Jin Kweon (3032235207) HW3

Jin Kweon

9/25/2017

Name: Jin Kweon

Lab: 11am - 1pm (section 102) (course number: 20980)

Problem 1

We define residual as $y - \hat{y} = e$.

$\sum (y_i - \hat{y_i}) = 0$. So, $\sum y_i = \sum \hat{y_i}$. So, $y_1 + ... + y_n = \hat{y_1} + ... + \hat{y_n}$.

And, it implies that we need to prove equation below:

$$\begin{pmatrix} 1 & 1 & ... & 1 \end{pmatrix} \begin{pmatrix} y_1 \\ . \\ . \\ . \\ y_n \end{pmatrix} = \begin{pmatrix} 1 & 1 & ... & 1 \end{pmatrix} \begin{pmatrix} \hat{y_1} \\ . \\ . \\ . \\ \hat{y_n} \end{pmatrix}$$

Since we can always define a normal equation (even when there is no model, but we have a model. And, b_0 and b_1 are the least square estimators), I will use it. (without proving normal equation as we proved it and used it as a fact in the class)

I will say

$$\hat{\beta} = \begin{pmatrix} b_0 \\ b_1 \end{pmatrix}$$

Normal equation is following: $X^T X\hat{\beta} = X^T y$ and since $\hat{y} = X\hat{\beta}$, I can say that $X^T\hat{y} = X^T y$.

Now let's define a design matrix (and we always need to include an intercept in our design matrix and that is a part of definition).

Let design matrix X with the dimension of (n) by (p+1) be following:

$$X = \begin{bmatrix} 1 & x_{11} & ... & x_{1p} \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ 1 & x_{n1} & ... & x_{np} \end{bmatrix}$$

.

So,

$$X^T\hat{y} = \begin{bmatrix} 1 & 1 & ... & 1 \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ x_{1p} & x_{2p} & ... & x_{np} \end{bmatrix} \hat{y} = \begin{bmatrix} [1 & 1 & ... & 1] \ \hat{y} \\ [. & . & . & .] \ \hat{y} \\ [. & . & . & .] \ \hat{y} \\ [. & . & . & .] \ \hat{y} \\ [x_{1p} & x_{2p} & ... & x_{np}] \ \hat{y} \end{bmatrix}$$

and this is equal to

$$X^T y = \begin{bmatrix} 1 & 1 & ... & 1 \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ x_{1p} & x_{2p} & ... & x_{np} \end{bmatrix} y = \begin{bmatrix} [1 & 1 & ... & 1] \ y \\ [. & . & . & .] \ y \\ [. & . & . & .] \ y \\ [. & . & . & .] \ y \\ [x_{1p} & x_{2p} & ... & x_{np}] \ y \end{bmatrix}$$

.

So, I can conlcude that

$$\begin{bmatrix} [1 & 1 & ... & 1] \ \hat{y} \\ [. & . & . & .] \ \hat{y} \\ [. & . & . & .] \ \hat{y} \\ [. & . & . & .] \ \hat{y} \\ [x_{1p} & x_{2p} & ... & x_{np}] \ \hat{y} \end{bmatrix} = \begin{bmatrix} [1 & 1 & ... & 1] \ y \\ [. & . & . & .] \ y \\ [. & . & . & .] \ y \\ [. & . & . & .] \ y \\ [x_{1p} & x_{2p} & ... & x_{np}] \ y \end{bmatrix}$$

and each element of vector is equal, so,

$$\begin{bmatrix} [1 & 1 & ... & 1] \ \hat{y} \end{bmatrix} = \begin{bmatrix} [1 & 1 & ... & 1] \ y \end{bmatrix}$$

is true and it implies

$$\begin{pmatrix} 1 & 1 & ... & 1 \end{pmatrix} \begin{pmatrix} y_1 \\ . \\ . \\ . \\ y_n \end{pmatrix} = \begin{pmatrix} 1 & 1 & ... & 1 \end{pmatrix} \begin{pmatrix} \hat{y_1} \\ . \\ . \\ . \\ \hat{y_n} \end{pmatrix}$$

Here is one the examples:

```
a <- lm(mpg ~ disp, data = mtcars)

round(sum(a$residuals), 10)
```

```
## [1] 0
```

Problem 2

We examine a response variable Y in terms of two predictors X (explanatory variable) and Z. There are n observations. Let **X** (design matrix) be a matrix formed by a constant term of **1**, and the vectors **x** and **z**.

part a

X^TX is a symmetric matrix with non-negative entries on the diagonal.

Thus, the missing values are 0, 0, and 7.

And, here are the full matrix: and this is equal to

$$X^TX = \begin{bmatrix} 30 & 0 & 0 \\ 0 & 10 & 7 \\ 0 & 7 & 15 \end{bmatrix}$$

.

Also, n will be 30, as element (1,1) of X^TX is 30 (add 1 30 times).

Part b

cor(X, Z) = $\frac{\sum (x_i - \bar{x})(z_i - \bar{z})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (z_i - \bar{z})^2}}$.

And, we have a matrix

$$X^TX = \begin{bmatrix} 30 & 0 & 0 \\ 0 & 10 & 7 \\ 0 & 7 & 15 \end{bmatrix} = \begin{bmatrix} - & - & - & 1 & - & - & - \\ - & - & - & x & - & - & - \\ - & - & - & z & - & - & - \end{bmatrix} \begin{bmatrix} | & | & | \\ | & | & | \\ | & | & | \\ 1 & x & z \\ | & | & | \\ | & | & | \\ | & | & | \end{bmatrix} = \begin{bmatrix} \sum 1 & \sum x_i & \sum z_i \\ \sum x_i & \sum x_i^2 & \sum x_i z_i \\ \sum z_i & \sum x_i z_i & \sum z_i^2 \end{bmatrix}$$

So, from this matrix, I can tell that there are n = 30 observations, and $\bar{x} = \frac{1}{30}\sum x_i = 0$. Also, $\bar{z} = \frac{1}{30}\sum z_i = 0$.

So, cor(X, Z) = $\frac{\sum (x_i - \bar{x})(z_i - \bar{z})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (z_i - \bar{z})^2}} = \frac{\sum x_i z_i}{\sqrt{\sum x_i^2 \sum z_i^2}} = \frac{7}{\sqrt{10}\sqrt{15}}$.

Part c

From the given equation $\hat{y_i} = \hat{\beta_0} + \hat{\beta_1} x_i + \hat{\beta_2} z_i$, I can say that $\hat{\beta_0}$ = -2, $\hat{\beta_1} = 1$, and $\hat{\beta_2} = 2$.

And, we know that $\bar{y} = \hat{\beta_0} + \hat{\beta_1}\bar{x} + \hat{\beta_2}\bar{z}$ (This is true if and only if we have an intercept, and this can be proved by using normal equation), so I can say that $\bar{y}$ = -2 + $\bar{x}$ + 2$\bar{z}$. And, we know what $\bar{x} = 0$ and $\bar{z} = 0$ are from X^TX.

Thus, $\bar{y}$ = -2.

Part d

$R^2 = \frac{TSS - RSS}{TSS} = \frac{REGSS}{REGSS + RSS}$ and since RSS is given as 12 in the question, we only need to seek for REGSS. And, since REGSS = $\sum (\hat{y_i} - \bar{y})^2 = \sum (-2 + x_i + 2z_i + 2)^2 = \sum (x_i + 2z_i)^2 = \sum x_i^2 + 4\sum x_i z_i + 4\sum z_i^2 = 10 + 28 + 60 = 98$. So, $R^2 = \frac{REGSS}{TSS} = \frac{98}{12 + 98} = \frac{98}{110} = \frac{49}{55}$.

Thus, $R^2 = \frac{49}{55}$.

Problem 3

Part a

```
set.seed(1)

x <- rnorm(100, 0, 1)
head(x)
```

```
## [1] -0.6264538  0.1836433 -0.8356286  1.5952808  0.3295078 -0.8204684
```

Part b

```
eps <- rnorm(100, 0, 0.5) #sd^2 = var
head(eps)
```

```
## [1] -0.31018334  0.02105794 -0.45546082  0.07901439 -0.32729232  0.88364363
```

Part c

So, I can say that

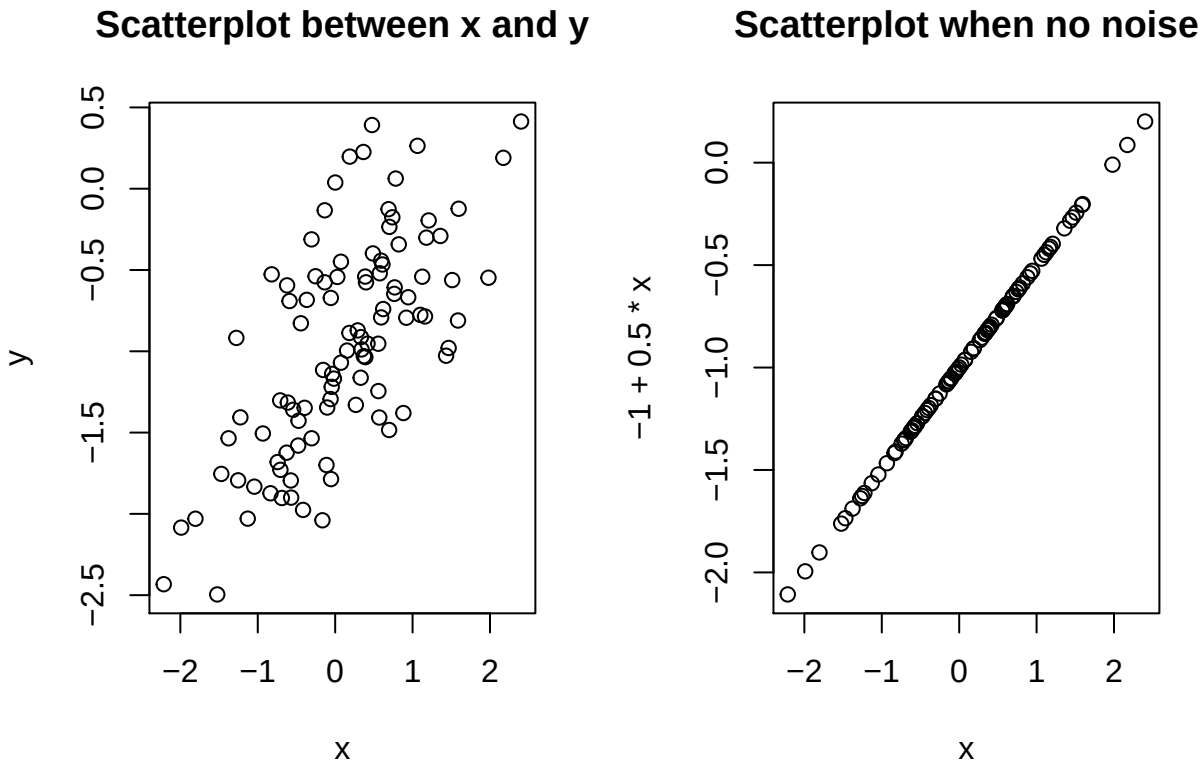
$$Y = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ . \\ y_{100} \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \\ . \\ . \\ . \\ -1 \end{bmatrix} + 0.5 \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ x_{100} \end{bmatrix} + \begin{bmatrix} e_1 \\ e_2 \\ . \\ . \\ . \\ e_{100} \end{bmatrix}$$

```
y <- -1 + 0.5*x + eps
head(y)
```

```
## [1] -1.6234102 -0.8871204 -1.8732751 -0.1233452 -1.1625384 -0.5265906
```

Part d

```
par(mfrow=c(1,2))
plot(x, y, main = "Scatterplot between x and y")
plot(x,-1+0.5*x, main = "Scatterplot when no noise")
```



COMMENT: They show the decent linear relationship between x and y, but definitely they do not form a perfect line. From the equation I created above, I can see that 0.5 slope (as x goes up by 1 unit, y goes up by around 0,5 unit) and the intercept will be around -1. Additionally, I plotted out the scatter when there is no normally distributed random noise, and as you can see, it will form a perfect line.

Part e

I will get solve it in two says: lm() and OLS least square estimator formula, $\hat{\beta} = (X^T X)^{-1} X^T Y$. Also, I can solve it in QR factorization.

```
par(mfrow=c(1,1))

lm <- lm(y ~ x)
beta0 <- lm$coefficients[1]
beta1 <- lm$coefficients[2]

int <- rep(1, nrow(as.matrix(x)))
newx <- cbind(int, x)
beta <- solve(t(newx) %*% newx) %*% t(newx) %*% y

#They are equal.
#lm$coefficients
```

```r
paste("Least square intercept will be", round(beta[1,1], 6),
      "and least square slope will be", round(beta[2,1], 6))
```

```
## [1] "Least square intercept will be -1.018846 and least square slope will be 0.49947"
```

```r
print("Original intercept will be -1 and slope will be 0.5")
```

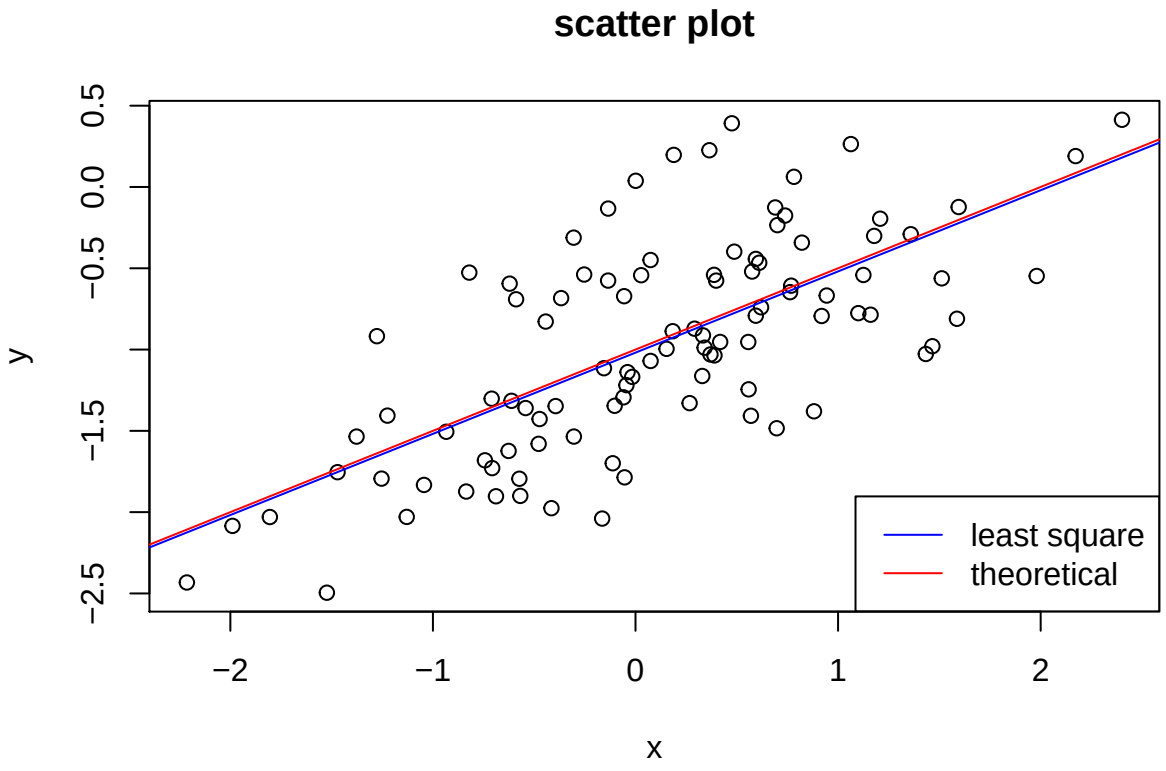
```
## [1] "Original intercept will be -1 and slope will be 0.5"
```

COMMENT: So, β_0 = -1 and $\beta_1 = 0.5$ and $\hat{\beta}_0 \approx -1.018846$ and $\hat{\beta}_1 \approx 0.49947$. The line is little bit gentle with least square estimators. Also, OLS estimators are almost the same as theoretical regression line's.

Part f

Theoretical regression line $= Y = -1 + 0.5X$ and model is $Y = -1 + 0.5X + \epsilon$

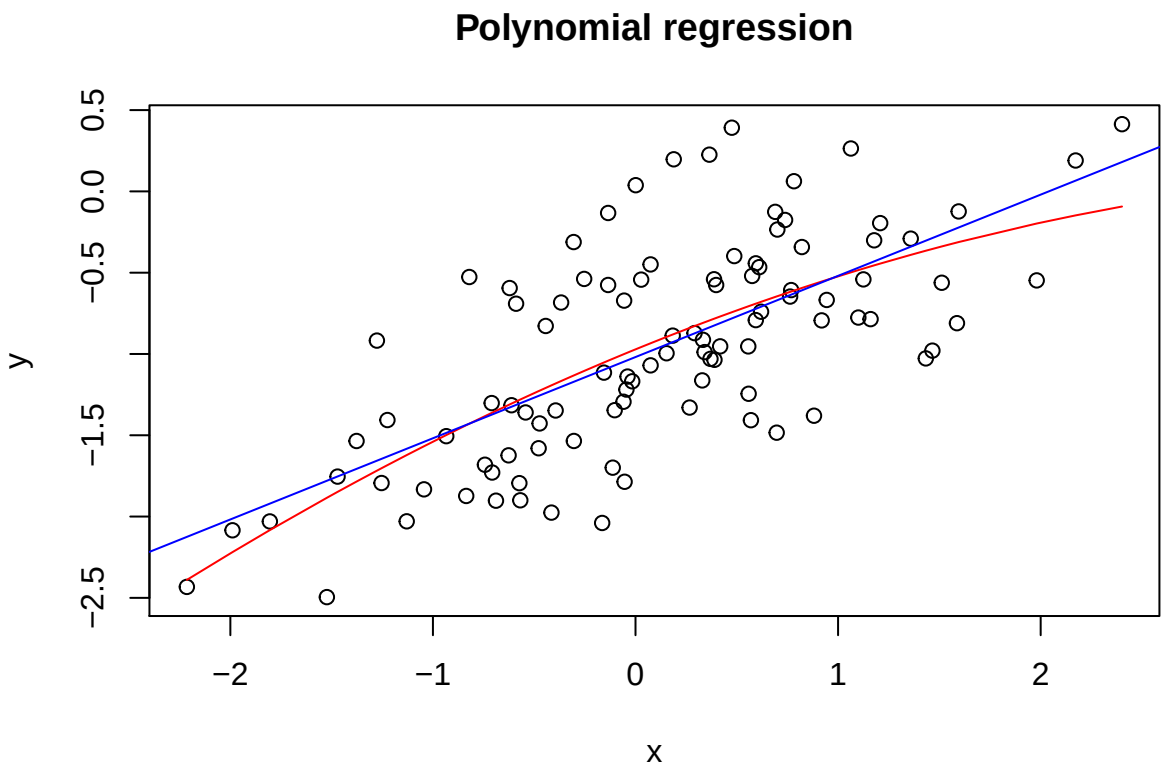
```r
plot(x, y, main = "scatter plot")
abline(a = -1, b = 0.5, col = "red")
abline(lm, col = "blue")
legend("bottomright", legend = c("least square", "theoretical"),
       lty = c(1,1), col = c("blue", "red"))
```



Part g

```r
lm2 <- lm(y ~ x + I(x^2)) #Or, do summary(lm(y ~ poly(x, 2, raw = T)))
y2 <- lm2$coefficients[1] + lm2$coefficients[2]*x + lm2$coefficients[3]*x^2
```

```r
plot(x, y, main = "Polynomial regression")

smoothingSpline = smooth.spline(x, y2, spar = 0.5)
lines(smoothingSpline, col = "red")
abline(lm, col = "blue")
```



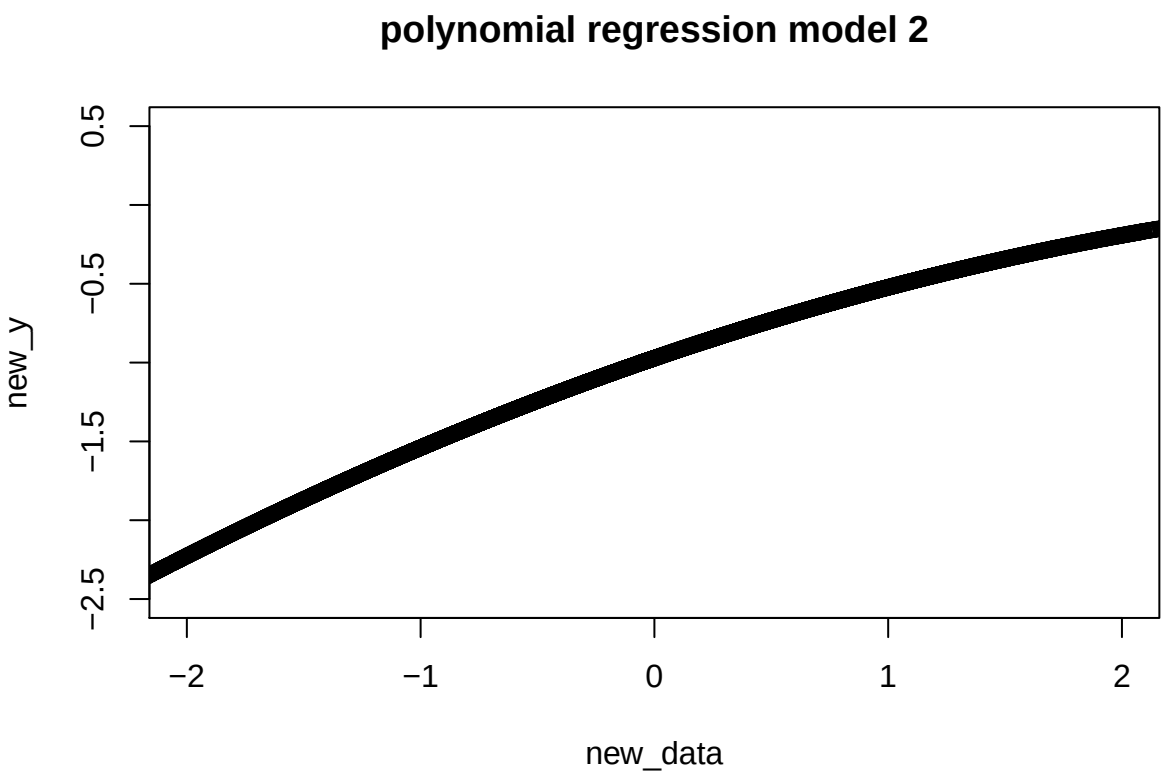

```r
summary(lm2)
```

```
##
## 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
```

```r
summary(lm)
```

```
##
## 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 -21.010  < 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
```

```r
set.seed(1)
new_data <- rnorm(10000, 0, 1)
new_y <- c(lm2$coefficients[1] + lm2$coefficients[2]*new_data +
            lm2$coefficients[3]*(new_data)^2)
plot(new_data, new_y, xlim = c(-2, 2), ylim = c(-2.5, 0.5),
     main = "polynomial regression model 2")
```



COMMENT: There is really no evidence that the quadratic term improves the model fit on both p-value analysis and graphs. P-value tells us to drop the quadratic term. Although adjusted R^2 goes up a little bit, I

would still say quadratic term does not improve the model fit that much. (but this might not be a good way to tell, since R^2 always goes up as you add the variables.) Also, it looks better when I use linear fit visually.

Part h

Let's say the new noise eps2 $\sim$ N(0, 0.04).

```
#Part a
set.seed(1)
x2 <- rnorm(100, 0, 1)
head(x2)
```

```
## [1] -0.6264538  0.1836433 -0.8356286  1.5952808  0.3295078 -0.8204684
```

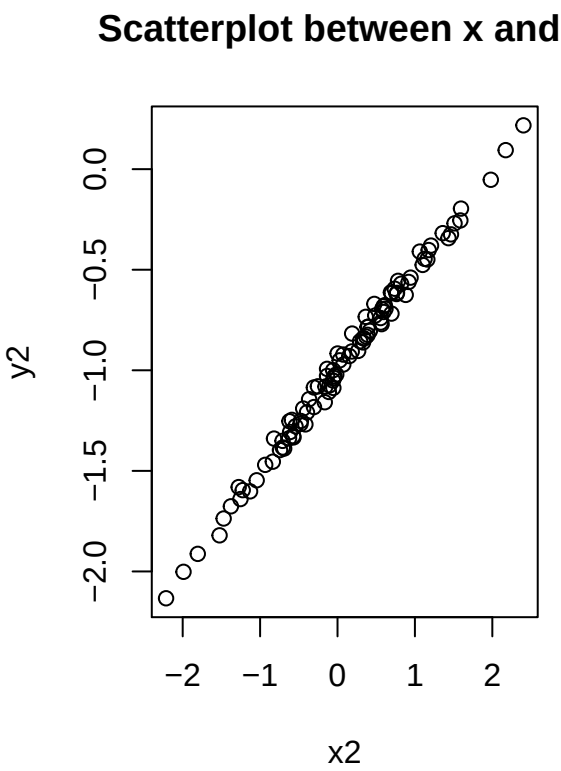
```
#Part b
eps2 <- rnorm(100, 0, 0.04) #sd^2 = var
head(eps2)
```

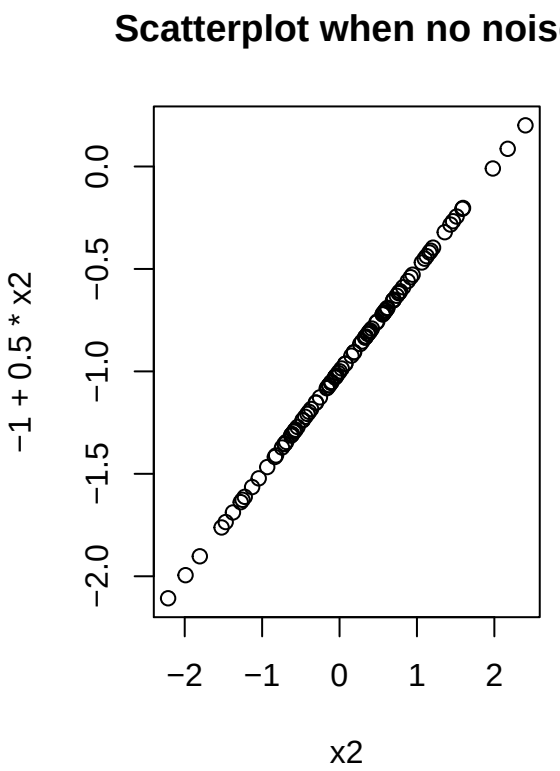
```
## [1] -0.024814667  0.001684635 -0.036436866  0.006321151 -0.026183386
## [6]  0.070691491
```

```
#Part c
y2 <- -1 + 0.5*x2 + eps2
head(y2)
```

```
## [1] -1.3380416 -0.9064937 -1.4542512 -0.1960384 -0.8614295 -1.3395427
```

```
#Part d
par(mfrow=c(1,2))
plot(x2, y2, main = "Scatterplot between x and y")
plot(x2,-1+0.5*x2, main = "Scatterplot when no noise")
```





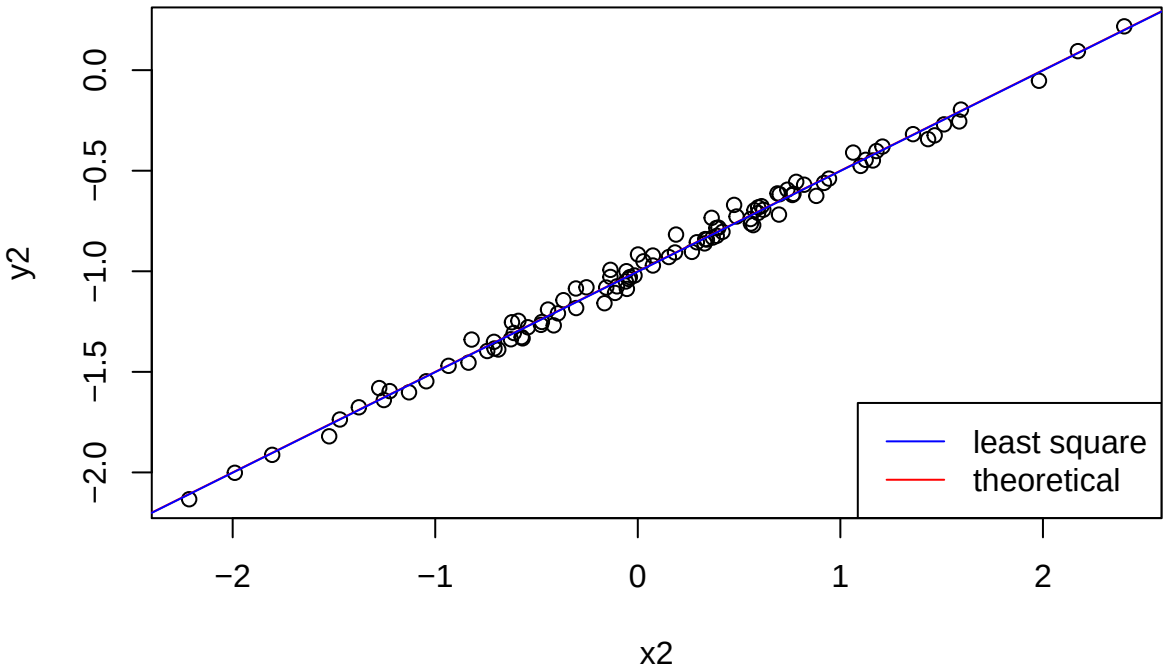

```
#Part e
par(mfrow=c(1,1))

lm2 <- lm(y2 ~ x2)
beta20 <- lm2$coefficients[1]
beta21 <- lm2$coefficients[2]

int <- rep(1, nrow(as.matrix(x)))
newx2 <- cbind(int, x2)
beta2 <- solve(t(newx2) %*% newx2) %*% t(newx2) %*% y2

#They are equal.
#lm$coefficients
paste("Least square intercept will be", round(beta2[1,1], 6),
      "and least square slope will be", round(beta2[2,1], 6))
```

```
## [1] "Least square intercept will be -1.001508 and least square slope will be 0.499958"
```

```
print("Original intercept will be -1 and slope will be 0.5")
```

```
## [1] "Original intercept will be -1 and slope will be 0.5"
```

```
#Part f
plot(x2, y2, main = "Scatter plot version2")
abline(a = -1, b = 0.5, col = "red")
abline(lm2, col = "blue")
legend("bottomright", legend = c("least square", "theoretical"),
       lty = c(1,1), col = c("blue", "red"))
```



COMMENT for part d: They show the decent linear relationship between x and y, but definitely they do not form a perfect line (yet form a closer to the perfect line than the one has original noise with variance 0.25). From the equation I created above, I can see that 0.5 slope (as x goes up by 1 unit, y goes up by around 0,5 unit) and the intercept will be around -1 more clearly. Additionally, I plotted out the scatter when there is no normally distributed random noise, and as you can see, it will form a perfect line.

COMMENT for part e: So, β_0 = -1 and β_1 = 0.5 and $\hat{\beta}_0 \approx -1.001508$ and $\hat{\beta}_1 \approx 0.499958$. The line is more gentle with least square estimators. Our estimators get closed to the theoretical regression line's.

COMMENT for part f: Variance is really small, the red and blue line are on the same position, and cannot tell the difference visually.

In overall, less noise means that a linear modeling with OLS estimators is more reliable.

Part i

Let's say the new noise eps3 $\sim$ N(0, 0.81).

```
#Part a
set.seed(1)
x3 <- rnorm(100, 0, 1)
head(x3)
```

```
## [1] -0.6264538  0.1836433 -0.8356286  1.5952808  0.3295078 -0.8204684
```

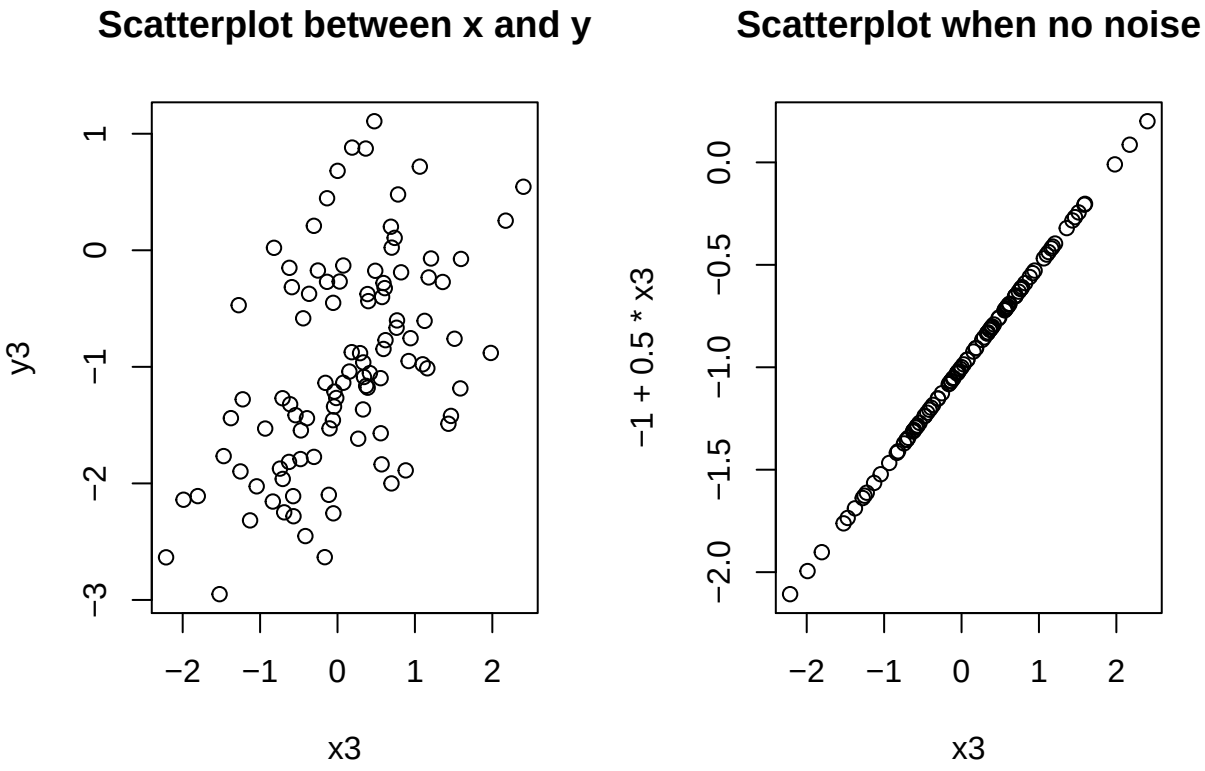
```
#Part b
eps3 <- rnorm(100, 0, 0.81) #sd^2 = var
head(eps3)
```

```
## [1] -0.50249701  0.03411386 -0.73784654  0.12800331 -0.53021356  1.43150269
```

```
#Part c
y3 <- -1 + 0.5*x3 + eps3
head(y3)
```

```
## [1] -1.81572391 -0.87406448 -2.15566084 -0.07435629 -1.36545968  0.02126850
```

```
#Part d
par(mfrow=c(1,2))
plot(x3, y3, main = "Scatterplot between x and y")
plot(x3,-1+0.5*x3, main = "Scatterplot when no noise")
```




```
#Part e
par(mfrow=c(1,1))

lm3 <- lm(y3 ~ x3)
beta30 <- lm3$coefficients[1]
beta31 <- lm3$coefficients[2]

int <- rep(1, nrow(as.matrix(x)))
newx3 <- cbind(int, x3)
beta3 <- solve(t(newx3) %*% newx3) %*% t(newx3) %*% y3

#They are equal.
```

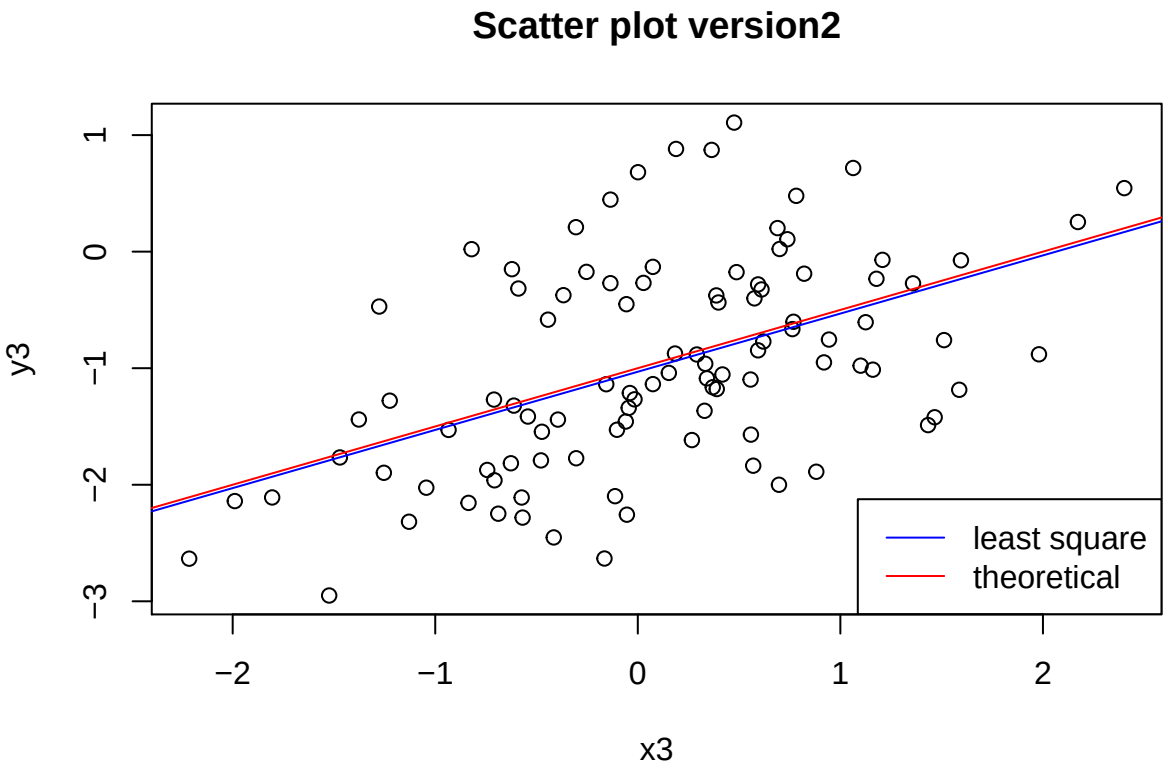
```r
#lm$coefficients
paste("Least square intercept will be", round(beta3[1,1], 6),
      "and least square slope will be", round(beta3[2,1], 6))
```

```
## [1] "Least square intercept will be -1.030531 and least square slope will be 0.499141"
```

```r
print("Original intercept will be -1 and slope will be 0.5")
```

```
## [1] "Original intercept will be -1 and slope will be 0.5"
```

```r
#Part f
plot(x3, y3, main = "Scatter plot version2")
abline(a = -1, b = 0.5, col = "red")
abline(lm3, col = "blue")
legend("bottomright", legend = c("least square", "theoretical"),
       lty = c(1,1), col = c("blue", "red"))
```



COMMENT for part d: They show weak linear relationship between x and y. I cannot really guess the slope and intercept from that scatter plot. Linear modeling will not be a good for this data. Additionally, I plotted out the scatter when there is no normally distributed random noise, and as you can see, it will form a perfect line.

COMMENT for part e: So, β_0 = -1 and β_1 = 0.5 and $\hat{\beta}_0 \approx -1.030531$ and $\hat{\beta}_1 \approx 0.499141$. OLS estimators are further from the theoretical regression line's. (slope relatively does not change although we change the noise, since the OLS line usually passes through the middle of the data unless they have significant outliers.) I doubt the linear model with these OLS estimators can be a good fit. The line is more gentle with least square estimators.

COMMENT for part f: I can see a gap between the least square and theoretical regression line well.

In overall, more noise means that a linear modeling with OLS estimators is less reliable.

Problem 4

$\hat{y} = X\hat{\beta}$, so if we X has a full rank/columns are linearly independent, then X can be decomposed into QR. And, for each y, the equation has a unique least square: $\hat{\beta} = R^{-1}Q^T y$.

Here is a proof using **normal equation**:

$X^T X\hat{\beta} = X^T y$, and then, $(QR)^T QR\hat{\beta} = (QR)^T y$. So, $R^T Q^T QR\hat{\beta} = R^T Q^T y$, and since Q is orthonormal $Q^T Q = I$. So, I have $R^T R\hat{\beta} = R^T Q^T y$, and I multiply $(R^T)^{-1}$ on both sides (there should be no 0 element on the diagonal), then, I will get $R\hat{\beta} = Q^T y$.

And, I am going to basically use this idea for this question.

Unless all the variables (i.e. the predictors and response) are mean-centered, X will be a matrix of dimension n $\times$ (p + 1), where n is the number of observations, and p is the number of predictors. In other words, X should include the column of 1's to account for the intercept term, except for when all variables are mean-centered.

==> R: we have a equation: $\hat{y} - \bar{y} = r_{xy}\frac{S_y}{S_x}(x - \bar{x})$, and we make $\bar{x}$ and $\bar{y}$ being zero, they will have no intercept.

```
ols_fit <- function(model, response){
  n <- nrow(model)
  q <- ncol(model)
  y_values <- response

  QR <- qr(model)
  Q <- qr.Q(QR)
  R <- qr.R(QR)

  coefficients <- backsolve(R, crossprod(Q, y_values))
  fitted_values <- model %*% coefficients
  residuals <- y_values - fitted_values

  #assigning names to all the elements in the list
  lists <- list(coefficients = coefficients, y_values = y_values,
                fitted_values = fitted_values,
```

```
                residuals = residuals, n = n, q = q)
  return(lists)
}

intercept <- rep(1, nrow(mtcars))
X <- as.matrix(cbind(intercept, mtcars[,c(3,4)]))
y <- mtcars$mpg


fit <- ols_fit(X, y)
names(fit)
```

```
## [1] "coefficients"  "y_values"      "fitted_values" "residuals"
## [5] "n"             "q"
```

```
fit$coefficients
```

```
##             [,1]
## [1,] 30.73590425
## [2,] -0.03034628
## [3,] -0.02484008
```

```
summary(fit$fitted_values)
```

```
##        V1
##  Min.   :11.32
##  1st Qu.:15.16
##  Median :21.64
##  Mean   :20.09
##  3rd Qu.:24.70
##  Max.   :27.15
```

```
summary(fit$residuals)
```

```
##        V1
##  Min.   :-4.7945
##  1st Qu.:-2.3036
##  Median :-0.8246
##  Mean   : 0.0000
##  3rd Qu.: 1.8582
##  Max.   : 6.9363
```

```
#Double check with lm function
lms <- lm(mpg ~ disp + hp, data = mtcars)$coefficients
names(lms) <- NULL #I took out the names since lm function automatically assigned the names.
identical((round(lms, 5)), as.numeric(round(fit$coefficients, 5)))
```

```
## [1] TRUE
```

Problem 5

Part a

Here is what coefficient of determination (R^2) is:

$R^2 = \frac{Regss}{TSS} = \frac{\sum(\hat{y}_i - \bar{y})^2}{\sum(y_i - \bar{y})^2}$. So, we need $\hat{y}_i$ (3rd list), $\bar{y}$ (mean of sum of 2nd list), and y_i (2nd list) from ols_fit() function.

```
R2 <- function(fit_list){
  yi <- fit_list[[2]]
  ybar <- mean(yi)
  yhat <- fit_list[[3]]

  R_2 <- sum((yhat - ybar)^2) / sum((yi - ybar)^2)

  return(R_2)
}

R2(fit)
```

```
## [1] 0.7482402
```

Part b

Here is what Residual standard error/relative standard error (RSE) is:

RSE $= \sqrt{\frac{RSS}{n - p - 1}} = \sqrt{\frac{\sum(y_i - \hat{y}_i)^2}{n - p - 1}}$, where p is the number of predictors/x/explanatory variables.

Basically, p + 1 (= q) is number of columns/variables including intercept.

So, we need $\hat{y}_i$ (3rd list), y_i (2nd list), number of observations (5th list), and q (6th list) from ols_fit() function.

```
RSE <- function(fit_list){
  yi <- fit_list[[2]]
  yhat <- fit_list[[3]]
  q <- fit_list[[6]]
  n <- fit_list[[5]]

  rse <- sqrt(sum((yi - yhat)^2) / (n - q))
  return(rse)
}

RSE(fit)
```

```
## [1] 3.126601
```

Problem 6

Part a

Use your ols_fit() and R2() functions!!! I did not use RSE() function just because the problem set did not say to use RSE() function.

```
web <- "https://raw.githubusercontent.com"
deep <- "/ucb-stat154/stat154-fall-2017/master/data/prostate.txt"
urls <- paste0(web, deep)
data <- download.file(urls, destfile = "prostate.txt")

data <- read.table("prostate.txt", header = T, stringsAsFactors = T)

intercept <- rep(1, nrow(data))
response <- as.matrix(data$lpsa)
predictor <- as.matrix(cbind(intercept, data$lcavol))


fit2 <- ols_fit(predictor, response)

#double check with lm() function.
lm(lpsa ~ lcavol, data = data)$coefficients
```

```
## (Intercept)      lcavol
##   1.5072979   0.7193201
```

```
fit2[[1]]
```

```
##           [,1]
## [1,] 1.5072979
## [2,] 0.7193201
```

```
paste("Fitted values for lpsa as a response and lcavol as a predictor a are",
      round(fit2[[1]][1,1], 6), "and", round(fit2[[1]][2,1], 6))
```

```
## [1] "Fitted values for lpsa as a response and lcavol as a predictor a are 1.507298 and 0.71932"
```

```
r2fit2 <- R2(fit2)
r2fit2
```

```
## [1] 0.5394319
```

```r
#doulbe check with lm() summary.
summary(lm(lpsa ~ lcavol, data = data))$r.squared
```

```
## [1] 0.5394319
```

```r
rse2 <- sqrt(sum((fit2[[2]] - fit2[[3]])^2) / (nrow(data) - fit2[[6]]))
rse2
```

```
## [1] 0.7874994
```

```r
paste("Coefficient of determination and RSE are", round(R2(fit2), 6),
      "and", round(rse2, 6), "respectively.")
```

```
## [1] "Coefficient of determination and RSE are 0.539432 and 0.787499 respectively."
```

Part b - 1

```r
response <- as.matrix(data$lpsa)
predictor <- cbind(predictor, data$lweight)

fit3 <- ols_fit(predictor, response)

#double check with lm() function.
lm(lpsa ~ lcavol + lweight, data = data)$coefficients
```

```
## (Intercept)      lcavol     lweight
##  -0.3026179   0.6775253   0.5109495
```

```r
fit3[[1]]
```

```
##             [,1]
## [1,] -0.3026179
## [2,]  0.6775253
## [3,]  0.5109495
```

```r
r2fit3 <- R2(fit3)
r2fit3
```

```
## [1] 0.5859345
```

```r
#doulbe check with lm() summary.
summary(lm(lpsa ~ lcavol + lweight, data = data))$r.squared
```

```
## [1] 0.5859345
```

```r
rse3 <- sqrt(sum((fit3[[2]] - fit3[[3]])^2) / (nrow(data) - fit3[[6]]))
rse3
```

```
## [1] 0.7506469
```

```r
paste("Coefficient of determination and RSE are", round(R2(fit3), 6),
      "and", round(rse3, 6), "respectively.")
```

```
## [1] "Coefficient of determination and RSE are 0.585935 and 0.750647 respectively."
```

Part b - 2

```
response <- as.matrix(data$lpsa)
predictor <- cbind(predictor, data$svi)

fit4 <- ols_fit(predictor, response)

#double check with lm() function.
lm(lpsa ~ lcavol + lweight + svi, data = data)$coefficients
```

```
## (Intercept)      lcavol     lweight         svi
##  -0.2680926   0.5516380   0.5085413   0.6661584
```

```
fit4[[1]]
```

```
##             [,1]
## [1,] -0.2680926
## [2,]  0.5516380
## [3,]  0.5085413
## [4,]  0.6661584
```

```
r2fit4 <- R2(fit4)
r2fit4
```

```
## [1] 0.6264403
```

```
#doulbe check with lm() summary.
summary(lm(lpsa ~ lcavol + lweight + svi, data = data))$r.squared
```

```
## [1] 0.6264403
```

```
rse4 <- sqrt(sum((fit4[[2]] - fit4[[3]])^2) / (nrow(data) - fit4[[6]]))
rse4
```

```
## [1] 0.7168094
```

```
paste("Coefficient of determination and RSE are", round(R2(fit4), 6),
      "and", round(rse4, 6), "respectively.")
```

```
## [1] "Coefficient of determination and RSE are 0.62644 and 0.716809 respectively."
```

Part b - 3

```
response <- as.matrix(data$lpsa)
predictor <- cbind(predictor, data$lbph)

fit5 <- ols_fit(predictor, response)

#double check with lm() function.
lm(lpsa ~ lcavol + lweight + svi + lbph, data = data)$coefficients
```

```
## (Intercept)      lcavol     lweight         svi        lbph
##   0.1455407   0.5496031   0.3908759   0.7117370   0.0900933
```

```
fit5[[1]]
```

```
##           [,1]
```

```
## [1,] 0.1455407
## [2,] 0.5496031
## [3,] 0.3908759
## [4,] 0.7117370
## [5,] 0.0900933
```

```
r2fit5 <- R2(fit5)
r2fit5
```

```
## [1] 0.6366035
```

```
#doulbe check with lm() summary.
summary(lm(lpsa ~ lcavol + lweight + svi + lbph, data = data))$r.squared
```

```
## [1] 0.6366035
```

```
rse5 <- sqrt(sum((fit5[[2]] - fit5[[3]])^2) / (nrow(data) - fit5[[6]]))
rse5
```

```
## [1] 0.7108232
```

```
paste("Coefficient of determination and RSE are", round(R2(fit5), 6),
      "and", round(rse5, 6), "respectively.")
```

```
## [1] "Coefficient of determination and RSE are 0.636603 and 0.710823 respectively."
```

Part b - 4

```
response <- as.matrix(data$lpsa)
predictor <- cbind(predictor, data$age)

fit6 <- ols_fit(predictor, response)

#double check with lm() function.
lm(lpsa ~ lcavol + lweight + svi + lbph + age, data = data)$coefficients
```

```
## (Intercept)      lcavol     lweight         svi        lbph         age
##  0.95099742  0.56560801  0.42369200  0.72095499  0.11183992 -0.01489225
```

```
fit6[[1]]
```

```
##             [,1]
## [1,]  0.95099742
## [2,]  0.56560801
## [3,]  0.42369200
## [4,]  0.72095499
## [5,]  0.11183992
## [6,] -0.01489225
```

```
r2fit6 <- R2(fit6)
r2fit6
```

```
## [1] 0.6441024
```

```
#doulbe check with lm() summary.
summary(lm(lpsa ~ lcavol + lweight + svi + lbph + age, data = data))$r.squared
```

```
## [1] 0.6441024
```

```
rse6 <- sqrt(sum((fit6[[2]] - fit6[[3]])^2) / (nrow(data) - fit6[[6]]))
rse6
```

```
## [1] 0.7073054
```

```
paste("Coefficient of determination and RSE are", round(R2(fit6), 6),
      "and", round(rse6, 6), "respectively.")
```

```
## [1] "Coefficient of determination and RSE are 0.644102 and 0.707305 respectively."
```

Part b - 5

```
response <- as.matrix(data$lpsa)
predictor <- cbind(predictor, data$lcp)

fit7 <- ols_fit(predictor, response)

#double check with lm() function.
lm(lpsa ~ lcavol + lweight + svi + lbph + age + lcp, data = data)$coefficients
```

```
## (Intercept)      lcavol     lweight         svi        lbph         age
##  0.93486843  0.58764668  0.41808376  0.78256448  0.11381222 -0.01511242
##         lcp
## -0.04118380
```

```
fit7[[1]]
```

```
##             [,1]
## [1,]  0.93486843
## [2,]  0.58764668
## [3,]  0.41808376
## [4,]  0.78256448
## [5,]  0.11381222
## [6,] -0.01511242
## [7,] -0.04118380
```

```
r2fit7 <- R2(fit7)
r2fit7
```

```
## [1] 0.645113
```

```
#doulbe check with lm() summary.
summary(lm(lpsa ~ lcavol + lweight + svi + lbph + age + lcp, data = data))$r.squared
```

```
## [1] 0.645113
```

```
rse7 <- sqrt(sum((fit7[[2]] - fit7[[3]])^2) / (nrow(data) - fit7[[6]]))
rse7
```

```
## [1] 0.7102135
```

```
paste("Coefficient of determination and RSE are", round(R2(fit7), 6),
      "and", round(rse7, 6), "respectively.")
```

```
## [1] "Coefficient of determination and RSE are 0.645113 and 0.710214 respectively."
```

Part b - 6

```
response <- as.matrix(data$lpsa)
predictor <- cbind(predictor, data$pgg45)

fit8 <- ols_fit(predictor, response)

#double check with lm() function.
lm(lpsa ~ lcavol + lweight + svi + lbph + age + lcp + pgg45, data = data)$coefficients
```

```
##  (Intercept)       lcavol      lweight          svi         lbph
##  0.953926041  0.591614545  0.448292433  0.757733506  0.107671072
##          age          lcp        pgg45
## -0.019336452 -0.104482266  0.005317704
```

```
fit8[[1]]
```

```
##              [,1]
## [1,]  0.953926041
## [2,]  0.591614545
## [3,]  0.448292433
## [4,]  0.757733506
## [5,]  0.107671072
## [6,] -0.019336452
## [7,] -0.104482266
## [8,]  0.005317704
```

```
r2fit8 <- R2(fit8)
r2fit8
```

```
## [1] 0.6544317
```

```
#doulbe check with lm() summary.
summary(lm(lpsa ~ lcavol + lweight + svi + lbph + age + lcp + pgg45, data = data))$r.squared
```

```
## [1] 0.6544317
```

```
rse8 <- sqrt(sum((fit8[[2]] - fit8[[3]])^2) / (nrow(data) - fit8[[6]]))
rse8
```

```
## [1] 0.7047533
```

```
paste("Coefficient of determination and RSE are", round(R2(fit8), 6),
      "and", round(rse8, 6), "respectively.")
```

```
## [1] "Coefficient of determination and RSE are 0.654432 and 0.704753 respectively."
```

Part b - 7

```
response <- as.matrix(data$lpsa)
predictor <- cbind(predictor, data$gleason)
```

```
fit9 <- ols_fit(predictor, response)

#double check with lm() function.
lm(lpsa ~ lcavol + lweight + svi + lbph + age + lcp + pgg45 + gleason,
   data = data)$coefficients
```

```
##  (Intercept)       lcavol      lweight          svi         lbph
##  0.669336698  0.587021826  0.454467424  0.766157326  0.107054031
##          age          lcp        pgg45      gleason
## -0.019637176 -0.105474263  0.004525231  0.045141598
```

```
fit9[[1]]
```

```
##               [,1]
##  [1,]  0.669336698
##  [2,]  0.587021826
##  [3,]  0.454467424
##  [4,]  0.766157326
##  [5,]  0.107054031
##  [6,] -0.019637176
##  [7,] -0.105474263
##  [8,]  0.004525231
##  [9,]  0.045141598
```

```
r2fit9 <- R2(fit9)
r2fit9
```

```
## [1] 0.6547541
```

```
#doulbe check with lm() summary.
summary(lm(lpsa ~ lcavol + lweight + svi + lbph + age + lcp + pgg45 + gleason, data = data))$r.squared
```

```
## [1] 0.6547541
```

```
rse9 <- sqrt(sum((fit9[[2]] - fit9[[3]])^2) / (nrow(data) - fit9[[6]]))
rse9
```

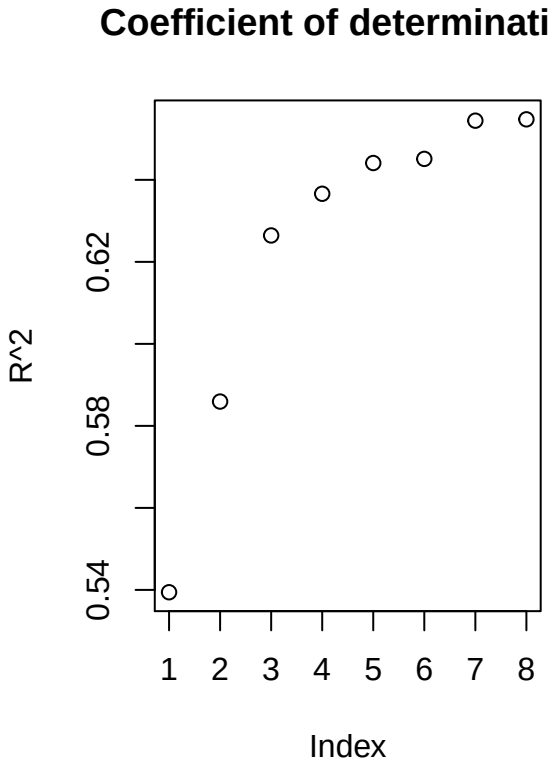
```
## [1] 0.7084155
```

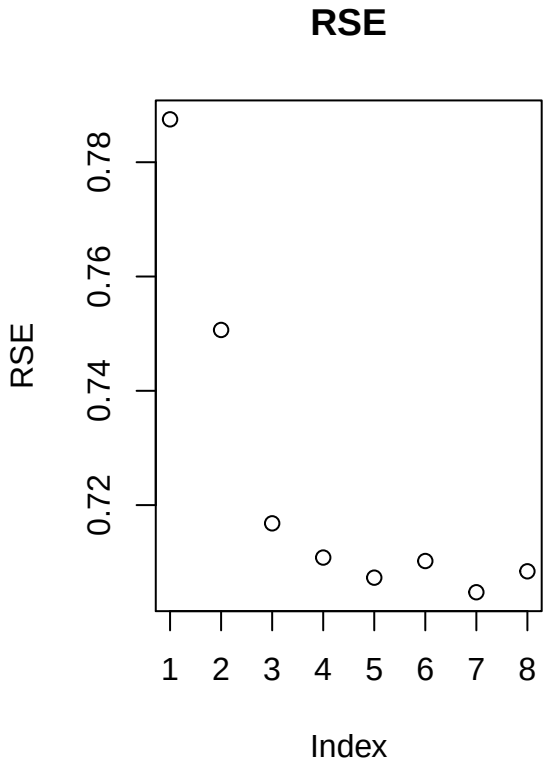
```
paste("Coefficient of determination and RSE are", round(R2(fit9), 6),
      "and", round(rse9, 6), "respectively.")
```

```
## [1] "Coefficient of determination and RSE are 0.654754 and 0.708416 respectively."
```

Part c

```
r2_list <- c(r2fit2, r2fit3, r2fit4, r2fit5, r2fit6, r2fit7, r2fit8, r2fit9)
rse_list <- c(rse2, rse3, rse4, rse5, rse6, rse7, rse8, rse9)

par(mfrow = c(1,2))
plot(r2_list, ylab = "R^2", main = "Coefficient of determination")
plot(rse_list, ylab = "RSE", main = "RSE")
```





COMMENT: R^2 strictly goes up, as we add variables. $R^2 = \frac{Regss}{TSS} = \frac{\sum(\hat{y}_i - \bar{y})^2}{\sum(y_i - \bar{y})^2}$. Since RSS strictly goes down as we add variables, we can say R^2 should go up always.

However, RSE is different. RSE $= \sqrt{\frac{RSS}{n - p - 1}} = \sqrt{\frac{\sum(y_i - \hat{y}_i)^2}{n - p - 1}}$, where p is the number of predictors/x/explanatory variables. Although RSE is a function of RSS and we know RSS goes down as we add variables, since the denominator changes as we add variables, RSE sometimes goes up although we add varaibles. But, in general, RSE should have a trend of going down.

Problem 7

Data importing

There are 5 missing values (marked as ? in the data set), so I would need to modify/change/clean it. Another problem of horsepower column is, its class is factor, which is hard to modify.

```
data2 <- download.file("http://www-bcf.usc.edu/~gareth/ISL/Auto.data", destfile = "auto.txt")

data2 <- read.table("auto.txt", header = T, stringsAsFactors = T)

data2$horsepower <- as.numeric(levels(data2$horsepower))[data2$horsepower]
```

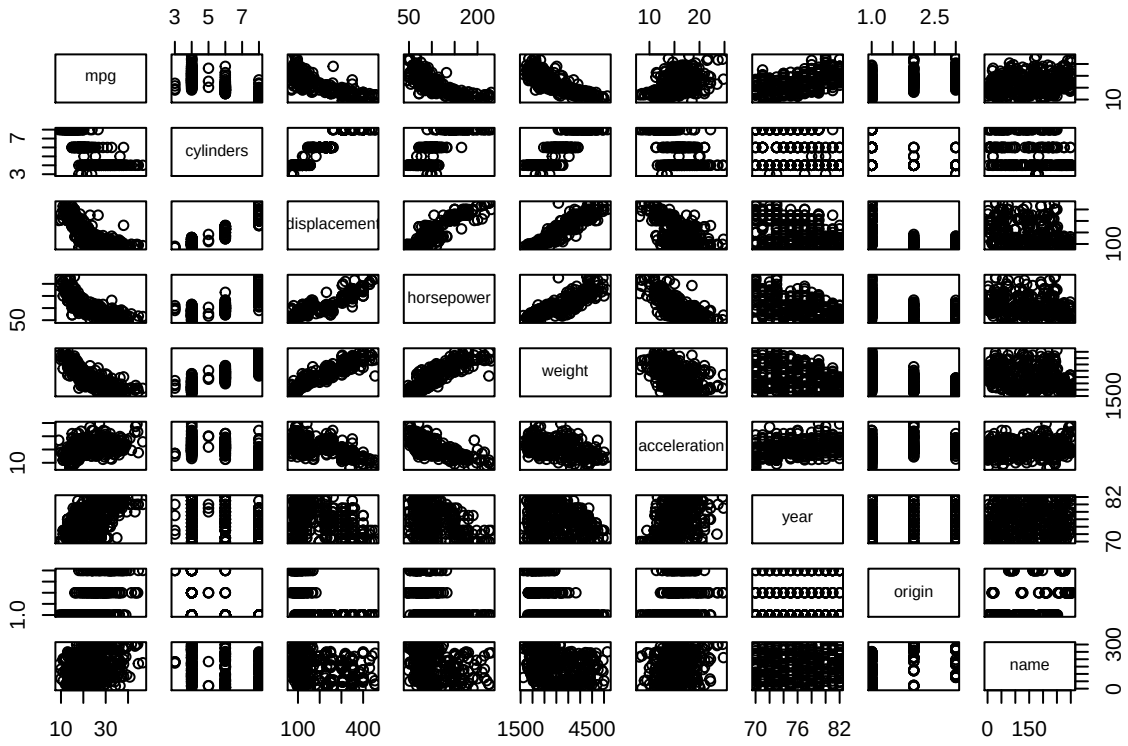
```
## Warning: NAs introduced by coercion
```

```
data2 <- na.omit(data2)
```

Part a

Scatterplot matrix

```
plot(data2)
```



Part b

```
cor_dat <- data2[,-ncol(data2)] #exclude name variable.
cor_dat$horsepower <- as.numeric(cor_dat$horsepower) #horsepower is factor.

cor(cor_dat)
```

```
##                     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
```

Part c

Year, origin, and cylinder should be considred as dummy variables, but since professor did not mention it, I will just regard them as quantitaitive/numerical.

```
lms <- lm(mpg ~., data = cor_dat)
summary(lms)
```

```
##
## Call:
## lm(formula = mpg ~ ., data = cor_dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.5903 -2.1565 -0.1169  1.8690 13.0604
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -17.218435   4.644294  -3.707  0.00024 ***
## cylinders     -0.493376   0.323282  -1.526  0.12780
## displacement   0.019896   0.007515   2.647  0.00844 **
## horsepower    -0.016951   0.013787  -1.230  0.21963
## weight        -0.006474   0.000652  -9.929  < 2e-16 ***
## acceleration   0.080576   0.098845   0.815  0.41548
## year           0.750773   0.050973  14.729  < 2e-16 ***
## origin         1.426141   0.278136   5.127 4.67e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.328 on 384 degrees of freedom
## Multiple R-squared:  0.8215, Adjusted R-squared:  0.8182
## F-statistic: 252.4 on 7 and 384 DF,  p-value: < 2.2e-16
```

COMMENT:

- **Is there a relationship between the predictors and the response?**

As we can see from the p-values, some of the predictors do not form a a good linear relationship with the response such as cylinders and horsepower (with significance level $\alpha = 0.05$). Also, I think year, origin, and cylinder should be regarded as categorical/dummy variables rather than just putting them into lm() function as quantitative variables. So, I would say some of the predictors shows a relationship with the response, but not all of them. Also, high R^2 implies that the model can be explained better when we included some of predictors.

- **Which predictors appear to have a statistically significant relationship to the response?**

By looking at p-values, I will say weight and year are the most significant predictors. Also, displacement and origin shows quite high significance, but again, since origin should be considered as dummy variable, this might not be correct. Also, all these linear modelings are based on the assumption that noise is normally distributed with constant variance. However, this assumption might not be appropriate.

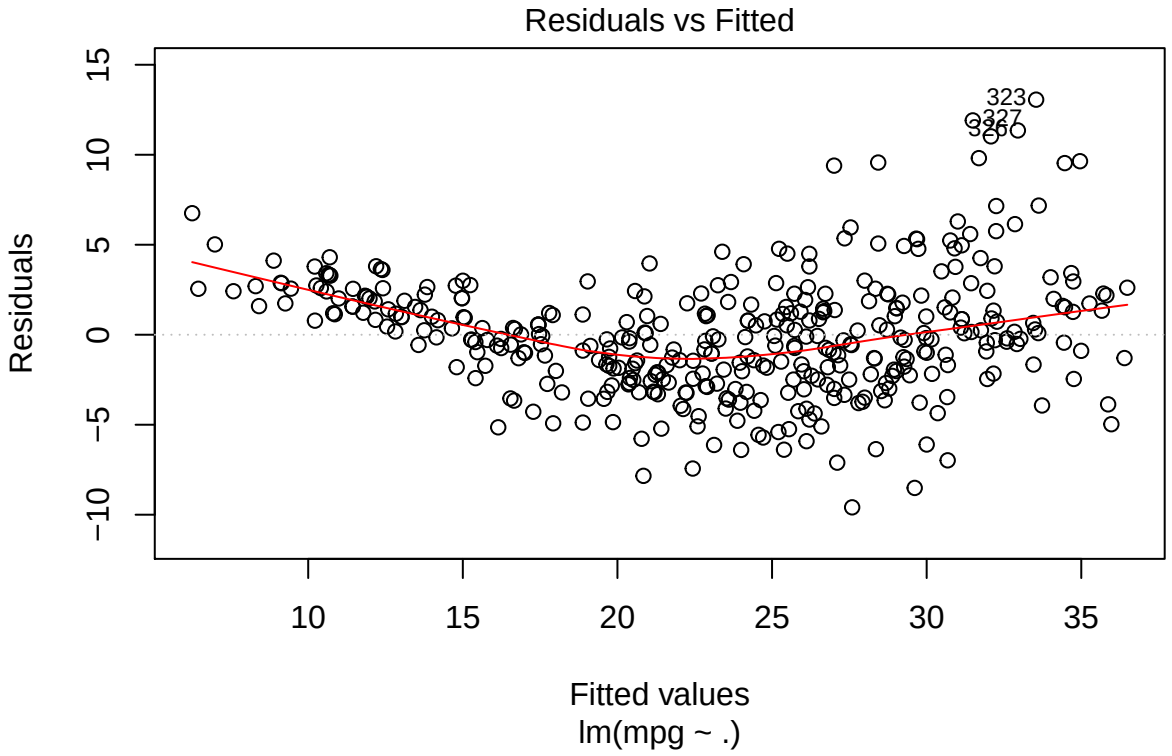
- **What does the coefficient for the year variable suggest?**

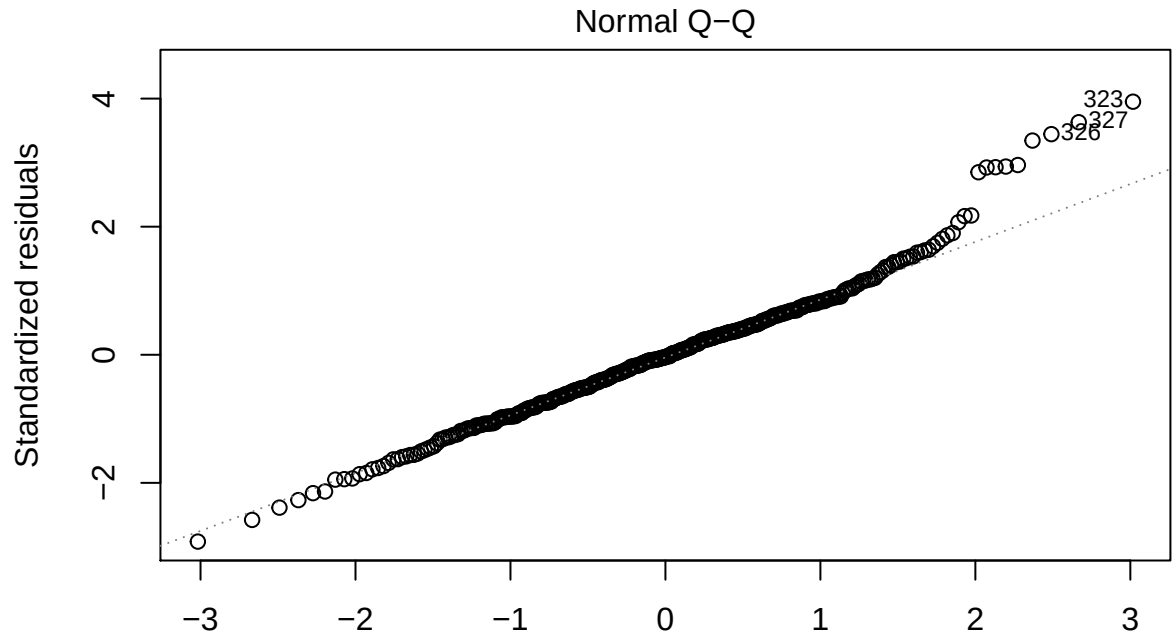
When we consider all the other variables being fixed, as year goes up by 1 unit, mpg will goes up by 0.7 unit. So, year and mpg are positively related.

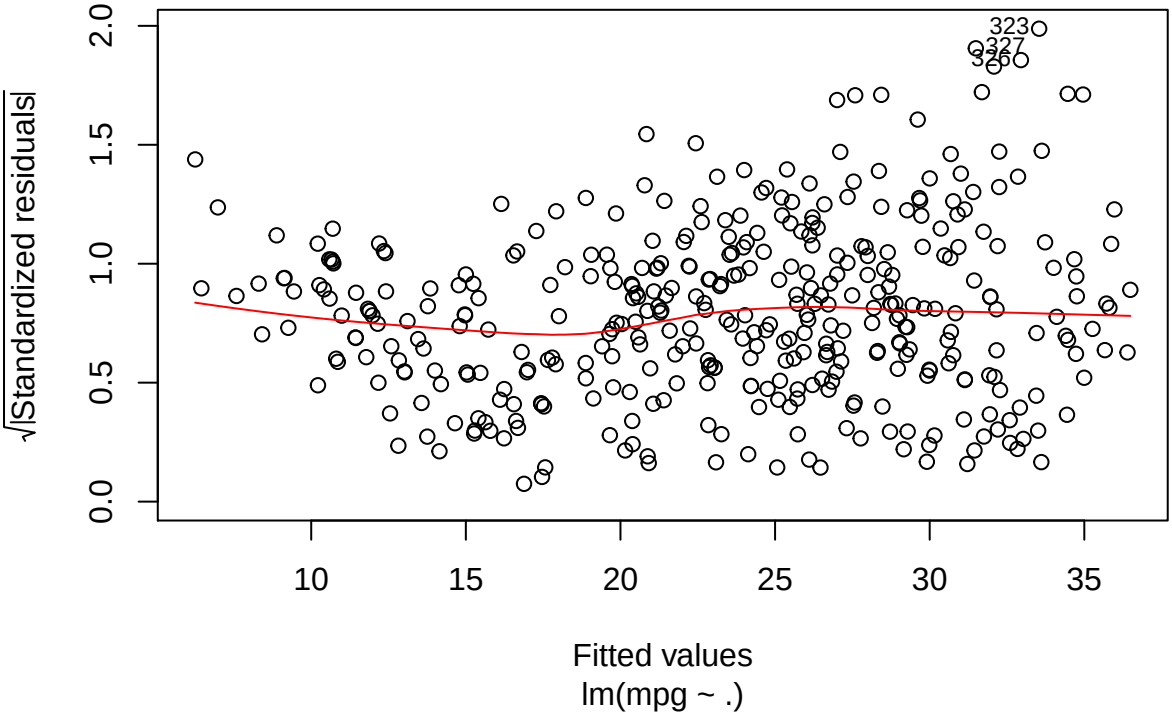
Part d

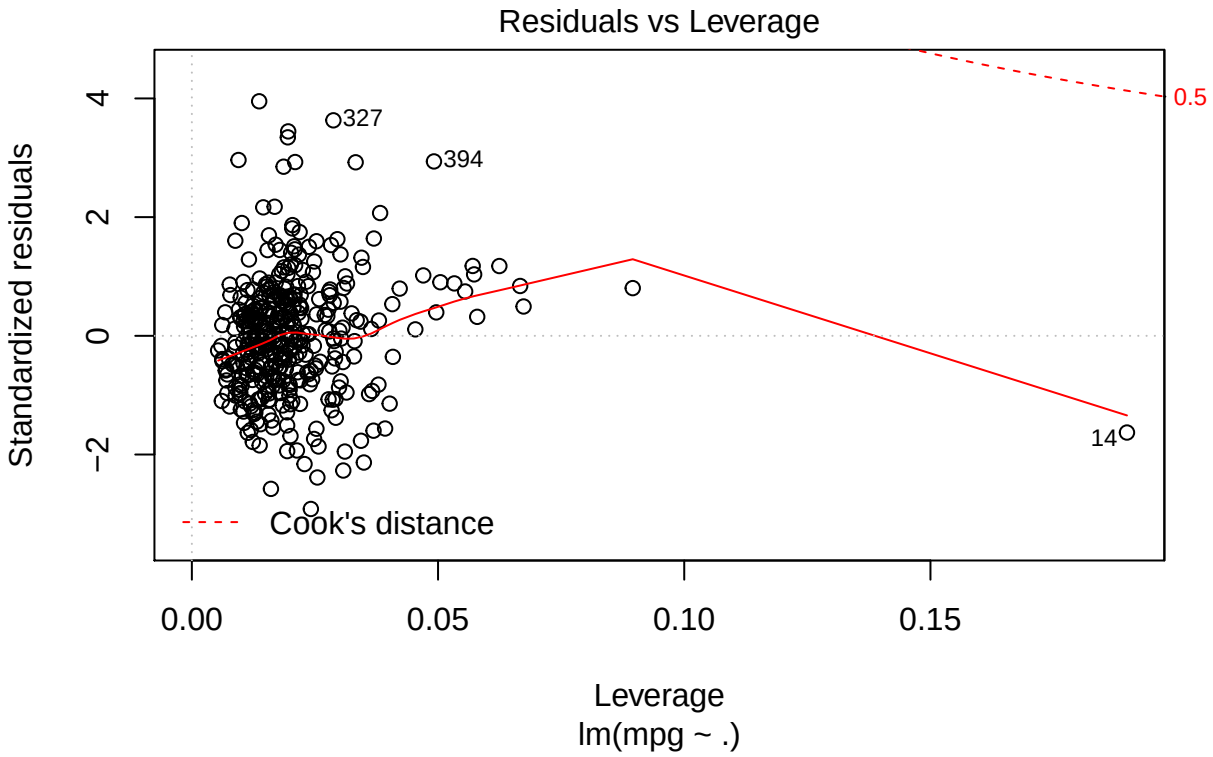
Leverage plot shows how each of the fitted value is influential. (and this helps to spot the outliers)

```
plot(lms)
```




Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(mpg ~ .)
323
327
326


Scale−Location
√|Standardized residuals|
Fitted values
lm(mpg ~ .)
323
327
326



COMMENT:

When I fit the plot with fitted values v.s. residuals, they show the pattern. This implies that our linear modeling does violate our assumption (we assumed that $\hat{y}$ is independent with residuals and variance of the noise is constant). Also, the variance of residual seems to increase as fitted values go up, and this is heteroscedasitic.

We can say that there are some outliers, but I will not say there is any unusual **large** outliers for this.

There is an observation on the far right side (showing 14 on the dot) with high leverage. However, this is not influential, as this is not an outlier. So, that is why we still have high R^2. So, we can say that one element can be far away with the other x's.

Part e

If we use *, we include both of main/single and interaction effects, and if we use :, we include only interaction effect.

```
lms2 <- lm(mpg ~ .*., data = cor_dat)
summary(lms2)
```

```
##
## Call:
## lm(formula = mpg ~ . * ., data = cor_dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.6303 -1.4481  0.0596  1.2739 11.1386
##
```

```
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 3.548e+01  5.314e+01   0.668  0.50475
## cylinders                   6.989e+00  8.248e+00   0.847  0.39738
## displacement               -4.785e-01  1.894e-01  -2.527  0.01192 *
## horsepower                  5.034e-01  3.470e-01   1.451  0.14769
## weight                      4.133e-03  1.759e-02   0.235  0.81442
## acceleration               -5.859e+00  2.174e+00  -2.696  0.00735 **
## year                        6.974e-01  6.097e-01   1.144  0.25340
## origin                     -2.090e+01  7.097e+00  -2.944  0.00345 **
## cylinders:displacement     -3.383e-03  6.455e-03  -0.524  0.60051
## cylinders:horsepower        1.161e-02  2.420e-02   0.480  0.63157
## cylinders:weight            3.575e-04  8.955e-04   0.399  0.69000
## cylinders:acceleration      2.779e-01  1.664e-01   1.670  0.09584 .
## cylinders:year             -1.741e-01  9.714e-02  -1.793  0.07389 .
## cylinders:origin            4.022e-01  4.926e-01   0.816  0.41482
## displacement:horsepower    -8.491e-05  2.885e-04  -0.294  0.76867
## displacement:weight         2.472e-05  1.470e-05   1.682  0.09342 .
## displacement:acceleration -3.479e-03  3.342e-03  -1.041  0.29853
## displacement:year           5.934e-03  2.391e-03   2.482  0.01352 *
## displacement:origin         2.398e-02  1.947e-02   1.232  0.21875
## horsepower:weight          -1.968e-05  2.924e-05  -0.673  0.50124
## horsepower:acceleration    -7.213e-03  3.719e-03  -1.939  0.05325 .
## horsepower:year            -5.838e-03  3.938e-03  -1.482  0.13916
## horsepower:origin           2.233e-03  2.930e-02   0.076  0.93931
## weight:acceleration         2.346e-04  2.289e-04   1.025  0.30596
## weight:year                -2.245e-04  2.127e-04  -1.056  0.29182
## weight:origin              -5.789e-04  1.591e-03  -0.364  0.71623
## acceleration:year           5.562e-02  2.558e-02   2.174  0.03033 *
## acceleration:origin         4.583e-01  1.567e-01   2.926  0.00365 **
## year:origin                 1.393e-01  7.399e-02   1.882  0.06062 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.695 on 363 degrees of freedom
## Multiple R-squared:  0.8893, Adjusted R-squared:  0.8808
## F-statistic: 104.2 on 28 and 363 DF,  p-value: < 2.2e-16
```

```
lms3 <- lm(mpg ~ .:., data = cor_dat)
summary(lms3)
```

```
##
## Call:
## lm(formula = mpg ~ .:., data = cor_dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.6303 -1.4481  0.0596  1.2739 11.1386
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 3.548e+01  5.314e+01   0.668  0.50475
## cylinders                   6.989e+00  8.248e+00   0.847  0.39738
## displacement               -4.785e-01  1.894e-01  -2.527  0.01192 *
## horsepower                  5.034e-01  3.470e-01   1.451  0.14769
```

```
## weight                     4.133e-03  1.759e-02   0.235  0.81442
## acceleration              -5.859e+00  2.174e+00  -2.696  0.00735 **
## year                       6.974e-01  6.097e-01   1.144  0.25340
## origin                    -2.090e+01  7.097e+00  -2.944  0.00345 **
## cylinders:displacement    -3.383e-03  6.455e-03  -0.524  0.60051
## cylinders:horsepower       1.161e-02  2.420e-02   0.480  0.63157
## cylinders:weight           3.575e-04  8.955e-04   0.399  0.69000
## cylinders:acceleration     2.779e-01  1.664e-01   1.670  0.09584 .
## cylinders:year            -1.741e-01  9.714e-02  -1.793  0.07389 .
## cylinders:origin           4.022e-01  4.926e-01   0.816  0.41482
## displacement:horsepower   -8.491e-05  2.885e-04  -0.294  0.76867
## displacement:weight        2.472e-05  1.470e-05   1.682  0.09342 .
## displacement:acceleration -3.479e-03  3.342e-03  -1.041  0.29853
## displacement:year          5.934e-03  2.391e-03   2.482  0.01352 *
## displacement:origin        2.398e-02  1.947e-02   1.232  0.21875
## horsepower:weight         -1.968e-05  2.924e-05  -0.673  0.50124
## horsepower:acceleration   -7.213e-03  3.719e-03  -1.939  0.05325 .
## horsepower:year           -5.838e-03  3.938e-03  -1.482  0.13916
## horsepower:origin          2.233e-03  2.930e-02   0.076  0.93931
## weight:acceleration        2.346e-04  2.289e-04   1.025  0.30596
## weight:year               -2.245e-04  2.127e-04  -1.056  0.29182
## weight:origin             -5.789e-04  1.591e-03  -0.364  0.71623
## acceleration:year          5.562e-02  2.558e-02   2.174  0.03033 *
## acceleration:origin        4.583e-01  1.567e-01   2.926  0.00365 **
## year:origin                1.393e-01  7.399e-02   1.882  0.06062 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.695 on 363 degrees of freedom
## Multiple R-squared:  0.8893, Adjusted R-squared:  0.8808
## F-statistic: 104.2 on 28 and 363 DF,  p-value: < 2.2e-16
```

COMMENT: Yes! the interaction does not seem to be that statistically significant, so we might need to investigate further. For some of them, for example, interaction between acceleration and origin seem to have pretty significant; however, I personally think it means nothing. As I said above, origin should be regarded as dummmy variable. It is quite subjective. Some people think any interaction would be statistically significant enough to investigate further, but I do not think we need to do that for most of interactions. (However, interaction between acceleartion and year seem pretty interesting, and I want to investigate more) Statisticians always investigate interaction effect first, and if there is some statistical significance, there is no use to investigate single effect.

Part f

```r
#Only X's were transformed.
log_dat <- log(cor_dat)
log_dat <- log_dat[,-1]
log_dat <- cbind(mpg = cor_dat$mpg, log_dat)

lms_log <- lm(mpg ~., data = log_dat)
summary(lms_log)
```

```
##
```

```
## Call:
## lm(formula = mpg ~ ., data = log_dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.5987 -1.8172 -0.0181  1.5906 12.8132
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -66.5643    17.5053  -3.803 0.000167 ***
## cylinders      1.4818     1.6589   0.893 0.372273
## displacement  -1.0551     1.5385  -0.686 0.493230
## horsepower    -6.9657     1.5569  -4.474 1.01e-05 ***
## weight       -12.5728     2.2251  -5.650 3.12e-08 ***
## acceleration  -4.9831     1.6078  -3.099 0.002082 **
## year          54.9857     3.5555  15.465  < 2e-16 ***
## origin         1.5822     0.5083   3.113 0.001991 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.069 on 384 degrees of freedom
## Multiple R-squared:  0.8482, Adjusted R-squared:  0.8454
## F-statistic: 306.5 on 7 and 384 DF,  p-value: < 2.2e-16
```

```
root_dat <- sqrt(cor_dat)
root_dat <- root_dat[,-1]
root_dat <- cbind(mpg = cor_dat$mpg, root_dat)

lms_root <- lm(mpg ~., data = root_dat)
summary(lms_root)
```

```
##
## Call:
## lm(formula = mpg ~ ., data = root_dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.5250 -1.9822 -0.1111  1.7347 13.0681
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -49.79814    9.17832  -5.426 1.02e-07 ***
## cylinders     -0.23699    1.53753  -0.154   0.8776
## displacement   0.22580    0.22940   0.984   0.3256
## horsepower    -0.77976    0.30788  -2.533   0.0117 *
## weight        -0.62172    0.07898  -7.872 3.59e-14 ***
## acceleration  -0.82529    0.83443  -0.989   0.3233
## year          12.79030    0.85891  14.891  < 2e-16 ***
## origin         3.26036    0.76767   4.247 2.72e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.21 on 384 degrees of freedom
## Multiple R-squared:  0.8338, Adjusted R-squared:  0.8308
## F-statistic: 275.3 on 7 and 384 DF,  p-value: < 2.2e-16
```

```r
square_dat <- (cor_dat)^2
square_dat <- square_dat[,-1]
square_dat <- as.data.frame(cbind(mpg = cor_dat$mpg, square_dat))

lms_square <- lm(mpg ~., data = square_dat)
summary(lms_square)
```

```
##
## Call:
## lm(formula = mpg ~ ., data = square_dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.6786 -2.3227 -0.0582  1.9073 12.9807
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.208e+00  2.356e+00   0.513 0.608382
## cylinders    -8.829e-02  2.521e-02  -3.502 0.000515 ***
## displacement  5.680e-05  1.382e-05   4.109 4.87e-05 ***
## horsepower   -3.621e-05  4.975e-05  -0.728 0.467201
## weight       -9.351e-07  8.978e-08 -10.416  < 2e-16 ***
## acceleration  6.278e-03  2.690e-03   2.334 0.020130 *
## year          4.999e-03  3.530e-04  14.160  < 2e-16 ***
## origin        4.129e-01  6.914e-02   5.971 5.37e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.539 on 384 degrees of freedom
## Multiple R-squared:  0.7981, Adjusted R-squared:  0.7944
## F-statistic: 216.8 on 7 and 384 DF,  p-value: < 2.2e-16
```

```r
#X and y are both tranformed.
log_dat2 <- log(cor_dat)

lms_log2 <- lm(mpg ~., data = log_dat2)
summary(lms_log2)
```

```
##
## Call:
## lm(formula = mpg ~ ., data = log_dat2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.41298 -0.07098  0.00055  0.06150  0.39532
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.155391   0.648230  -0.240  0.81068
## cylinders    -0.082815   0.061429  -1.348  0.17841
## displacement  0.006625   0.056970   0.116  0.90748
## horsepower   -0.294389   0.057652  -5.106 5.18e-07 ***
## weight       -0.569666   0.082397  -6.914 1.98e-11 ***
## acceleration -0.179239   0.059536  -3.011  0.00278 **
```

```
## year          2.243989   0.131661  17.044  < 2e-16 ***
## origin        0.044848   0.018821   2.383  0.01767 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1136 on 384 degrees of freedom
## Multiple R-squared:  0.8903, Adjusted R-squared:  0.8883
## F-statistic: 445.3 on 7 and 384 DF,  p-value: < 2.2e-16
```

```
root_dat2 <- sqrt(cor_dat)

lms_root2 <- lm(mpg ~., data = root_dat2)
summary(lms_root2)
```

```
##
## Call:
## lm(formula = mpg ~ ., data = root_dat2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.98667 -0.17280 -0.00315  0.16145  1.02245
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.949286   0.847481  -2.300 0.021979 *
## cylinders    -0.108552   0.141968  -0.765 0.444964
## displacement  0.019707   0.021182   0.930 0.352752
## horsepower   -0.090896   0.028428  -3.197 0.001502 **
## weight       -0.061414   0.007292  -8.422 7.48e-16 ***
## acceleration -0.107258   0.077048  -1.392 0.164699
## year          1.266015   0.079308  15.963  < 2e-16 ***
## origin        0.272324   0.070883   3.842 0.000143 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2964 on 384 degrees of freedom
## Multiple R-squared:  0.8662, Adjusted R-squared:  0.8638
## F-statistic: 355.1 on 7 and 384 DF,  p-value: < 2.2e-16
```

```
square_dat2 <- as.data.frame((cor_dat)^2)

lms_square2 <- lm(mpg ~., data = square_dat2)
summary(lms_square2)
```

```
##
## Call:
## lm(formula = mpg ~ ., data = square_dat2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -501.89 -145.36  -18.91  111.41 1034.08
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -7.523e+02  1.456e+02  -5.165 3.87e-07 ***
```

```
## cylinders    -3.746e+00  1.559e+00  -2.403 0.016713 *
## displacement  3.356e-03  8.547e-04   3.926 0.000102 ***
## horsepower    1.279e-04  3.076e-03   0.042 0.966851
## weight       -4.833e-05  5.551e-06  -8.707  < 2e-16 ***
## acceleration  4.892e-01  1.663e-01   2.941 0.003474 **
## year          2.731e-01  2.183e-02  12.513  < 2e-16 ***
## origin        2.608e+01  4.275e+00   6.101 2.57e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 218.8 on 384 degrees of freedom
## Multiple R-squared:  0.7055, Adjusted R-squared:  0.7001
## F-statistic: 131.4 on 7 and 384 DF,  p-value: < 2.2e-16
```

```
#Only y's are transformed.
log_mpg <- log(cor_dat[,1])
log_dat3 <- cbind(mpg = log_mpg, cor_dat[,-1])

lms_log3 <- lm(mpg ~., data = log_dat3)
summary(lms_log3)
```

```
##
## Call:
## lm(formula = mpg ~ ., data = log_dat3)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.40955 -0.06533  0.00079  0.06785  0.33925
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.751e+00  1.662e-01  10.533  < 2e-16 ***
## cylinders    -2.795e-02  1.157e-02  -2.415  0.01619 *
## displacement  6.362e-04  2.690e-04   2.365  0.01852 *
## horsepower   -1.475e-03  4.935e-04  -2.989  0.00298 **
## weight       -2.551e-04  2.334e-05 -10.931  < 2e-16 ***
## acceleration -1.348e-03  3.538e-03  -0.381  0.70339
## year          2.958e-02  1.824e-03  16.211  < 2e-16 ***
## origin        4.071e-02  9.955e-03   4.089 5.28e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1191 on 384 degrees of freedom
## Multiple R-squared:  0.8795, Adjusted R-squared:  0.8773
## F-statistic: 400.4 on 7 and 384 DF,  p-value: < 2.2e-16
```

```
root_mpg <- sqrt(cor_dat[,1])
root_dat3 <- cbind(mpg = root_mpg, cor_dat[,-1])

lms_root3 <- lm(mpg ~., data = root_dat3)
summary(lms_root3)
```

```
##
## Call:
## lm(formula = mpg ~ ., data = root_dat3)
```

```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.98891 -0.18946  0.00505  0.16947  1.02581
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.075e+00  4.290e-01   2.506   0.0126 *
## cylinders    -5.942e-02  2.986e-02  -1.990   0.0474 *
## displacement  1.752e-03  6.942e-04   2.524   0.0120 *
## horsepower   -2.512e-03  1.274e-03  -1.972   0.0493 *
## weight       -6.367e-04  6.024e-05 -10.570  < 2e-16 ***
## acceleration  2.738e-03  9.131e-03   0.300   0.7644
## year          7.381e-02  4.709e-03  15.675  < 2e-16 ***
## origin        1.217e-01  2.569e-02   4.735 3.09e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3074 on 384 degrees of freedom
## Multiple R-squared:  0.8561, Adjusted R-squared:  0.8535
## F-statistic: 326.3 on 7 and 384 DF,  p-value: < 2.2e-16
```

```
square_mpg <- (cor_dat[,1])^2
square_dat3 <- as.data.frame(cbind(mpg = square_mpg, cor_dat[,-1]))

lms_square3 <- lm(mpg ~., data = square_dat3)
summary(lms_square3)
```

```
##
## Call:
## lm(formula = mpg ~ ., data = square_dat3)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -483.45 -141.87  -19.62  103.58 1042.84
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.878e+03  2.928e+02  -6.412 4.22e-10 ***
## cylinders    -1.436e+01  2.038e+01  -0.704  0.48157
## displacement  1.328e+00  4.738e-01   2.802  0.00534 **
## horsepower   -3.587e-01  8.693e-01  -0.413  0.68009
## weight       -3.522e-01  4.111e-02  -8.567 2.62e-16 ***
## acceleration  9.278e+00  6.232e+00   1.489  0.13740
## year          4.081e+01  3.214e+00  12.698  < 2e-16 ***
## origin        9.509e+01  1.754e+01   5.422 1.04e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 209.8 on 384 degrees of freedom
## Multiple R-squared:  0.7292, Adjusted R-squared:  0.7243
## F-statistic: 147.8 on 7 and 384 DF,  p-value: < 2.2e-16
```

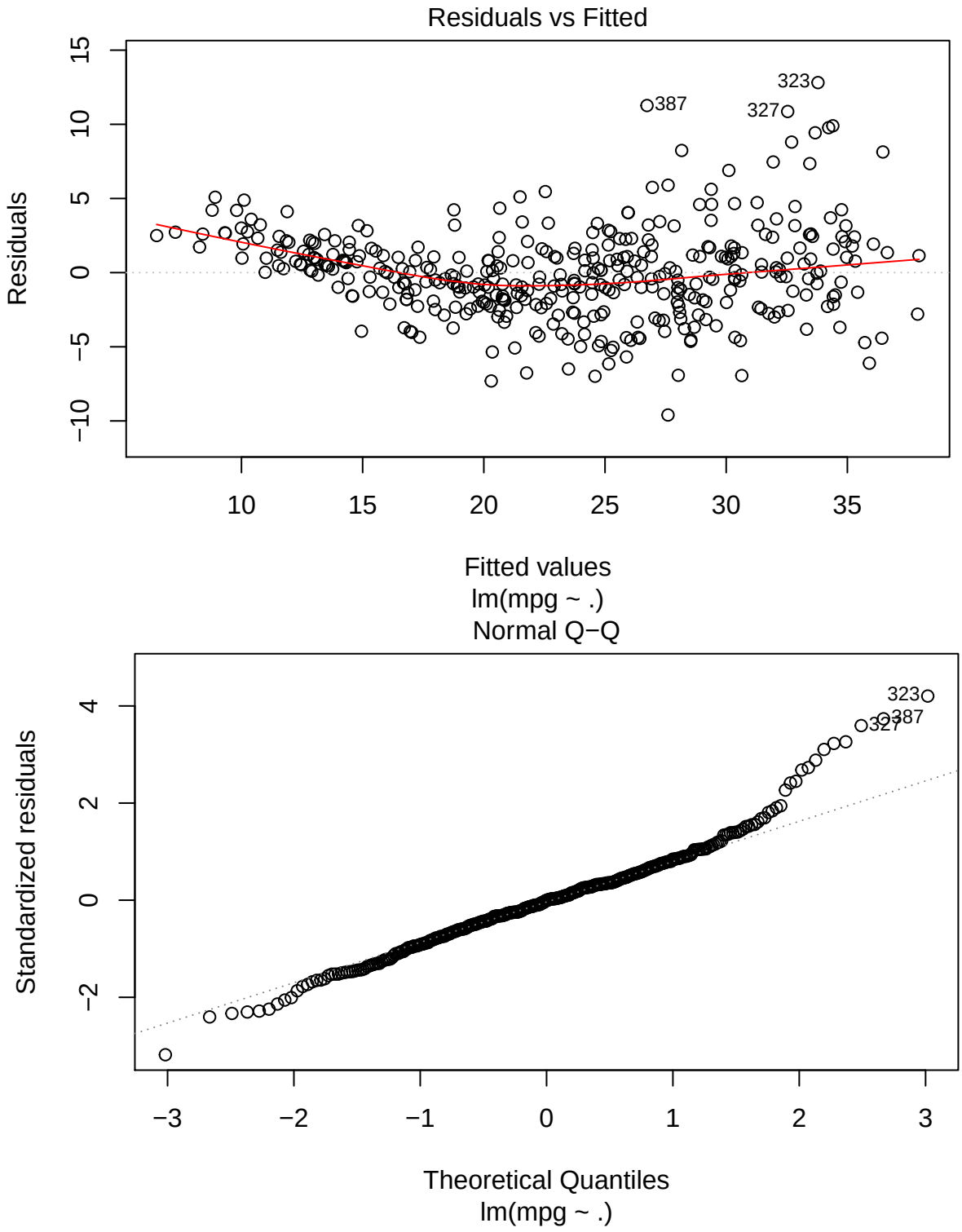
COMMENT: I think log transformation on both X and Y makes the linear modelling better when I look at the plot of fitted values v.s. residuals, as I cannot see a pattern anymore. Square root transformation on

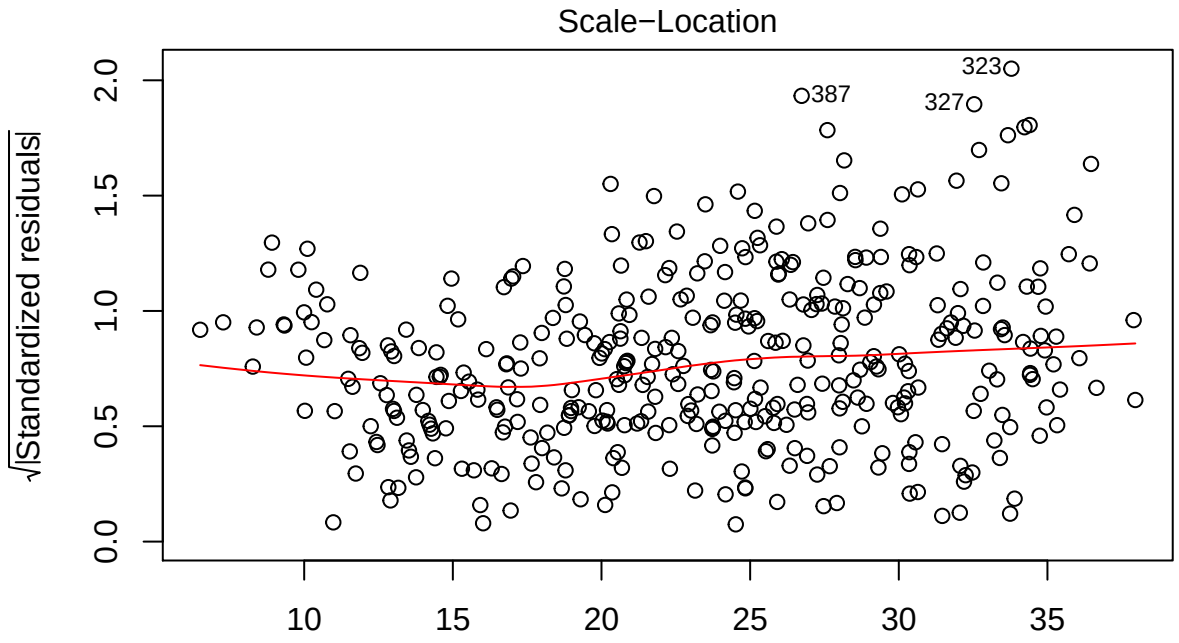
both X and Y also does decent job of making better linear modeling. However, square transformation on both X and Y makes the linear modeling worse. (with our assumption of variance of noise is constant) Also, R^2 back up my reasoning.

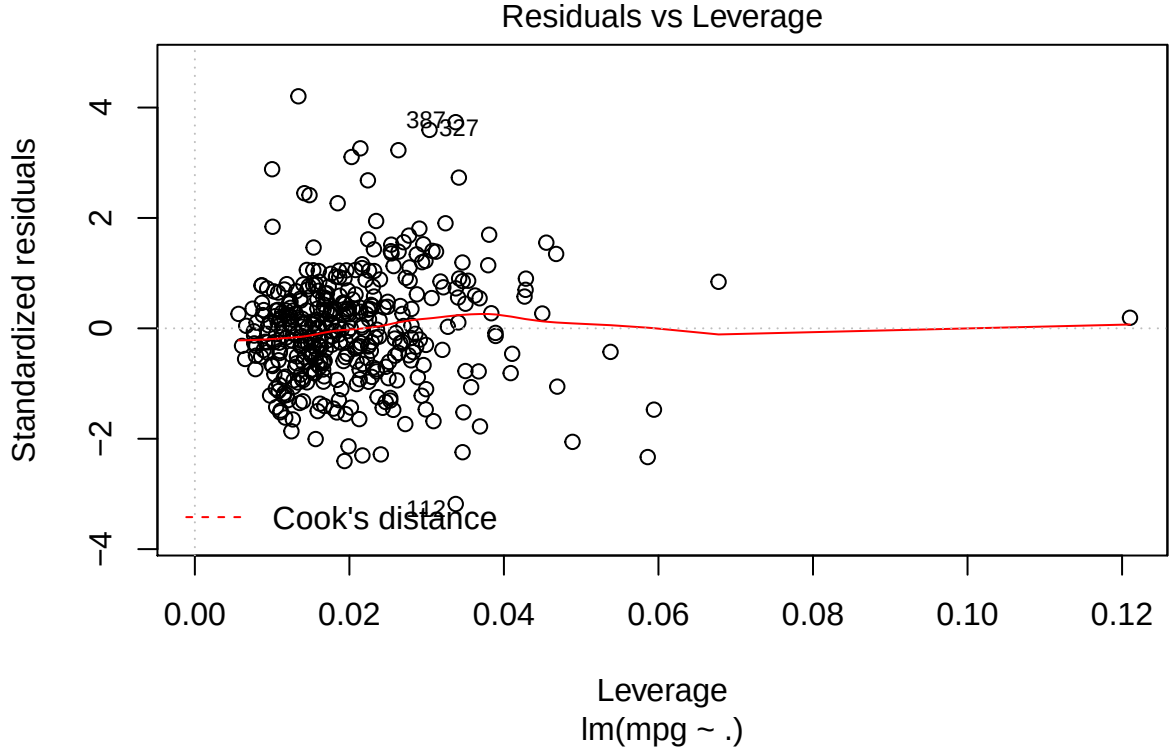
I added diagnostic plots and lm() summary to help interpretations.

Extra-7F for interpretation!

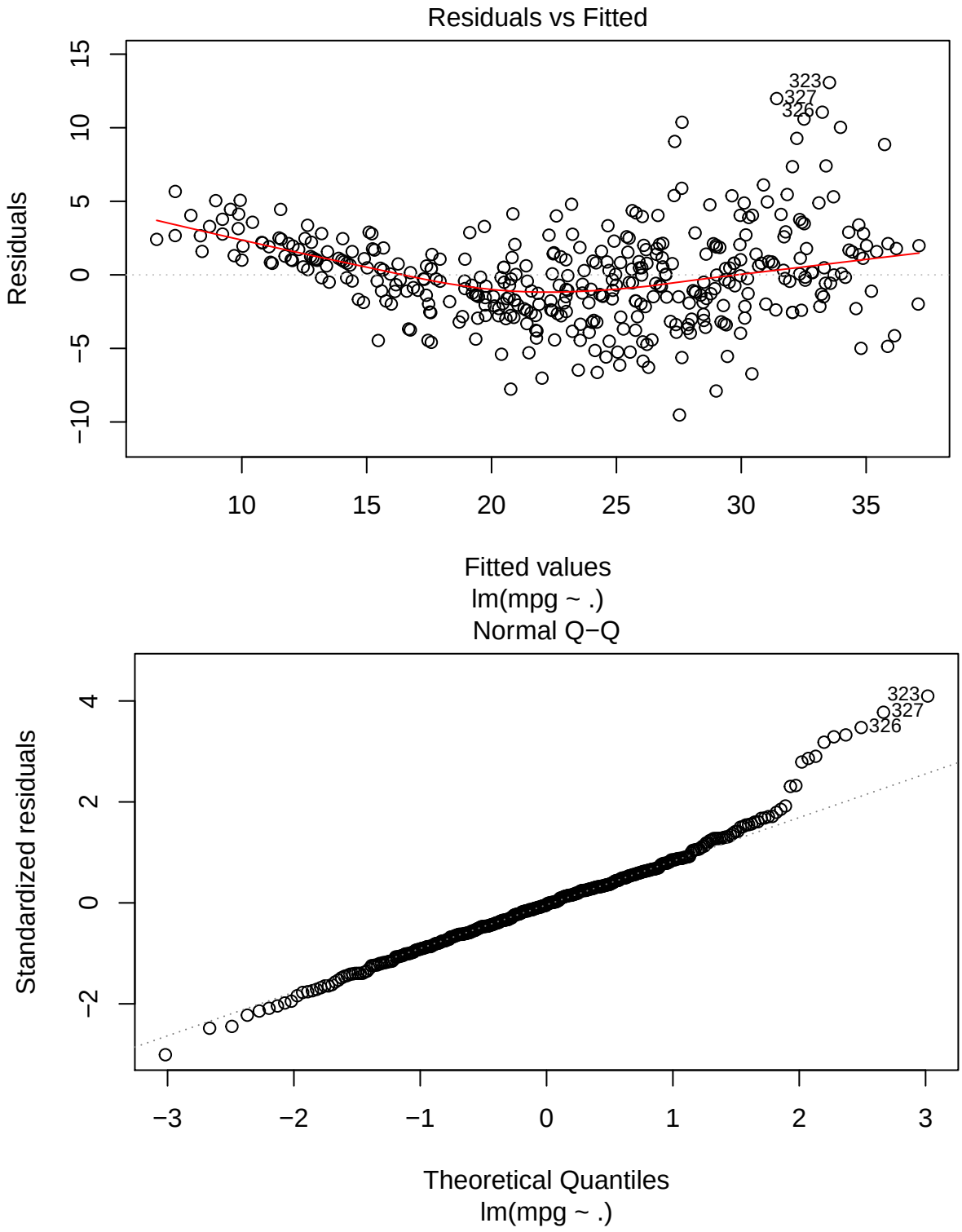
```
#Diagnostic plot for log transformation on x only
plot(lms_log)
```

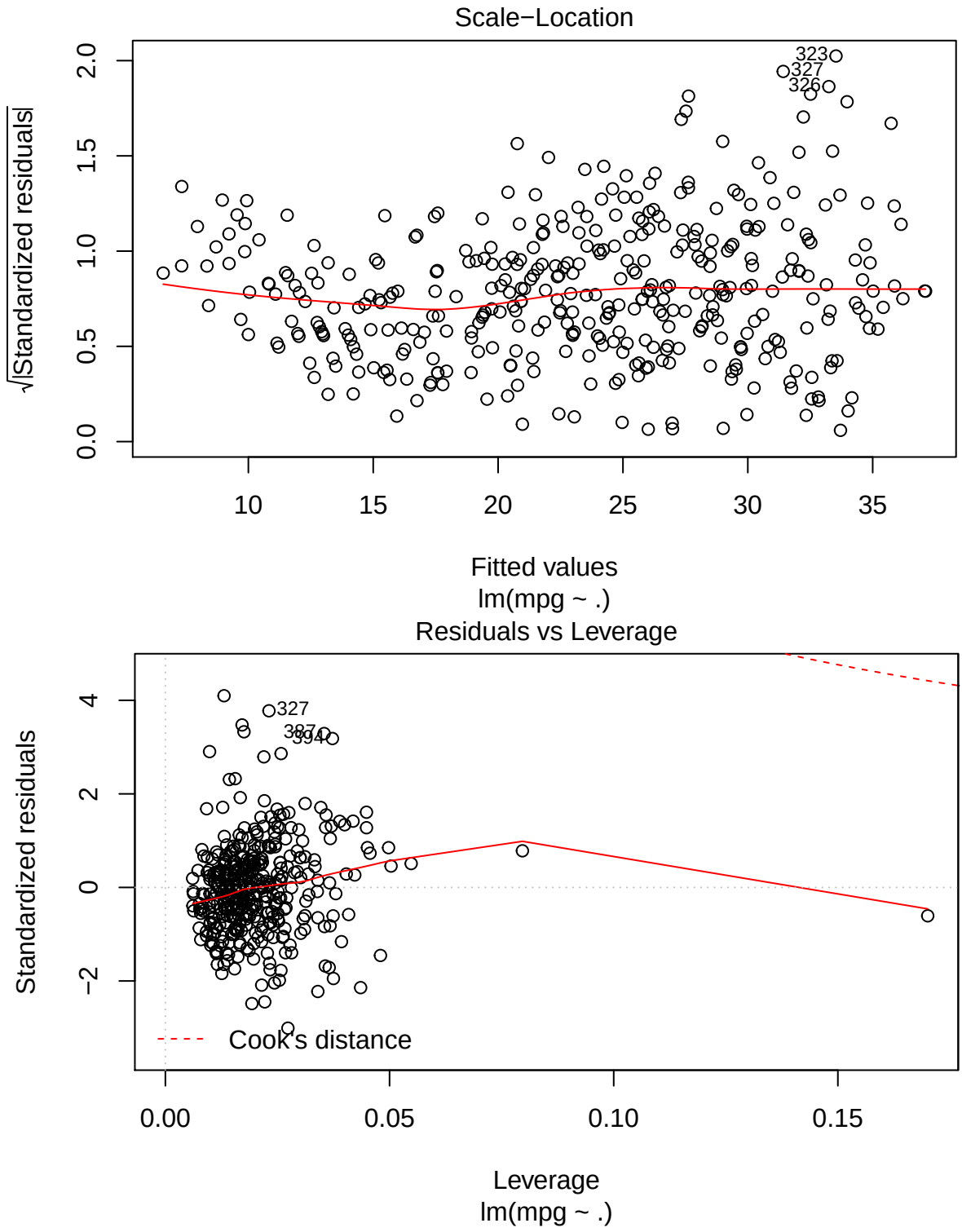

Residuals vs Fitted
Residuals
Fitted values
lm(mpg ~ .)
323
387
327
Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(mpg ~ .)



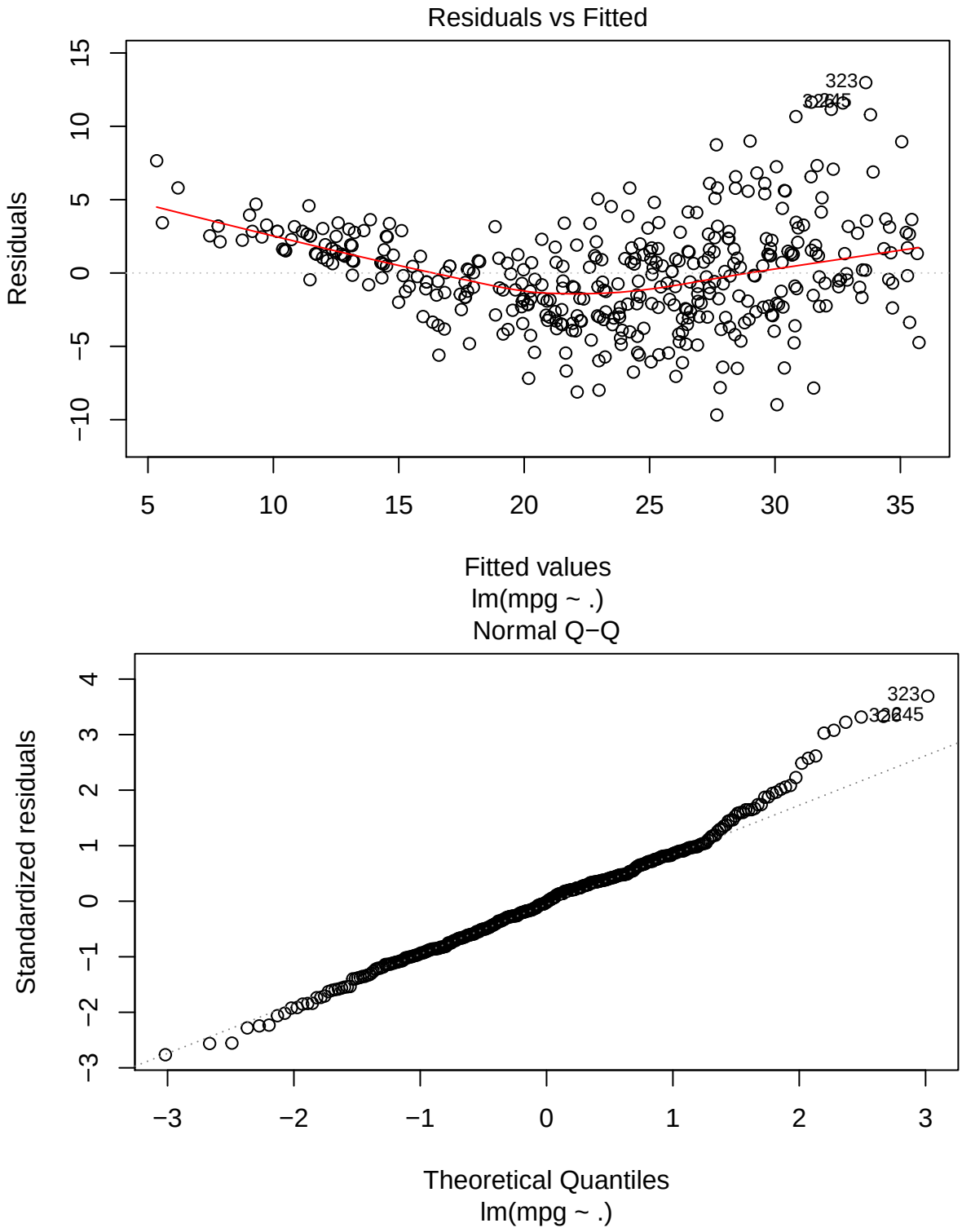


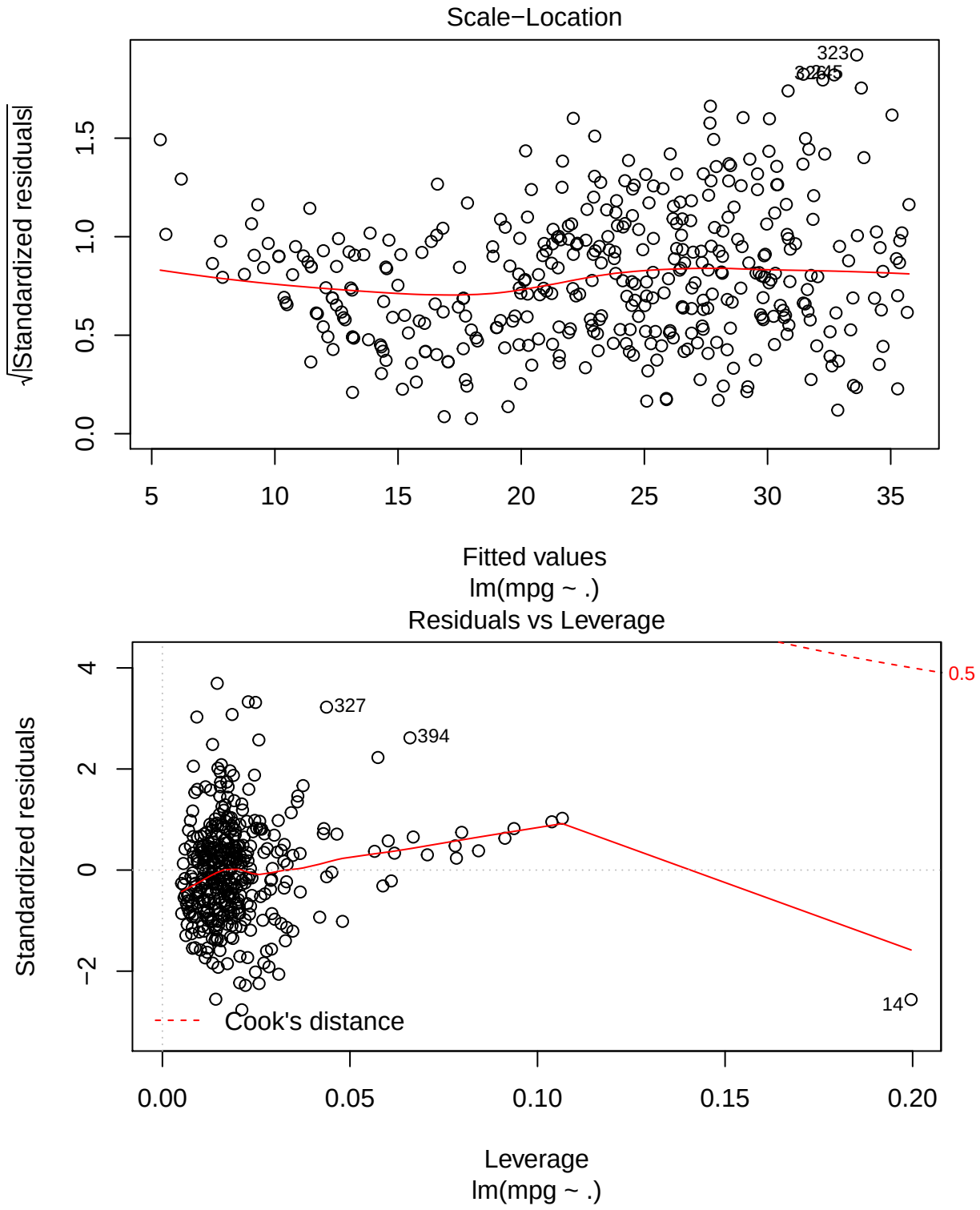

```
#Diagnostic plot for square root transformation on x only
plot(lms_root)
```


Residuals vs Fitted
Residuals
Fitted values
lm(mpg ~ .)
323
327
326
Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(mpg ~ .)

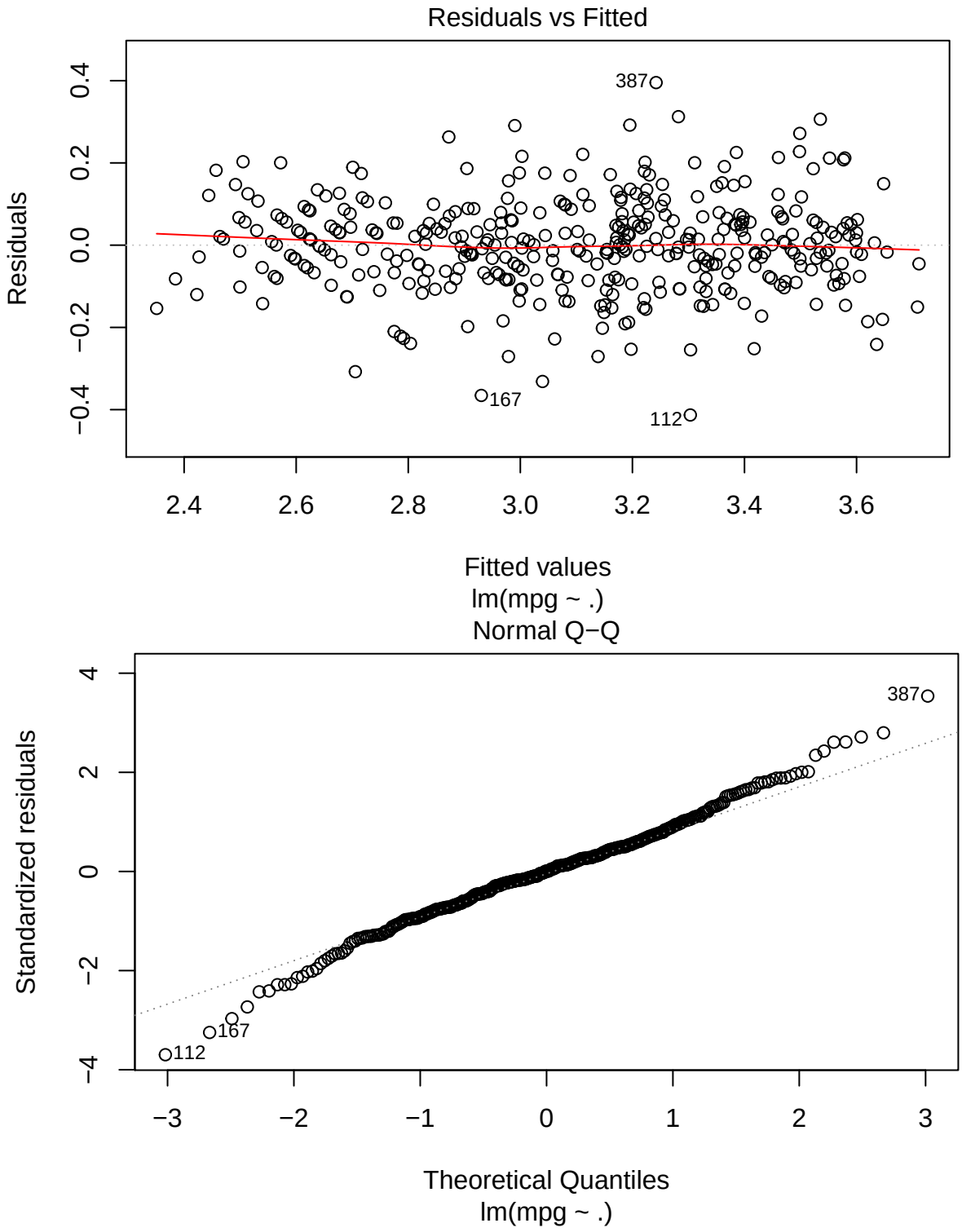


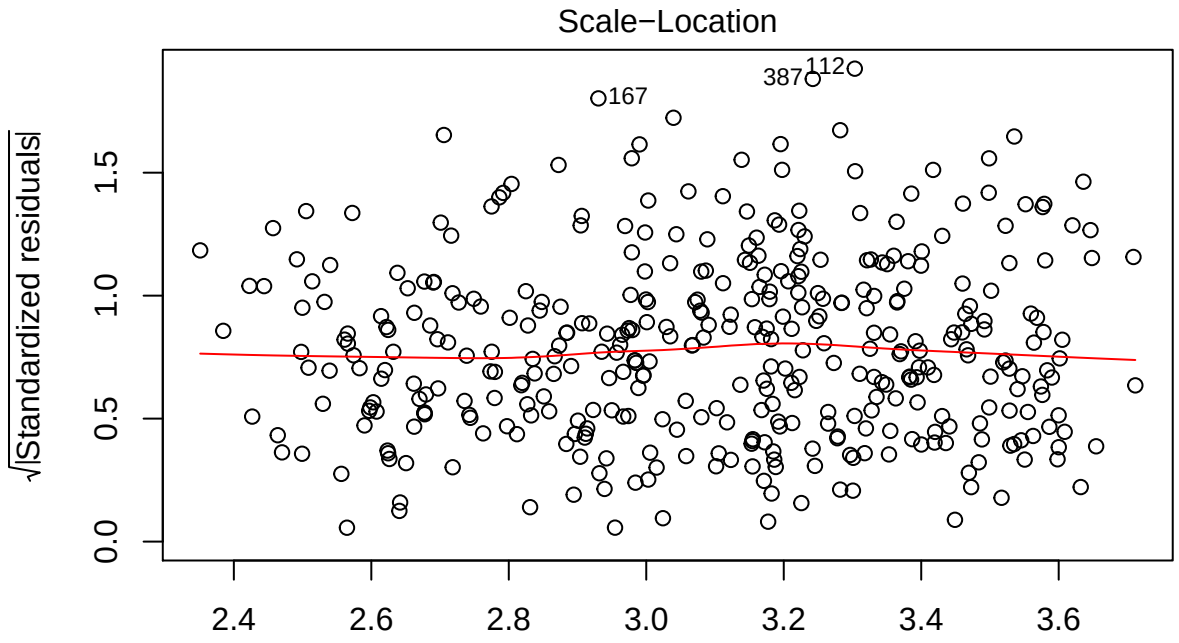

```
#Diagnostic plot for square transformation on x only
plot(lms_square)
```

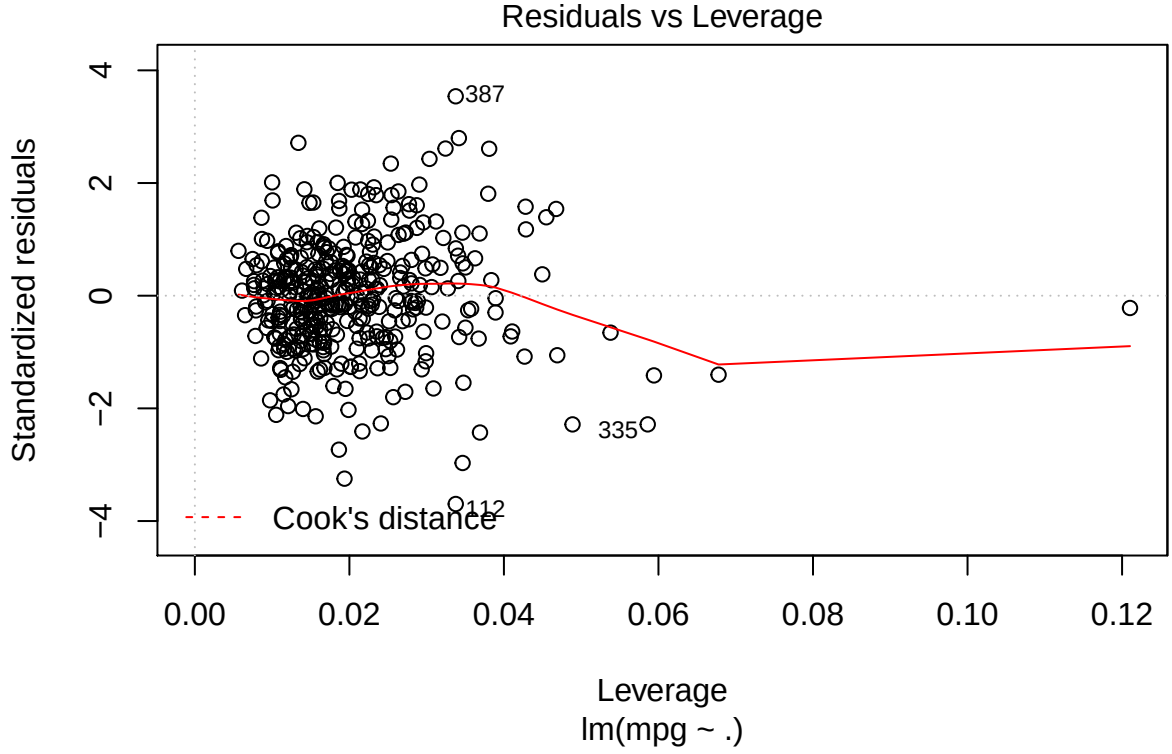

Residuals vs Fitted
Residuals
Fitted values
lm(mpg ~ .)
Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(mpg ~ .)



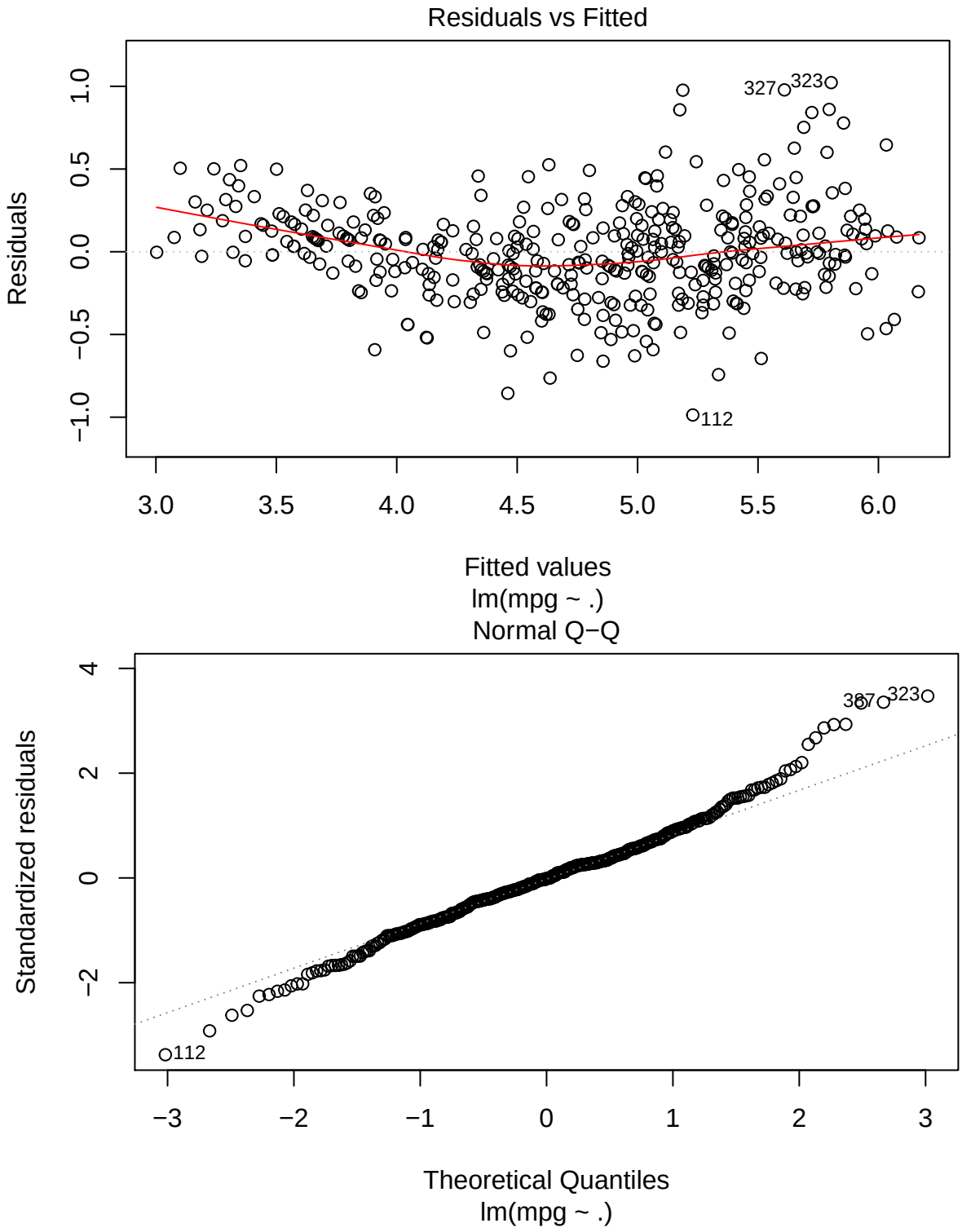

```
#Diagonostic plot for log transformation on both x and y
plot(lms_log2)
```

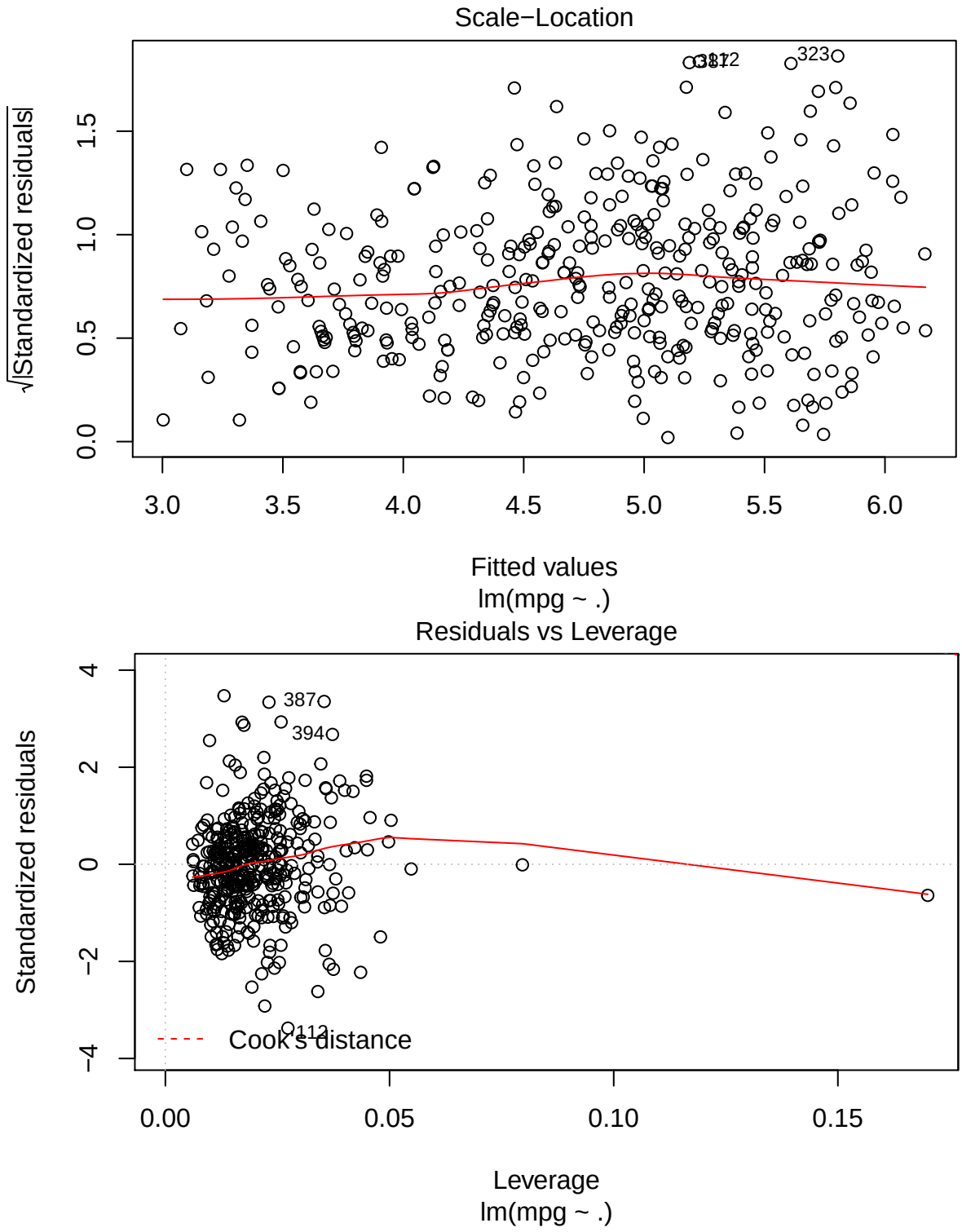

Residuals vs Fitted
Residuals
Fitted values
lm(mpg ~ .)
387
167
112
2.4
2.6
2.8
3.0
3.2
3.4
3.6
−0.4
−0.2
0.0
0.2
0.4
Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(mpg ~ .)
−3
−2
−1
0
1
2
3
−4
−2
0
2
4



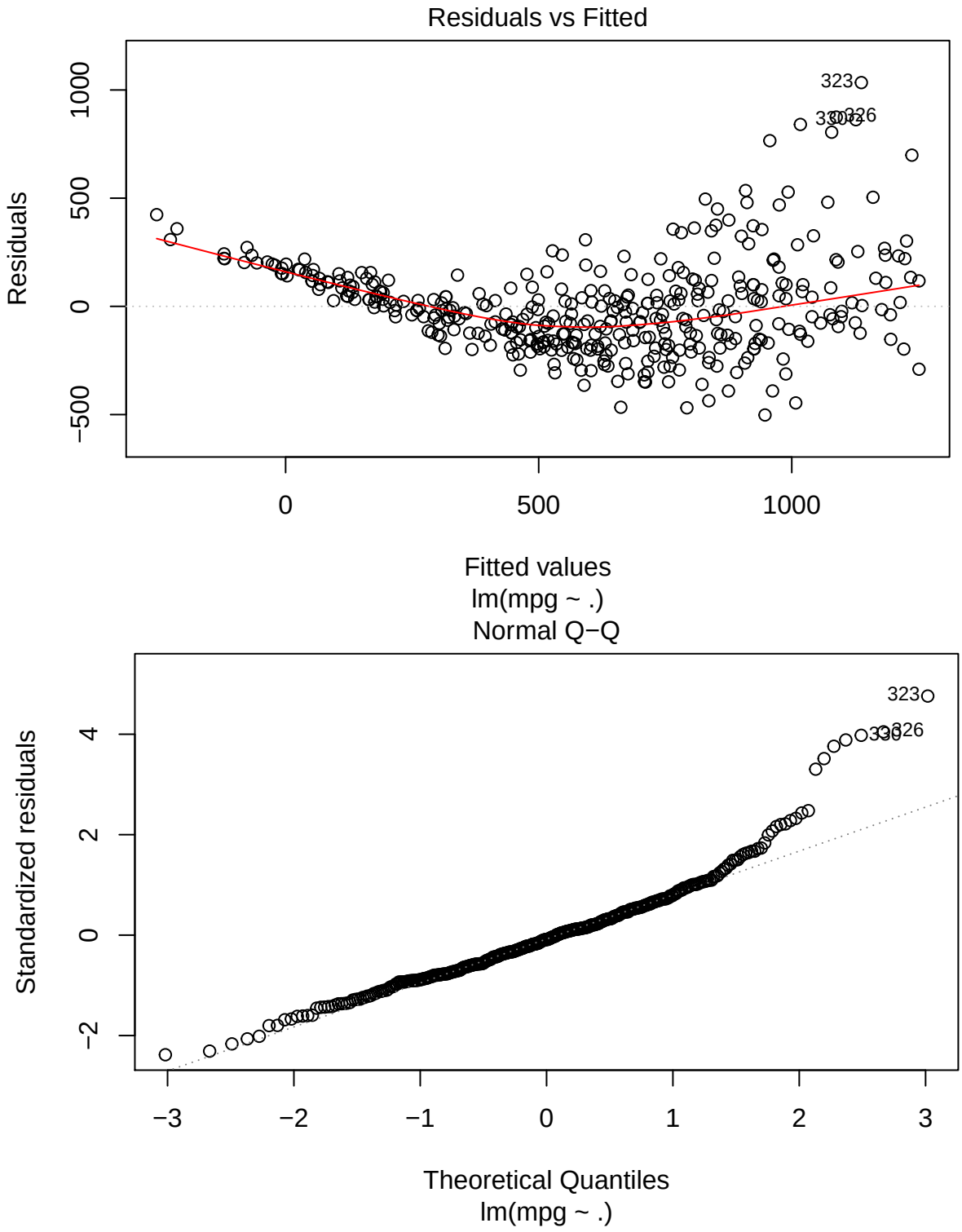


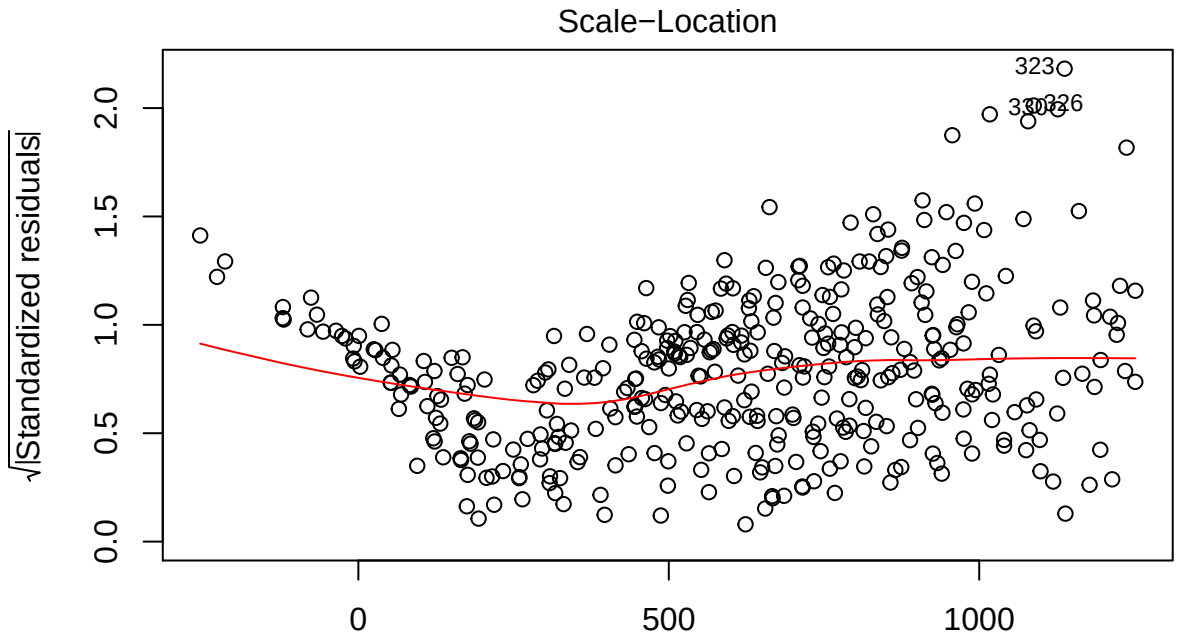

```
#Diagnostic plot for square root transformation on both x and y
plot(lms_root2)
```

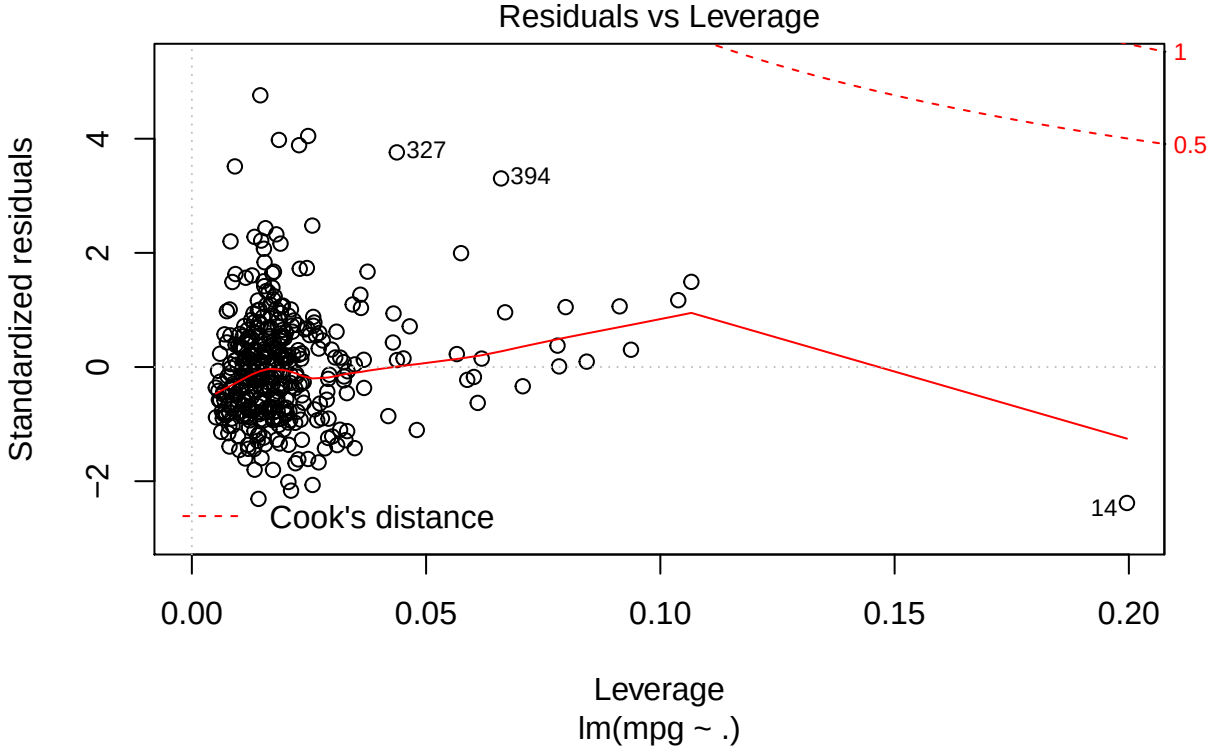

Residuals vs Fitted
Residuals
Fitted values
lm(mpg ~ .)
327
323
112
Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(mpg ~ .)
327
323
112



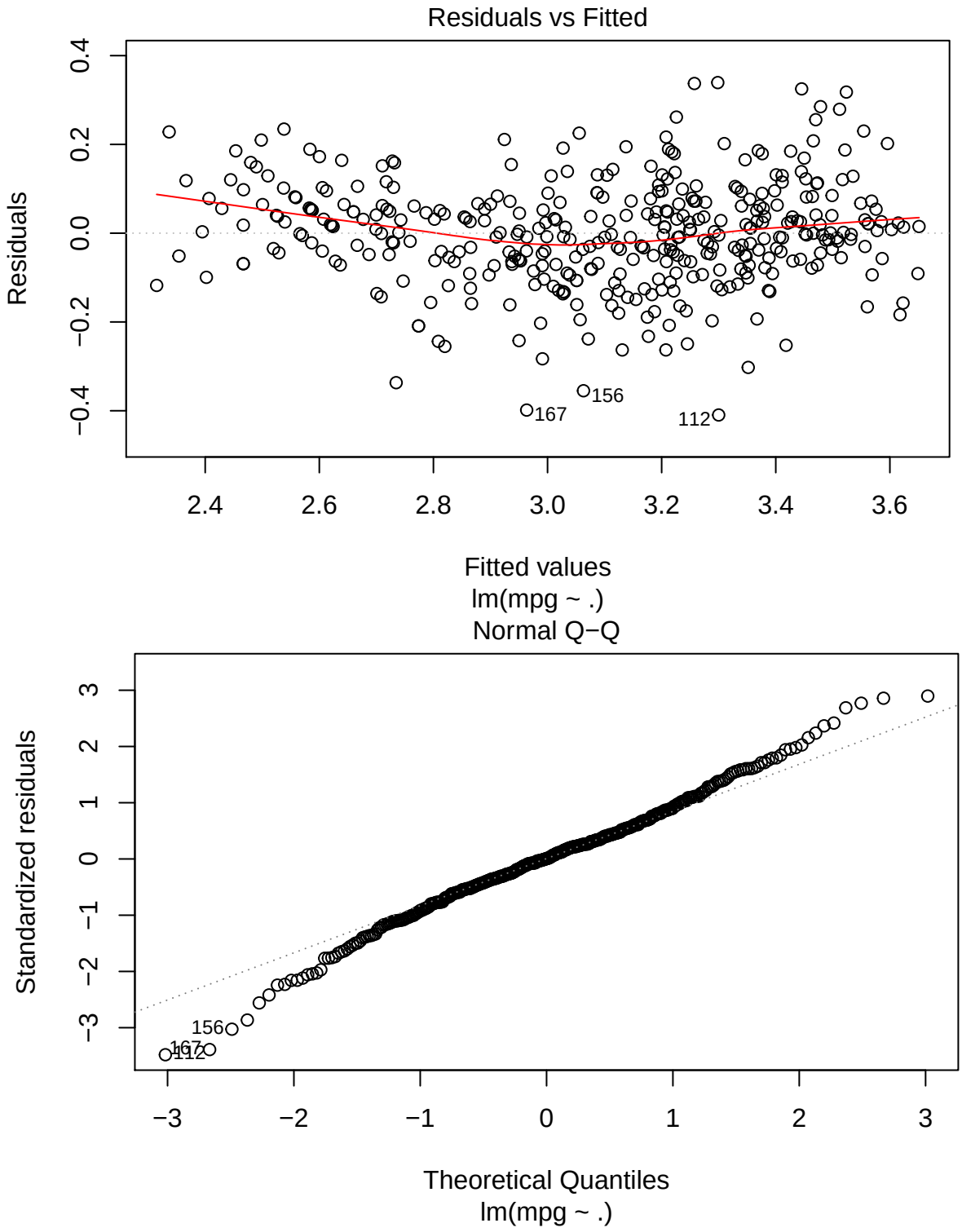

```
#Diagnostic plot for square transformation on both x and y
plot(lms_square2)
```

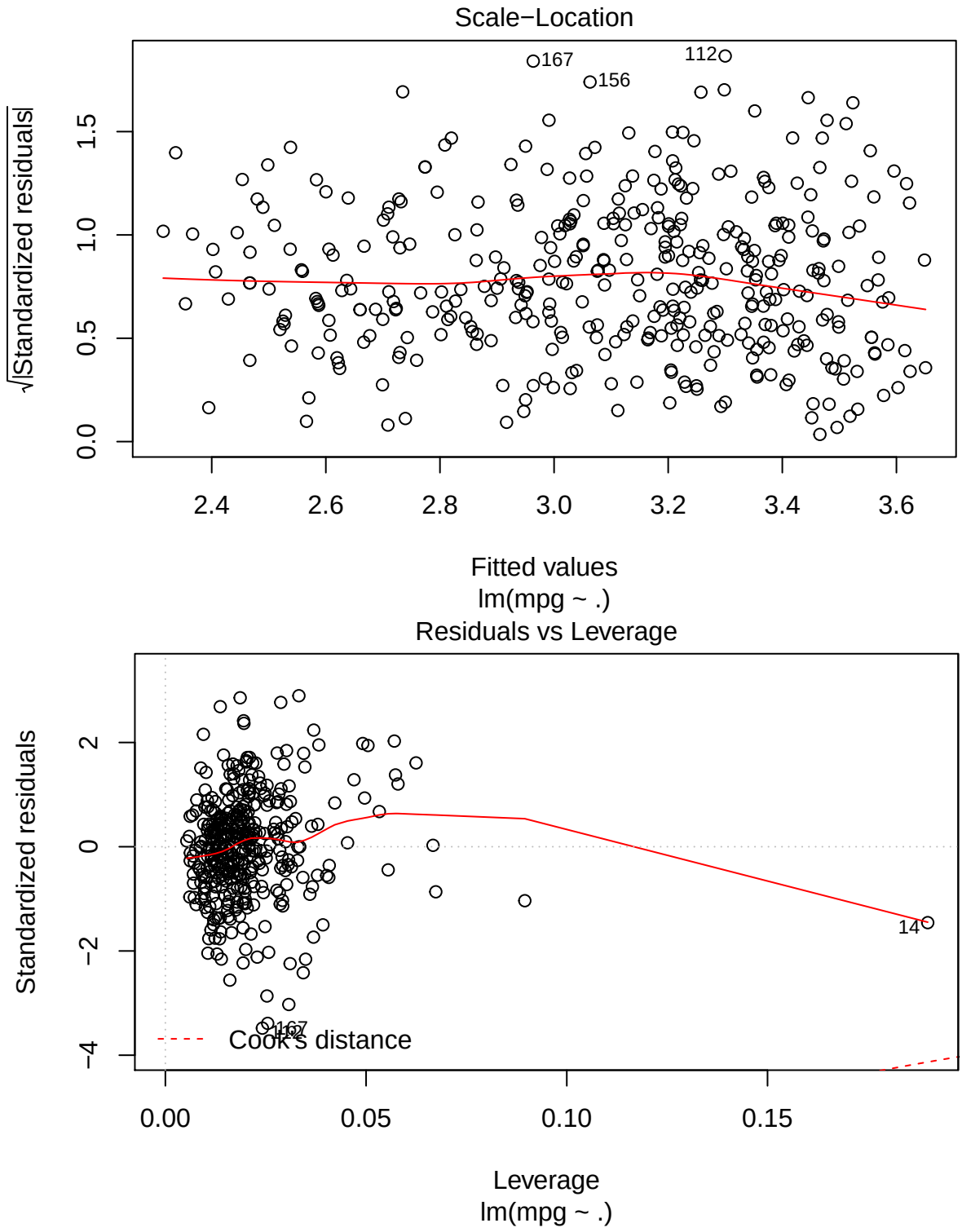

Residuals vs Fitted
Residuals
Fitted values
lm(mpg ~ .)
Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(mpg ~ .)



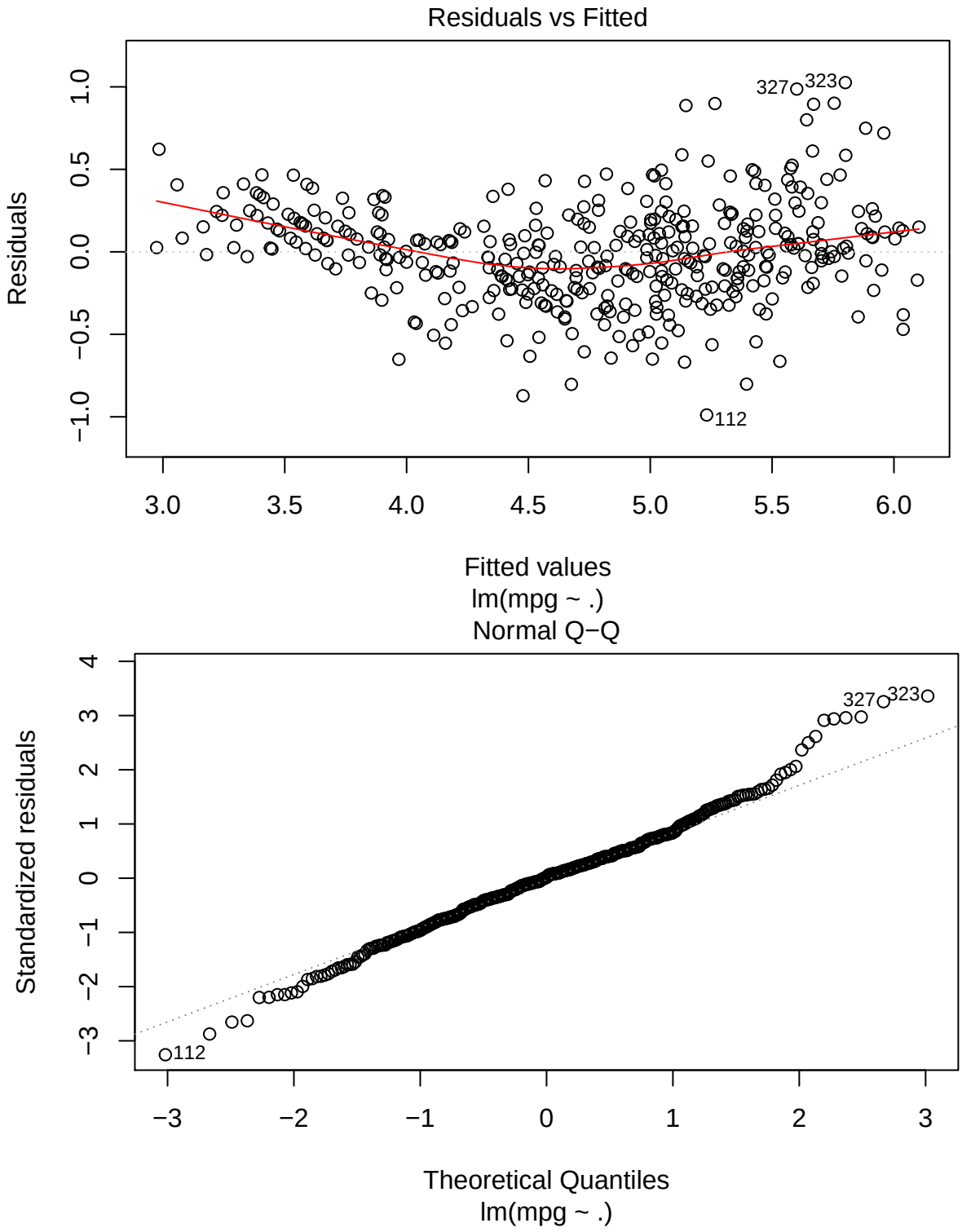


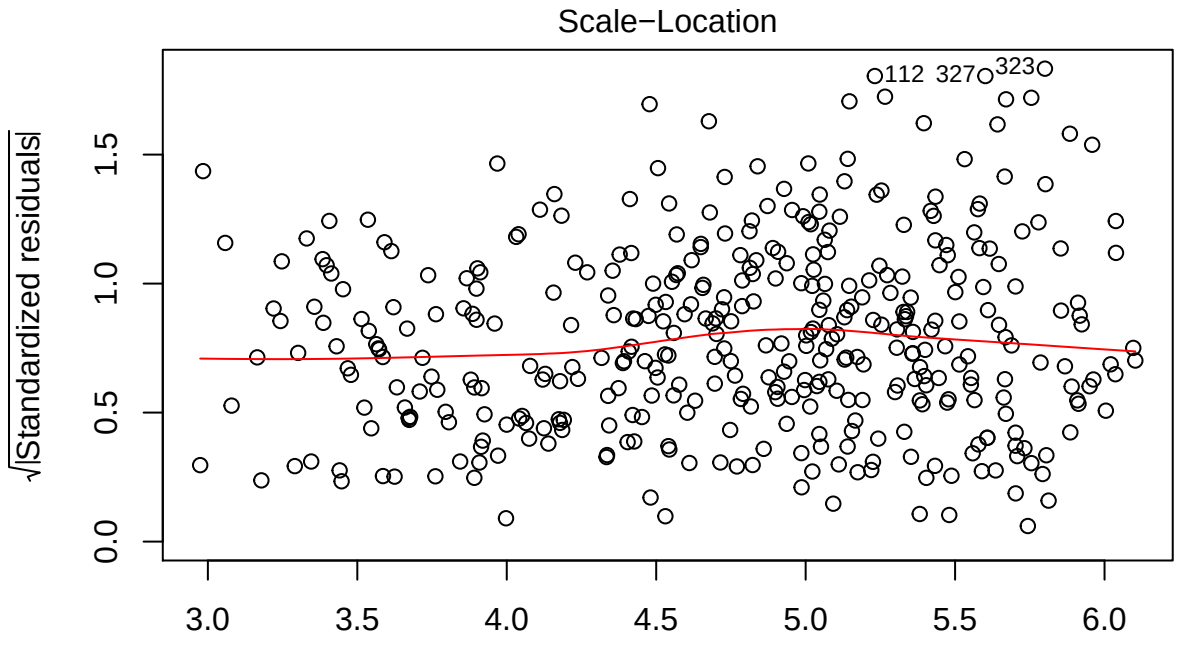

```
#Diagonostic plot for log transformation on y only
plot(lms_log3)
```

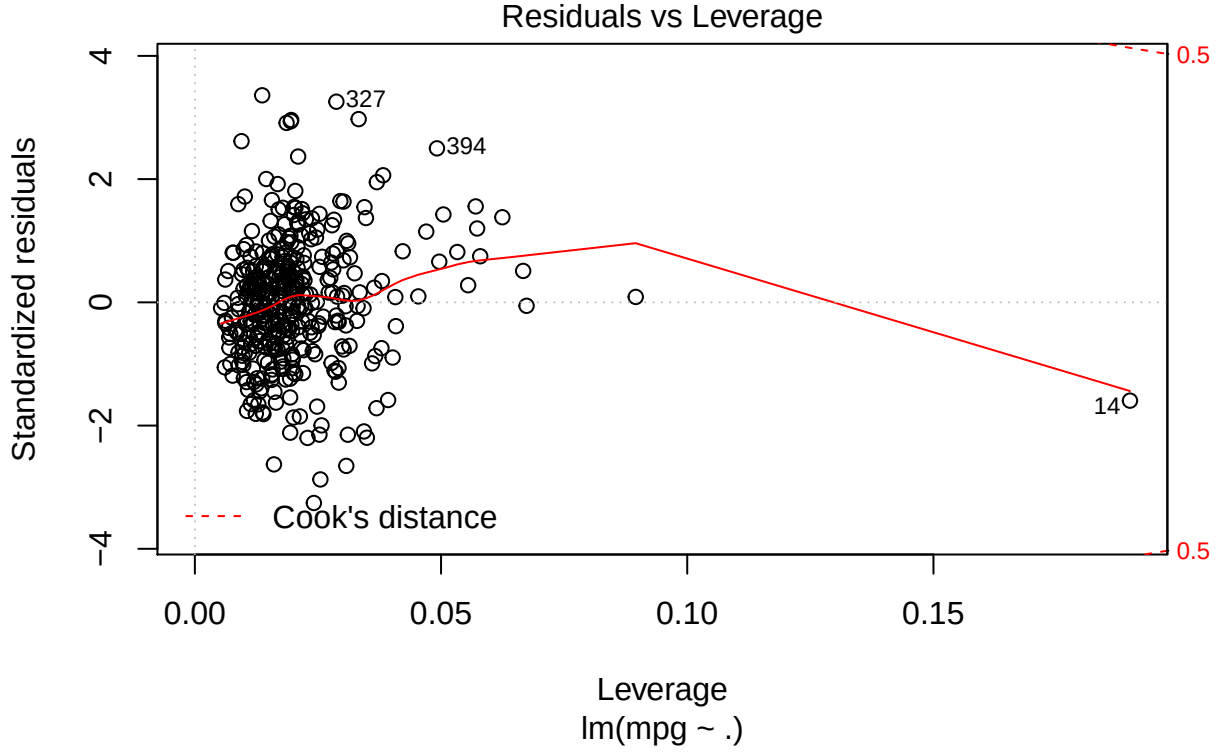

Residuals vs Fitted
Residuals
0.4
0.2
0.0
−0.2
−0.4
2.4
2.6
2.8
3.0
3.2
3.4
3.6
156
167
112
Fitted values
lm(mpg ~ .)
Normal Q−Q
Standardized residuals
3
2
1
0
−1
−2
−3
156
167
112
Theoretical Quantiles
lm(mpg ~ .)



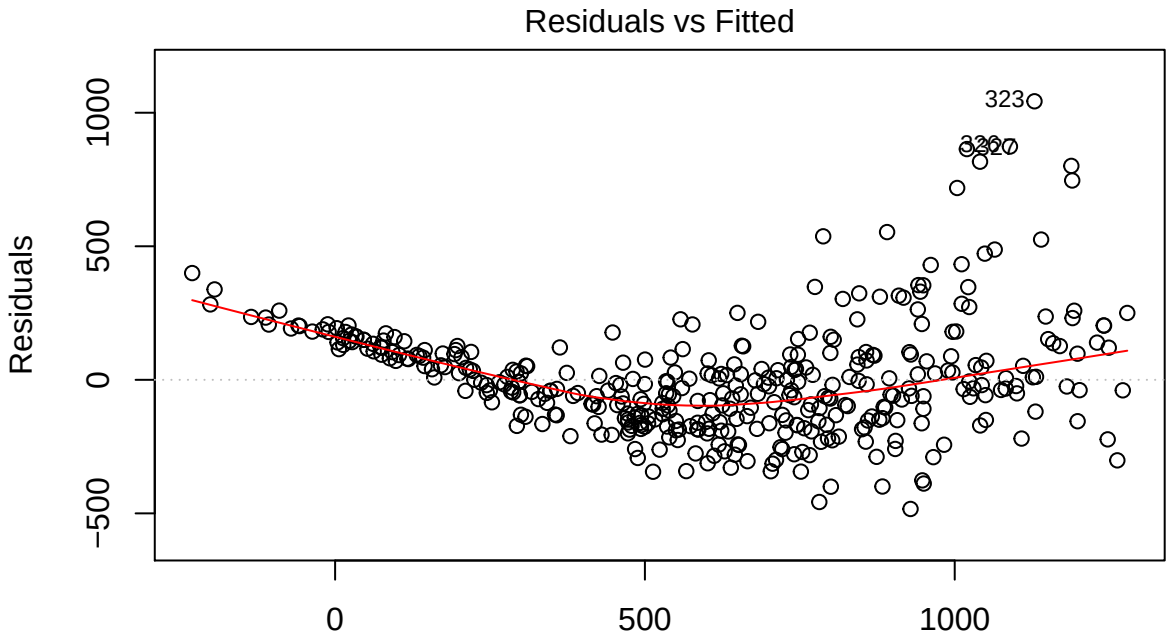

```
#Diagnostic plot for square root transformation on y only
plot(lms_root3)
```

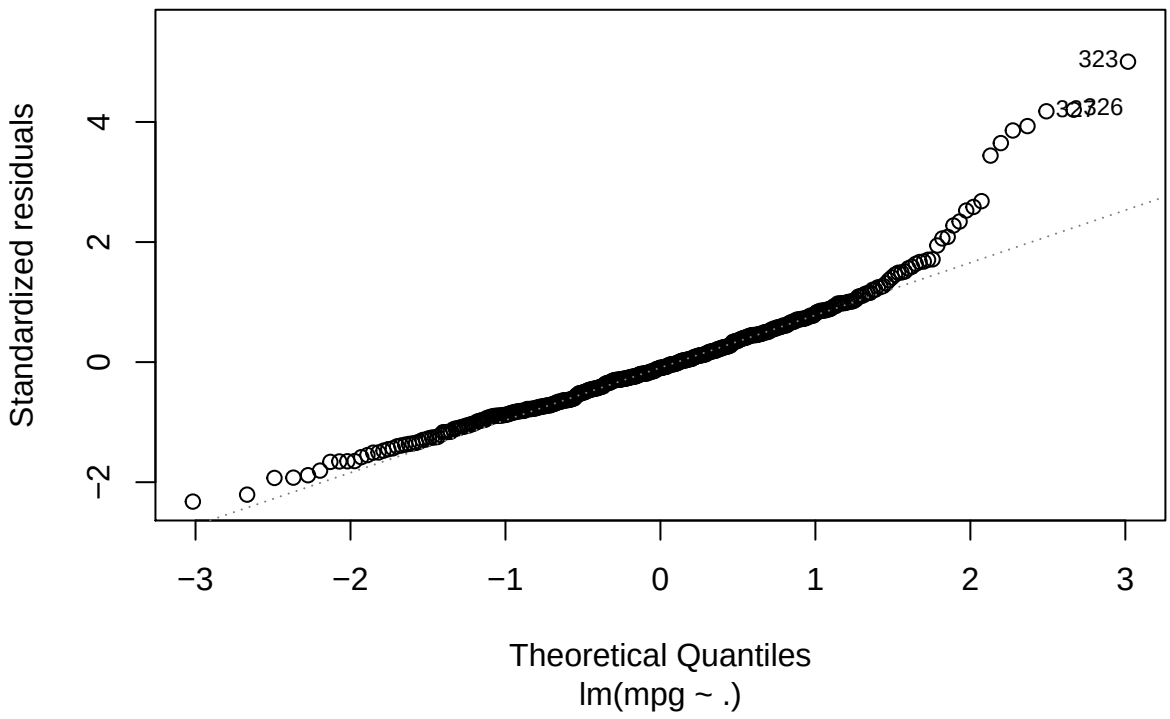

Residuals vs Fitted
Residuals
Fitted values
lm(mpg ~ .)
327
323
112
Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(mpg ~ .)

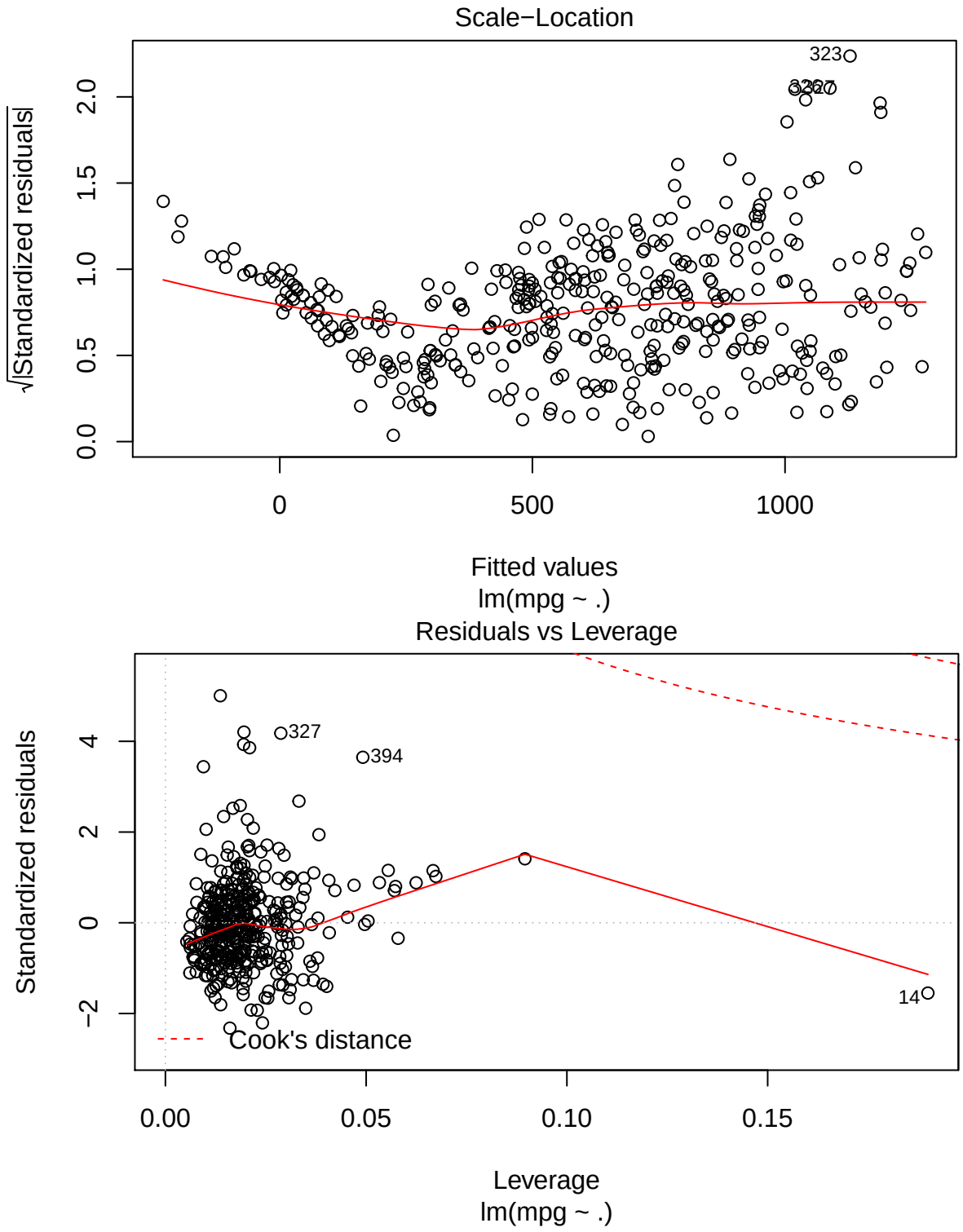





```
#Diagnostic plot for square transformation on y only
plot(lms_square3)
```


Residuals vs Fitted
Residuals
1000
500
0
−500
323
0
500
1000
Fitted values
lm(mpg ~ .)


Normal Q−Q
Standardized residuals
4
2
0
−2
323
326
−3
−2
−1
0
1
2
3
Theoretical Quantiles
lm(mpg ~ .)


Scale−Location
√|Standardized residuals|
323
0.0
0.5
1.0
1.5
2.0
0
500
1000
Fitted values
lm(mpg ~ .)
Residuals vs Leverage
Standardized residuals
327
394
14
1
0.5
Cook's distance
−2
0
2
4
0.00
0.05
0.10
0.15
Leverage
lm(mpg ~ .)