Matthew Bradley
Ayanna Fisher
Ben Howell
Hayley Zorkic

HW 2

Question 8

```
require(tidyverse)
```

```
## Loading required package: tidyverse
```

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

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.1.1     v forcats 0.5.1
```

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

```
require(ISLR2)
```

```
## Loading required package: ISLR2
```

```
df <- Auto

model <- lm(mpg ~ horsepower, data = df)

summary(model)
```

Creating lm

```
##
## Call:
## lm(formula = mpg ~ horsepower, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.5710  -3.2592  -0.3435   2.7630  16.9240
##
## Coefficients:
```

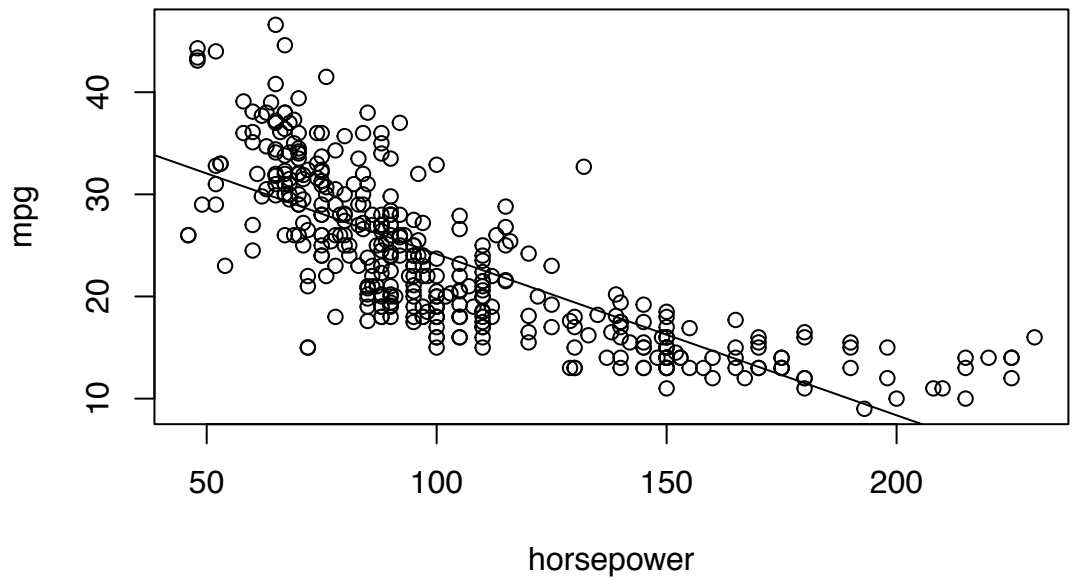
```
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 39.935861   0.717499   55.66   <2e-16 ***
## horsepower  -0.157845   0.006446  -24.49   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.906 on 390 degrees of freedom
## Multiple R-squared:  0.6059, Adjusted R-squared:  0.6049
## F-statistic: 599.7 on 1 and 390 DF,  p-value: < 2.2e-16
```

There is a clear relationship between `horsepower` and `mpg` in the `Auto` dataset. The `horsepower` coefficient of `-0.16` indicates that for every one additional unit of `horsepower` corresponds with less `mpg`. The negative relationship is fairly strong, though it is interesting that it's more of a non-linear relationship at higher `horsepower` values, where the difference between 150 units of `horsepower` and 200 units of `horsepower` is fairly small, whereas the difference between 75 units and 100 units is far more pronounced.

```r
new <- data.frame(horsepower = c(98))

pred <- predict(model, newdata = new, interval = "confidence") %>%
  data.frame()
```

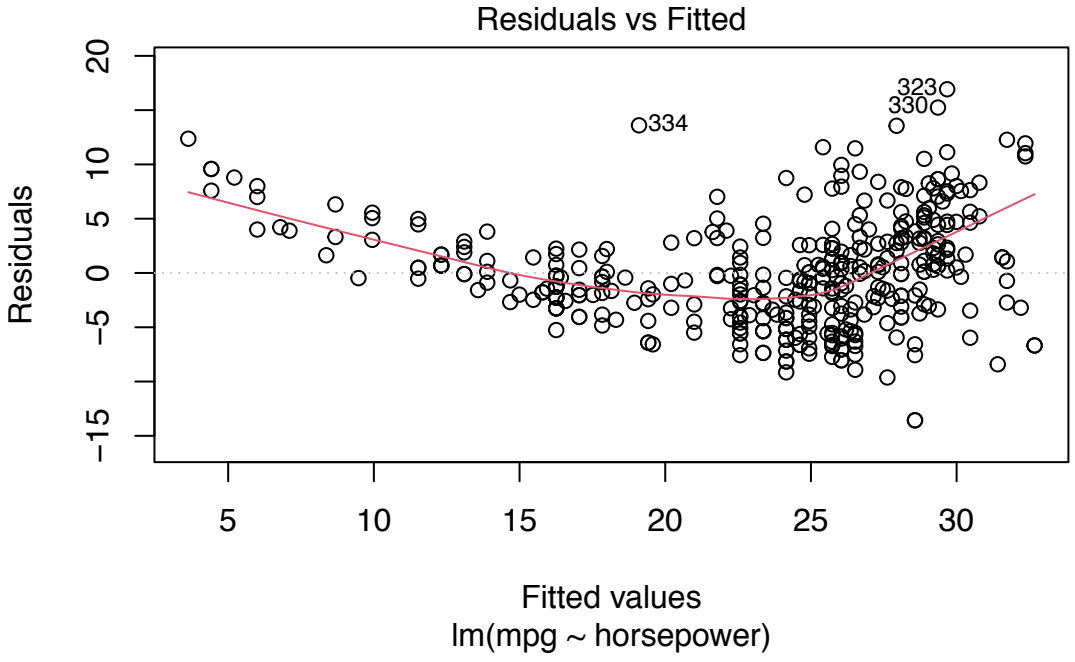
The predicted `mpg` of a 98 `horsepower` is 24.4670772, while the confidence interval ranges from 23.973079 to 24.9610753. Below is a plot of `mpg` as a function of `horsepower`, with the least squares regression line plotted over it.

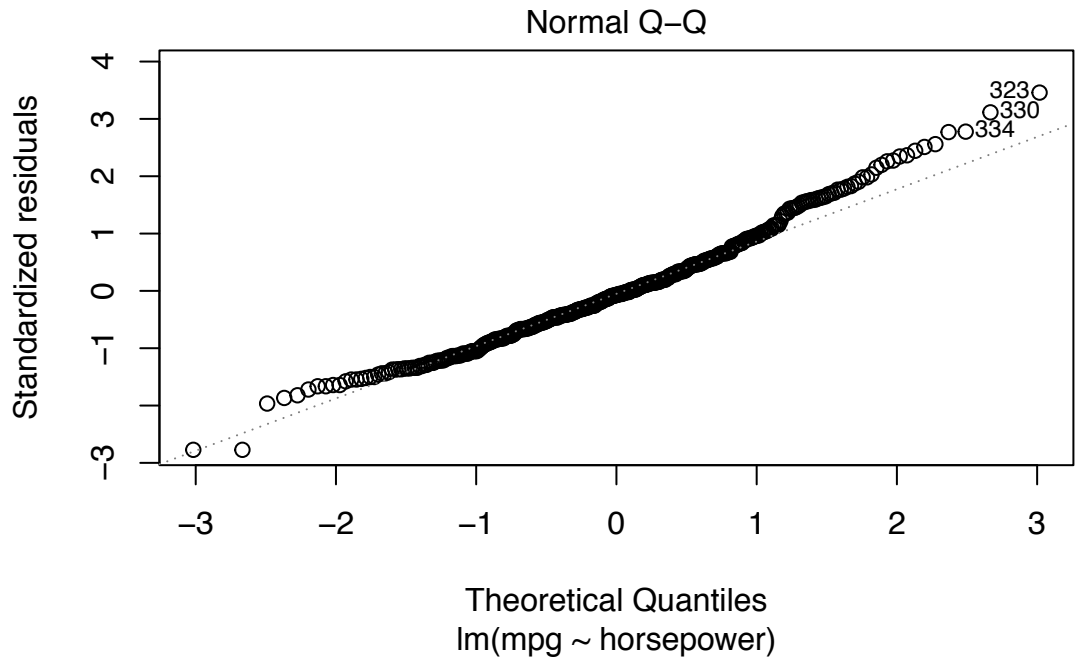
```r
# df %>%
#   ggplot(aes(x = horsepower, y = mpg)) +
#   geom_point() +
#   geom_smooth(method = "lm", formula = mpg ~ horsepower) +
#   theme_minimal() +
#   labs(title = "Horsepower vs MPG") +
#   theme(plot.title = element_text(hjust = 0.5, size = 14, face = "bold"))

with(df, plot(horsepower, mpg)) +
abline(model)
```

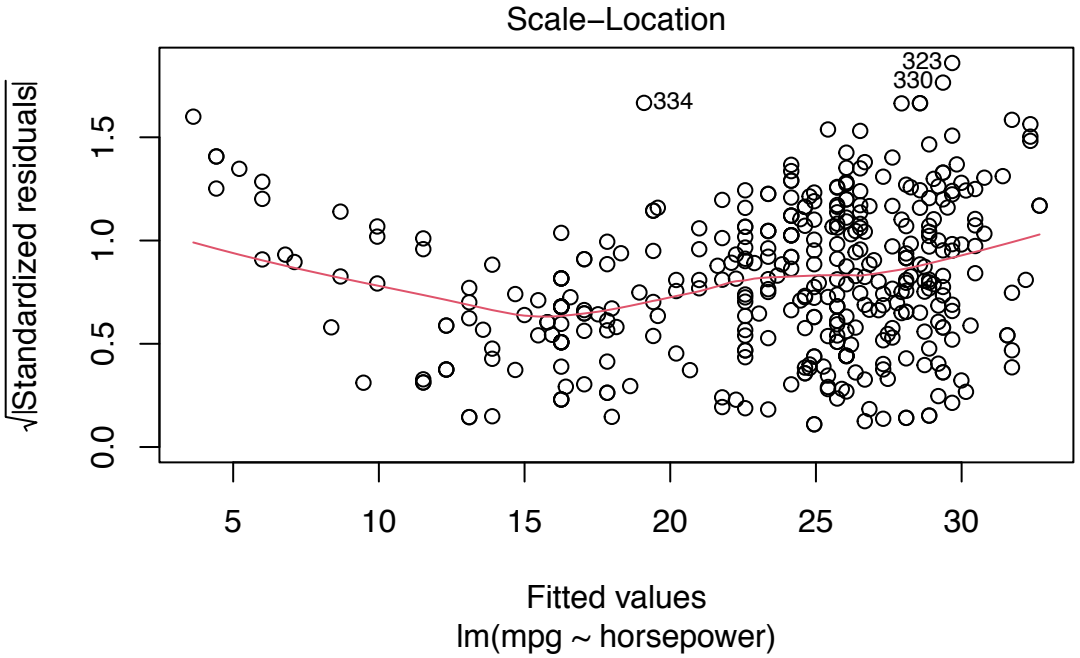


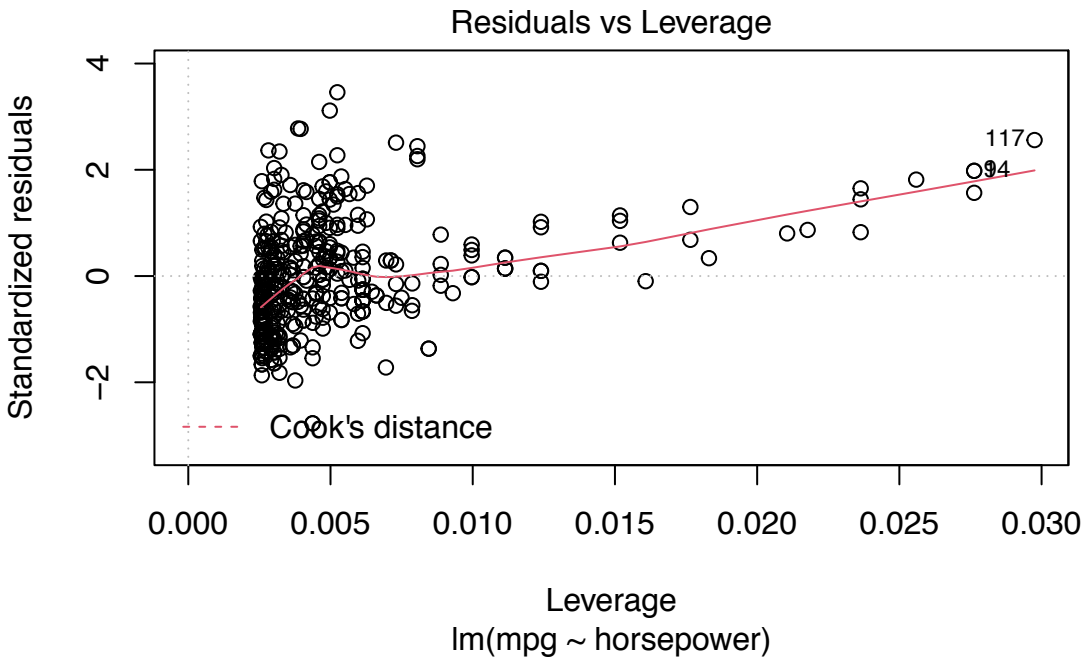

```
## integer(0)
```

```
plot(model)
```




Normal Q−Q
Standardized residuals
323
330
334
−3
−2
−1
0
1
2
3
4
Theoretical Quantiles
lm(mpg ~ horsepower)


Scale−Location
√|Standardized residuals|
323
330
334
0.0
0.5
1.0
1.5
5
10
15
20
25
30
Fitted values
lm(mpg ~ horsepower)

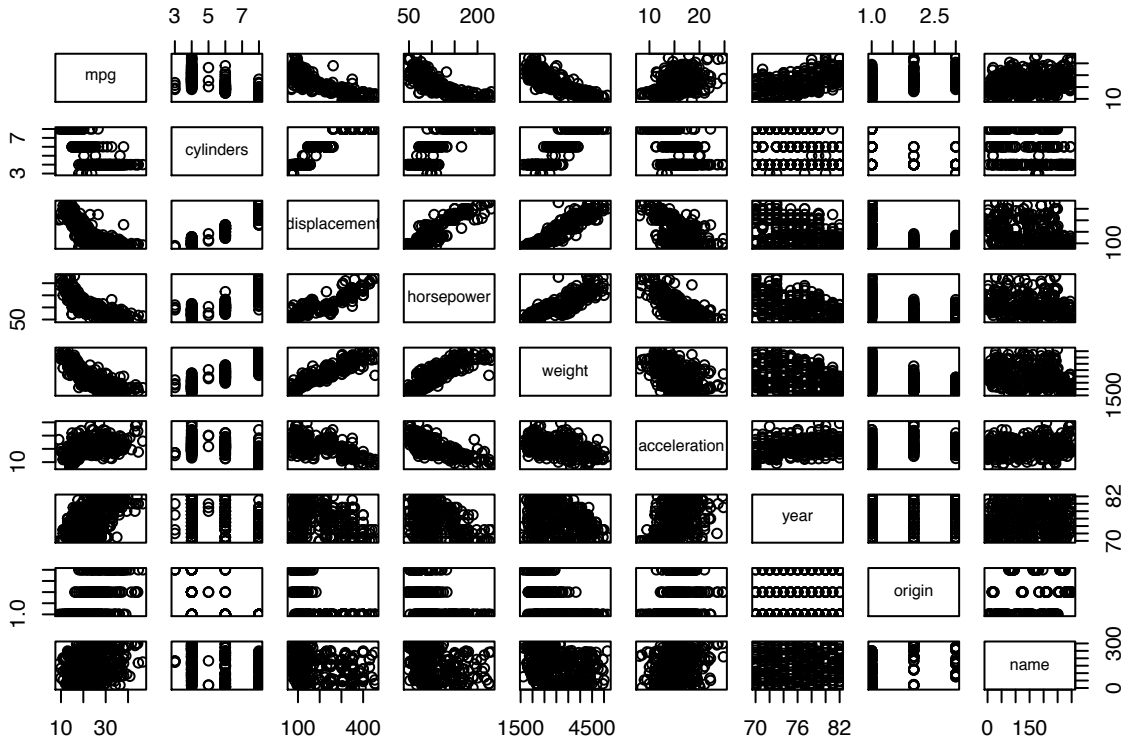


From evaluating the diagnostic plots, it's clear that the linear model is not a good fit for the current data set. The residuals are not evenly or consistently distributed over the distribution of fitted values. For example, the model consistently under predicts low values of `mpg`. The large positive residual shows that the actual values are well above the prediction. A similar phenomenon occurs at higher values too; given the `horsepower vs mpg` and line of best fit plot, the non-linear relationship between the two variables ends up under-predicting values at both extremes.

Question 9:

A:

```
Auto <- read.csv("/Users/matthewbradley/Downloads/Auto.csv")
Auto$name <- factor(Auto$name)
pairs(Auto)
```



B:

```
cor(Auto[,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
```

C:

```
linearModel <- lm(mpg ~ cylinders + displacement + horsepower + weight + acceleration +
                    year + as.factor(origin), data = Auto)

summary(linearModel)
```

```
##
## Call:
## lm(formula = mpg ~ cylinders + displacement + horsepower + weight +
##     acceleration + year + as.factor(origin), data = Auto)
##
## 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 ***
## as.factor(origin)2  2.630e+00  5.664e-01   4.643 4.72e-06 ***
## as.factor(origin)3  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
```

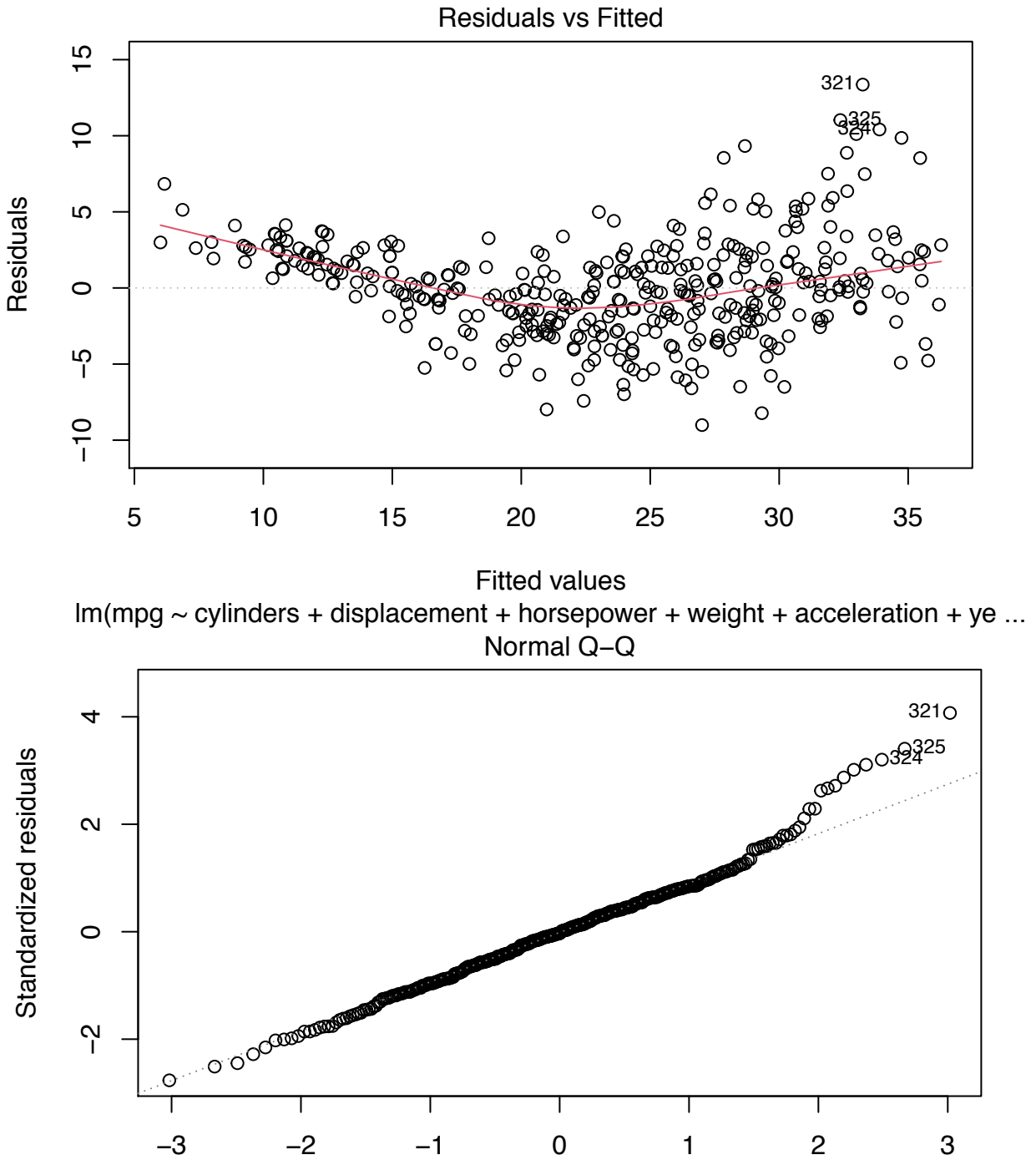
i. There is a relationship between predictor and response. Our R-squared is .8205, meaning we can predict about 82% of the variation of the data using our predictors. The F statistic is also extremely small (<2.2e-16), which allows us to reject the null hypothesis that there is no relationship.

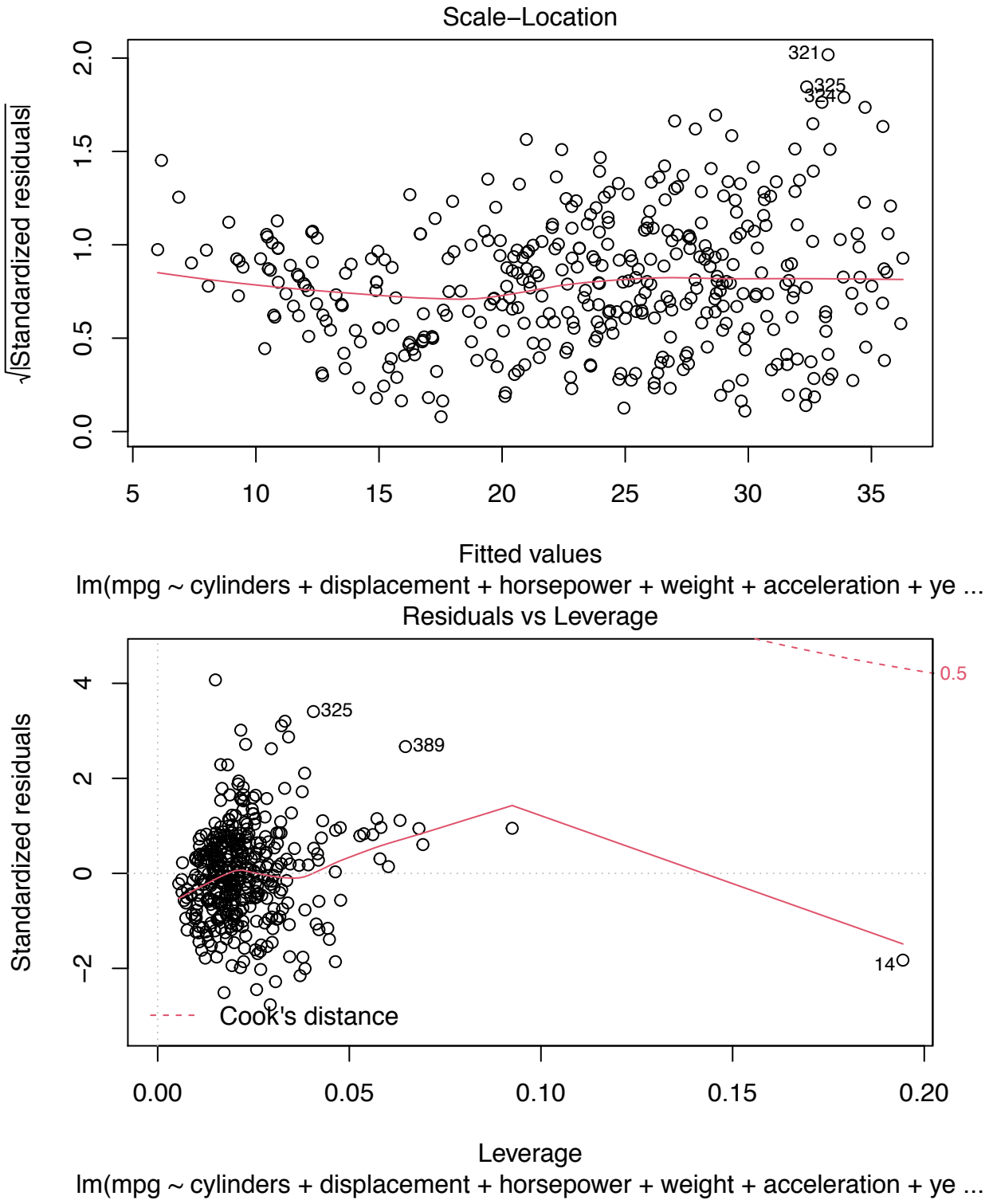
ii. Displacement, weight, year, and origin all appear to have statistically significant relationships with the response ($p<0.05$ for these variables)

iii. The year coefficient (0.777) suggests that for every increase of 1 year, the response variable (mpg) increases by 0.777.

D:

```
plot(linearModel)
```


Residuals vs Fitted
Residuals
Fitted values
lm(mpg ~ cylinders + displacement + horsepower + weight + acceleration + ye ...
321
325
324
Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(mpg ~ cylinders + displacement + horsepower + weight + acceleration + ye ...



The residual plot has a "U" shape, suggesting there could be non linearity in the data. There also do appear to be large outliers on the right hand side of the plot (marked 323, 326, and 327).

The Normal Q-Q plot also shows outliers on the right side, marked with the same numbers. This suggests these data points may be skewing the normality of the data.

The leverage plot suggests that point 14 has high leverage, suggesting it may be a particularly influential

point for our model.

E:

```
linearModel <- lm(mpg ~ cylinders + displacement + horsepower + weight + acceleration + year +
                    as.factor(origin) + cylinders*displacement + cylinders:horsepower +
                    cylinders:weight +cylinders*acceleration + cylinders*year +
                    cylinders:as.factor(origin), data = Auto)

summary(linearModel)
```

```
##
## Call:
## lm(formula = mpg ~ cylinders + displacement + horsepower + weight +
##     acceleration + year + as.factor(origin) + cylinders * displacement +
##     cylinders:horsepower + cylinders:weight + cylinders * acceleration +
##     cylinders * year + cylinders:as.factor(origin), data = Auto)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.4976 -1.7194  0.0678  1.3838 12.0082
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  -4.084e+01  1.515e+01  -2.695  0.00736 **
## cylinders                     5.057e+00  2.709e+00   1.867  0.06271 .
## displacement                  2.243e-02  2.524e-02   0.889  0.37483
## horsepower                   -1.526e-01  5.552e-02  -2.748  0.00628 **
## weight                       -1.185e-02  2.593e-03  -4.569 6.67e-06 ***
## acceleration                  5.202e-02  2.988e-01   0.174  0.86188
## year                          1.441e+00  1.700e-01   8.477 5.33e-16 ***
## as.factor(origin)2            1.451e+00  3.277e+00   0.443  0.65832
## as.factor(origin)3           -3.900e+00  2.837e+00  -1.375  0.16998
## cylinders:displacement       -1.752e-03  3.727e-03  -0.470  0.63859
## cylinders:horsepower          1.592e-02  8.105e-03   1.964  0.05027 .
## cylinders:weight              1.089e-03  3.765e-04   2.893  0.00404 **
## cylinders:acceleration       -5.631e-03  5.417e-02  -0.104  0.91725
## cylinders:year               -1.305e-01  3.180e-02  -4.105 4.96e-05 ***
## cylinders:as.factor(origin)2  1.177e-01  7.636e-01   0.154  0.87755
## cylinders:as.factor(origin)3  1.347e+00  6.563e-01   2.053  0.04080 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.85 on 376 degrees of freedom
## Multiple R-squared:  0.8718, Adjusted R-squared:  0.8667
## F-statistic: 170.4 on 15 and 376 DF,  p-value: < 2.2e-16
```

I examined the interaction effects of the "cylinder" variable with all other variables. There do appear to be several significant interaction effects (with weight, year, and origin).

F:

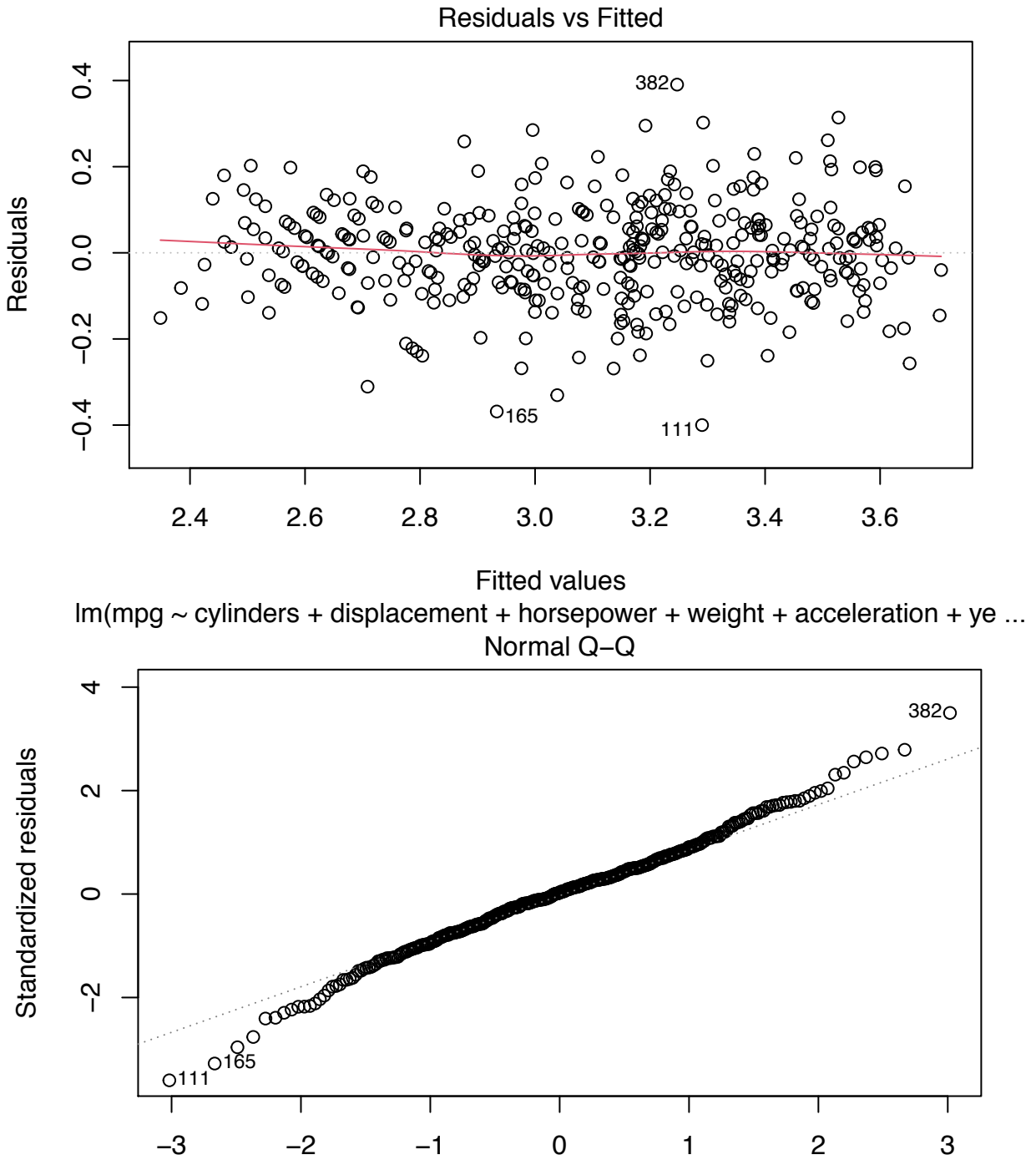
```
loggedData <- log(Auto[1:7])
origin = Auto[,8]
```

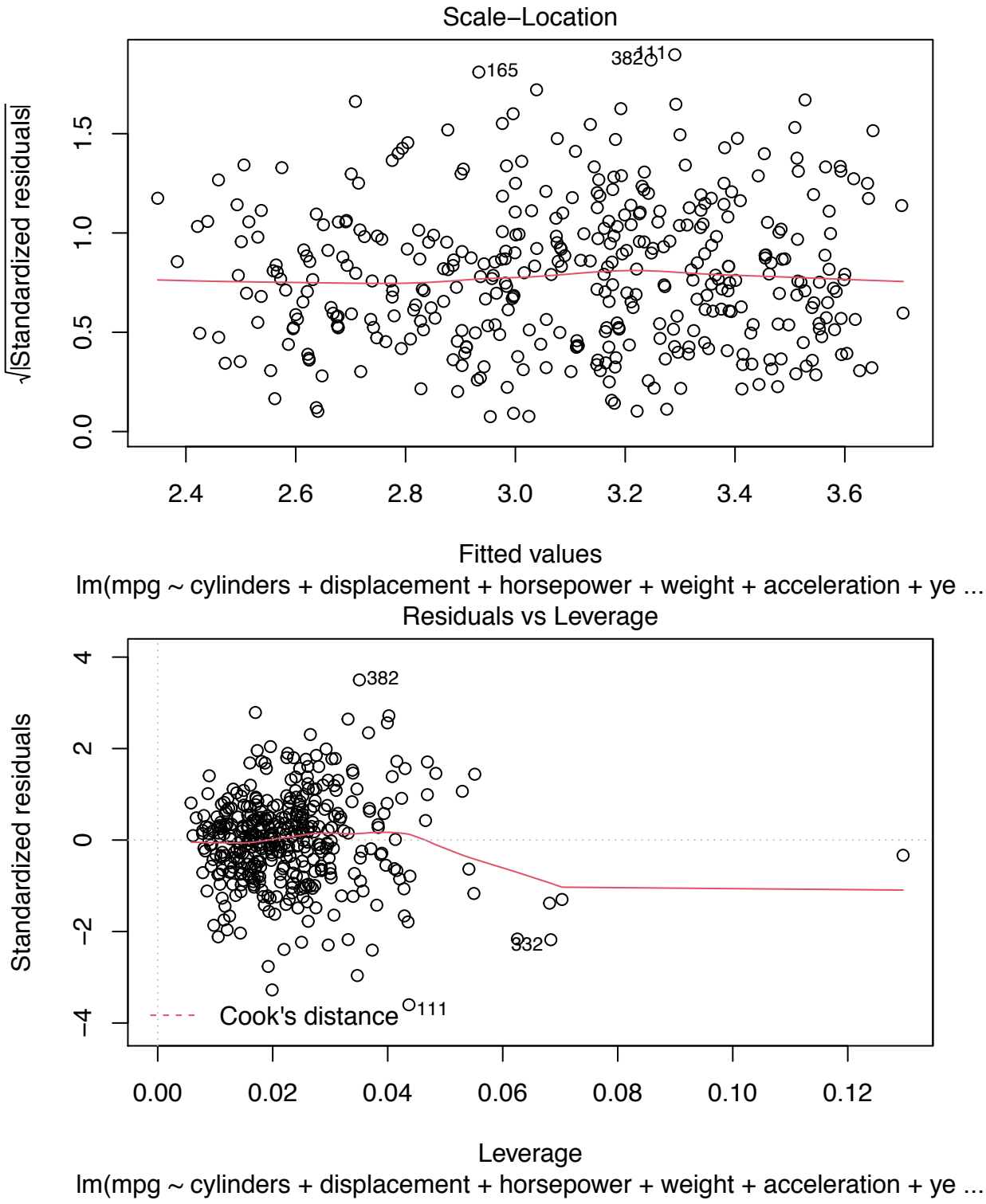
```r
loggedData <- cbind(loggedData, origin)
logModel <- lm(mpg ~ cylinders + displacement + horsepower + weight + acceleration + year +
                 as.factor(origin), data = loggedData)
summary(logModel)
```

Log transformation:

```
##
## Call:
## lm(formula = mpg ~ cylinders + displacement + horsepower + weight +
##     acceleration + year + as.factor(origin), data = loggedData)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.39999 -0.06970  0.00294  0.06304  0.39059
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)        -0.24514    0.65266  -0.376  0.70741
## cylinders          -0.08546    0.06145  -1.391  0.16510
## displacement        0.02303    0.05871   0.392  0.69503
## horsepower         -0.28422    0.05830  -4.875 1.60e-06 ***
## weight             -0.59696    0.08572  -6.964 1.45e-11 ***
## acceleration       -0.17066    0.05998  -2.845  0.00468 **
## year                2.27962    0.13521  16.859  < 2e-16 ***
## as.factor(origin)2  0.05004    0.02103   2.379  0.01785 *
## as.factor(origin)3  0.04736    0.02074   2.284  0.02293 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1136 on 383 degrees of freedom
## Multiple R-squared:  0.8907, Adjusted R-squared:  0.8884
## F-statistic: 390.2 on 8 and 383 DF,  p-value: < 2.2e-16
```

```r
plot(logModel)
```


Residuals vs Fitted
Residuals
Fitted values
lm(mpg ~ cylinders + displacement + horsepower + weight + acceleration + ye ...
382
165
111
Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(mpg ~ cylinders + displacement + horsepower + weight + acceleration + ye ...



We have a slightly higher R-squared for the model using log values, suggesting that it may have helped limit the effects of non linearity and outliers. However, the Q-Q plot shows new outliers as well, so we would need to look deeper to determine if this model is truly better.

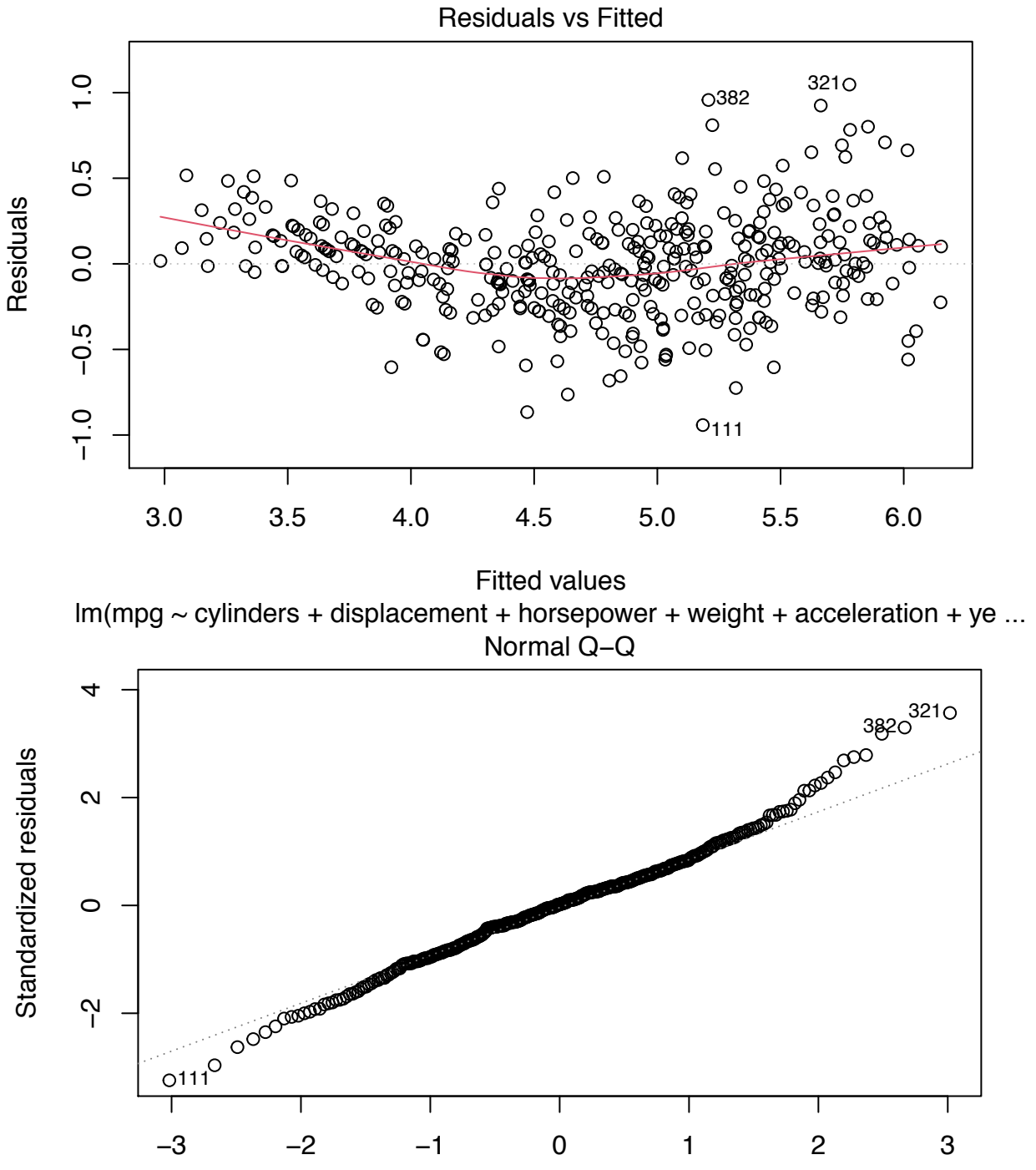
```
squareRootData <- sqrt(Auto[1:7])
```

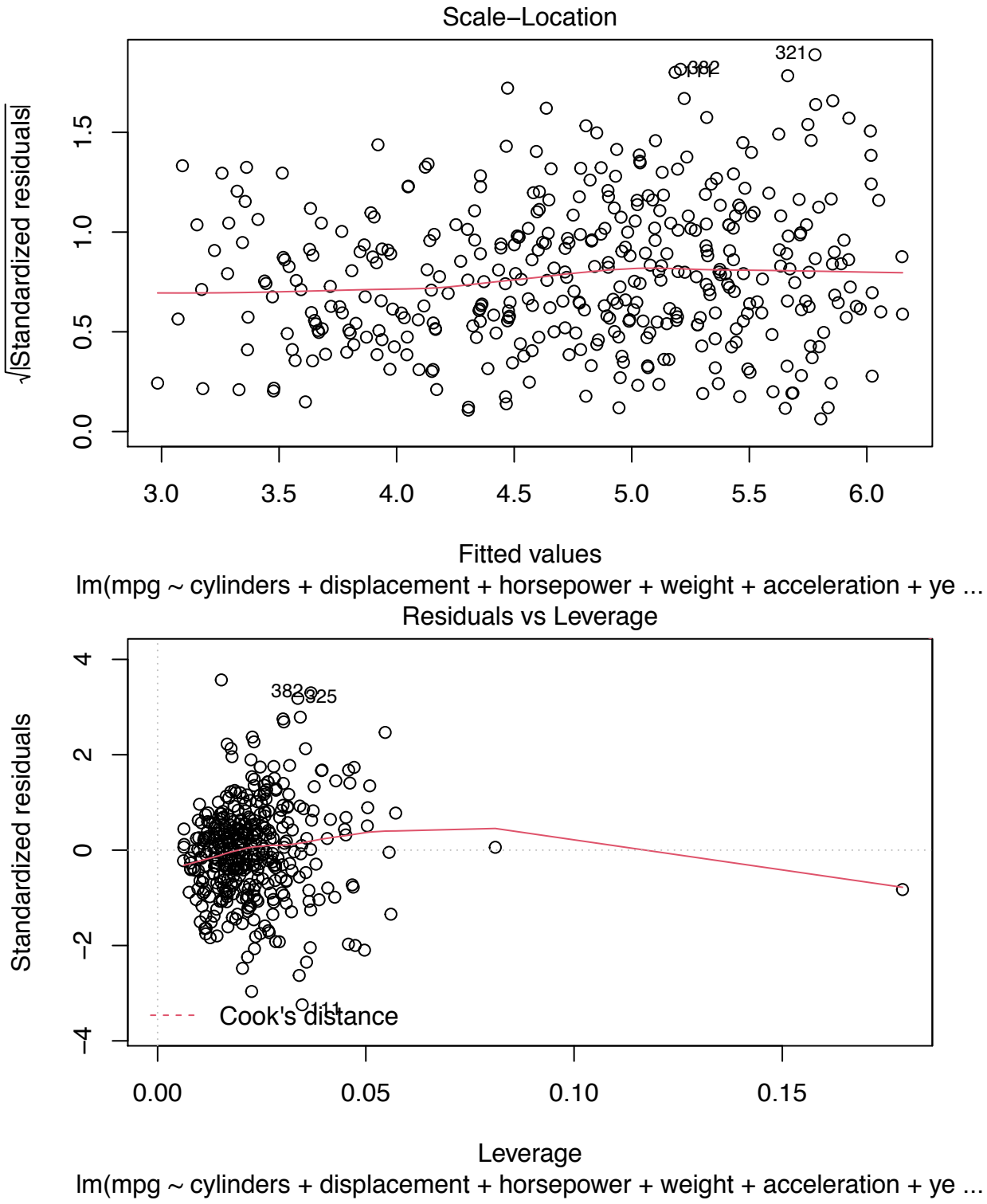
```r
squareRootData <- cbind(squareRootData, origin)
sqrtModel <- lm(mpg ~ cylinders + displacement + horsepower + weight + acceleration + year +
                  as.factor(origin), data = squareRootData)
summary(sqrtModel)
```

Square root transformation:

```
##
## Call:
## lm(formula = mpg ~ cylinders + displacement + horsepower + weight +
##     acceleration + year + as.factor(origin), data = squareRootData)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.94193 -0.18566  0.00493  0.16375  1.04694
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)        -1.983035   0.857373  -2.313 0.021256 *
## cylinders          -0.118364   0.141690  -0.835 0.404029
## displacement        0.029209   0.021802   1.340 0.181127
## horsepower         -0.087893   0.028402  -3.095 0.002116 **
## weight             -0.064363   0.007463  -8.625  < 2e-16 ***
## acceleration       -0.097928   0.077020  -1.271 0.204336
## year                1.298207   0.081175  15.993  < 2e-16 ***
## as.factor(origin)2  0.190208   0.052807   3.602 0.000357 ***
## as.factor(origin)3  0.196275   0.051778   3.791 0.000174 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2956 on 383 degrees of freedom
## Multiple R-squared:  0.8673, Adjusted R-squared:  0.8645
## F-statistic: 312.8 on 8 and 383 DF,  p-value: < 2.2e-16
```

```r
plot(sqrtModel)
```


Residuals vs Fitted
Residuals
1.0
0.5
0.0
−0.5
−1.0
382
321
111
3.0
3.5
4.0
4.5
5.0
5.5
6.0
Fitted values
lm(mpg ~ cylinders + displacement + horsepower + weight + acceleration + ye ...
Normal Q−Q
Standardized residuals
4
2
0
−2
321
382
111
−3
−2
−1
0
1
2
3
Theoretical Quantiles
lm(mpg ~ cylinders + displacement + horsepower + weight + acceleration + ye ...



This model (square root values) has a similar R-squared to our original model, and this transformation did not seem to help much with outliers. Therefore, it would probably be unhelpful to use a square root transformation on the variables.

HW2_Q10

HW 10

```
library(tidyr)
```

```
## Warning: package 'tidyr' was built under R version 4.1.2
```

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.1.2
```

```
##
## 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
```

```
library(tidyverse)
```

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

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.4     v stringr 1.4.0
## v readr   2.0.2     v forcats 0.5.1
```

```
## Warning: package 'ggplot2' was built under R version 4.1.2
```

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

```
setwd("C:/Users/Ayanna/Downloads")
cars <- read_csv("carseats.csv")
```

```
## Rows: 400 Columns: 11
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (3): ShelveLoc, Urban, US
## dbl (8): Sales, CompPrice, Income, Advertising, Population, Price, Age, Educ...
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

10.a

Fit a multiple regression model to predict Sales using Price, Urban, and US.

```
sales_model <- lm(Sales ~ Price + Urban + US, data = cars)
summary(sales_model)
```

```
##
## Call:
## lm(formula = Sales ~ Price + Urban + US, data = cars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.9206 -1.6220 -0.0564  1.5786  7.0581
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 13.043469   0.651012  20.036  < 2e-16 ***
## Price       -0.054459   0.005242 -10.389  < 2e-16 ***
## UrbanYes    -0.021916   0.271650  -0.081    0.936
## USYes        1.200573   0.259042   4.635 4.86e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.472 on 396 degrees of freedom
## Multiple R-squared:  0.2393, Adjusted R-squared:  0.2335
## F-statistic: 41.52 on 3 and 396 DF,  p-value: < 2.2e-16
```

10.b

Provide an interpretation of each coefficient in the model. Be careful—some of the variables in the model are qualitative! Price = - 0.0545 Urban = - 0.0219 US = + 1.2006

Interpretation:

For every one unit increase in Price (with Urban and US factors being constant) there is a change in Sales of -0.0545 units.

If a store is in an urban area (with Price and US factors being constant) there is a change in Sales of - 0.0219 units. That being said, since the value is greater than alpha (p = 0.05), we can conclude that there is not a statistically significant relationship between urban area and the sale of carseats (fail to reject the null).

If a store is in the US (with Price and Urban being fixed factors) there is a change in sales of 1.2006 units.

10.c

Write out the model in equation form, being careful to handle the qualitative variables properly.

Sales multiple regression equation: Sales = 13.0435 - 0.0545(Price) - 0.0219(Urban) + 1.2006(US) Urban is 1 if the store IS in an urban location, otherwise 0. US is 1 if the store is in the US, otherwise 0.

10.d

For which of the predictors can you reject the null hypothesis H0 : βj = 0?

From the information given in the previous questions, we are able to reject the null for Price and US predictors, there is enough statistically significant evidence that these factors do affect the sales of carseats. That being said, we fail to reject the null for the Urban factor as the pvalue is >0.05.

10.e

On the basis of your response to the previous question, fit a smaller model that only uses the predictors for which there is evidence of association with the outcome.

```
new_model <- lm(Sales ~ Price + US, data = cars)
summary(new_model)
```

```
##
## Call:
## lm(formula = Sales ~ Price + US, data = cars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.9269 -1.6286 -0.0574  1.5766  7.0515
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 13.03079    0.63098  20.652  < 2e-16 ***
## Price       -0.05448    0.00523 -10.416  < 2e-16 ***
## USYes        1.19964    0.25846   4.641 4.71e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.469 on 397 degrees of freedom
## Multiple R-squared:  0.2393, Adjusted R-squared:  0.2354
## F-statistic: 62.43 on 2 and 397 DF,  p-value: < 2.2e-16
```

10.f

How well do the models in (a) and (e) fit the data? model for 10.a: R^2 = 0.2393 adjusted R^2 = 0.2335

model for 10.e: R^2 = 0.2393 adjusted R^2 = 0.2354

Both models explain 23.93% of the variation occuring in the Sales of carseats in this df.

10.g

Using the model from (e), obtain 95 % confidence intervals for the coefficient(s).

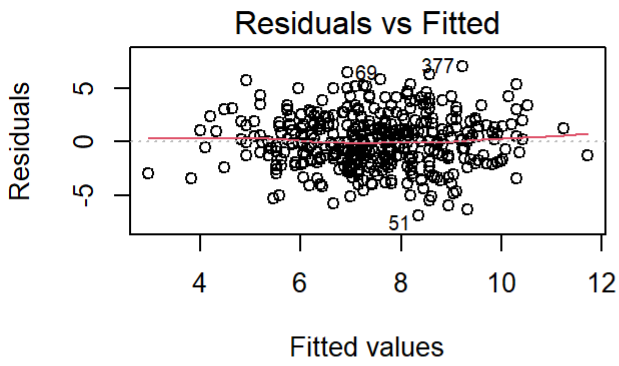
```
confint(new_model)
```

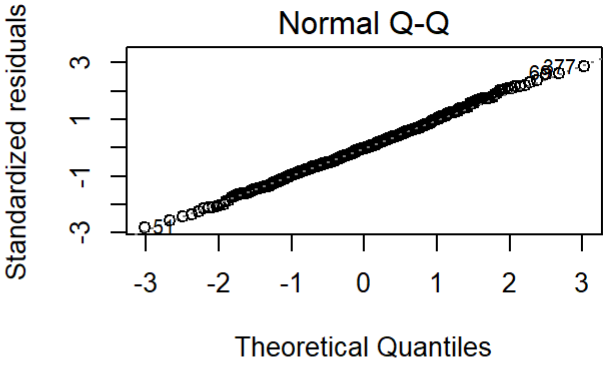
```
##                   2.5 %      97.5 %
## (Intercept) 11.79032020 14.27126531
## Price       -0.06475984 -0.04419543
## USYes        0.69151957  1.70776632
```

10.h

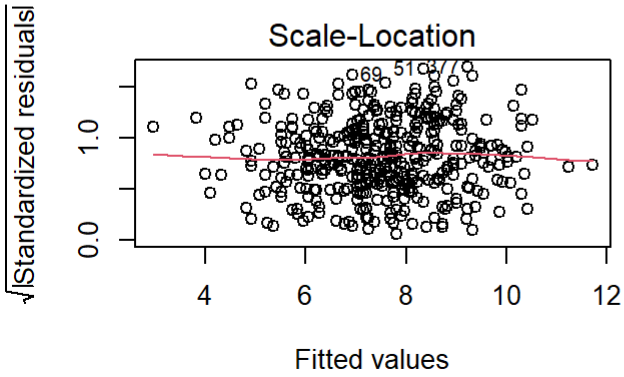
Is there evidence of outliers or high leverage observations in the model from (e)?

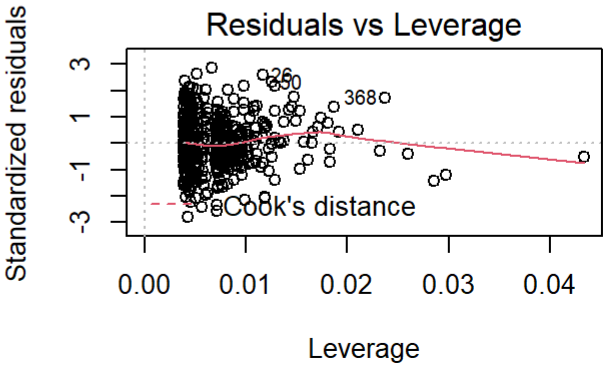
```
par(mfrow=c(2,2))
plot(new_model)
```





The





residuals vs leverage plot shows that there ARE influential data points in the regression model (outliers).

Question 13:

In this exercise you will create some simulated data and will fit simple linear regression models to it. Make sure to use set.seed(1) prior to starting part (a) to ensure consistent result

(a) Using the rnorm() function, create a vector, x, containing 100 observations drawn from a N (0, 1) distribution. This represents a feature, X.

```
set.seed(1)
x <- rnorm(100)
```

(b) Using the rnorm() function, create a vector, eps, containing 100 observations drawn from a N (0, 0.25) distribution—a normal distribution with mean zero and variance 0.25.

```
eps <- rnorm(100, 0, sqrt(0.25))
```

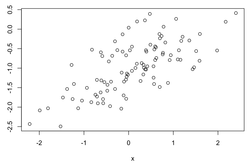
(c) Using x and eps, generate a vector y according to the model Y = −1 + 0.5X + ε. What is the length of the vector y? What are the values of β 0 and β 1 in this linear model?

```
y = -1 + (0.5 * x) + eps
```

The length of vector y is length of 100. The values of B0 is -1, and B1 is 0.5.

(d) Create a scatterplot displaying the relationship between x and y. Comment on what you observe.

```
plot(x, y)
```



From this plot, we observe a weak,

positive, linear relationship between x and y.

(e) Fit a least squares linear model to predict y using x . Comment on the model obtained. How do ˆβ 0 and ˆβ 1 compare to β 0 and β 1 ?

```
lm.fit <-  lm(y ~ x)
summary(lm.fit)
```

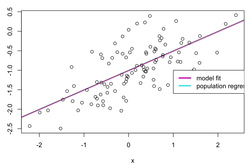
```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.93842 -0.30688 -0.06975  0.26970  1.17309
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.01885    0.04849 -21.018  < 2e-16 ***
## x            0.49947    0.05386   9.273 4.58e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4814 on 98 degrees of freedom
## Multiple R-squared:  0.4674,    Adjusted R-squared:  0.4619
## F-statistic: 85.99 on 1 and 98 DF,  p-value: 4.583e-15
```

A simple linear regression was used to test if x significantly predicted y. The fitted regression model was: y = -1.01885 + 5.2503(x). The overall regression was statistically significant (R^2 = 0.47, F(1, 98)= 85.99, p<0.005). It was found that x significantly predicted y (B=0.49947, p<0.005).

Thus, the linear regression model closely fits the true coefficient values.

(f) Display the least squares line on the scatterplot obtained in (d). Draw the population regression line on the plot, in a different color. Use the legend() command to create an appropriate legend.

```
plot(x, y)
abline(lm.fit, lwd = 1, col = 6)
abline(-1, 0.5, lwd = 1, col = 1)
legend(-1, legend = c("model fit", "population regression"), col = 6:1, lwd = 3)
```



(g) Now fit a polynomial regression model that predicts y using x and x 2 . Is there evidence that the quadratic term improves the model fit? Explain your answer.

```
lm.fit_sq <- lm(y ~  x + I(x^2))
summary(lm.fit_sq)
```

```
##
## Call:
## lm(formula = y ~ x + I(x^2))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.98252 -0.31270 -0.06441  0.29014  1.13500
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.97164    0.05883 -16.517  < 2e-16 ***
## x            0.50858    0.05399   9.420  2.4e-15 ***
## I(x^2)      -0.05946    0.04238  -1.403    0.164
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.479 on 97 degrees of freedom
## Multiple R-squared:  0.4779,    Adjusted R-squared:  0.4672
## F-statistic:  44.4 on 2 and 97 DF,  p-value: 2.038e-14
```

There is a slight increase in model fit on the training data in the polynomial proven by the slight increase in the adjusted R^2 value.

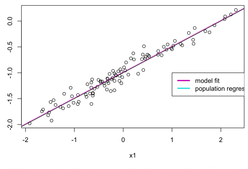
Interestingly enough, the model output suggests the y and x^2 relationship is not significant (B=-0.05946, p>0.05).

(h) Repeat (a)–(f) after modifying the data generation process in such a way that there is less noise in the data. The model (3.39) should remain the same. You can do this by decreasing the variance of the normal distribution used to generate the error term ε in (b). Describe your results.

```
set.seed(1)
eps1 <-  rnorm(100, 0, 0.125)
x1 <-  rnorm(100)
y1 <-  -1 + 0.5*x1 + eps1
plot(x1, y1)
lm.fit1 <- lm(y1 ~ x1)
summary(lm.fit1)
```

```
##
## Call:
## lm(formula = y1 ~ x1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.29052 -0.07545  0.00067  0.07288  0.28664
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.98639    0.01129  -87.34   <2e-16 ***
## x1           0.49988    0.01184   42.22   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1128 on 98 degrees of freedom
## Multiple R-squared:  0.9479,    Adjusted R-squared:  0.9474
## F-statistic:  1782 on 1 and 98 DF,  p-value: < 2.2e-16
```

```
plot(x1, y1)
abline(lm.fit1, lwd = 1, col = 6)
abline(-1, 0.5, lwd = 1, col = 1)
legend(-1, legend = c("model fit", "population regression"), col = 6:1, lwd = 3)
```



There is a slight increase in model fit

on the training data in the less noisy data proven by the slight increase in the adjusted R^2 value.

A multiple linear regression was used to test if x1 significantly predicted y1. The fitted regression model was: y1 = -0.98639 + 0.49988(x1). The overall regression was statistically significant (R^2 = 0.95, F(1, 98)= 1782, p<0.005). It was found that x1 significantly predicted y1 (B=0.49988, p<0.005).

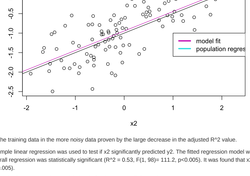
Thus, the linear regression model closely fits the true coefficient values.

(i) Repeat (a)–(f) after modifying the data generation process in such a way that there is more noise in the data. The model (3.39) should remain the same. You can do this by increasing the variance of the normal distribution used to generate the error term ε in (b). Describe your results.

```
set.seed(1)
eps2 <-  rnorm(100, 0, 0.5)
x2 <-  rnorm(100)
y2 <-  -1 + 0.5*x2 + eps2
plot(x2, y2)
lm.fit2 <-  lm(y2 ~ x2)
summary(lm.fit2)
```

```
##
## Call:
## lm(formula = y2 ~ x2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.16298 -0.30181  0.00268  0.29152  1.14656
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.94557    0.04517 -20.93   <2e-16 ***
## x2           0.49953    0.04736  10.55   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4514 on 98 degrees of freedom
## Multiple R-squared:  0.5317,    Adjusted R-squared:  0.5269
## F-statistic: 111.2 on 1 and 98 DF,  p-value: < 2.2e-16
```

```
plot(x2, y2)
abline(lm.fit2, lwd = 1, col = 6)
abline(-1, 0.5, lwd = 1, col = 1)
legend(-1, legend = c("model fit", "population regression"), col = 6:1, lwd = 3)
```



There is a slight increase in model fit

on the training data in the more noisy data proven by the large decrease in the adjusted R^2 value.

A simple linear regression was used to test if x2 significantly predicted y2. The fitted regression model was: y2 = -0.98639 + 0.49988(x2). The overall regression was statistically significant (R^2 = 0.53, F(1, 98)= 111.2, p<0.005). It was found that x2 significantly predicted y2 (B=0.49953, p<0.005).

Thus, the linear regression model closely fits the true coefficient values.

(j) What are the confidence intervals for β 0 and β 1 based on the original data set, the noisier data set, and the less noisy dataset? Comment on your result.

```
confint(lm.fit)
```

```
##                   2.5 %     97.5 %
## (Intercept) -1.1150804 -0.9226122
## x            0.3925794  0.6063602
```

```
confint(lm.fit1)
```

```
##                 2.5 %     97.5 %
## (Intercept) -1.008805 -0.9639819
## x1           0.476387  0.5233799
```

```
confint(lm.fit2)
```

```
##                 2.5 %     97.5 %
## (Intercept) -1.0352283 -0.8559176
## x2           0.4055479  0.5935197
```

All of the confidence intervals around x include 0.5. The range of the x CI's decrease in size from x1, x, x2. Additionally, all of the confidence intervals around the intercept include -1.0. These are all close to our original coefficients.