," Male": 2 2 2 2 1 1 1 2 1 2 ...
## $ cgain       : num  2174 0 0 0 0 ...
## $ closs       : num  0 0 0 0 0 0 0 0 0 0 ...
## $ hrperweek   : num  40 13 40 40 40 40 16 45 50 40 ...
## $ country     : Factor w/ 42 levels " ?"," Cambodia",..: 40 40 40 40 6 40 24 40 40 40 ...
## $ class       : Factor w/ 4 levels " <=50K"," <=50K.",..: 1 1 1 1 1 1 1 3 3 3 ...
```

```r
# create a summary table of the data set
summary(adult.data)
```

```
##       age                    workclass         fnlwgt
##  Min.   :17.00    Private          :33906   Min.   :  12285
##  1st Qu.:28.00    Self-emp-not-inc: 3862   1st Qu.: 117550
##  Median :37.00    Local-gov        : 3136   Median : 178144
##  Mean   :38.64    ?                : 2799   Mean   : 189664
##  3rd Qu.:48.00    State-gov        : 1981   3rd Qu.: 237642
##  Max.   :90.00    Self-emp-inc     : 1695   Max.   :1490400
##                  (Other)          : 1463
##            edu            edunum                        status
##   HS-grad     :15784   Min.   : 1.00    Divorced              : 6633
##   Some-college:10878   1st Qu.: 9.00    Married-AF-spouse     :   37
##   Bachelors   : 8025   Median :10.00    Married-civ-spouse    :22379
##   Masters     : 2657   Mean   :10.08    Married-spouse-absent:  628
##   Assoc-voc   : 2061   3rd Qu.:12.00    Never-married         :16117
##   11th        : 1812   Max.   :16.00    Separated             : 1530
##  (Other)      : 7625                    Widowed               : 1518
##            occupation             relationship                  race
##   Prof-specialty : 6172    Husband        :19716    Amer-Indian-Eskimo:  470
##   Craft-repair   : 6112    Not-in-family :12583    Asian-Pac-Islander: 1519
##   Exec-managerial: 6086    Other-relative: 1506    Black             : 4685
##   Adm-clerical   : 5611    Own-child      : 7581    Other             :  406
##   Sales          : 5504    Unmarried      : 5125    White             :41762
##   Other-service  : 4923    Wife           : 2331
##  (Other)         :14434
##       sex            cgain            closs            hrperweek
##   Female:16192   Min.   :    0   Min.   :   0.0   Min.   : 1.00
##   Male  :32650   1st Qu.:    0   1st Qu.:   0.0   1st Qu.:40.00
##                  Median :    0   Median :   0.0   Median :40.00
##                  Mean   : 1079   Mean   :  87.5   Mean   :40.42
##                  3rd Qu.:    0   3rd Qu.:   0.0   3rd Qu.:45.00
##                  Max.   :99999   Max.   :4356.0   Max.   :99.00
##
##           country          class
##   United-States:43832    <=50K :24720
##   Mexico       :  951    <=50K.:12435
##   ?            :  857    >50K  : 7841
##   Philippines  :  295    >50K. : 3846
##   Germany      :  206
##   Puerto-Rico  :  184
##  (Other)       : 2517
```

From the summarization above, workclass, occupation, and country has a level “?” indicating missing data, and most of the strings has a white space in front of them. Next I will check the number of missing data and remove white space. If the missing data is not a big portion of the data set, it will be removed.

2.3 Clean the data set

```
# remove all white space in the data set
adult.data <- as.data.frame(apply(adult.data, 2,
                                  function(x) gsub('\\s+', '', x)))
# check missing data
missing <- adult.data %>%
  filter(workclass == "?" | occupation == "?" | country == "?")
nrow(missing)
```

```
## [1] 3620
```

```
# remove all data (7.4% of the entire data set)
adult.data <- adult.data %>%
  filter(workclass != "?", occupation != "?", country != "?") %>%
  droplevels()
# change class column to a binomial factor that lower than 50K as "lower",
# higher than 50K as "higher"
adult.data$class <- as.factor(case_when((
  adult.data$class == "<=50K" | adult.data$class == "<=50K.") ~ "lower",
  (adult.data$class == ">50K" | adult.data$class == ">50K.") ~ "higher"))
# summary the data set again to get an overview
adult.data[ , c(1, 3, 5, 11:13)] <- lapply(adult.data[ , c(1, 3, 5, 11:13)],
                                           as.numeric)
adult.data[ , -c(1, 3, 5, 11:13)] <- lapply(adult.data[ , -c(1, 3, 5, 11:13)],
                                            as.factor)
summary(adult.data)
```

```
##       age                   workclass          fnlwgt
##  Min.   :17.00   Federal-gov     : 1406   Min.   :  13492
##  1st Qu.:28.00   Local-gov       : 3100   1st Qu.: 117388
##  Median :37.00   Private         :33307   Median : 178316
##  Mean   :38.55   Self-emp-inc    : 1646   Mean   : 189735
##  3rd Qu.:47.00   Self-emp-not-inc: 3796   3rd Qu.: 237926
##  Max.   :90.00   State-gov       : 1946   Max.   :1490400
##                  Without-pay     :   21
##            edu             edunum                      status
##  HS-grad     :14783   Min.   : 1.00   Divorced             : 6297
##  Some-college: 9899   1st Qu.: 9.00   Married-AF-spouse    :   32
##  Bachelors   : 7570   Median :10.00   Married-civ-spouse   :21055
##  Masters     : 2514   Mean   :10.12   Married-spouse-absent:  552
##  Assoc-voc   : 1959   3rd Qu.:13.00   Never-married        :14598
##  11th        : 1619   Max.   :16.00   Separated            : 1411
##  (Other)     : 6878                   Widowed              : 1277
##            occupation           relationship                   race
##  Craft-repair   : 6020   Husband       :18666   Amer-Indian-Eskimo:  435
##  Prof-specialty : 6008   Not-in-family :11702   Asian-Pac-Islander: 1303
##  Exec-managerial: 5984   Other-relative: 1349   Black             : 4228
```

```
##  Adm-clerical   : 5540   Own-child     : 6626   Other              :  353
##  Sales          : 5408   Unmarried     : 4788   White              :38903
##  Other-service  : 4808   Wife          : 2091
##  (Other)        :11454
##      sex            cgain          closs            hrperweek
##  Female:14695   Min.   :    0   Min.   :   0.00   Min.   : 1.00
##  Male  :30527   1st Qu.:    0   1st Qu.:   0.00   1st Qu.:40.00
##                 Median :    0   Median :   0.00   Median :40.00
##                 Mean   : 1101   Mean   :  88.59   Mean   :40.94
##                 3rd Qu.:    0   3rd Qu.:   0.00   3rd Qu.:45.00
##                 Max.   :99999   Max.   :4356.00   Max.   :99.00
##
##           country         class
##  United-States:41292   higher:11208
##  Mexico       :  903   lower :34014
##  Philippines  :  283
##  Germany      :  193
##  Puerto-Rico  :  175
##  Canada       :  163
##  (Other)      : 2213
```

Question 3

Split the combined data set 70%/30% so you retain 30% for validation and tuning using random sampling with replacement. Ues a fixed seed so you produce the same results each time you run the code. Going forward you will use the 70% data set for training and the 30% data set for validation and to determine accuracy.

```r
# generate random numbers as the row index of training data
set.seed(500)
train.sample <- sample.int(nrow(adult.data),
                           0.7 * nrow(adult.data), replace = TRUE)
# split the data set
adult.data.training <- adult.data[train.sample, ]
adult.data.testing <- adult.data[-train.sample, ]
# check the proportion of lower and higher in both training and testing data
prop.table(table(adult.data.training$class))
```

```
##
##    higher     lower
## 0.2453325 0.7546675
```

```r
prop.table(table(adult.data.testing$class))
```

```
##
##    higher     lower
## 0.2499107 0.7500893
```

As the proportion table shows, training and testing data are fairly even.

Question 4

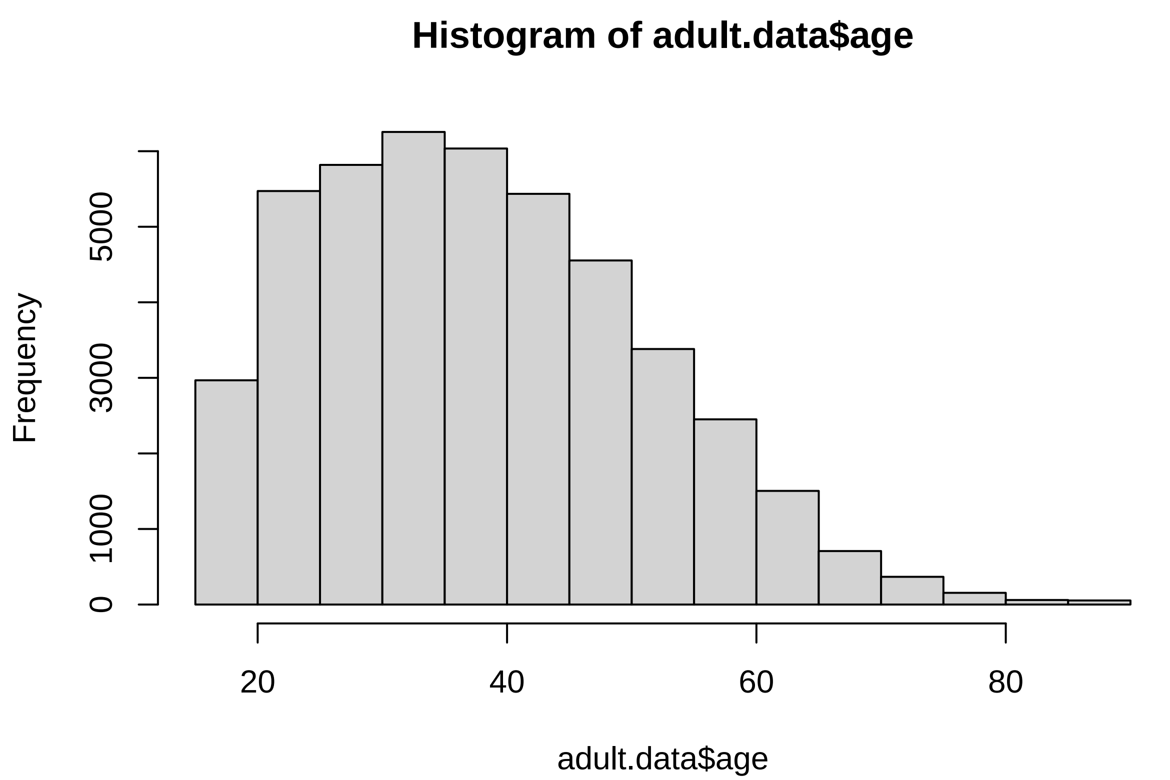
Using the Naive Bayes Classification algorithm from the KlaR package, build a binary classifier that predicts whether an individual earns more than or less than US$50,000. Only use the features age, education, workclass, sex, race, and naive-country. Ignore any other features in your model. You need to transform continuous variables into categorical variables by binning (use equal size bins from min to max).

4.1 Check the distribution of age

```
# make a copy of training data and testing data for Naive Bayes
adult.data.training.nb <- adult.data.training
adult.data.testing.nb <- adult.data.training
# check the distribution of age
summary(adult.data$age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   17.00   28.00   37.00   38.55   47.00   90.00
```

```
hist(adult.data$age)
```



Therefore, all the subject will be binned to 5 groups based on their age: 17-20, 20-40, 40-60, 60-80, 80-90

4.2 Bin the data

```
# assign bins to subjects
# training data
adult.data.training.nb <- adult.data.training.nb %>%
  mutate(age = case_when(age <= 20 ~ 1,
                         ((age > 20) & (age <= 40)) ~ 2,
                         ((age > 40) & (age <= 60)) ~ 3,
                         ((age > 60) & (age <= 80)) ~ 4,
                         (age > 80) ~ 5))
# testing data
adult.data.testing.nb <- adult.data.testing.nb %>%
  mutate(age = case_when(age <= 20 ~ 1,
                         ((age > 20) & (age <= 40)) ~ 2,
                         ((age > 40) & (age <= 60)) ~ 3,
                         ((age > 60) & (age <= 80)) ~ 4,
                         (age > 80) ~ 5))
# convert them to factors
adult.data.training.nb$age <- as.factor(adult.data.training.nb$age)
adult.data.testing.nb$age <- as.factor(adult.data.testing.nb$age)
```

4.3 Create a Naive Bayes classifier and make prediction

```
# create classifier
adult.model.nb <- NaiveBayes(class ~ age + edu + workclass +
                               sex + race + country,
                             data = adult.data.training.nb)
# make prediction
adult.pred.nb <- adult.model.nb %>% predict(adult.data.testing.nb)
```

Question 5

Build a confusion matrix for the classifier from (4) and comment on it, e.g., explain what it means.

```
confusionMatrix(adult.pred.nb$class, adult.data.testing.nb$class)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction higher lower
##     higher   2965  1878
##     lower    4801 22011
##
##                Accuracy : 0.789
##                  95% CI : (0.7845, 0.7935)
##     No Information Rate : 0.7547
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3473
```

```
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.38179
##             Specificity : 0.92139
##          Pos Pred Value : 0.61222
##          Neg Pred Value : 0.82094
##              Prevalence : 0.24533
##          Detection Rate : 0.09367
##    Detection Prevalence : 0.15299
##       Balanced Accuracy : 0.65159
##
##        'Positive' Class : higher
##
```

From the confusion matrix, the classifier made 31655 predictions, overall prediction acuuracy is 0.789. True positive rate (sensitivity) is 0.382, true negative rate (specificity) is 0.921.

Question 6

Create a full logistic regression model of the same features as in (4) (i.e., do not eliminate any features regardless of p-value). Be sure to either use dummy coding for categorical features or convert them to factor variables and ensure that the glm function does the dummy coding.

6.1 Prepare the data set

```
# create a copy for glm
adult.data.training.glm <- adult.data.training
adult.data.testing.glm <- adult.data.testing

# create a function dummy codes
dummy.adult <- function(data){
  for (i in c(2, 4, 9, 10, 14)){
    contrasts(data[ , i])
  }
}

# convert workclass, education, sex, race, and native-country to dummy codes
dummy.adult(adult.data.training.glm)
dummy.adult(adult.data.testing.glm)
```

6.2 Create glm model and make prediction

```
# create glm model
adult.model.glm <- glm(class ~ age + edu + workclass + sex + race + country,
                       data = adult.data.training.glm, family = binomial)
# summary the model
summary(adult.model.glm)
```

```
##
## Call:
## glm(formula = class ~ age + edu + workclass + sex + race + country,
##     family = binomial, data = adult.data.training.glm)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.2861   0.0457   0.4324   0.6713   2.5422
##
## Coefficients:
##                                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)                         6.606849   0.838005   7.884 3.17e-15 ***
## age                                -0.044021   0.001238 -35.549  < 2e-16 ***
## edu11th                            -0.046704   0.209214  -0.223  0.82335
## edu12th                            -0.577148   0.260959  -2.212  0.02699 *
## edu1st-4th                          0.780747   0.542493   1.439  0.15010
## edu5th-6th                          0.294448   0.326302   0.902  0.36686
## edu7th-8th                          0.330929   0.232331   1.424  0.15434
## edu9th                              0.157967   0.266224   0.593  0.55294
## eduAssoc-acdm                      -2.164448   0.168246 -12.865  < 2e-16 ***
## eduAssoc-voc                       -1.869455   0.165814 -11.274  < 2e-16 ***
## eduBachelors                       -2.747946   0.154695 -17.764  < 2e-16 ***
## eduDoctorate                       -3.842487   0.198754 -19.333  < 2e-16 ***
## eduHS-grad                         -1.310254   0.153724  -8.523  < 2e-16 ***
## eduMasters                         -3.181917   0.160531 -19.821  < 2e-16 ***
## eduPreschool                       11.304790 104.763413   0.108  0.91407
## eduProf-school                     -3.882137   0.184619 -21.028  < 2e-16 ***
## eduSome-college                    -1.746428   0.154769 -11.284  < 2e-16 ***
## workclassLocal-gov                  0.287861   0.091917   3.132  0.00174 **
## workclassPrivate                    0.311021   0.077986   3.988 6.66e-05 ***
## workclassSelf-emp-inc              -0.287628   0.101910  -2.822  0.00477 **
## workclassSelf-emp-not-inc           0.561512   0.090768   6.186 6.16e-10 ***
## workclassState-gov                  0.621503   0.103850   5.985 2.17e-09 ***
## workclassWithout-pay                0.709266   0.828763   0.856  0.39210
## sexMale                            -1.281511   0.037735 -33.961  < 2e-16 ***
## raceAsian-Pac-Islander             -0.403454   0.228911  -1.762  0.07799 .
## raceBlack                           0.044642   0.193589   0.231  0.81762
## raceOther                          -0.327420   0.281591  -1.163  0.24493
## raceWhite                          -0.412150   0.184628  -2.232  0.02559 *
## countryCanada                      -1.378441   0.824866  -1.671  0.09470 .
## countryChina                       -0.415768   0.852985  -0.487  0.62596
## countryColumbia                     0.117627   0.958165   0.123  0.90230
## countryCuba                        -0.640684   0.845519  -0.758  0.44861
## countryDominican-Republic           0.917165   1.101997   0.832  0.40525
## countryEcuador                     -1.090993   0.918272  -1.188  0.23480
## countryEl-Salvador                  0.024720   0.905422   0.027  0.97822
## countryEngland                     -1.412496   0.843861  -1.674  0.09416 .
## countryFrance                      -1.363131   0.912342  -1.494  0.13515
## countryGermany                     -0.883741   0.827934  -1.067  0.28579
## countryGreece                      -0.856882   0.972739  -0.881  0.37837
## countryGuatemala                    1.073008   1.300579   0.825  0.40936
## countryHaiti                        1.112476   1.304274   0.853  0.39369
## countryHoland-Netherlands          11.215452 624.194339   0.018  0.98566
## countryHonduras                     0.270356   1.349732   0.200  0.84124
```

```
## countryHong                        -0.779997   1.018877  -0.766  0.44395
## countryHungary                      0.605843   1.070059   0.566  0.57127
## countryIndia                       -0.607900   0.816414  -0.745  0.45651
## countryIran                        -0.777485   0.862335  -0.902  0.36727
## countryIreland                     -1.703801   0.942272  -1.808  0.07058 .
## countryItaly                       -1.404667   0.857796  -1.638  0.10152
## countryJamaica                     -0.636759   0.913149  -0.697  0.48560
## countryJapan                       -1.056319   0.843943  -1.252  0.21070
## countryLaos                         0.614153   1.201259   0.511  0.60917
## countryMexico                      -0.049130   0.819812  -0.060  0.95221
## countryNicaragua                   12.098549 147.755836   0.082  0.93474
## countryOutlying-US(Guam-USVI-etc)  -0.162819   1.131933  -0.144  0.88563
## countryPeru                        12.372147 148.607943   0.083  0.93365
## countryPhilippines                 -0.822004   0.809255  -1.016  0.30975
## countryPoland                      -0.064124   0.879820  -0.073  0.94190
## countryPortugal                    -1.187375   0.918289  -1.293  0.19600
## countryPuerto-Rico                 -0.185946   0.870270  -0.214  0.83081
## countryScotland                    13.063842 229.884940   0.057  0.95468
## countrySouth                       -0.772071   0.850607  -0.908  0.36405
## countryTaiwan                      -0.421831   0.865205  -0.488  0.62587
## countryThailand                     0.294153   1.067959   0.275  0.78298
## countryTrinadad&Tobago             12.361548 266.038904   0.046  0.96294
## countryUnited-States               -0.821696   0.797825  -1.030  0.30305
## countryVietnam                      0.322121   0.955433   0.337  0.73601
## countryYugoslavia                  -1.302676   0.943690  -1.380  0.16746
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 35273  on 31654  degrees of freedom
## Residual deviance: 27639  on 31587  degrees of freedom
## AIC: 27775
##
## Number of Fisher Scoring iterations: 13
```

From the summarization above, AIC of this model is 27775. Age, education, workclass, sex, and race are significant related to the class.

```r
# make prediction
adult.pred.glm <- adult.model.glm %>%
  predict(adult.data.testing.glm, type = "response")
head(adult.pred.glm)
```

```
##         2         3         4         6         7         8
## 0.4265359 0.8052532 0.9226908 0.7054372 0.9869610 0.7414754
```

The output is the probability of the class, however, it didn't indicate which classess do these probabilities refers to. Next I will use contrasts() function to check.

```r
contrasts(adult.data$class)
```

```
##        lower
```

```
## higher     0
## lower      1
```

From the result, 1 is for lower class. Therefore, probability > 0.5 will be lower class, assign the binomial result as below.

```
adult.pred.glm <- ifelse(adult.pred.glm > 0.5, "lower", "higher")
```

Question 7

Build a confusion matrix for the classifier from (5) and comment on it, e.g., explain what it means.

```
confusionMatrix(as.factor(adult.pred.glm), adult.data.testing.glm$class)
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction higher lower
##     higher   2045  1100
##     lower    3551 15696
##
##                Accuracy : 0.7923
##                  95% CI : (0.7869, 0.7976)
##     No Information Rate : 0.7501
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3512
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.36544
##             Specificity : 0.93451
##          Pos Pred Value : 0.65024
##          Neg Pred Value : 0.81550
##              Prevalence : 0.24991
##          Detection Rate : 0.09133
##    Detection Prevalence : 0.14045
##       Balanced Accuracy : 0.64997
##
##        'Positive' Class : higher
##
```

From the confusion matrix, the classifier made 31655 predictions, overall prediction acuuracy is 0.792. True positive rate (sensitivity) is 0.365, true negative rate (specificity) is 0.935. The accuracy is slightly higher than Naive Bayes.

Question 8

Build a function called predictEarningsClass() that predicts whether an individual makes more or less than US$50,000 and that combines the two predictive models from (4) and (6) into a simple ensemble. If the two models disagree on a prediction, then the prediction should be the one from the model with the higher accuracy – make sure you do not hard code that as the training data may change over time and the same model may not be the more accurate forever.

```
predictEarningsClass <- function(newdata){
  # make two copies of new data for each model
  data1 <- newdata
  data2 <- newdata
  # prepare new data
  data1 <- data1 %>% mutate(age = case_when(age <= 20 ~ 1,
                                            ((age > 20) & (age <= 40)) ~ 2,
                                            ((age > 40) & (age <= 60)) ~ 3,
                                            ((age > 60) & (age <= 80)) ~ 4,
                                            (age > 80) ~ 5))
  data1[ , 1] <- as.factor(data1[ , 1])
  # make predictions using both models, Naive Bayes is pred1,
  # logistic regression is pred2
  pred1 <- adult.model.nb %>% predict(data1)
  pred2 <- adult.model.glm %>% predict(data2,  type = "response")
  pred2 <- ifelse(pred2 > 0.5, "lower", "higher")
  # calculate accuracy for both models
  accuracy.nb <- mean(adult.pred.nb$class == adult.data.testing.nb$class)
  accuracy.glm <- mean(adult.pred.glm == adult.data.testing.glm$class)
  # return which model is better
  better.model <- which.max(c(accuracy.nb, accuracy.glm))
  # check if two models are making same predictions
  if(any(adult.pred.glm != adult.pred.nb$class)){
    print("Having different predictions")
    # if better model is Naive Bayes, output is Naive Bayes results
      if (better.model == 1){
        print("Taking prediction from Naive Bayes")
        pred.final <- pred1
      }
    # if better model is logistic regression, return its results
    else{
      print("Taking prediction from logistic regression")
      pred.final <- pred2
    }
  }
  # if two models are making same predictions, return one of them (pred1)
  else{
    pred.final <- pred1
  }
  pred.final
}
```

Question 9

Using the ensemble model from (8), predict whether a 47-year-old black female adult who is a local government worker with a Bacherlor's degree who immigrated from Honduras earns more or less than US$50,000.

```
# create a data frame for the new data
new.sbj <- data.frame("age" = 47, "workclass" = "Federal-gov",
                      "fnlwgt" = NA, "edu" = "Bachelors",
```

```
                      "edunum" = NA, "status" = NA, "occupation" = NA,
                      "relationship" = NA, "race" = "Black",
                      "sex" = "Female", "cgain" = NA, "closs" = NA,
                      "hrperweek" = NA, "country" = "Honduras", "class" = NA)
# check column type
sapply(new.sbj, class)
```

```
##          age    workclass       fnlwgt          edu       edunum       status
##    "numeric"  "character"    "logical"  "character"    "logical"    "logical"
##   occupation relationship         race          sex        cgain        closs
##    "logical"    "logical"  "character"  "character"    "logical"    "logical"
##    hrperweek      country        class
##    "logical"  "character"    "logical"
```

```
# correct the column type
new.sbj[ , -c(1, 3, 5, 11:13)] <-
  lapply(new.sbj[ , -c(1, 3, 5, 11:13)], as.factor)
# make prediction
new.sbj.pred <- predictEarningsClass(new.sbj)
```

```
## [1] "Having different predictions"
## [1] "Taking prediction from logistic regression"
```

```
new.sbj.pred
```

```
##       1
## "lower"
```

From the information above, two models had different predictions, the function took result from logistic regression model as the final resutl. This new subject will likely to make less than $50,000.

Problem 2

Question 1

Load and then explore this data set on car sales into a dataframe called cars.df. Exclude name (manufacturer and model) from the data – do not use in any of the modeling going forward.

```
# read in data file
cars.df <- read.csv("CarDataSet.csv", header = TRUE)
# remove name column
cars.df <- cars.df[ , -1]
# rename the columns for further use
colnames(cars.df) <- c("year", "price", "driven",
                       "fuel", "seller", "transmission", "owner")
```

Question 2

Are there outliers in any one of the features in the data set? How do you identify outliers? Remove them but create a second data set with outliers removed called cars.no.df. Keep the original data set cars.df.

2.1 Convert price and driven columns to z score

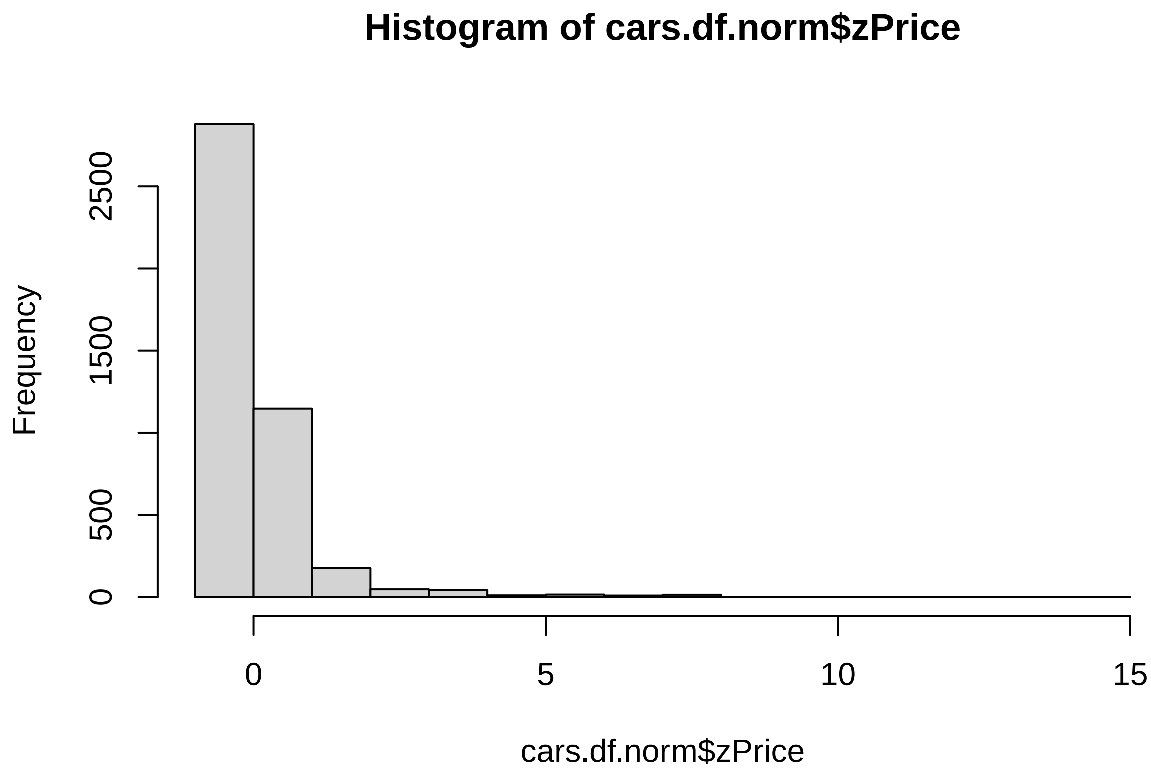
```r
# extract price and driven columns to a new data frame cars.df.norm and calculate z scores
cars.df.norm <- as.data.frame(scale(cars.df[ , 2:3]))
# rename the columns of cars.df.norm
colnames(cars.df.norm) <- c("zPrice", "zDriven")
# summarize the new normalized data frame
summary(cars.df.norm)
```

```
##      zPrice            zDriven
##  Min.   :-0.8368   Min.   :-1.4196
##  1st Qu.:-0.5105   1st Qu.:-0.6692
##  Median :-0.2664   Median :-0.1333
##  Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.1657   3rd Qu.: 0.5099
##  Max.   :14.5120   Max.   :15.8730
```

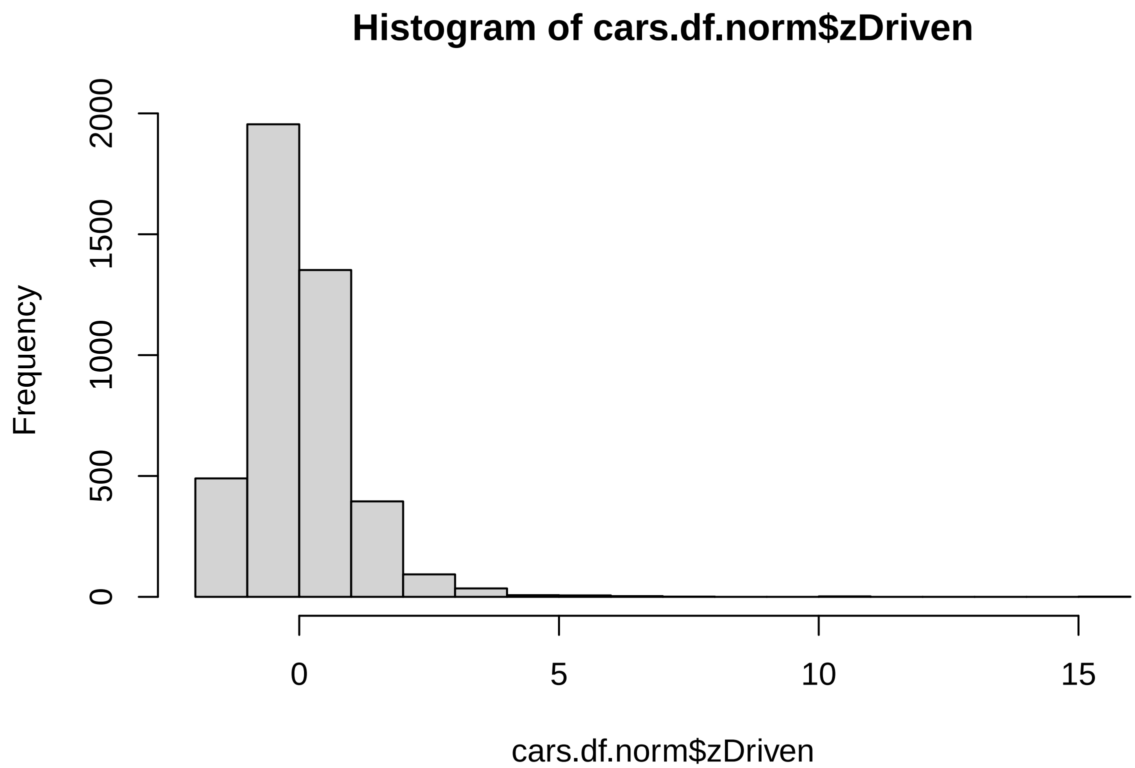
From the summary table, both columns mean values are zero and don't have large IQR, however, both columns have extrame maximum values. Next I will plot figures to check the distribution of these two columns.

2.2 Detect outliers

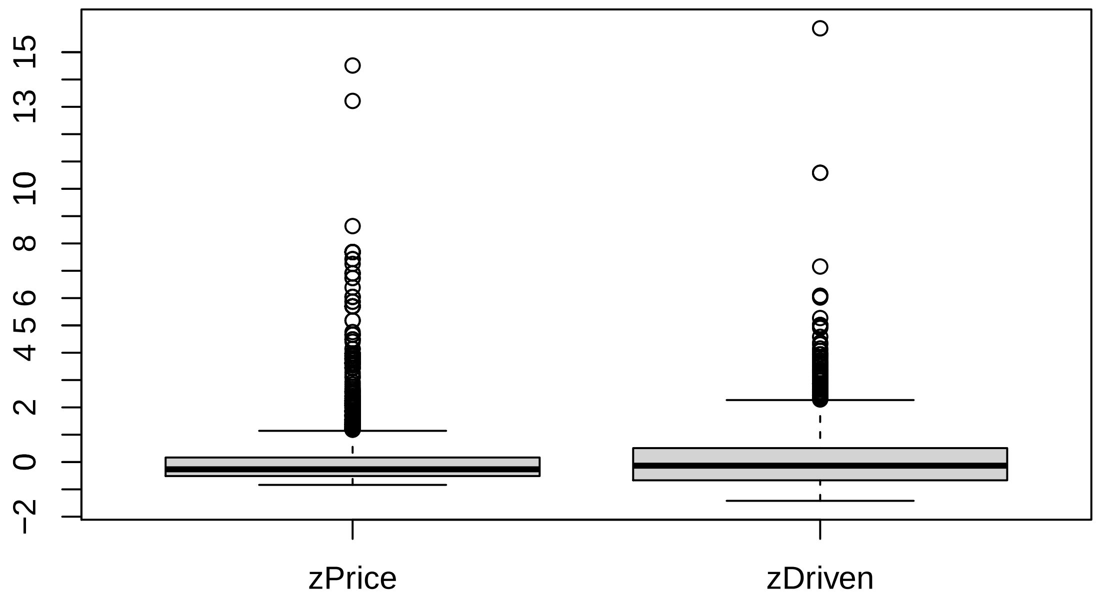
```r
# plot histograms to see the distributions
hist(cars.df.norm$zPrice)
```



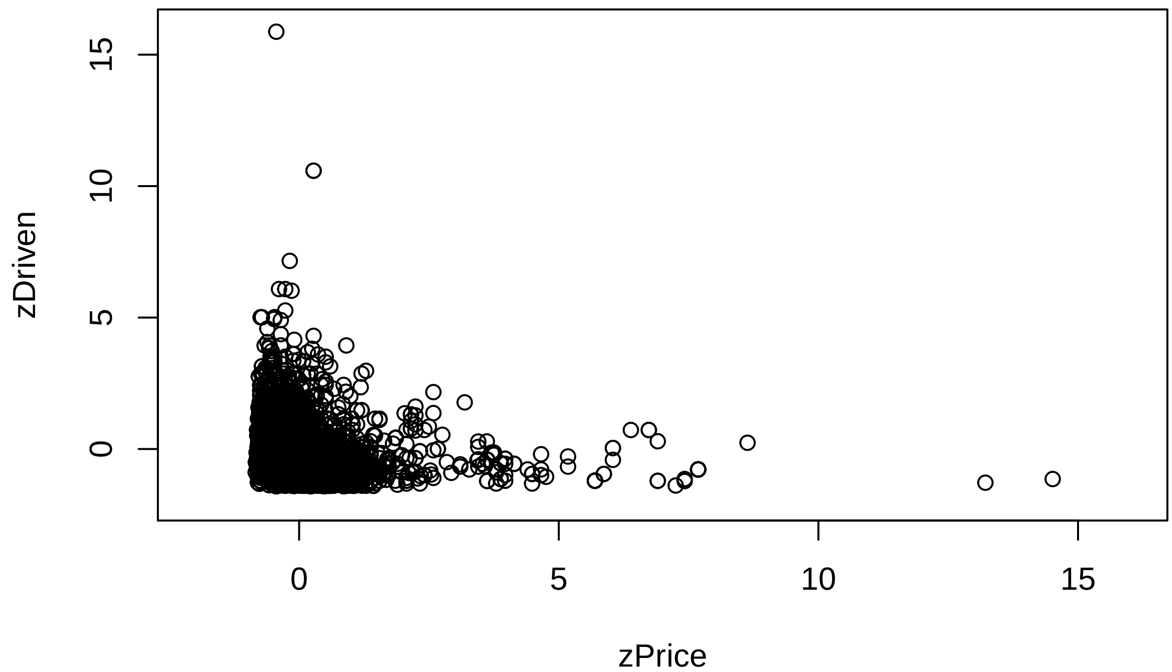

```
hist(cars.df.norm$zDriven)
```




```r
# plot box plot and scatter plot to visualize outliers
boxplot(cars.df.norm)
axis(2, at = seq(-5, 15, 1))
```




```
plot(cars.df.norm, xlim = c(-2, 16), ylim = c(-2, 16))
```



From plots above, z score larger than 2 are likely to be outliers.

2.3 Remove outliers

```
# define outliers as 2 standard deviation away from mean
outliers <- cars.df.norm %>% filter_all(any_vars(abs(.) > 2))
# check how many outliers we have
nrow(outliers)
```

```
## [1] 286
```

```
# add normalized price and driven columns back to the main data, as identifiers of outliers
cars.no.df <- cbind(cars.df, cars.df.norm)
# remove outliers
cars.no.df <- cars.no.df %>%
  anti_join(outliers, by = c("zPrice", "zDriven")) %>%
  dplyr::select(-c(zPrice, zDriven))
```

Question 3

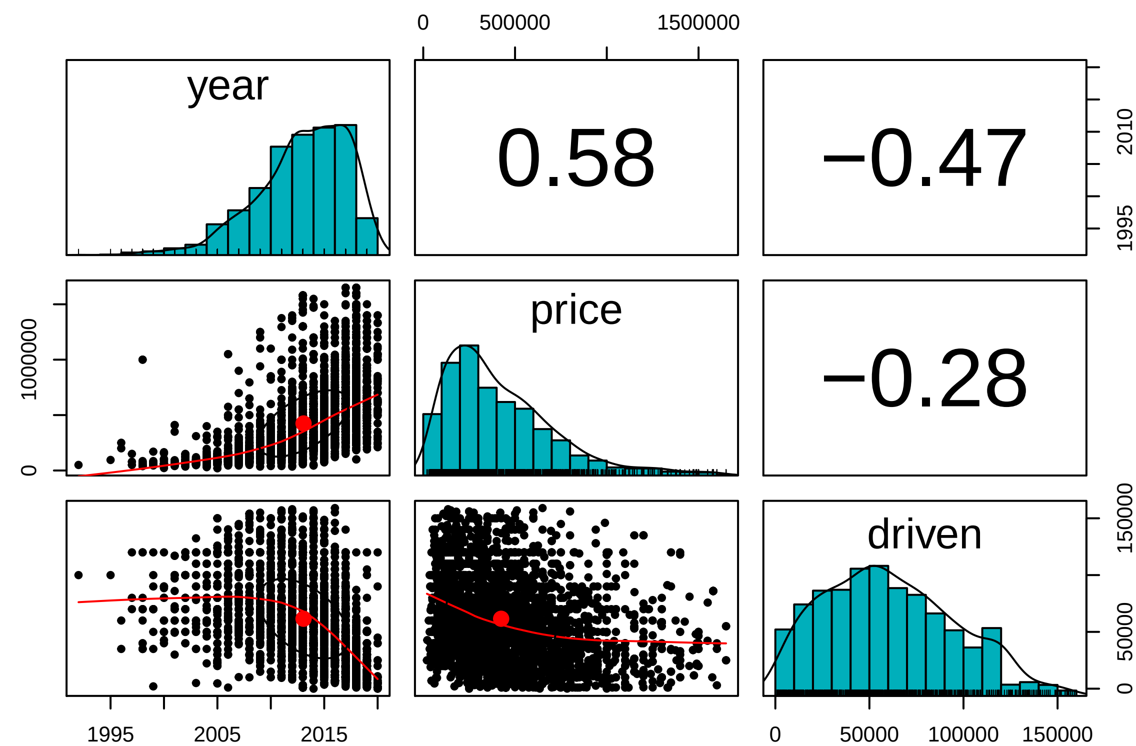
Using pairs.panel, what are the distributions of each of the features in the data set with outliers removed (cars.no.df)? Are they reasonable normal so you can apply a statistical learner such as regression? Can you normalize features through a log, inverse, or square-root transform? State which features should be transformed and then transform as needed and build a new data set, cars.tx.

3.1 Create a pair panel

```r
# check column types
sapply(cars.no.df, class)
```

```
##         year        price       driven         fuel       seller transmission
##    "integer"    "integer"    "integer"  "character"  "character"  "character"
##        owner
##  "character"
```

```r
# create pairs panels
pairs.panels(cars.no.df[, 1:3], method = "pearson",
             hist.col = "#00AFBB", density = TRUE, ellipses = TRUE)
```




```r
# perform shpiro test to check the normality
shapiro.test(cars.no.df$price)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  cars.no.df$price
## W = 0.89778, p-value < 2.2e-16
```

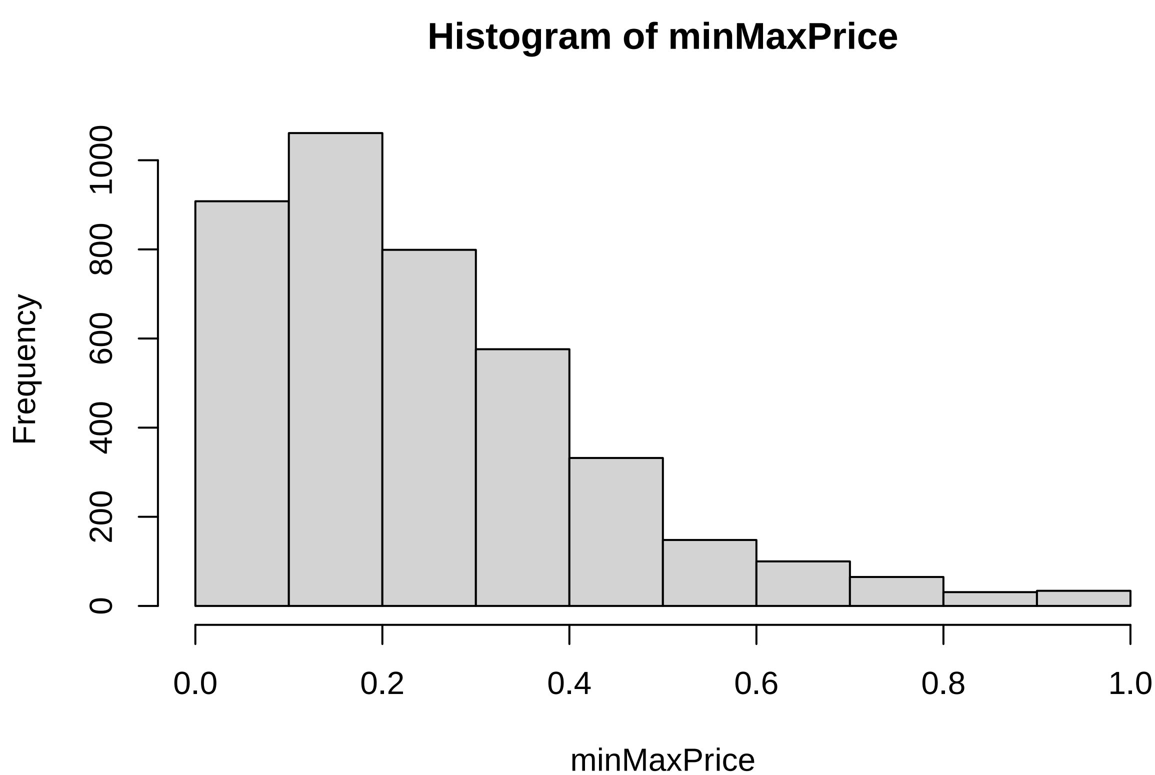
```r
shapiro.test(cars.no.df$driven)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  cars.no.df$driven
## W = 0.97485, p-value < 2.2e-16
```

Form the panel above, the driven data is fairly normal, should be okay for regression. Price data is a bit skewed. I will try different transformation for these two data in the next step see if there is any improvement. Results from Shapiro test indicate both columns are not normally distributed.

3.2 Try different transformations

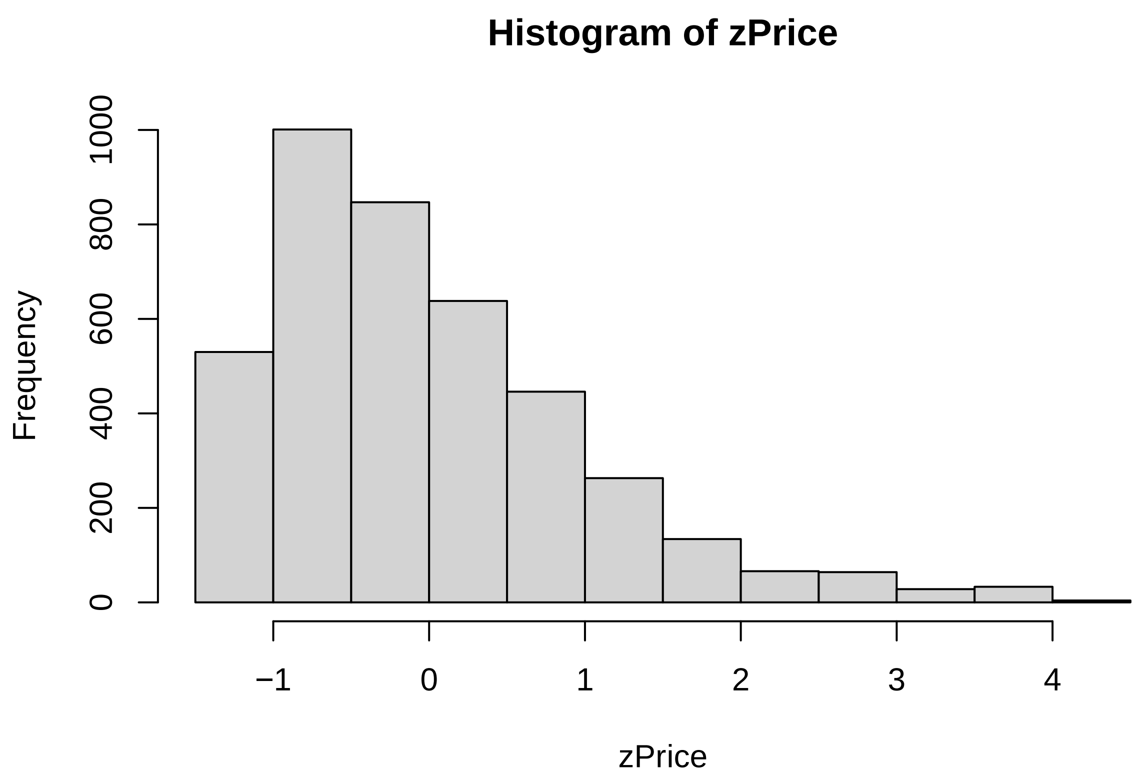
```r
# create a min-max function
minmax <- function(x){
  return((x - min(x)) / (max(x) - min(x)))
}
# transform price using min-max
minMaxPrice <- minmax(cars.no.df$price)
# plot a histogram to check the distribution
hist(minMaxPrice)
```



```r
# preform Shapiro test to check the normality
shapiro.test(minMaxPrice)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  minMaxPrice
## W = 0.89778, p-value < 2.2e-16
```

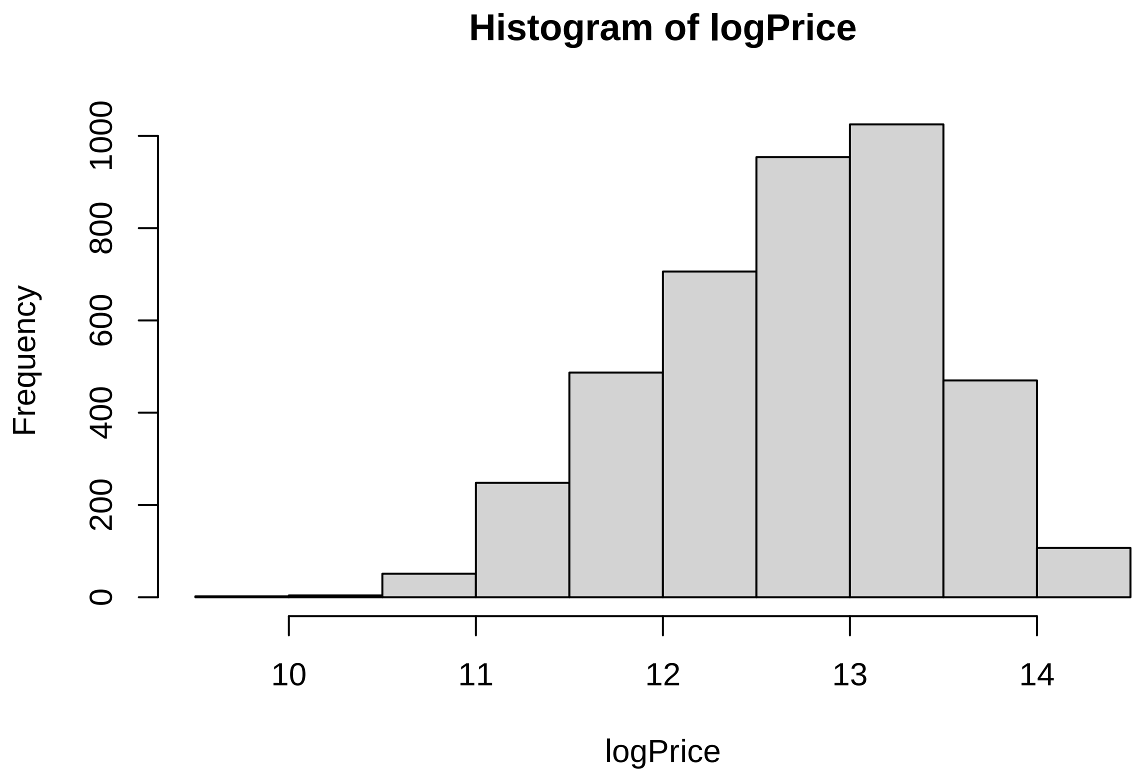
```r
# transform price using z-score
zPrice <- scale(cars.no.df$price)
hist(zPrice)
```




```r
shapiro.test(zPrice)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  zPrice
## W = 0.89778, p-value < 2.2e-16
```

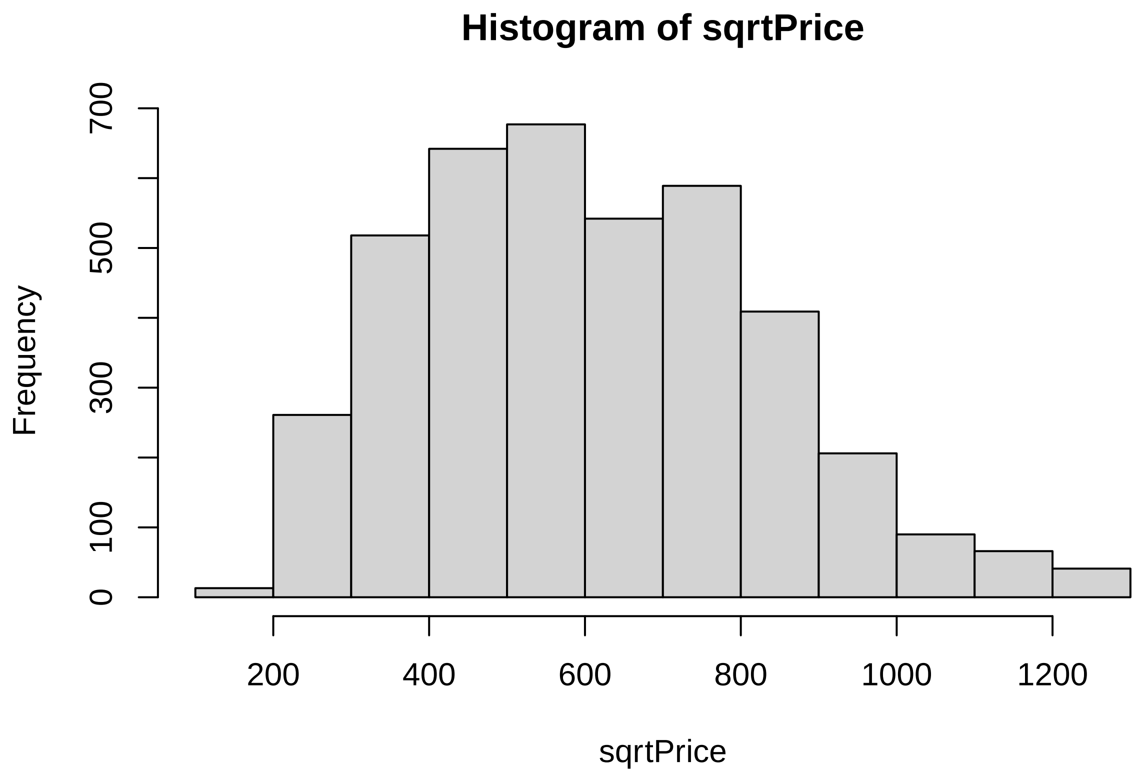
```r
# transform price using log
logPrice <- log(cars.no.df$price)
hist(logPrice)
```




```
shapiro.test(logPrice)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  logPrice
## W = 0.98464, p-value < 2.2e-16
```

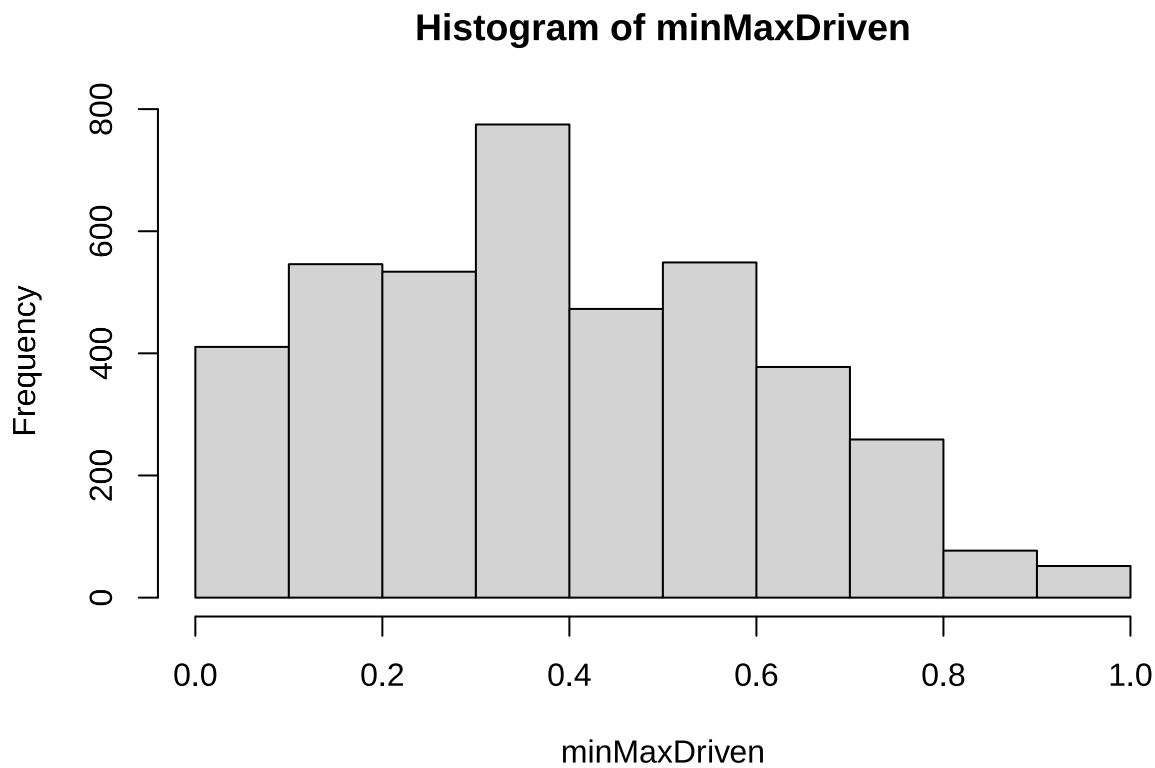
```
# transfrom price using squared root
sqrtPrice <- sqrt(cars.no.df$price)
hist(sqrtPrice)
```




```r
shapiro.test(sqrtPrice)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  sqrtPrice
## W = 0.97969, p-value < 2.2e-16
```

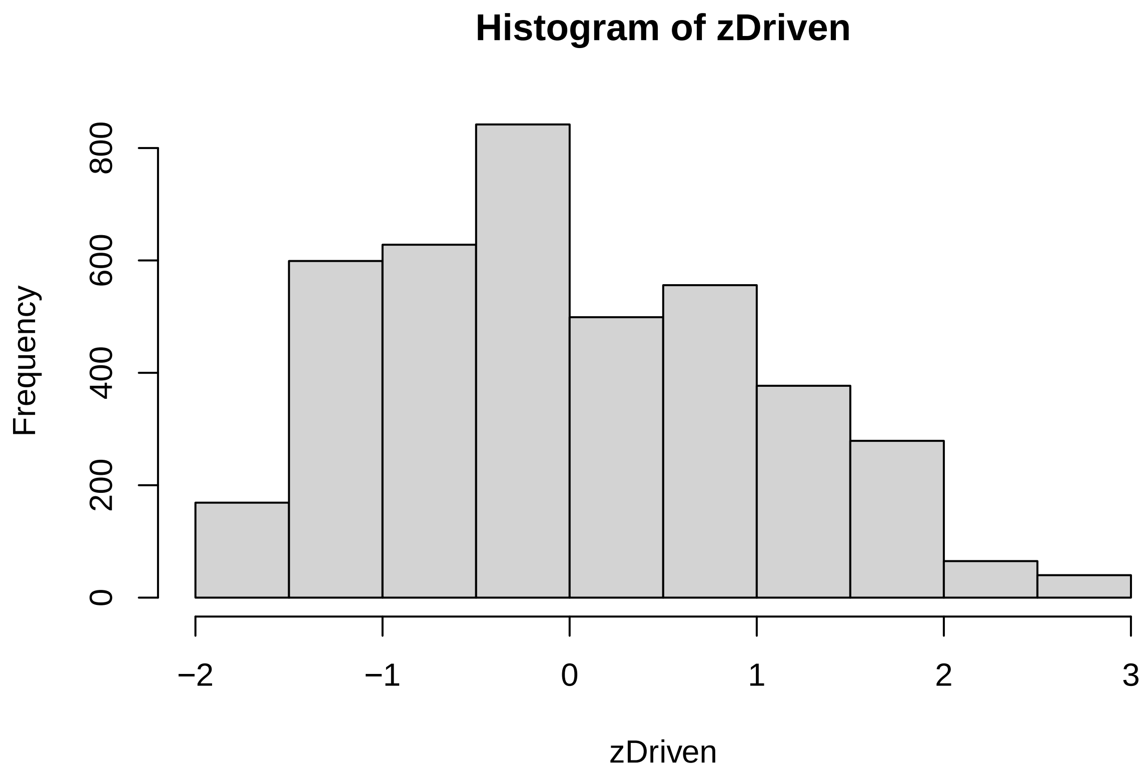
```r
# transform driven using min-max
minMaxDriven <- minmax(cars.no.df$driven)
hist(minMaxDriven)
```




```r
shapiro.test(minMaxDriven)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  minMaxDriven
## W = 0.97485, p-value < 2.2e-16
```

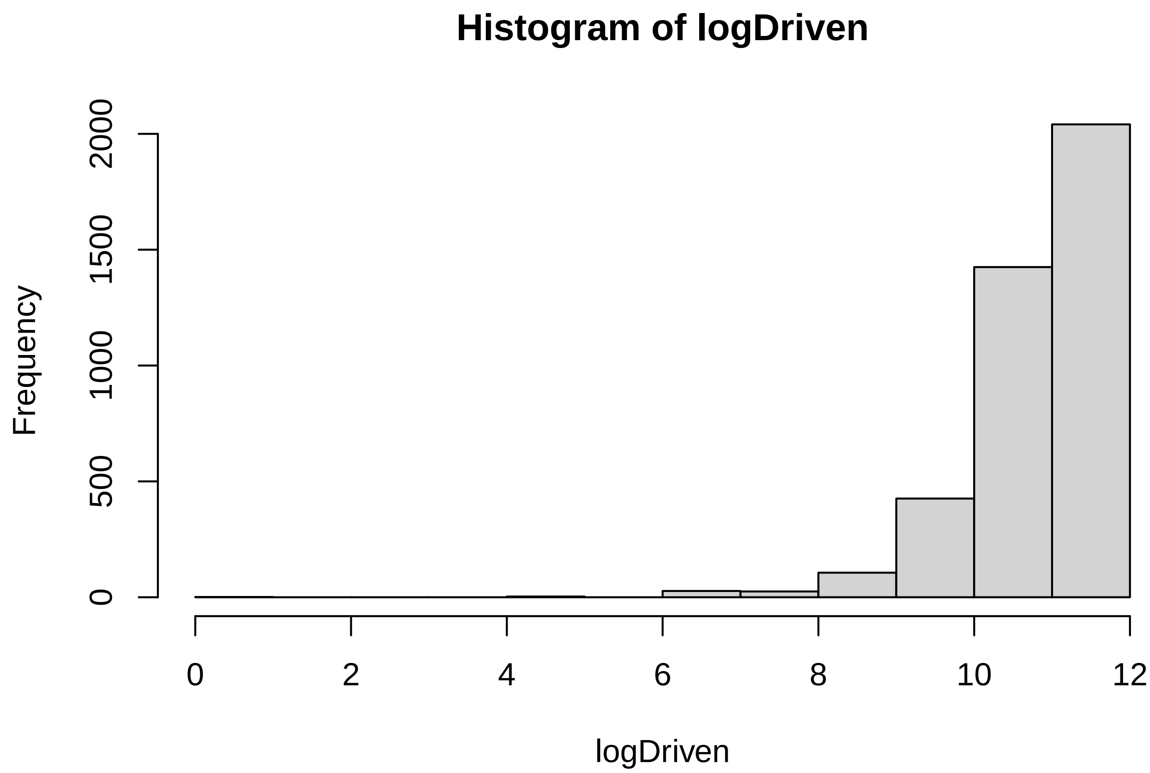
```r
# transform driven using z-score
zDriven <- scale(cars.no.df$driven)
hist(zDriven)
```




```r
shapiro.test(zDriven)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  zDriven
## W = 0.97485, p-value < 2.2e-16
```

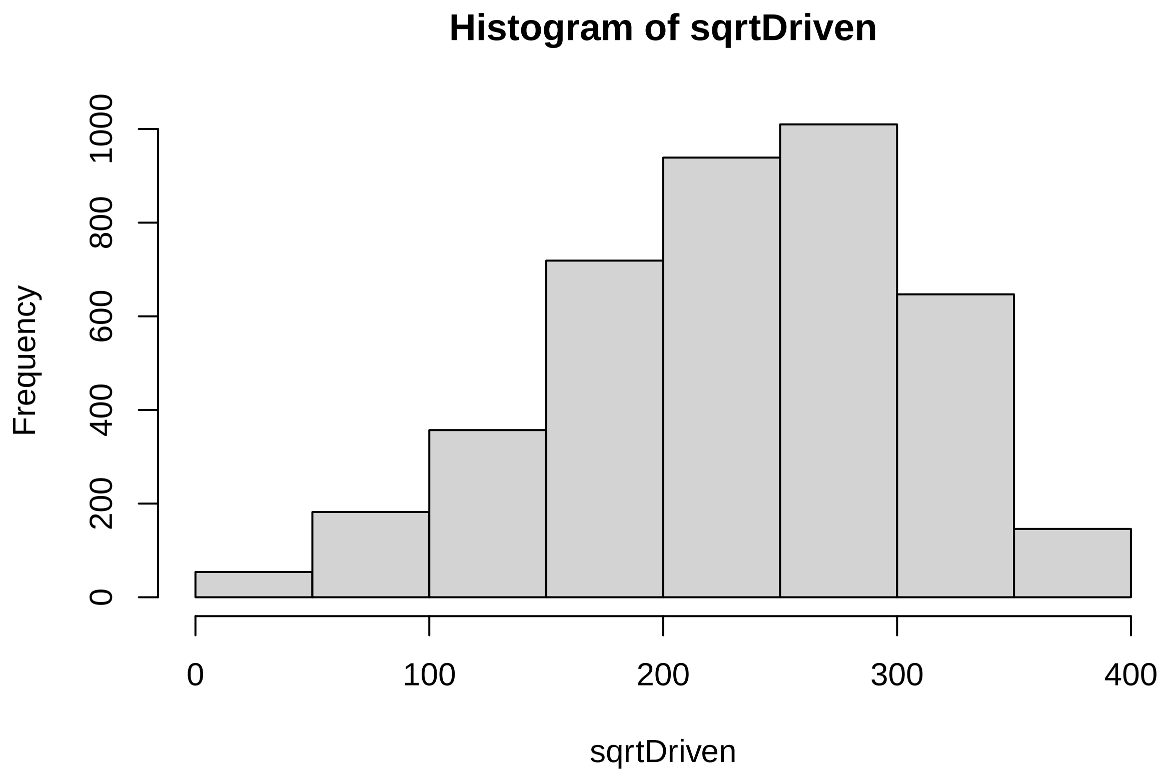
```r
# thransform driven using log
logDriven <- log(cars.no.df$driven)
hist(logDriven)
```




```
shapiro.test(logDriven)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  logDriven
## W = 0.84171, p-value < 2.2e-16
```

```
# transfrom driven using squared root
sqrtDriven <- sqrt(cars.no.df$driven)
hist(sqrtDriven)
```




```
shapiro.test(sqrtDriven)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  sqrtDriven
## W = 0.9837, p-value < 2.2e-16
```

From all attempts above, Shapiro test showed there was no actual improvement of normality. But from the distribution figures, log transformation for price and squraed rood transformation for driven seemed better.

3.3 Create the tranformed data set

```
# make a copy of cars.no.df
cars.tx <- cars.no.df
# replace price column with log transformed price
cars.tx$price <- logPrice
# replace driven column with squared root transformed driven
cars.tx$driven <- sqrtDriven
```

Question 4

What are the correlations to the response variale (car sales price) for cars.no.df? Are there collinearities? Build a full correlation matrix.

```r
cor.matrix <- cor(cars.no.df[, 1:3])
round(cor.matrix, 2)
```

```
##         year price driven
## year    1.00  0.58  -0.47
## price   0.58  1.00  -0.28
## driven -0.47 -0.28   1.00
```

From the matrix, year and price has strong positive relationship (0.58), year and drive has fairly strong negative relationship (-0.47).

Question 5

Split the each of the three data set, cars.no.dr, cars.df, and cars.tx 75z5/25% so you retian 25% for testing using random sampling without replacement. Call the data sets, cars.training and cars.testing, cars.no.training and cars.no.testing, and cars.tx.training and cars.tx.testing.

5.1 Prepare three datasets

```r
# check column types
sapply(cars.df, class)
```

```
##          year        price       driven         fuel       seller transmission 
##     "integer"    "integer"    "integer"  "character"  "character"  "character" 
##         owner 
##   "character"
```

```r
sapply(cars.no.df, class)
```

```
##          year        price       driven         fuel       seller transmission 
##     "integer"    "integer"    "integer"  "character"  "character"  "character" 
##         owner 
##   "character"
```

```r
sapply(cars.tx, class)
```

```
##          year        price       driven         fuel       seller transmission 
##     "integer"    "numeric"    "numeric"  "character"  "character"  "character" 
##         owner 
##   "character"
```

```r
# convert fuel, seller, transmission, and owner to factor columns
cars.df[ ,4:7] <- lapply(cars.df[ , 4:7], factor)
cars.no.df[ , 4:7] <- lapply(cars.no.df[ , 4:7], factor)
cars.tx[ , 4:7] <- lapply(cars.tx[ , 4:7], factor)
```

5.2 Convert categorical columns to dummy codes

```r
# create a function to convert all columns together
dummy.cars <- function(data){
  for (i in 4:7){
    contrasts(data[ , i])
  }
}
# convert all categorical columns to dummy codes in these three data sets
dummy.cars(cars.df)
dummy.cars(cars.no.df)
dummy.cars(cars.tx)
```

5.3 Split three datasets

```r
# split original dataset
set.seed(400)
train.sample.df <- sample(nrow(cars.df), nrow(cars.df) * 0.75)
cars.df.training <- cars.df[train.sample.df, ]
cars.df.testing <- cars.df[-train.sample.df, ]
# split dataset without outliers
set.seed(400)
train.sample.no <- sample(nrow(cars.no.df), nrow(cars.no.df) * 0.75)
cars.no.training <- cars.no.df[train.sample.no, ]
cars.no.testing <- cars.no.df[-train.sample.no, ]
# split transformed dataset
set.seed(400)
train.sample.tx <- sample(nrow(cars.tx), nrow(cars.tx) * 0.75)
cars.tx.training <- cars.tx[train.sample.tx, ]
cars.tx.testing <- cars.tx[-train.sample.tx, ]
```

Question 6

Build three full multiple regression models for predicting km-driven: one with cars.training, one with cars.no.training, and one with cars.tx.training, i.e., regression models that contains all features regardless of their p-values. Call the model reg.full, reg.no, and reg.tx.

```r
# build model with full training data
reg.full <- lm(driven ~ ., data = cars.df.training)
# summary the data
summary(reg.full)
```

```
##
## Call:
## lm(formula = driven ~ ., data = cars.df.training)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -123400  -19700   -4715   15794  748699
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)                8.084e+06  3.773e+05  21.423  < 2e-16 ***
## year                      -3.987e+03  1.874e+02 -21.277  < 2e-16 ***
## price                     -8.749e-03  1.552e-03  -5.637 1.88e-08 ***
## fuelDiesel                 2.027e+04  7.286e+03   2.782  0.00543 **
## fuelElectric              -2.771e+04  3.751e+04  -0.739  0.46003
## fuelLPG                    6.178e+03  1.127e+04   0.548  0.58359
## fuelPetrol                -1.226e+04  7.279e+03  -1.684  0.09225 .
## sellerIndividual           9.073e+03  1.639e+03   5.535 3.36e-08 ***
## sellerTrustmark Dealer    -7.940e+02  4.185e+03  -0.190  0.84953
## transmissionManual        -1.052e+03  2.485e+03  -0.423  0.67214
## ownerFourth & Above Owner  9.219e+03  5.058e+03   1.822  0.06847 .
## ownerSecond Owner          7.066e+03  1.643e+03   4.299 1.76e-05 ***
## ownerTest Drive Car       -2.201e+04  1.172e+04  -1.878  0.06054 .
## ownerThird Owner           1.636e+04  2.803e+03   5.837 5.85e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 36680 on 3241 degrees of freedom
## Multiple R-squared:  0.3376, Adjusted R-squared:  0.3349
## F-statistic: 127.1 on 13 and 3241 DF,  p-value: < 2.2e-16
```

```r
# build model with training data without outliers
reg.no <- lm(driven ~ ., data = cars.no.training)
summary(reg.no)
```

```
##
## Call:
## lm(formula = driven ~ ., data = cars.no.training)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -126448  -18259   -2767   16991  113139
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                5.920e+06  3.139e+05  18.857  < 2e-16 ***
## year                      -2.911e+03  1.561e+02 -18.648  < 2e-16 ***
## price                     -1.759e-02  2.423e-03  -7.259 4.92e-13 ***
## fuelDiesel                 1.533e+04  5.153e+03   2.975  0.00296 **
## fuelElectric              -2.109e+04  2.784e+04  -0.758  0.44875
## fuelLPG                    7.145e+03  8.689e+03   0.822  0.41096
## fuelPetrol                -1.203e+04  5.128e+03  -2.346  0.01904 *
## sellerIndividual           8.348e+03  1.260e+03   6.627 4.05e-11 ***
## sellerTrustmark Dealer    -6.242e+02  3.415e+03  -0.183  0.85496
## transmissionManual        -1.141e+03  2.040e+03  -0.559  0.57597
## ownerFourth & Above Owner  8.106e+03  3.865e+03   2.097  0.03606 *
## ownerSecond Owner          8.235e+03  1.264e+03   6.517 8.36e-11 ***
## ownerTest Drive Car       -1.874e+04  7.196e+03  -2.604  0.00925 **
## ownerThird Owner           1.357e+04  2.188e+03   6.205 6.21e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 27280 on 3026 degrees of freedom
## Multiple R-squared:  0.393,  Adjusted R-squared:  0.3904
```

```
## F-statistic: 150.7 on 13 and 3026 DF,  p-value: < 2.2e-16
```

```r
# build model with transformed data
reg.tx <- lm(driven ~ ., data = cars.tx.training)
summary(reg.tx)
```

```
##
## Call:
## lm(formula = driven ~ ., data = cars.tx.training)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -343.66  -36.27    2.94   40.16  207.49
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)              14973.3904   793.5805  18.868  < 2e-16 ***
## year                        -7.2836     0.4045 -18.005  < 2e-16 ***
## price                       -7.7591     2.4763  -3.133  0.00174 **
## fuelDiesel                  28.4630    11.3156   2.515  0.01194 *
## fuelElectric               -32.8389    61.0368  -0.538  0.59060
## fuelLPG                     12.6067    19.0436   0.662  0.50803
## fuelPetrol                 -27.4562    11.2399  -2.443  0.01463 *
## sellerIndividual            15.2944     2.7765   5.509 3.92e-08 ***
## sellerTrustmark Dealer      -7.6169     7.4844  -1.018  0.30890
## transmissionManual           7.3625     4.3850   1.679  0.09325 .
## ownerFourth & Above Owner   12.6251     8.4803   1.489  0.13666
## ownerSecond Owner           17.3960     2.7631   6.296 3.50e-10 ***
## ownerTest Drive Car       -115.0083    15.7190  -7.317 3.24e-13 ***
## ownerThird Owner            27.3606     4.7953   5.706 1.27e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 59.78 on 3026 degrees of freedom
## Multiple R-squared:  0.4031, Adjusted R-squared:  0.4005
## F-statistic: 157.2 on 13 and 3026 DF,  p-value: < 2.2e-16
```

Question 7

Build three ideal multiple regression models for cars.training, cars.no.training, and cars.tx.training using backward elimination based on p-value for predicting km-driven.

7.1. Compare between features

```r
anova(reg.full)
```

```
## Analysis of Variance Table
##
## Response: driven
##                 Df     Sum Sq    Mean Sq  F value    Pr(>F)
```

```
## year            1 1.2538e+12 1.2538e+12 931.8124 < 2.2e-16 ***
## price           1 3.0784e+09 3.0784e+09   2.2878    0.1305
## fuel            4 8.2990e+11 2.0748e+11 154.1918 < 2.2e-16 ***
## seller          2 7.4399e+10 3.7200e+10  27.6460 1.244e-12 ***
## transmission    1 8.7143e+08 8.7143e+08   0.6476    0.4210
## owner           4 6.0585e+10 1.5146e+10  11.2564 4.525e-09 ***
## Residuals    3241 4.3610e+12 1.3456e+09
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova(reg.no)
```

```
## Analysis of Variance Table
##
## Response: driven
##                Df     Sum Sq    Mean Sq   F value    Pr(>F)
## year            1 8.1060e+11 8.1060e+11 1089.5099 < 2.2e-16 ***
## price           1 9.5192e+08 9.5192e+08    1.2795    0.2581
## fuel            4 5.3526e+11 1.3382e+11  179.8580 < 2.2e-16 ***
## seller          2 5.8774e+10 2.9387e+10   39.4981 < 2.2e-16 ***
## transmission    1 9.8292e+08 9.8292e+08    1.3211    0.2505
## owner           4 5.1355e+10 1.2839e+10   17.2562 5.257e-14 ***
## Residuals    3026 2.2514e+12 7.4401e+08
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova(reg.tx)
```

```
## Analysis of Variance Table
##
## Response: driven
##                Df   Sum Sq Mean Sq  F value    Pr(>F)
## year            1  4339952 4339952 1214.273 < 2.2e-16 ***
## price           1    57787   57787   16.168 5.938e-05 ***
## fuel            4  2259783  564946  158.066 < 2.2e-16 ***
## seller          2   259902  129951   36.359 2.490e-16 ***
## transmission    1     3313    3313    0.927    0.3357
## owner           4   383062   95765   26.794 < 2.2e-16 ***
## Residuals    3026 10815278    3574
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From the tables above, for all three models, price and transmission are not affecting the predictions a lot. Next I will try to remove the one with higher p value first, then try to remove both see which could improve the model more.

7.2 Try different eliminations based on p-value

```r
# full data
# remove transmission first since it has higher p value (0.98 vs. 0.15)
```

```r
reg.full.1 <- lm(driven ~ year + price + fuel + seller + owner,
                 data = cars.df.training)
summary(reg.full.1)$r.squared
```

```
## [1] 0.3375659
```

```r
# remove both transmission and price
reg.full.2 <- lm(driven ~ year + fuel + seller + owner,
                 data = cars.df.training)
summary(reg.full.2)$r.squared
```

```
## [1] 0.3294548
```

```r
# data without outliers
# remove price first since it has higher p value (0.46 vs. 0.13)
reg.no.1 <- lm(driven ~ year + fuel + seller + transmission + owner,
               data = cars.no.training)
summary(reg.no.1)$r.squared
```

```
## [1] 0.3824773
```

```r
# remove both transmission and price
reg.no.2 <- lm(driven ~ year + fuel + seller + owner,
               data = cars.no.training)
summary(reg.no.2)$r.squared
```

```
## [1] 0.3816934
```

```r
# transformed data
# remove transmission first since it has higher p value (0.91 vs. 0.10)
reg.tx.1 <- lm(driven ~ year + price + fuel + seller + owner,
               data = cars.tx.training)
summary(reg.tx.1)$r.squared
```

```
## [1] 0.4025439
```

```r
# remove both transmission and price
reg.tx.2 <- lm(driven ~ year + fuel + seller + transmission + owner,
               data = cars.tx.training)
summary(reg.tx.2)$r.squared
```

```
## [1] 0.4011633
```

From the summary tables above, for all three datasets, the first model of each has a slightly higher r squared, indicating they fit each of their dataset better. Next I will further check AIC values.

7.3 Compare between eliminated models for each data set

```r
# compare two models for full data
model.sel(reg.full.1, reg.full.2)
```

```
## Model selection table
##            (Intrc) fuel owner     price sellr  year             family df
## reg.full.1 8096000    +     + -0.008414     + -3994 gaussian(identity) 14
## reg.full.2 8879000    +     +               + -4385 gaussian(identity) 13
##               logLik    AICc delta weight
## reg.full.1 -38821.90 77671.9   0.0      1
## reg.full.2 -38841.71 77709.5  37.6      0
## Models ranked by AICc(x)
```

```r
# compare two models for data without outliers
model.sel(reg.no.1, reg.no.2)
```

```
## Model selection table
##          (Intrc) fuel owner sellr trnsm  year             family df    logLik
## reg.no.1 7056000    +     +     +     + -3481 gaussian(identity) 14 -35382.69
## reg.no.2 7097000    +     +     +       -3500 gaussian(identity) 13 -35384.62
##             AICc delta weight
## reg.no.1 70793.5  0.00  0.715
## reg.no.2 70795.4  1.84  0.285
## Models ranked by AICc(x)
```

```r
# compare two models for transformed data
model.sel(reg.tx.1, reg.tx.2)
```

```
## Model selection table
##          (Intrc) fuel owner  price sellr   year trnsm             family df
## reg.tx.1   14810    +     + -8.907     + -7.189       gaussian(identity) 14
## reg.tx.2   16610    +     +            + -8.148     + gaussian(identity) 14
##             logLik    AICc delta weight
## reg.tx.1 -16743.81 33515.8  0.00  0.971
## reg.tx.2 -16747.32 33522.8  7.02  0.029
## Models ranked by AICc(x)
```

From the result, for all three datasets, the first model of each is better since they have higher weight and lower AIC. Combined with the r squared results from step 7.2, the final models should be the first model of each.

7.4 Make decision of ideal models

```r
reg.full.ideal <- reg.full.1
reg.no.ideal <- reg.no.1
reg.tx.ideal <- reg.tx.1
# print the formula
formula(reg.full.ideal)
```

```
## driven ~ year + price + fuel + seller + owner
```

```r
formula(reg.no.ideal)
```

```
## driven ~ year + fuel + seller + transmission + owner
```

```r
formula(reg.tx.ideal)
```

```
## driven ~ year + price + fuel + seller + owner
```

Question 8

Provide an analysis of the six models (using their respective testing data sets), including Adjusted R-Squared and RMSE. Which of these models is the best? Why?

8.1 Make predictions using these six models

```r
# models with all features
pred.full <- reg.full %>% predict(cars.df.testing)
pred.no <- reg.no %>% predict(cars.no.testing)
pred.tx <- reg.tx %>% predict(cars.tx.testing)
# models after elimination
pred.full.ideal <- reg.full.ideal %>% predict(cars.df.testing)
pred.no.ideal <- reg.no.ideal %>% predict(cars.no.testing)
pred.tx.ideal <- reg.tx.ideal %>% predict(cars.tx.testing)
```

8.2 Overall comparison between 6 models

```r
summary(cars.df.testing$driven)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     101   35000   60000   68566   90000  560000
```

```r
summary(pred.full)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -44634   46775   67254   66157   84477  149243
```

```r
summary(pred.full.ideal)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -42676   46776   67321   66181   84559  149279
```

```r
summary(cars.no.testing$driven)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1000   34125   58341   61833   90000  156040
```

```r
summary(pred.no)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   -5885   44876   62298   62110   78510  129568
```

```r
summary(pred.no.ideal)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   -4157   44839   63327   62297   77260  131415
```

```r
summary(cars.tx.testing$driven)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   31.62  184.73  241.54  236.59  300.00  395.02
```

```r
summary(pred.tx)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##    24.3   200.8   238.0   237.3   271.7   389.8
```

```r
summary(pred.tx.ideal)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   30.92  199.99  237.46  237.41  272.09  390.04
```

From this comparison table, for full data set, both pred.full and pred.ideal were not able to capture the extrame values, and on the lower end, they make negative predictions which doesn't make sense in real life. For data without outliers, both models did fairly well from Q1 to the Q3, but same as full data models, they made negative predictions on the lower end. For transformed data, both models did pretty good on the entire data set predictions. Next I will further compare adjusted r squraed value, MAE (mean absolute error), and RMSE (rooted mean squared error). Since tx data was transformed so they are on a different scale compared to full data and data without outliers, I will also compare RSE (residual standard error) to assess their accuracy.

8.3 Compare adjusted r squared, MAE, RMSE, and RSE

```r
# create function for MAE calculation
MAE <- function(actual, predicted){
  mean(abs(actual - predicted))
}
# create function for RMSE calculation
RMSE <- function(actual, predicted){
  sqrt(mean(actual - predicted) ^ 2)
}
# create function for RSE calculation
RSE <- function(model, dataset){
  sigma(model) / mean(dataset$driven)
}
```

```r
# create a table put everything together
data.frame(model = c("full", "full.ideal", "no", "no.ideal", "tx", "tx.ideal"),
           AdjR2 = c(summary(reg.full)$adj.r.squared,
                     summary(reg.full.ideal)$adj.r.squared,
                     summary(reg.no)$adj.r.squared,
                     summary(reg.no.ideal)$adj.r.squared,
                     summary(reg.tx)$adj.r.squared,
                     summary(reg.tx.ideal)$adj.r.squared),
           MAE = c(MAE(cars.df.testing$driven, pred.full),
                   MAE(cars.df.testing$driven, pred.full.ideal),
                   MAE(cars.no.testing$driven, pred.no),
                   MAE(cars.no.testing$driven, pred.no.ideal),
                   MAE(cars.tx.testing$driven, pred.tx),
                   MAE(cars.tx.testing$driven, pred.tx.ideal)),
           RMSE = c(RMSE(cars.df.testing$driven, pred.full),
                    RMSE(cars.df.testing$driven, pred.full.ideal),
                    RMSE(cars.no.testing$driven, pred.no),
                    RMSE(cars.no.testing$driven, pred.no.ideal),
                    RMSE(cars.tx.testing$driven, pred.tx),
                    RMSE(cars.tx.testing$driven, pred.tx.ideal)),
           RSE = c(RSE(reg.full, cars.df.testing),
                   RSE(reg.full.ideal, cars.df.testing),
                   RSE(reg.no, cars.no.testing),
                   RSE(reg.no.ideal, cars.no.testing),
                   RSE(reg.tx, cars.tx.testing),
                   RSE(reg.tx.ideal, cars.tx.testing)))
```

```
##        model     AdjR2         MAE         RMSE       RSE
## 1       full 0.3349456 26511.03837 2408.9166974 0.5349863
## 2 full.ideal 0.3351140 26495.97516 2385.4613887 0.5349186
## 3         no 0.3904400 19881.70857  277.6965953 0.4411345
## 4   no.ideal 0.3800293 20075.89576  464.1505100 0.4448856
## 5         tx 0.4005356    43.05276    0.7497174 0.2526908
## 6   tx.ideal 0.4001754    43.11965    0.8199734 0.2527667
```

From the table above. Models trained with full data set (full and full.ideal) have the lowest adjusted r squared values with the highest MAE and RMSE, meaning they are not ideal models. Comparing between models trained with dataset without outliers (no and no.ideal) and models trained with transformed dataset (tx and tx.ideal), the models trained with transformed dataset (tx and tx.ideal) have the highest adjusted r squared values meaning they fit the data better. Since tx and tx.ideal are trained with transformed data, it is not comparable for MAE and RMSE values. Therefore, I use RSE values to compare between no and tx models, which gives a measure of error of prediction. Since tx and tx.ideal have lower RSE, meaning they have higher accuracy compare to no and no.ideal, I then only make selection between tx and tx.ideal. Given that tx has a silghtly higher adjusted r squared value and slightly lower RSE and RMSE, tx should be the best model.

Question 9

Using each of the prediction models, what are the predicted odometer readings (km_driven) of a 2004 vehicle that was sold by a dealer for 87,000, has a Diesel engine, a manual transmission, and is second owner? Why are the predictions different?

9.1 Create a data frame for new data

```
# new data frame
new.car <- data.frame("year" = 2004, "price" = 87000,
                      "driven" = NA, "fuel" = "Diesel", "seller" = "Dealer",
                      "transmission" = "Manual", "owner" = "Second Owner")
# convert column types
sapply(new.car, class)
```

```
##         year        price       driven         fuel       seller transmission
##    "numeric"    "numeric"    "logical"  "character"  "character"  "character"
##        owner
##  "character"
```

```
new.car[ ,4:7] <- lapply(new.car[ , 4:7], factor)
# transform the data as needed
new.car.tx <- new.car
new.car.tx$price <- log(new.car.tx$price)
```

9.2 Make predictions using each model

```
new.car.pred1 <- reg.full %>% predict(new.car, interval = "prediction")
new.car.pred2 <- reg.full.ideal %>% predict(new.car, interval = "prediction")
new.car.pred3 <- reg.no %>% predict(new.car, interval = "prediction")
new.car.pred4 <- reg.no.ideal %>% predict(new.car, interval = "prediction")
new.car.pred5 <- reg.tx %>% predict(new.car.tx, interval = "prediction")
new.car.pred6 <- reg.tx.ideal %>% predict(new.car.tx, interval = "prediction")
# create a data frame with all the predictions
# for tx models, square them
new.car.pred <- data.frame(
  model = c("full", "full.ideal", "no", "no.ideal", "tx", "tx.ideal"),
  predict = c(new.car.pred1[1], new.car.pred2[1], new.car.pred3[1],
              new.car.pred4[1], new.car.pred5[1] ^ 2, new.car.pred6[1] ^ 2))
new.car.pred
```

```
##        model  predict
## 1       full 118855.0
## 2 full.ideal 118940.9
## 3         no 107024.2
## 4   no.ideal 104842.0
## 5         tx 117024.7
## 6   tx.ideal 116956.9
```

The predictions made by models trained from data without outliers (no and no.ideal) were the lowest. In Question 8, I considered full and full.ideal are the least accurate models and tx as the best one. However, in this prediction, results from full and full.ideal model are very similar to tx model.

Question 10

For each of the predictions, calculate the 95% prediction interval for the kilometers driven.

```
# create a data frame with all the predictions and 95% of interval
# predictions and calculations were made from Question 9
# square all numbers generated by tx models
new.car.pred <- new.car.pred %>%
  mutate(lower = c(new.car.pred1[2], new.car.pred2[2],
                   new.car.pred3[2], new.car.pred4[2],
                   new.car.pred5[2] ^ 2, new.car.pred6[2] ^ 2),
         upper = c(new.car.pred1[3], new.car.pred2[3],
                   new.car.pred3[3], new.car.pred4[3],
                   new.car.pred5[3] ^ 2, new.car.pred6[3] ^ 2))
new.car.pred
```

```
##        model  predict    lower    upper
## 1       full 118855.0 46771.50 190938.5
## 2 full.ideal 118940.9 46867.62 191014.2
## 3         no 107024.2 53414.75 160633.7
## 4   no.ideal 104842.0 50779.88 158904.0
## 5         tx 117024.7 50427.42 211247.7
## 6   tx.ideal 116956.9 50367.09 211189.0
```