Homework 2

4375 Machine Learning with Dr. Mazidi

Cris Chou

9/8/2021

This homework gives practice in using linear regression in two parts:

- Part 1 Simple Linear Regression (one predictor)
- Part 2 Multiple Linear Regression (many predictors)

You will need to install package ISLR at the console, not in your script.

Problem 1: Simple Linear Regression

Step 1: Initial data exploration

- Load library ISLR (install.packages() at console if needed)
- Use names() and summary() to learn more about the Auto data set
- Divide the data into 75% train, 25% test, using seed 1234

```
# your code here

#install.packages("ISLR")
df <- ISLR::Auto
names(df)
```

```
## [1] "mpg"          "cylinders"    "displacement" "horsepower"   "weight"
## [6] "acceleration" "year"         "origin"       "name"
```

```
summary(df)
```

```
##       mpg          cylinders      displacement     horsepower        weight
##  Min.   : 9.00   Min.   :3.000   Min.   : 68.0   Min.   : 46.0   Min.   :1613
##  1st Qu.:17.00   1st Qu.:4.000   1st Qu.:105.0   1st Qu.: 75.0   1st Qu.:2225
##  Median :22.75   Median :4.000   Median :151.0   Median : 93.5   Median :2804
##  Mean   :23.45   Mean   :5.472   Mean   :194.4   Mean   :104.5   Mean   :2978
##  3rd Qu.:29.00   3rd Qu.:8.000   3rd Qu.:275.8   3rd Qu.:126.0   3rd Qu.:3615
##  Max.   :46.60   Max.   :8.000   Max.   :455.0   Max.   :230.0   Max.   :5140
##
##   acceleration        year           origin                      name
##  Min.   : 8.00   Min.   :70.00   Min.   :1.000   amc matador       :  5
```

```
##  1st Qu.:13.78   1st Qu.:73.00   1st Qu.:1.000   ford pinto        :  5
##  Median :15.50   Median :76.00   Median :1.000   toyota corolla    :  5
##  Mean   :15.54   Mean   :75.98   Mean   :1.577   amc gremlin       :  4
##  3rd Qu.:17.02   3rd Qu.:79.00   3rd Qu.:2.000   amc hornet        :  4
##  Max.   :24.80   Max.   :82.00   Max.   :3.000   chevrolet chevette:  4
##                                                  (Other)           :365
```

```
set.seed(1234)
i <- sample(1:nrow(df), nrow(df)*0.75, replace=FALSE)
train <- df[i,]
test <- df[-i,]
```

Step 2: Create and evaluate a linear model

- Use the lm() function to perform simple linear regression on the train data with mpg as the response and horsepower as the predictor
- Use the summary() function to evaluate the model
- Calculate the MSE by extracting the residuals from the model like this: mse <- mean(lm1$residuals^2)
- Print the MSE
- Calculate and print the RMSE by taking the square root of MSE

```
# your code here
lm1 <- lm(mpg~horsepower, data=train)
summary(lm1)
```

```
##
## Call:
## lm(formula = mpg ~ horsepower, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.3675  -3.1682  -0.2885   2.8518  17.1357
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 39.648595   0.814676   48.67   <2e-16 ***
## horsepower  -0.156681   0.007276  -21.53   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.853 on 292 degrees of freedom
## Multiple R-squared:  0.6136, Adjusted R-squared:  0.6123
## F-statistic: 463.7 on 1 and 292 DF,  p-value: < 2.2e-16
```

```
mse <- mean(lm1$residuals^2)
print(mse)
```

```
## [1] 23.39176
```

```
rmse <- sqrt(mse)
print(rmse)
```

```
## [1] 4.836502
```

Step 3 (No code. Write your answers in white space)

- Write the equation for the model, y = wx + b, filling in the parameters w, b and variable names x, y
- Is there a strong relationship between horsepower and mpg?
- Is it a positive or negative correlation?
- Comment on the RSE, R^2, and F-statistic, and how each indicates the strength of the model
- Comment on the RMSE and whether it indicates that a good model was created

Equation: y = 39.648595+ x * -.156681 where x is the mpg and y will be the predicted horsepower Correlation: There is a negative correlation the RSE, R^2 don't show a strong correlation since the R^2 is a bit low. The F statistic shows that it is still statistically significant since the P value is low and the F statistic is greater than 1. The RMSE shows that we were off by on average around 4.853 units of horsepower which shows that the created model is ok but not the best.

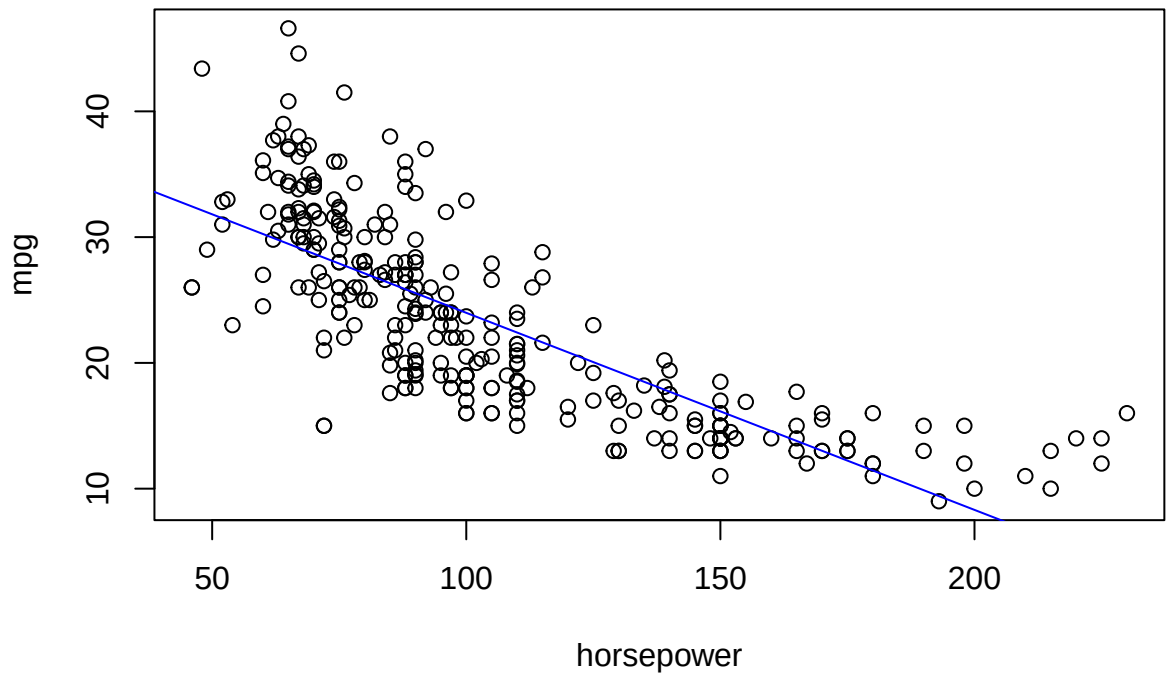
Step 4: Examine the model graphically

- Plot train$mpg~train$horsepower
- Draw a blue abline()
- Comment on how well the data fits the line
- Predict mpg for horsepower of 98. Hint: See the Quick Reference 5.10.3 on page 96
- Comment on the predicted value given the graph you created

Your commentary here: The predicted value 24.29381 is near the actual values of vehicles with a horsepower of 98. Looking at the graph at 98, the predictor line has many values near it and not too many values that are extremely far away.

```
# your code here

plot(train$mpg~train$horsepower,xlab = "horsepower", ylab = "mpg")
abline(lm(train$mpg~train$horsepower), pch=19,col= "blue")
```




```
pred <- predict(lm1,data.frame(horsepower=98))
print(pred)
```

```
##        1
## 24.29381
```

Step 5: Evaluate on the test data

- Test on the test data using the predict function
- Find the correlation between the predicted values and the mpg values in the test data
- Print the correlation
- Calculate the mse on the test results
- Print the mse
- Compare this to the mse for the training data
- Comment on the correlation and the mse in terms of whether the model was able to generalize well to the test data

Your commentary here:

```
# your code here

pred2 <- predict(lm1,newdata=test)
corr <- cor(pred2,test$mpg)
print(corr)
```

```
## [1] 0.7642101
```

```
lm2 <- lm(mpg~horsepower,data=test)
mse2 <- mean(lm2$residuals^2)
print(mse2)
```

```
## [1] 25.25282
```

```
rmse2 <- sqrt(mse2)
print(rmse2)
```

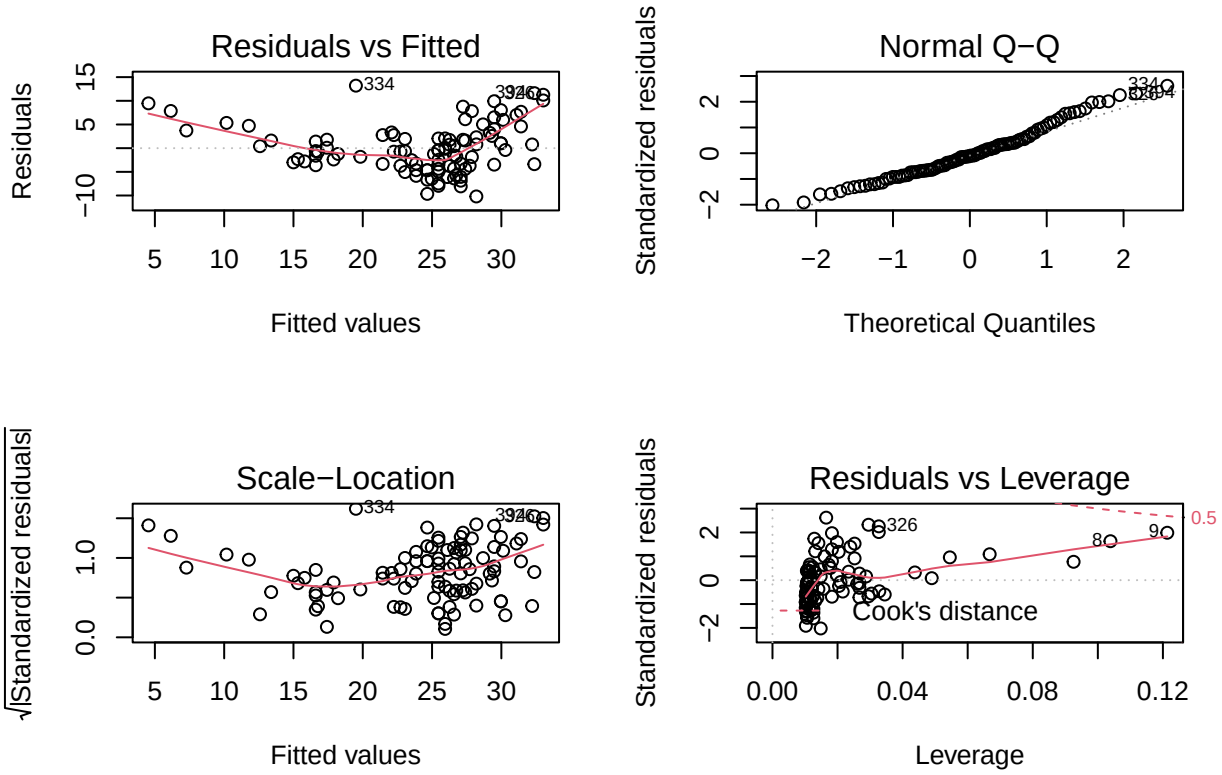
```
## [1] 5.025218
```

Step 6: Plot the residuals

- Plot the linear model in a 2x2 arrangement
- Do you see evidence of non-linearity from the residuals?

Your commentary here: No, the patterns are mostly random which show that the linear model can be a good fit.

```
# your code here
par(mfrow=c(2,2))
plot(lm2)
```



Step 7: Create a second model

- Create a second linear model with log(mpg) predicted by horsepower
- Run summary() on this second model
- Compare the summary statistic R^2 of the two models

Your commentary here: The R^2 for this linear model(lm3) is higher than both lm1 and lm2.

```
# your code here
logMpg <- log(train$mpg)
lm3 <- lm(logMpg~train$horsepower)
summary(lm3)
```

```
##
## Call:
## lm(formula = logMpg ~ train$horsepower)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.62229 -0.12814  0.01443  0.12330  0.61150
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)       3.8631645  0.0319324  120.98   <2e-16 ***
## train$horsepower -0.0074003  0.0002852  -25.95   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1902 on 292 degrees of freedom
## Multiple R-squared:  0.6975, Adjusted R-squared:  0.6965
## F-statistic: 673.3 on 1 and 292 DF,  p-value: < 2.2e-16
```

```
summary(lm2)
```

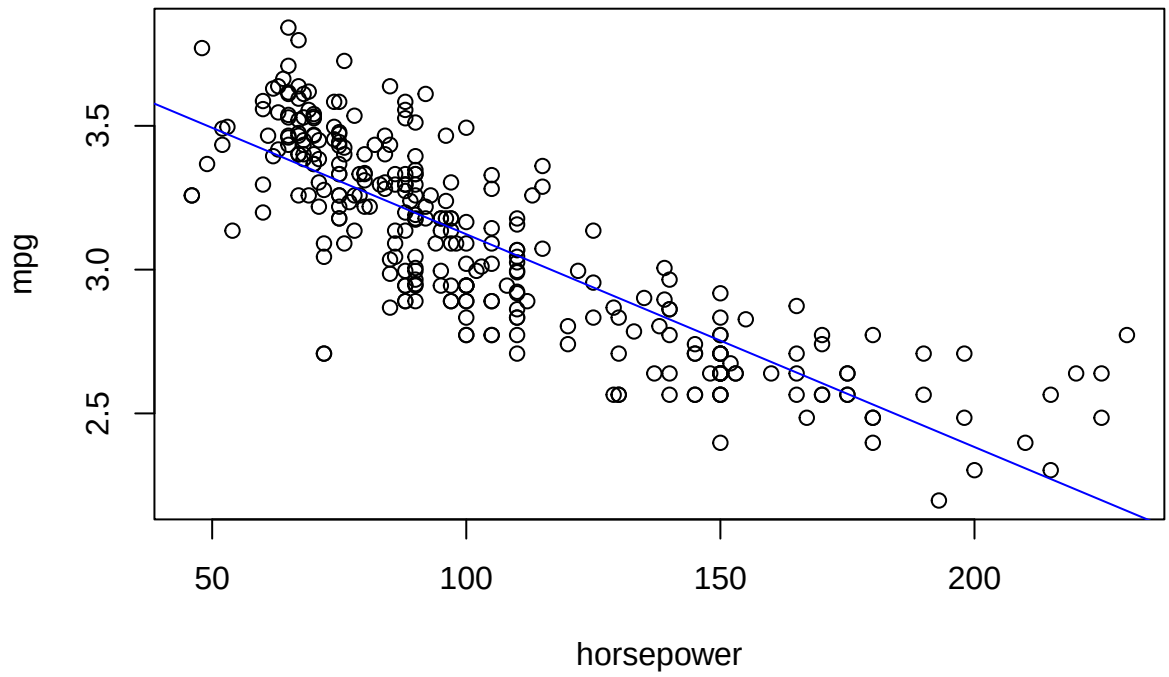
```
##
## Call:
## lm(formula = mpg ~ horsepower, data = test)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.1967  -3.5773  -0.5345   2.7130  13.1946
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 40.75077    1.51608   26.88   <2e-16 ***
## horsepower  -0.16095    0.01386  -11.61   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.077 on 96 degrees of freedom
## Multiple R-squared:  0.584,  Adjusted R-squared:  0.5797
## F-statistic: 134.8 on 1 and 96 DF,  p-value: < 2.2e-16
```

Step 8: Evaluate the second model graphically

- Plot log(train$mpg)~train$horsepower
- Draw a blue abline()
- Comment on how well the line fits the data compared to model 1 above

Your commentary here: This line fits better than the one from model 1.

```
# your code here
plot(logMpg~train$horsepower,xlab = "horsepower", ylab = "mpg")
abline(lm(logMpg~train$horsepower), pch=19,col= "blue")
```



Step 9: Predict and evaluate on the second model

- Predict on the test data using lm2
- Find the correlation of the predictions and log() of test mpg, remembering to compare pred with log(test$mpg)
- Output this correlation
- Compare this correlation with the correlation you got for model
- Calculate and output the MSE for the test data on lm2, and compare to model 1. Hint: Compute the residuals and mse like this:

```
residuals <- pred - log(test$mpg)
mse <- mean(residuals^2)
```

Your commentary here: The correlation is noticeably higher than the other models. The MSE is much higher than model 1.

```
# your code here
pred3 <- predict(lm2,newdata=test)
corr2 <- cor(pred3,log(test$mpg))
print(corr2)
```

```
## [1] 0.814936
```

```
residuals <- pred3 -log(test$mpg)
mse3 <- mean(residuals^2)
print(mse3)
```

```
## [1] 475.6735
```

```
rmse3 <- sqrt(mse3)
print(rmse3)
```

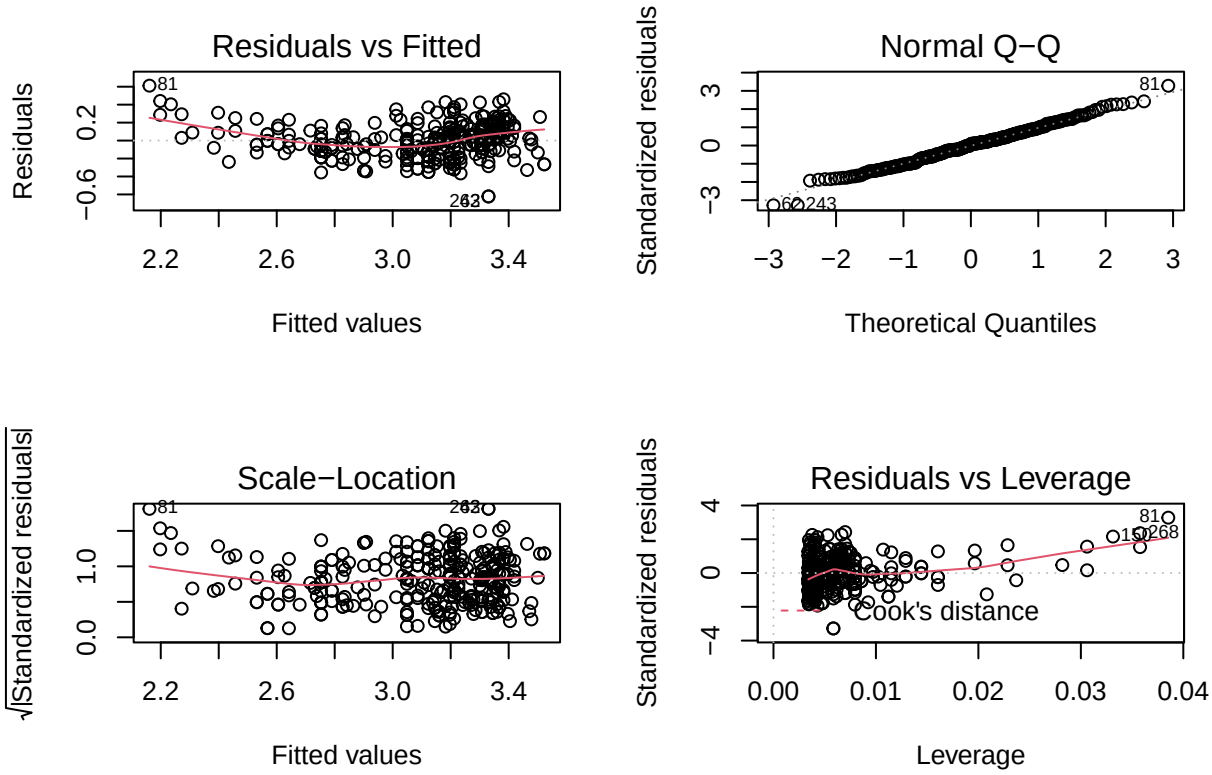
```
## [1] 21.80994
```

Step 10: Plot the residuals of the second model

- Plot the second linear model in a 2x2 arrangement
- How does it compare to the first set of graphs?

Your commentary here: The second set is much better than the first set since the points are more clustered together Visibly seen by having much more concentrated areas resulting in darker, more filled areas on each graph.

```
# your code here
par(mfrow=c(2,2))
plot(lm3)
```



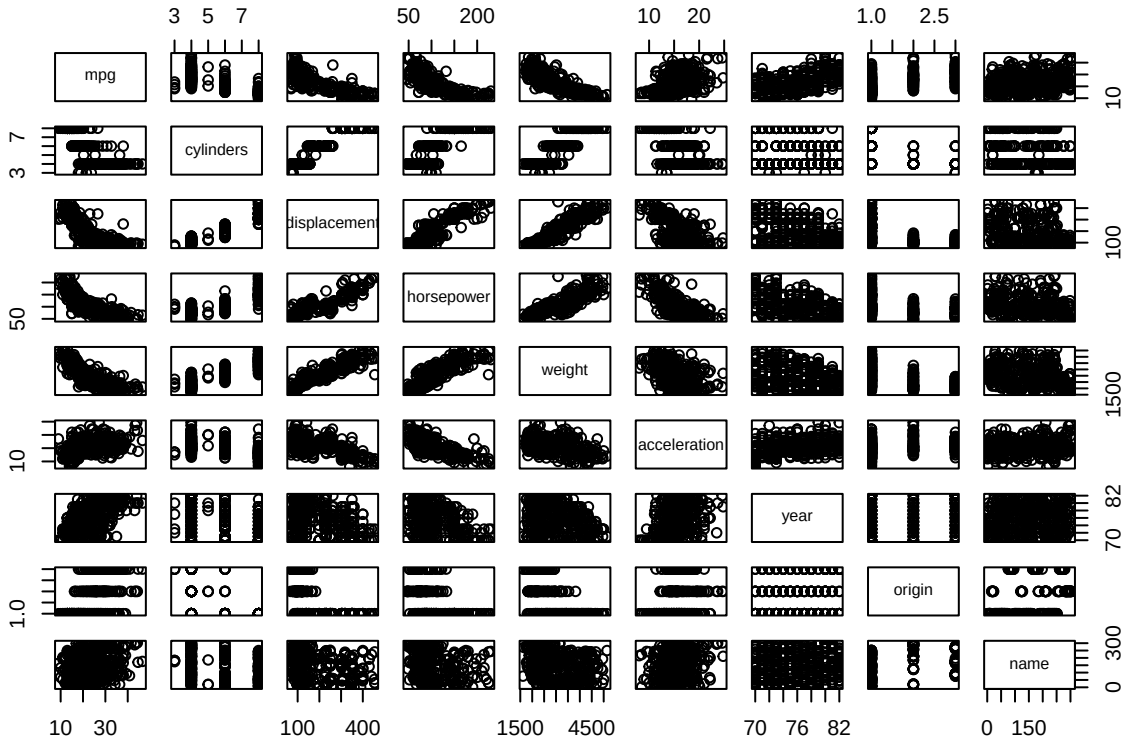
Problem 2: Multiple Linear Regression

Step 1: Data exploration

- Produce a scatterplot matrix of correlations which includes all the variables in the data set using the command "pairs(Auto)"
- List any possible correlations that you observe, listing positive and negative correlations separately, with at least 3 in each category.

Your commentary here: Positive correlations: (horsepower, weight), (weight, displacement), (displacement, horsepower) Negative correlations: (displacement, mpg), (horsepower, acceleration), (weight, mpg)

```
# your code here
pairs(df)
```



Step 2: Data visualization

- Display the matrix of correlations between the variables using function cor(), excluding the "name" variable since is it qualitative
- Write the two strongest positive correlations and their values below. Write the two strongest negative correlations and their values as well.

Your commentary here: Strongest positive correlations: (weight, displacement), (displacement, cylinders) Strongest negative correlation: (displacement, mpg), (mpg, weight)

```
# your code here

cor(df[,c(1:8)])
```

```
##                     mpg  cylinders displacement horsepower     weight
## mpg           1.0000000 -0.7776175   -0.8051269 -0.7784268 -0.8322442
## cylinders    -0.7776175  1.0000000    0.9508233  0.8429834  0.8975273
## displacement -0.8051269  0.9508233    1.0000000  0.8972570  0.9329944
## horsepower   -0.7784268  0.8429834    0.8972570  1.0000000  0.8645377
## weight       -0.8322442  0.8975273    0.9329944  0.8645377  1.0000000
## acceleration  0.4233285 -0.5046834   -0.5438005 -0.6891955 -0.4168392
## year          0.5805410 -0.3456474   -0.3698552 -0.4163615 -0.3091199
## origin        0.5652088 -0.5689316   -0.6145351 -0.4551715 -0.5850054
##              acceleration       year     origin
```

```
## mpg             0.4233285  0.5805410  0.5652088
## cylinders      -0.5046834 -0.3456474 -0.5689316
## displacement   -0.5438005 -0.3698552 -0.6145351
## horsepower     -0.6891955 -0.4163615 -0.4551715
## weight         -0.4168392 -0.3091199 -0.5850054
## acceleration    1.0000000  0.2903161  0.2127458
## year            0.2903161  1.0000000  0.1815277
## origin          0.2127458  0.1815277  1.0000000
```

Step 3: Build a third linear model

- Convert the origin variable to a factor
- Use the lm() function to perform multiple linear regression with mpg as the response and all other variables except name as predictors
- Use the summary() function to print the results
- Which predictors appear to have a statistically significant relationship to the response?

Your commentary here: The ones with statical significance seem to be weight, year, origins, and displacement.

```
# your code here


df$origin <- factor(df$origin)
levels(df$origin) <- c("American","European","Japanese")

lm4 <-  lm(mpg~(cylinders+displacement+horsepower+weight+acceleration+year+origin),data=df)
summary(lm4)
```

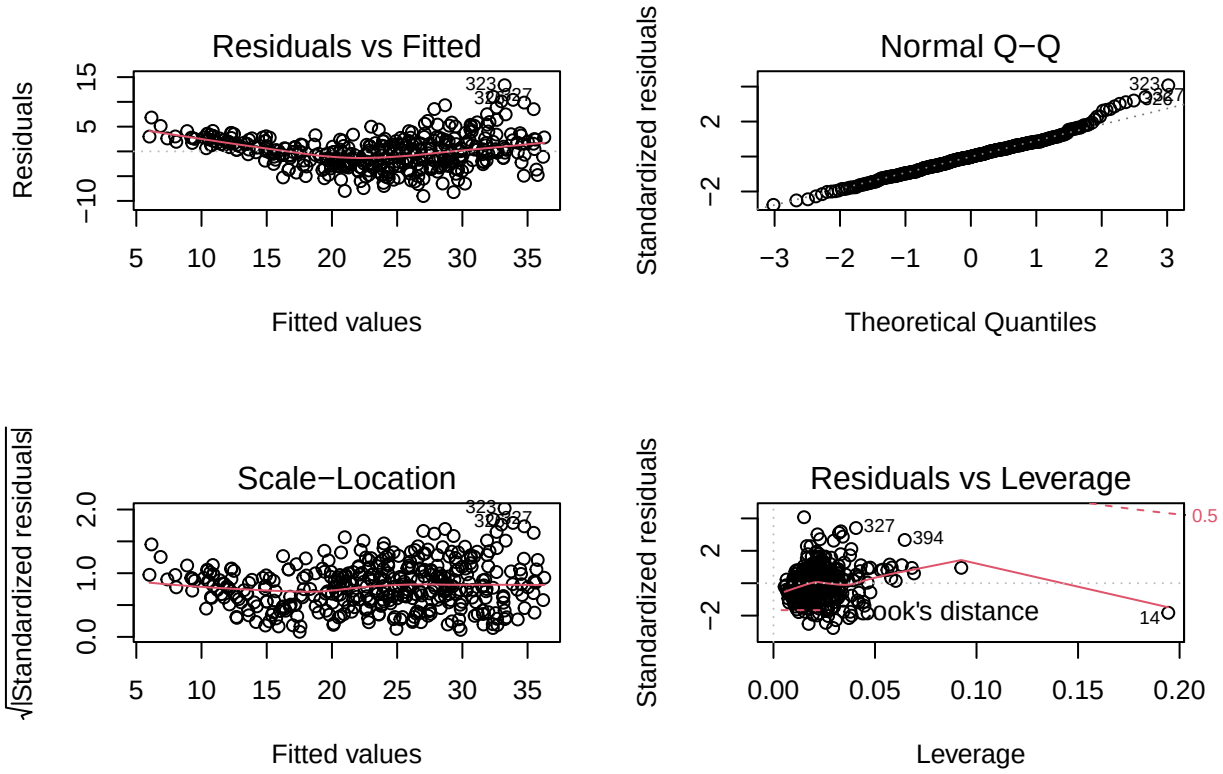
```
##
## Call:
## lm(formula = mpg ~ (cylinders + displacement + horsepower + weight +
##     acceleration + year + origin), data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.0095 -2.0785 -0.0982  1.9856 13.3608
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -1.795e+01  4.677e+00  -3.839 0.000145 ***
## cylinders      -4.897e-01  3.212e-01  -1.524 0.128215
## displacement    2.398e-02  7.653e-03   3.133 0.001863 **
## horsepower     -1.818e-02  1.371e-02  -1.326 0.185488
## weight         -6.710e-03  6.551e-04 -10.243  < 2e-16 ***
## acceleration    7.910e-02  9.822e-02   0.805 0.421101
## year            7.770e-01  5.178e-02  15.005  < 2e-16 ***
## originEuropean  2.630e+00  5.664e-01   4.643 4.72e-06 ***
## originJapanese  2.853e+00  5.527e-01   5.162 3.93e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.307 on 383 degrees of freedom
## Multiple R-squared:  0.8242, Adjusted R-squared:  0.8205
## F-statistic: 224.5 on 8 and 383 DF,  p-value: < 2.2e-16
```

Step 4: Plot the residuals of the third model

- Use the plot() function to produce diagnostic plots of the linear regression fit
- Comment on any problems you see with the fit
- Are there any leverage points?
- Display a row from the data set that seems to be a leverage point.

Your commentary here: Although all the points are close to the line of fit, the patterns for the residuals aren't too random. The leverage points seem to be 320, 323, 327, and 394.

```
# your code here
par(mfrow=c(2,2))
plot(lm4)
```




```
df[c(320,323,327,394),]
```

```
##      mpg cylinders displacement horsepower weight acceleration year   origin
## 322 32.2         4          108         75   2265         15.2   80 Japanese
## 325 40.8         4           85         65   2110         19.2   80 Japanese
## 329 30.0         4          146         67   3250         21.8   80 European
## NA    NA        NA           NA         NA     NA           NA   NA     <NA>
##                   name
## 322     toyota corolla
## 325         datsun 210
## 329 mercedes-benz 240d
## NA                <NA>
```

Step 5: Create and evaluate a fourth model

- Use the * and + symbols to fit linear regression models with interaction effects, choosing whatever variables you think might get better results than your model in step 3 above
- Compare the summaries of the two models, particularly R^2
- Run anova() on the two models to see if your second model outperformed the previous one, and comment below on the results

Your commentary here: My model(lm5) outperformed the previous model(lm4). The R^2 of my model(lm5) was .8807 compared to the previous models' .8242. The results of anova also show a smaller Res.df and RSS.

```
# your code here
lm5 <- lm(mpg~(weight+year+origin+displacement+weight*year*origin*displacement), data=df)
summary(lm5)
```

```
##
## Call:
## lm(formula = mpg ~ (weight + year + origin + displacement + weight *
##     year * origin * displacement), data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.3209 -1.5571  0.0617  1.3664 13.5827
##
## Coefficients:
##                                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)                             -1.584e+02  4.868e+01  -3.255 0.001239
## weight                                   4.849e-02  1.878e-02   2.582 0.010222
## year                                     2.785e+00  6.439e-01   4.326 1.96e-05
## originEuropean                          -7.526e+02  3.061e+02  -2.459 0.014407
## originJapanese                           1.315e+02  2.138e+02   0.615 0.538946
## displacement                             2.857e-01  2.241e-01   1.275 0.203066
## weight:year                             -7.721e-04  2.500e-04  -3.088 0.002169
## weight:originEuropean                    2.124e-01  1.274e-01   1.667 0.096390
## weight:originJapanese                   -1.148e-01  9.217e-02  -1.246 0.213595
## year:originEuropean                      9.880e+00  4.027e+00   2.453 0.014614
## year:originJapanese                     -1.661e+00  2.794e+00  -0.595 0.552463
## weight:displacement                     -1.124e-04  6.114e-05  -1.839 0.066712
## year:displacement                       -4.581e-03  3.029e-03  -1.512 0.131285
## originEuropean:displacement              9.759e+00  2.846e+00   3.428 0.000676
## originJapanese:displacement              2.250e+00  2.362e+00   0.953 0.341432
## weight:year:originEuropean              -2.672e-03  1.662e-03  -1.607 0.108853
## weight:year:originJapanese               1.432e-03  1.202e-03   1.192 0.234210
## weight:year:displacement                 1.722e-06  8.222e-07   2.094 0.036956
## weight:originEuropean:displacement      -3.163e-03  1.081e-03  -2.924 0.003664
## weight:originJapanese:displacement      -4.528e-04  8.901e-04  -0.509 0.611231
## year:originEuropean:displacement        -1.300e-01  3.767e-02  -3.451 0.000623
## year:originJapanese:displacement        -2.796e-02  3.067e-02  -0.911 0.362682
## weight:year:originEuropean:displacement  4.130e-05  1.415e-05   2.919 0.003728
## weight:year:originJapanese:displacement  5.723e-06  1.153e-05   0.496 0.619993
##
## (Intercept)                             **
## weight                                  *
```

```
## year                                     ***
## originEuropean                           *
## originJapanese
## displacement
## weight:year                              **
## weight:originEuropean                    .
## weight:originJapanese
## year:originEuropean                      *
## year:originJapanese
## weight:displacement                      .
## year:displacement
## originEuropean:displacement              ***
## originJapanese:displacement
## weight:year:originEuropean
## weight:year:originJapanese
## weight:year:displacement                 *
## weight:originEuropean:displacement       **
## weight:originJapanese:displacement
## year:originEuropean:displacement         ***
## year:originJapanese:displacement
## weight:year:originEuropean:displacement **
## weight:year:originJapanese:displacement
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.779 on 368 degrees of freedom
## Multiple R-squared:  0.8807, Adjusted R-squared:  0.8733
## F-statistic: 118.1 on 23 and 368 DF,  p-value: < 2.2e-16
```

```
summary(lm4)
```

```
##
## Call:
## lm(formula = mpg ~ (cylinders + displacement + horsepower + weight +
##     acceleration + year + origin), data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.0095 -2.0785 -0.0982  1.9856 13.3608
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -1.795e+01  4.677e+00  -3.839 0.000145 ***
## cylinders      -4.897e-01  3.212e-01  -1.524 0.128215
## displacement    2.398e-02  7.653e-03   3.133 0.001863 **
## horsepower     -1.818e-02  1.371e-02  -1.326 0.185488
## weight         -6.710e-03  6.551e-04 -10.243  < 2e-16 ***
## acceleration    7.910e-02  9.822e-02   0.805 0.421101
## year            7.770e-01  5.178e-02  15.005  < 2e-16 ***
## originEuropean  2.630e+00  5.664e-01   4.643 4.72e-06 ***
## originJapanese  2.853e+00  5.527e-01   5.162 3.93e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 3.307 on 383 degrees of freedom
## Multiple R-squared:  0.8242, Adjusted R-squared:  0.8205
## F-statistic: 224.5 on 8 and 383 DF,  p-value: < 2.2e-16
```

```
anova(lm5,lm4)
```

```
## Analysis of Variance Table
##
## Model 1: mpg ~ (weight + year + origin + displacement + weight * year *
##     origin * displacement)
## Model 2: mpg ~ (cylinders + displacement + horsepower + weight + acceleration +
##     year + origin)
##   Res.Df    RSS  Df Sum of Sq      F    Pr(>F)
## 1    368 2841.1
## 2    383 4187.4 -15   -1346.3 11.625 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```