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,

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

and this is equal to

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

.

So, I can conlcude that

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

and each element of vector is equal, so,

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

is true and it implies

$$\begin{pmatrix} 1 & 1 & \dots & 1 \end{pmatrix} \begin{pmatrix} y_1 \\ . \\ . \\ . \\ y_n \end{pmatrix} = \begin{pmatrix} 1 & 1 & \dots & 1 \end{pmatrix} \begin{pmatrix} \hat{y}_1 \\ . \\ . \\ . \\ \hat{y}_n \end{pmatrix}$$

Here is one the examples:

```r
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

$X^T X$ 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

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

.

Part b

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

And, we have a matrix

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

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

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 $X^T X$.

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

3

Problem 3

Part a

```r
set.seed(1)

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

```
##   [1] -0.626453811  0.183643324 -0.835628612  1.595280802  0.329507772
##   [6] -0.820468384  0.487429052  0.738324705  0.575781352 -0.305388387
##  [11]  1.511781168  0.389843236 -0.621240581 -2.214699887  1.124930918
##  [16] -0.044933609 -0.016190263  0.943836211  0.821221195  0.593901321
##  [21]  0.918977372  0.782136301  0.074564983 -1.989351696  0.619825748
##  [26] -0.056128740 -0.155795507 -1.470752384 -0.478150055  0.417941560
##  [31]  1.358679552 -0.102787727  0.387671612 -0.053805041 -1.377059557
##  [36] -0.414994563 -0.394289954 -0.059313397  1.100025372  0.763175748
##  [41] -0.164523596 -0.253361680  0.696963375  0.556663199 -0.688755695
##  [46] -0.707495157  0.364581962  0.768532925 -0.112346212  0.881107726
##  [51]  0.398105880 -0.612026393  0.341119691 -1.129363096  1.433023702
##  [56]  1.980399899 -0.367221476 -1.044134626  0.569719627 -0.135054604
##  [61]  2.401617761 -0.039240003  0.689739362  0.028002159 -0.743273209
##  [66]  0.188792300 -1.804958629  1.465554862  0.153253338  2.172611670
##  [71]  0.475509529 -0.709946431  0.610726353 -0.934097632 -1.253633400
##  [76]  0.291446236 -0.443291873  0.001105352  0.074341324 -0.589520946
##  [81] -0.568668733 -0.135178615  1.178086997 -1.523566800  0.593946188
##  [86]  0.332950371  1.063099837 -0.304183924  0.370018810  0.267098791
##  [91] -0.542520031  1.207867806  1.160402616  0.700213650  1.586833455
##  [96]  0.558486426 -1.276592208 -0.573265414 -1.224612615 -0.473400636
```

Part b

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

```
##   [1] -0.310183339  0.021057937 -0.455460824  0.079014386 -0.327292322
##   [6]  0.883643635  0.358353738  0.455087115  0.192092679  0.841088040
##  [11] -0.317868227 -0.230822365  0.716141119 -0.325348177 -0.103690372
##  [16] -0.196403965 -0.159996434 -0.139556651  0.247094166 -0.088665241
```

```
## [21] -0.252978731  0.671519413 -0.107289704 -0.089778265 -0.050095371
## [26]  0.356333154 -0.036782202 -0.018817086 -0.340830239 -0.162135136
## [31]  0.030080220 -0.294447243  0.265748096 -0.759197041  0.153278930
## [36] -0.768224912 -0.150488063 -0.264139952 -0.326047390 -0.028448389
## [41] -0.957179713  0.588291656 -0.832486218 -0.231765201 -0.557960053
## [46] -0.375409501  1.043583273  0.008697810 -0.643150265 -0.820302767
## [51]  0.225093551 -0.009279916 -0.159034187 -0.464681074 -0.743730155
## [56] -0.537596148  0.500014402 -0.310633347 -0.692213424  0.934645311
## [61]  0.212550189 -0.119323550  0.529241524  0.443211326 -0.309621524
## [66]  1.103051232 -0.127513515 -0.712247325 -0.072199801  0.103769170
## [71]  1.153989200  0.052901184  0.228499403 -0.038576468 -0.167000421
## [76] -0.017363014  0.393819803  1.037622504  0.513696219  0.603954199
## [81] -0.615661711  0.491947785  0.109962402 -0.733625015  0.260511371
## [86] -0.079377302  0.732293656 -0.383041000 -0.215105877 -0.463054749
## [91] -0.088551981  0.201005890 -0.365874087  0.415186584 -0.604041393
## [96] -0.523992206  0.720578853 -0.507923733  0.205987356 -0.190538026
```

Part c

So, I can say that

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

```r
y <- -1 + 0.5*x + eps
y
```

```
##   [1] -1.62341024 -0.88712040 -1.87327513 -0.12334521 -1.16253844
##   [6] -0.52659056 -0.39793174 -0.17575053 -0.52001665 -0.31160615
##  [11] -0.56197764 -1.03590075 -0.59447917 -2.43269812 -0.54122491
##  [16] -1.21887077 -1.16809157 -0.66763855 -0.34229524 -0.79171458
##  [21] -0.79349005  0.06258756 -1.07000721 -2.08445411 -0.74018250
##  [26] -0.67173122 -1.11467996 -1.75419328 -1.57990527 -0.95316436
##  [31] -0.29058000 -1.34584111 -0.54041610 -1.78609956 -1.53525085
##  [36] -1.97572219 -1.34763304 -1.29379665 -0.77603470 -0.64686051
##  [41] -2.03944151 -0.53838918 -1.48400453 -0.95343360 -1.90233790
##  [46] -1.72915708  0.22587425 -0.60703573 -1.69932337 -1.37974890
##  [51] -0.57585351 -1.31529311 -0.98847434 -2.02936262 -1.02721830
##  [56] -0.54739620 -0.68359634 -1.83270066 -1.40735361 -0.13288199
##  [61]  0.41335907 -1.13894355 -0.12588879 -0.54278759 -1.68125813
##  [66]  0.19744738 -2.02999283 -0.97946989 -0.99557313  0.19007500
##  [71]  0.39174396 -1.30207203 -0.46613742 -1.50562528 -1.79381712
##  [76] -0.87163990 -0.82782613  0.03817518 -0.44913312 -0.69080627
##  [81] -1.89999608 -0.57564152 -0.30099410 -2.49540841 -0.44251553
##  [86] -0.91290212  0.26384357 -1.53513296 -1.03009647 -1.32950535
##  [91] -1.35981200 -0.19506021 -0.78567278 -0.23470659 -0.81062467
##  [96] -1.24474899 -0.91771725 -1.79455644 -1.40631895 -1.42723834
```

Part d

```r
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")
```


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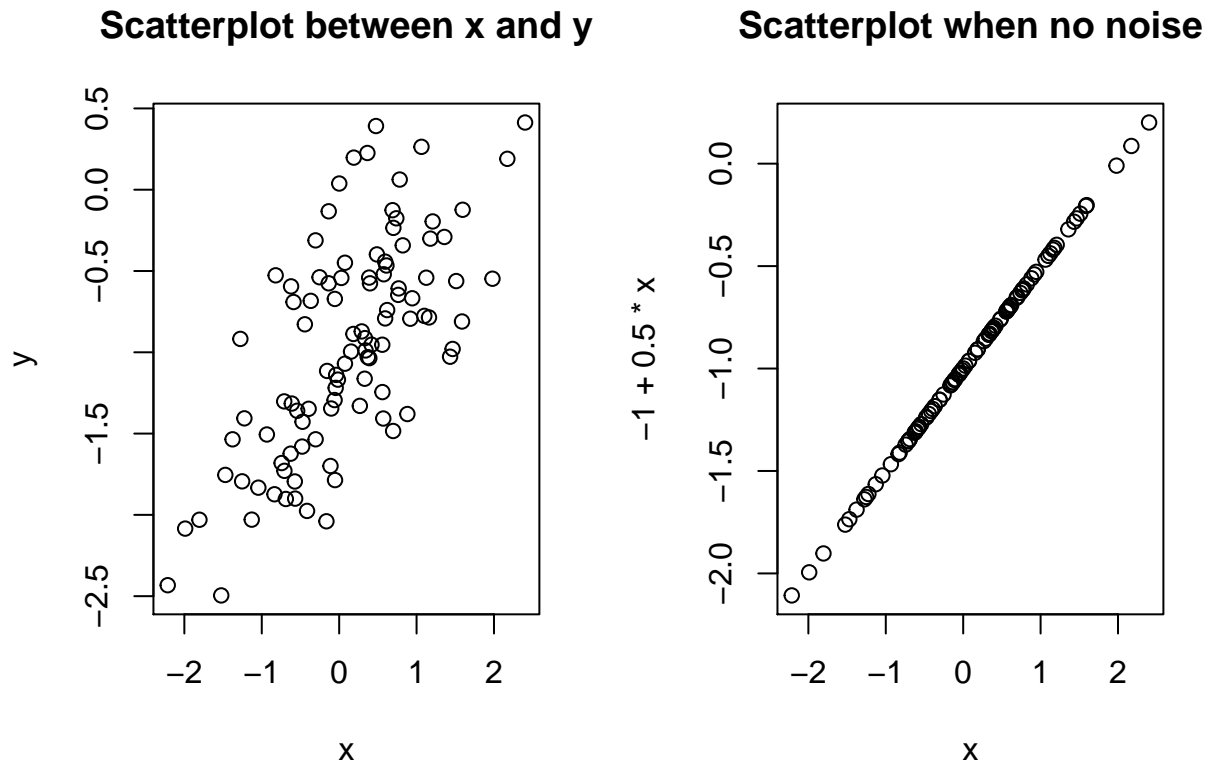
Comment: They show the decent linear relationship between x and y, but definitely they do not form a perfect line. From the equation I created above, I can see that 0.5 slope (as x goes up by 1 unit, y goes up by around 0,5 unit) and the intercept will be around -1. Additionally, I plotted out the scatter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

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

```
## [1] "Least square intercept will be -1.018846 and least square slope will be 0.49947"
```
```
print("Original intercept will be -1 and slope will be 0.5")
```
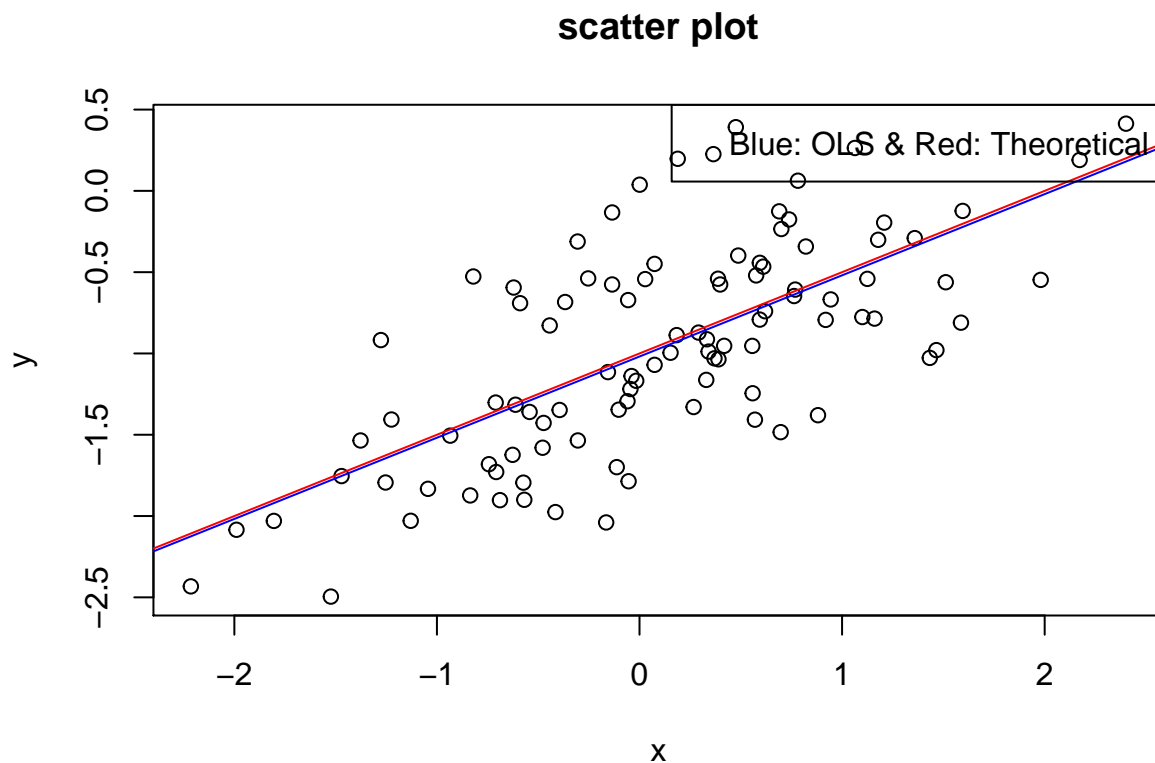```
[1] "Original intercept will be -1 and slope will be 0.5"
```
Comment: So, $\beta_0 = -1$ and $\beta_1 = 0.5$ and $\hat{\beta}_0 \approx -1.018846$ and $\hat{\beta}_1 \approx 0.49947$. The line is more gentle with least square estimators.

**Part f**

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

```
plot(x, y, main = "scatter plot")
abline(a = -1, b = 0.5, col = "red")
abline(lm, col = "blue")
legend("topright", legend = "Blue: OLS & Red: Theoretical")
```
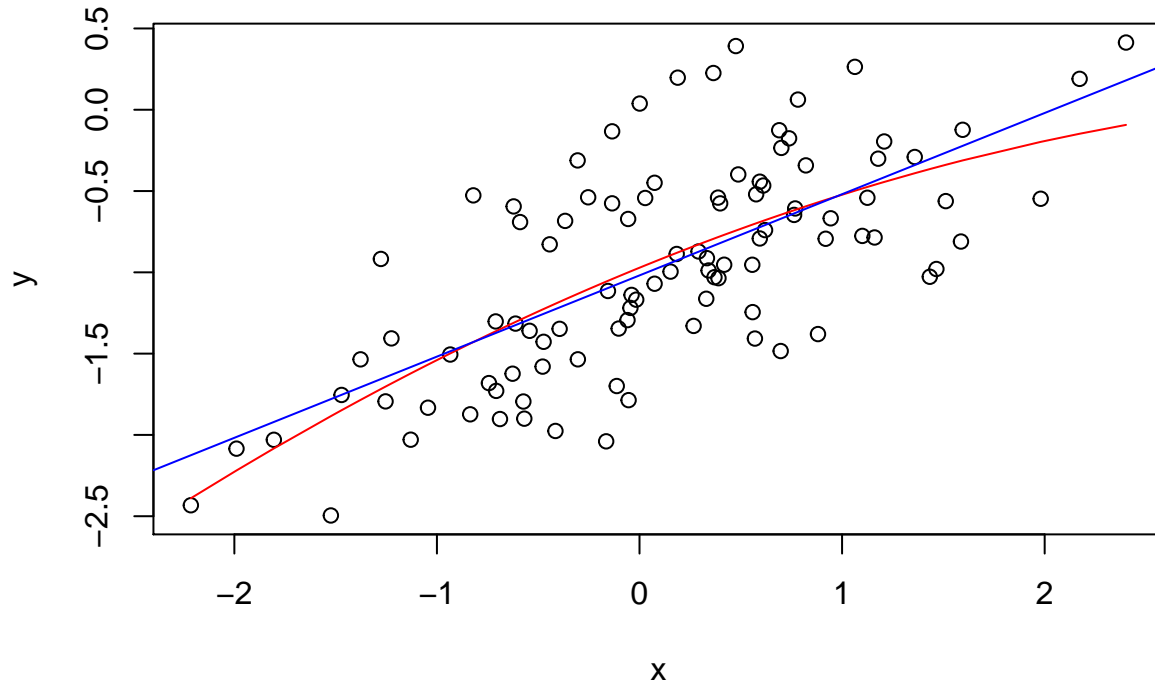


**Part g**

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

smoothingSpline = smooth.spline(x, y2, spar = 0.5)
```

```
lines(smoothingSpline, col = "red")
abline(lm, col = "blue")
```

Polynomial regression


```
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
```

```
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")
```

polynomial regression model 2
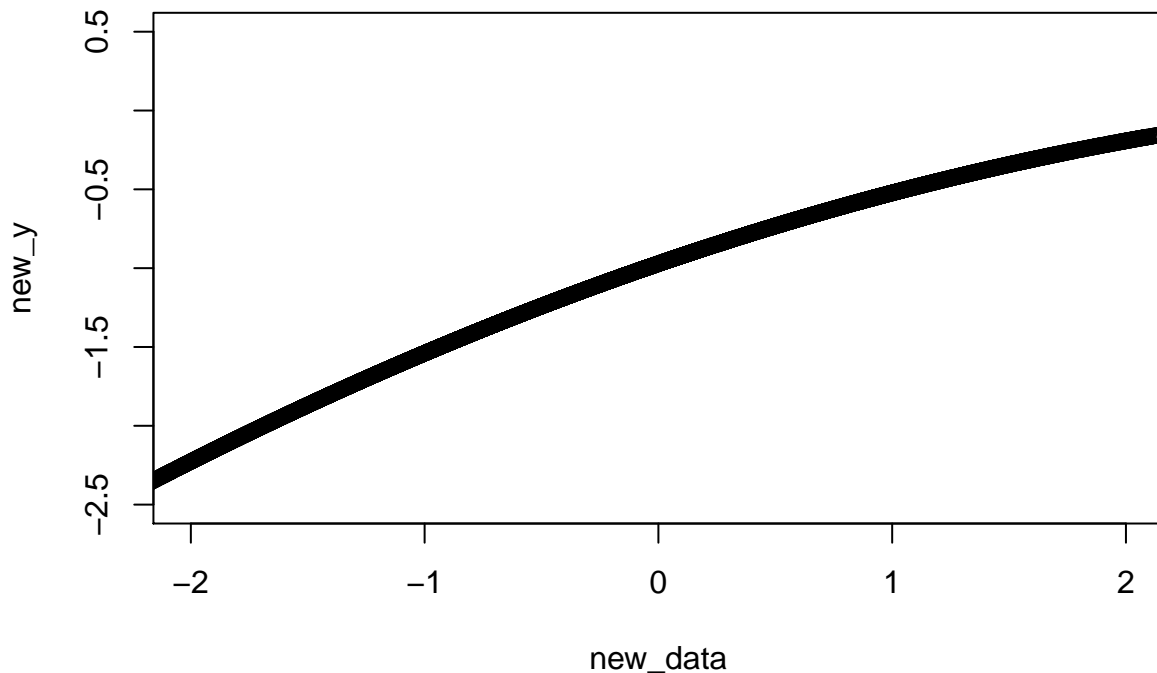
Comment: There is really no evidence that the quadratic term improves the model fit on both p-value analysis and graphs. P-value tells us to drop the quadratic term. Although adjusted R^2 goes up a little bit, I would still say quadratic term does not improve the model fit that much. (but this might 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 ~ N(0, 0.04).

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

```
##    [1] -0.626453811  0.183643324 -0.835628612  1.595280802  0.329507772
##    [6] -0.820468384  0.487429052  0.738324705  0.575781352 -0.305388387
##   [11]  1.511781168  0.389843236 -0.621240581 -2.214699887  1.124930918
##   [16] -0.044933609 -0.016190263  0.943836211  0.821221195  0.593901321
##   [21]  0.918977372  0.782136301  0.074564983 -1.989351696  0.619825748
##   [26] -0.056128740 -0.155795507 -1.470752384 -0.478150055  0.417941560
##   [31]  1.358679552 -0.102787727  0.387671612 -0.053805041 -1.377059557
##   [36] -0.414994563 -0.394289954 -0.059313397  1.100025372  0.763175748
##   [41] -0.164523596 -0.253361680  0.696963375  0.556663199 -0.688755695
##   [46] -0.707495157  0.364581962  0.768532925 -0.112346212  0.881107726
##   [51]  0.398105880 -0.612026393  0.341119691 -1.129363096  1.433023702
##   [56]  1.980399899 -0.367221476 -1.044134626  0.569719627 -0.135054604
##   [61]  2.401617761 -0.039240003  0.689739362  0.028002159 -0.743273209
##   [66]  0.188792300 -1.804958629  1.465554862  0.153253338  2.172611670
##   [71]  0.475509529 -0.709946431  0.610726353 -0.934097632 -1.253633400
##   [76]  0.291446236 -0.443291873  0.001105352  0.074341324 -0.589520946
##   [81] -0.568668733 -0.135178615  1.178086997 -1.523566800  0.593946188
##   [86]  0.332950371  1.063099837 -0.304183924  0.370018810  0.267098791
##   [91] -0.542520031  1.207867806  1.160402616  0.700213650  1.586833455
##   [96]  0.558486426 -1.276592208 -0.573265414 -1.224612615 -0.473400636
```

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

```
##    [1] -0.0248146671  0.0016846349 -0.0364368659  0.0063211509 -0.0261833858
##    [6]  0.0706914908  0.0286682990  0.0364069692  0.0153674143  0.0672870432
##   [11] -0.0254294582 -0.0184657892  0.0572912895 -0.0260278541 -0.0082952297
##   [16] -0.0157123172 -0.0127997147 -0.0111645321  0.0197675333 -0.0070932193
##   [21] -0.0202382985  0.0537215530 -0.0085831763 -0.0071822612 -0.0040076296
##   [26]  0.0285066523 -0.0029425762 -0.0015053669 -0.0272664192 -0.0129708109
##   [31]  0.0024064176 -0.0235557795  0.0212598477 -0.0607357633  0.0122623144
##   [36] -0.0614579929 -0.0120390451 -0.0211311962 -0.0260837912 -0.0022758711
##   [41] -0.0765743770  0.0470633325 -0.0665988974 -0.0185412161 -0.0446368042
##   [46] -0.0300327600  0.0834866618  0.0006958248 -0.0514520212 -0.0656242214
##   [51]  0.0180074841 -0.0007423933 -0.0127227350 -0.0371744859 -0.0594984124
##   [56] -0.0430076919  0.0400011521 -0.0248506678 -0.0553770739  0.0747716249
##   [61]  0.0170040151 -0.0095458840  0.0423393219  0.0354569061 -0.0247697219
##   [66]  0.0882440986 -0.0102010812 -0.0569797860 -0.0057759841  0.0083015336
##   [71]  0.0923191360  0.0042320947  0.0182799522 -0.0030861174 -0.0133600337
##   [76] -0.0013890411  0.0315055842  0.0830098003  0.0410956976  0.0483163359
##   [81] -0.0492529369  0.0393558228  0.0087969921 -0.0586900012  0.0208409097
##   [86] -0.0063501842  0.0585834925 -0.0306432800 -0.0172084702 -0.0370443799
##   [91] -0.0070841585  0.0160804712 -0.0292699269  0.0332149267 -0.0483233115
##   [96] -0.0419193765  0.0576463083 -0.0406338986  0.0164789885 -0.0152430420
```

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

```
##   [1] -1.33804157 -0.90649370 -1.45425117 -0.19603845 -0.86142950
##   [6] -1.33954270 -0.72761717 -0.59443068 -0.69674191 -1.08540715
##  [11] -0.26953887 -0.82354417 -1.25332900 -2.13337780 -0.44582977
##  [16] -1.03817912 -1.02089485 -0.53924643 -0.56962187 -0.71014256
##  [21] -0.56074961 -0.55521030 -0.97130068 -2.00185811 -0.69409476
##  [26] -0.99955772 -1.08084033 -1.73688156 -1.26634145 -0.80400003
##  [31] -0.31825381 -1.07494964 -0.78490435 -1.08763828 -1.67626746
##  [36] -1.26895527 -1.20918402 -1.05078789 -0.47607111 -0.62068800
##  [41] -1.15883618 -1.07961751 -0.71811721 -0.74020962 -1.38901465
##  [46] -1.38378034 -0.73422236 -0.61503771 -1.10762513 -0.62507036
##  [51] -0.78293958 -1.30675559 -0.84216289 -1.60185603 -0.34298656
##  [56] -0.05280774 -1.14360959 -1.54691798 -0.77051726 -0.99275568
##  [61]  0.21781290 -1.02916589 -0.61279100 -0.95054201 -1.39640633
##  [66] -0.81735975 -1.91268040 -0.32420236 -0.92914931  0.09460737
##  [71] -0.66992610 -1.35074112 -0.67635687 -1.47013493 -1.64017673
##  [76] -0.85566592 -1.19014035 -0.91643752 -0.92173364 -1.24644414
##  [81] -1.33358730 -1.02823348 -0.40215951 -1.82047340 -0.68218600
##  [86] -0.83987500 -0.40986659 -1.18273524 -0.83219907 -0.90349498
##  [91] -1.27834417 -0.37998563 -0.44906862 -0.61667825 -0.25490658
##  [96] -0.76267616 -1.58064980 -1.32726661 -1.59582732 -1.25194336
```

```r
#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")
```


Scatterplot between x and y   **Scatterplot when no noise**
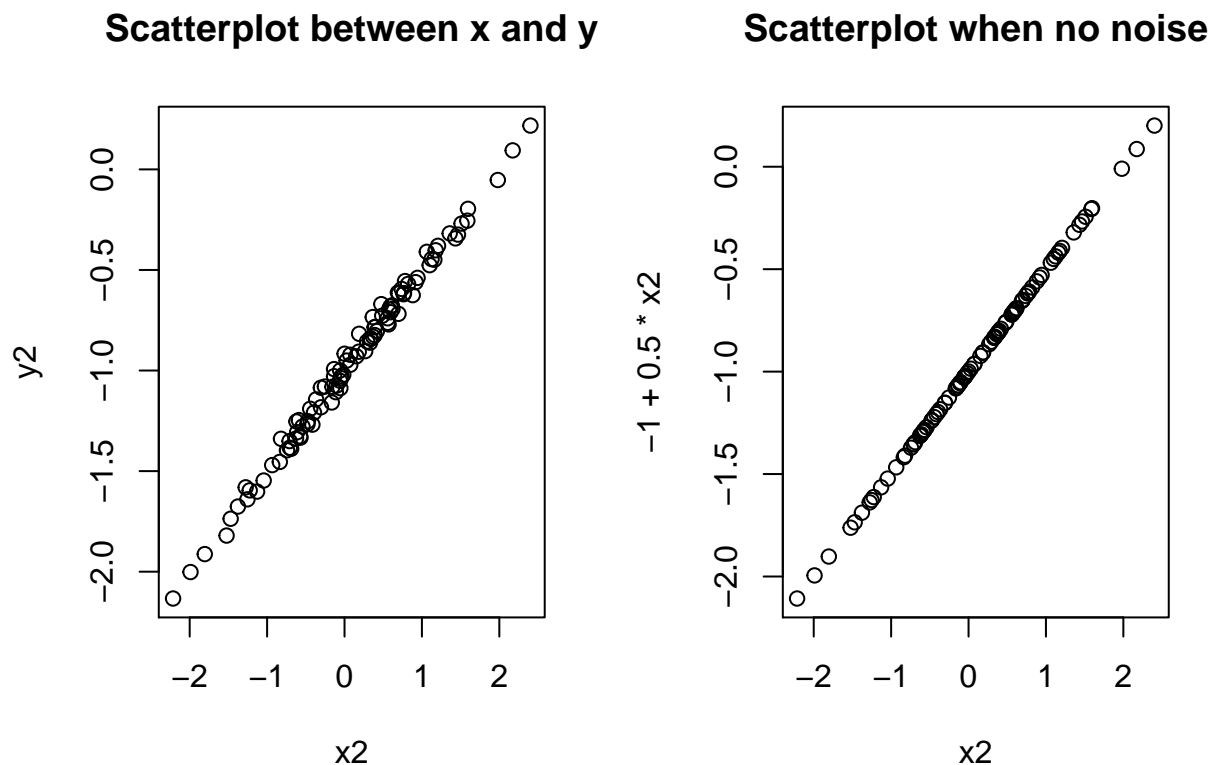
```r
#Part e
par(mfrow=c(1,1))

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

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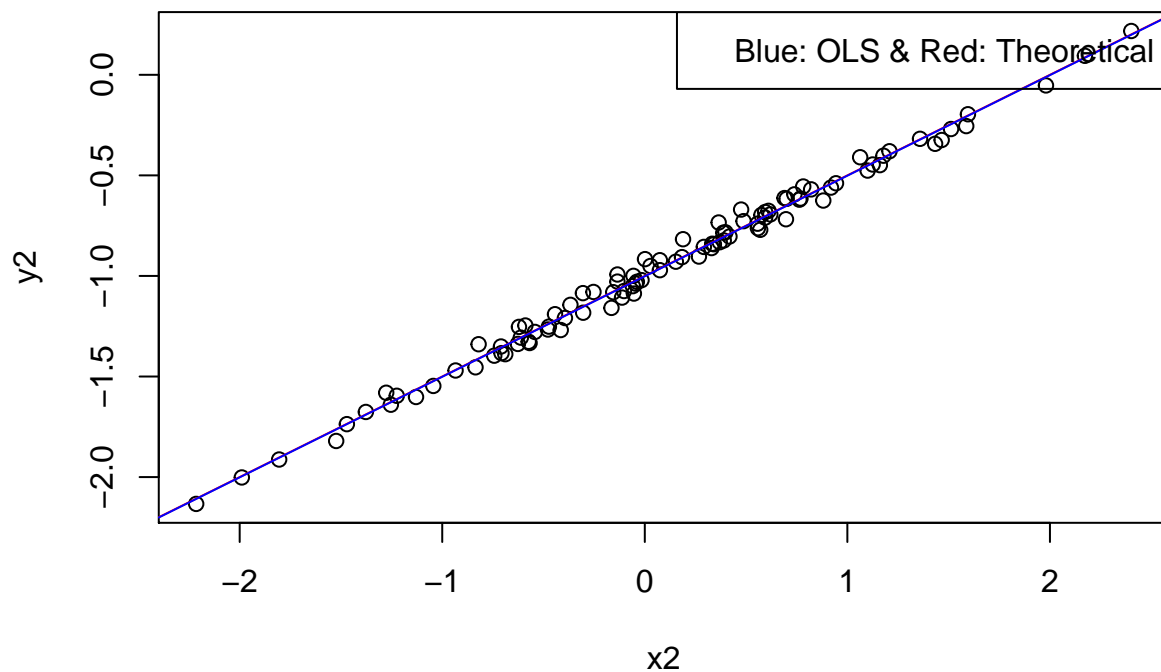
```
## [1] "Least square intercept will be -1.001508 and least square slope will be 0.499958"
```

```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(x2, y2, main = "Scatter plot version2")
abline(a = -1, b = 0.5, col = "red")
abline(lm2, col = "blue")
legend("topright", legend = "Blue: OLS & Red: Theoretical")
```

Scatter plot version2

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, $\beta_0 = $ -1 and $\beta_1 = 0.5$ and $\hat{\beta}_0 \approx -1.001508$ and $\hat{\beta}_1 \approx 0.499958$. The line is more gentle with least square estimators.

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

the difference.

Part i

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

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

```
##   [1] -0.626453811  0.183643324 -0.835628612  1.595280802  0.329507772
##   [6] -0.820468384  0.487429052  0.738324705  0.575781352 -0.305388387
##  [11]  1.511781168  0.389843236 -0.621240581 -2.214699887  1.124930918
##  [16] -0.044933609 -0.016190263  0.943836211  0.821221195  0.593901321
##  [21]  0.918977372  0.782136301  0.074564983 -1.989351696  0.619825748
##  [26] -0.056128740 -0.155795507 -1.470752384 -0.478150055  0.417941560
##  [31]  1.358679552 -0.102787727  0.387671612 -0.053805041 -1.377059557
##  [36] -0.414994563 -0.394289954 -0.059313397  1.100025372  0.763175748
##  [41] -0.164523596 -0.253361680  0.696963375  0.556663199 -0.688755695
##  [46] -0.707495157  0.364581962  0.768532925 -0.112346212  0.881107726
##  [51]  0.398105880 -0.612026393  0.341119691 -1.129363096  1.433023702
##  [56]  1.980399899 -0.367221476 -1.044134626  0.569719627 -0.135054604
##  [61]  2.401617761 -0.039240003  0.689739362  0.028002159 -0.743273209
##  [66]  0.188792300 -1.804958629  1.465554862  0.153253338  2.172611670
##  [71]  0.475509529 -0.709946431  0.610726353 -0.934097632 -1.253633400
##  [76]  0.291446236 -0.443291873  0.001105352  0.074341324 -0.589520946
##  [81] -0.568668733 -0.135178615  1.178086997 -1.523566800  0.593946188
##  [86]  0.332950371  1.063099837 -0.304183924  0.370018810  0.267098791
##  [91] -0.542520031  1.207867806  1.160402616  0.700213650  1.586833455
##  [96]  0.558486426 -1.276592208 -0.573265414 -1.224612615 -0.473400636
```

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

```
##   [1] -0.50249701  0.03411386 -0.73784654  0.12800331 -0.53021356
##   [6]  1.43150269  0.58053306  0.73724113  0.31119014  1.36256263
##  [11] -0.51494653 -0.37393223  1.16014861 -0.52706405 -0.16797840
##  [16] -0.31817442 -0.25919422 -0.22608178  0.40029255 -0.14363769
##  [21] -0.40982554  1.08786145 -0.17380932 -0.14544079 -0.08115450
##  [26]  0.57725971 -0.05958717 -0.03048368 -0.55214499 -0.26265892
##  [31]  0.04872996 -0.47700453  0.43051192 -1.22989921  0.24831187
##  [36] -1.24452436 -0.24379066 -0.42790672 -0.52819677 -0.04608639
##  [41] -1.55063113  0.95303248 -1.34862767 -0.37545963 -0.90389529
##  [46] -0.60816339  1.69060490  0.01409045 -1.04190343 -1.32889048
##  [51]  0.36465155 -0.01503346 -0.25763538 -0.75278334 -1.20484285
##  [56] -0.87090576  0.81002333 -0.50322602 -1.12138575  1.51412540
##  [61]  0.34433131 -0.19330415  0.85737127  0.71800235 -0.50158687
##  [66]  1.78694300 -0.20657189 -1.15384067 -0.11696368  0.16810605
##  [71]  1.86946250  0.08569992  0.37016903 -0.06249388 -0.27054068
##  [76] -0.02812808  0.63798808  1.68094846  0.83218788  0.97840580
##  [81] -0.99737197  0.79695541  0.17813909 -1.18847252  0.42202842
```

```
##  [86] -0.12859123  1.18631572 -0.62052642 -0.34847152 -0.75014869
##  [91] -0.14345421  0.32562954 -0.59271602  0.67260227 -0.97854706
##  [96] -0.84886737  1.16733774 -0.82283645  0.33369952 -0.30867160
```

```r
#Part c
y3 <- -1 + 0.5*x3 + eps3
y3
```

```
##   [1] -1.81572391 -0.87406448 -2.15566084 -0.07435629 -1.36545968
##   [6]  0.02126850 -0.17575242  0.10640348 -0.40091918  0.20986843
##  [11] -0.75905594 -1.17901061 -0.15047168 -2.63441399 -0.60551294
##  [16] -1.34064123 -1.26728936 -0.75416367 -0.18909685 -0.84668703
##  [21] -0.95033686  0.47892960 -1.13652683 -2.14011664 -0.77124163
##  [26] -0.45080466 -1.13748492 -1.76585987 -1.79122002 -1.05368814
##  [31] -0.27193027 -1.52839840 -0.37565228 -2.25680173 -1.44021791
##  [36] -2.45202164 -1.44093564 -1.45756342 -0.97818409 -0.66449852
##  [41] -2.63289293 -0.17364836 -2.00014599 -1.09712803 -2.24827313
##  [46] -1.96191097  0.87289588 -0.60164309 -2.09807654 -1.88833662
##  [51] -0.43629551 -1.32104666 -1.08707554 -2.31746489 -1.48833100
##  [56] -0.88070581 -0.37358741 -2.02529334 -1.83652593  0.44659810
##  [61]  0.54514019 -1.21292415  0.20224095 -0.26799657 -1.87322347
##  [66]  0.88133915 -2.10905121 -1.42106324 -1.04033701  0.25441189
##  [71]  1.10721727 -1.26927330 -0.32446779 -1.52954269 -1.89735738
##  [76] -0.88240497 -0.58365786  0.68150113 -0.13064146 -0.31635467
##  [81] -2.28170634 -0.27063390 -0.23281741 -2.95025592 -0.28099848
##  [86] -0.96211604  0.71786564 -1.77261838 -1.16346212 -1.61659930
##  [91] -1.41471422 -0.07043656 -1.01251471  0.02270909 -1.18513033
##  [96] -1.56962416 -0.47095836 -2.10946915 -1.27860679 -1.54537192
```
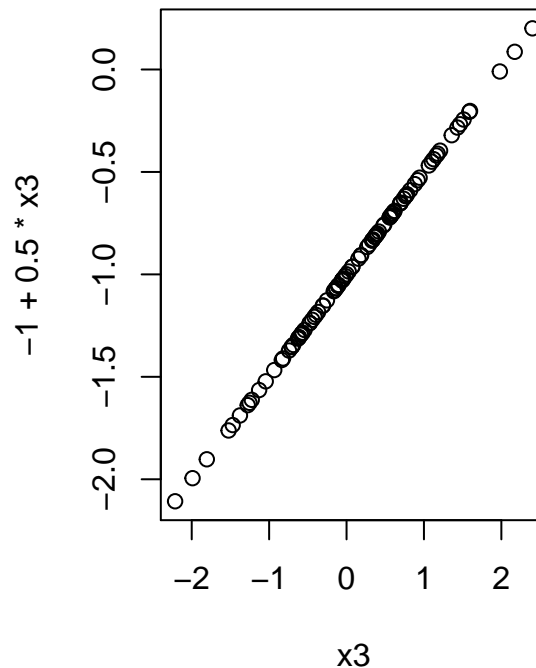
```r
#Part d
par(mfrow=c(1,2))
plot(x3, y3, main = "Scatterplot between x and y")
plot(x3,-1+0.5*x3, main = "Scatterplot when no noise")
```
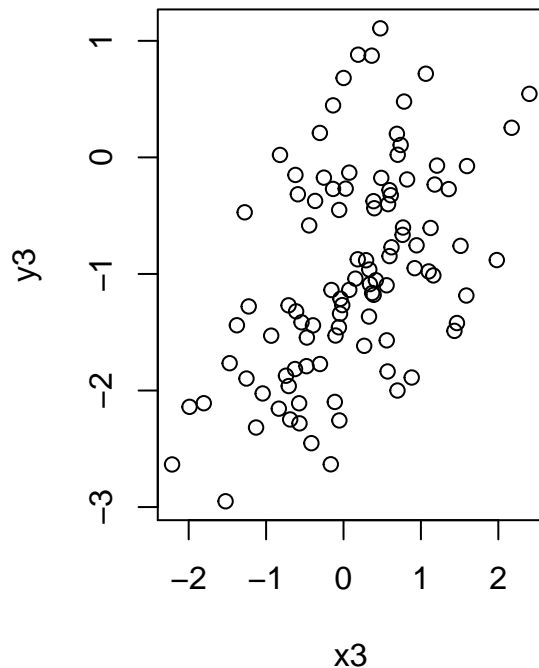
Scatterplot between x and y   **Scatterplot when no noise**


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

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

```
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(x3, y3, main = "Scatter plot version2")
abline(a = -1, b = 0.5, col = "red")
abline(lm3, col = "blue")
legend("topright", legend = "Blue: OLS & Red: Theoretical")
```

Scatter plot version2
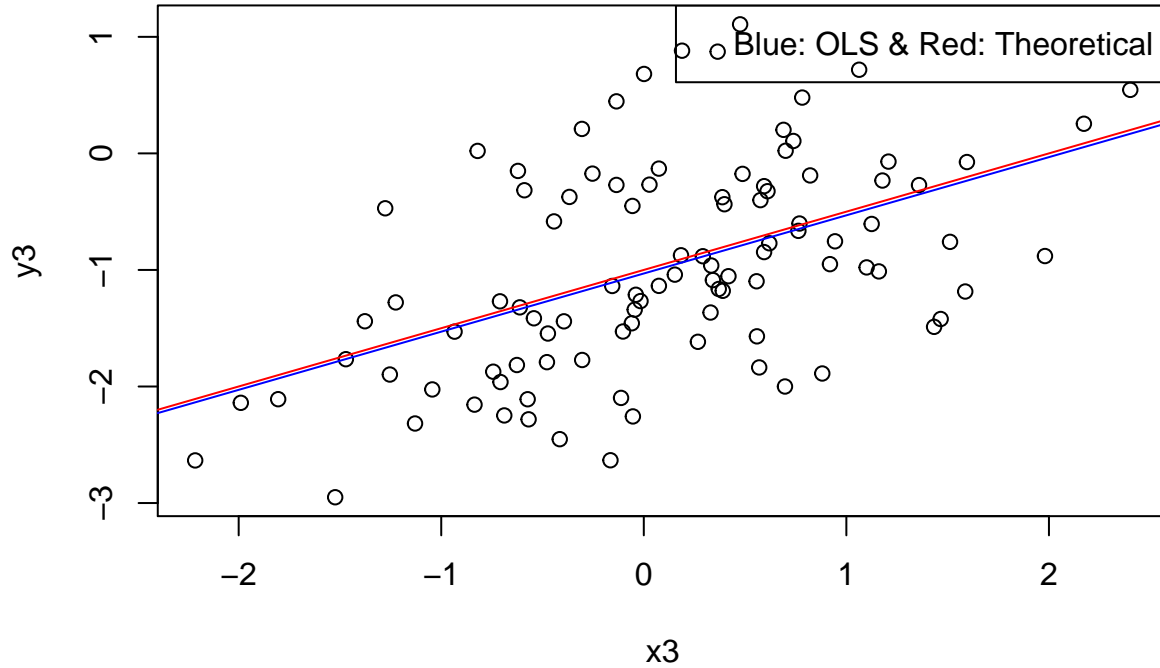
Comment for part d: They show weak linear relationship between x and y. I cannot really guess the slope and intercept from that scatter plot. Additionally, I plotted out the scatter when there is no normally distributed random noise, and as you can see, it will form a perfect line.

Comment for part e: So, $\beta_0 = -1$ and $\beta_1 = 0.5$ and $\hat{\beta}_0 \approx -1.030531$ and $\hat{\beta}_1 \approx 0.499141$. The line is more gentle with least square estimators.

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: $\hat{\beta} = R^{-1}Q^Ty$.

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

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

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 × (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.

```r
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"
```

```r
fit$coefficients
```

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

```r
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:

$\text{RSE} = \sqrt{\frac{RSS}{n - p - 1}} = \sqrt{\frac{\sum (y_i - \hat{y}_i)^2}{n - p - 1}}$, 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.

```
data <- download.file("https://raw.githubusercontent.com/ucb-stat154/stat154-fall-2017/master/data/pros

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)
fit2
```

```
## $coefficients
##           [,1]
## [1,] 1.5072979
## [2,] 0.7193201
##
## $y_values
##           [,1]
##  [1,] -0.43078
##  [2,] -0.16252
##  [3,] -0.16252
##  [4,] -0.16252
##  [5,]  0.37156
##  [6,]  0.76547
##  [7,]  0.76547
##  [8,]  0.85442
##  [9,]  1.04732
## [10,]  1.04732
## [11,]  1.26695
## [12,]  1.26695
## [13,]  1.26695
## [14,]  1.34807
## [15,]  1.39872
## [16,]  1.44692
## [17,]  1.47018
## [18,]  1.49290
## [19,]  1.55814
## [20,]  1.59939
## [21,]  1.63900
## [22,]  1.65823
## [23,]  1.69562
## [24,]  1.71380
## [25,]  1.73166
## [26,]  1.76644
## [27,]  1.80006
## [28,]  1.81645
## [29,]  1.84845
## [30,]  1.89462
## [31,]  1.92425
## [32,]  2.00821
## [33,]  2.00821
## [34,]  2.02155
## [35,]  2.04769
## [36,]  2.08567
## [37,]  2.15756
## [38,]  2.19165
## [39,]  2.21375
## [40,]  2.27727
## [41,]  2.29757
## [42,]  2.30757
## [43,]  2.32728
## [44,]  2.37491
## [45,]  2.52172
## [46,]  2.55334
## [47,]  2.56879
```

```
## [48,]   2.56879
## [49,]   2.59152
## [50,]   2.59152
## [51,]   2.65676
## [52,]   2.67759
## [53,]   2.68444
## [54,]   2.69124
## [55,]   2.70471
## [56,]   2.71800
## [57,]   2.78809
## [58,]   2.79423
## [59,]   2.80639
## [60,]   2.81241
## [61,]   2.84200
## [62,]   2.85359
## [63,]   2.85359
## [64,]   2.88200
## [65,]   2.88200
## [66,]   2.88759
## [67,]   2.92047
## [68,]   2.96269
## [69,]   2.96269
## [70,]   2.97298
## [71,]   3.01308
## [72,]   3.03735
## [73,]   3.05636
## [74,]   3.07501
## [75,]   3.27526
## [76,]   3.33755
## [77,]   3.39283
## [78,]   3.43560
## [79,]   3.45789
## [80,]   3.51304
## [81,]   3.51601
## [82,]   3.53076
## [83,]   3.56530
## [84,]   3.57094
## [85,]   3.58768
## [86,]   3.63099
## [87,]   3.68009
## [88,]   3.71235
## [89,]   3.98434
## [90,]   3.99360
## [91,]   4.02981
## [92,]   4.12955
## [93,]   4.38515
## [94,]   4.68444
## [95,]   5.14312
## [96,]   5.47751
## [97,]   5.58293
##
## $fitted_values
##            [,1]
##  [1,] 1.0902228
```

```
##  [2,] 0.7921122
##  [3,] 1.1398508
##  [4,] 0.6412560
##  [5,] 2.0478067
##  [6,] 0.7521398
##  [7,] 2.0375549
##  [8,] 2.0058927
##  [9,] 0.9487251
## [10,] 1.6678096
## [11,] 1.6904672
## [12,] 0.5383208
## [13,] 2.6678706
## [14,] 2.5697688
## [15,] 2.3747770
## [16,] 2.6158847
## [17,] 1.2084093
## [18,] 3.1534521
## [19,] 1.1029545
## [20,] 1.6384455
## [21,] 2.3326477
## [22,] 2.9885499
## [23,] 1.1154647
## [24,] 2.7889172
## [25,] 1.7844249
## [26,] 2.5480959
## [27,] 1.8761823
## [28,] 1.2192263
## [29,] 2.2555899
## [30,] 3.2406035
## [31,] 1.7124329
## [32,] 1.6384455
## [33,] 2.4246921
## [34,] 1.5144554
## [35,] 1.5000685
## [36,] 2.4484081
## [37,] 2.5309684
## [38,] 1.8363328
## [39,] 3.4213790
## [40,] 2.0809609
## [41,] 1.9536911
## [42,] 2.5447029
## [43,] 1.9260973
## [44,] 2.7816144
## [45,] 2.5763081
## [46,] 2.7041935
## [47,] 3.4694974
## [48,] 2.3439757
## [49,] 2.7630262
## [50,] 2.3854655
## [51,] 2.2927404
## [52,] 2.7014636
## [53,] 1.8761823
## [54,] 3.0373210
## [55,] 3.7757390
```

```
## [56,] 2.4186389
## [57,] 2.2083183
## [58,] 1.8408711
## [59,] 1.8974027
## [60,] 2.2706811
## [61,] 1.8363328
## [62,] 2.9440807
## [63,] 3.5039212
## [64,] 2.9709026
## [65,] 2.9985722
## [66,] 2.5565091
## [67,] 2.9623899
## [68,] 3.0886046
## [69,] 1.1862746
## [70,] 2.3661104
## [71,] 2.8481683
## [72,] 2.3417243
## [73,] 2.3812091
## [74,] 2.8300997
## [75,] 3.6647017
## [76,] 3.7667764
## [77,] 2.9537752
## [78,] 3.3326859
## [79,] 3.4122736
## [80,] 3.5066052
## [81,] 2.5631695
## [82,] 3.3154214
## [83,] 3.3868863
## [84,] 3.4333431
## [85,] 2.6311251
## [86,] 3.8831039
## [87,] 2.9633408
## [88,] 2.7529126
## [89,] 3.5268567
## [90,] 2.6311251
## [91,] 3.8425643
## [92,] 3.3292659
## [93,] 3.5431666
## [94,] 4.2558228
## [95,] 3.5986834
## [96,] 3.5807840
## [97,] 4.0047534
##
## $residuals
##              [,1]
## [1,] -1.521002808
## [2,] -0.954632229
## [3,] -1.302370790
## [4,] -0.803776050
## [5,] -1.676246665
## [6,]  0.013330247
## [7,] -1.272084915
## [8,] -1.151472672
## [9,]  0.098594873
```

```
## [10,] -0.620489617
## [11,] -0.423517195
## [12,]  0.728629237
## [13,] -1.400920552
## [14,] -1.221698808
## [15,] -0.976057015
## [16,] -1.168964710
## [17,]  0.261770663
## [18,] -1.660552144
## [19,]  0.455185513
## [20,] -0.039055531
## [21,] -0.693647658
## [22,] -1.330319872
## [23,]  0.580155313
## [24,] -1.075117170
## [25,] -0.052764935
## [26,] -0.781655908
## [27,] -0.076122275
## [28,]  0.597223671
## [29,] -0.407139912
## [30,] -1.345983533
## [31,]  0.211817142
## [32,]  0.369764469
## [33,] -0.416482079
## [34,]  0.507094617
## [35,]  0.547621451
## [36,] -0.362738064
## [37,] -0.373408392
## [38,]  0.355317197
## [39,] -1.207629042
## [40,]  0.196309078
## [41,]  0.343878894
## [42,] -0.237132875
## [43,]  0.401182662
## [44,] -0.406704416
## [45,] -0.054588148
## [46,] -0.150853486
## [47,] -0.900707387
## [48,]  0.224814273
## [49,] -0.171506248
## [50,]  0.206054535
## [51,]  0.364019648
## [52,] -0.023873594
## [53,]  0.808257725
## [54,] -0.346081000
## [55,] -1.071029017
## [56,]  0.299361144
## [57,]  0.579771721
## [58,]  0.953358863
## [59,]  0.908987277
## [60,]  0.541728895
## [61,]  1.005667197
## [62,] -0.090490710
## [63,] -0.650331172
```

```
## [64,] -0.088902647
## [65,] -0.116572231
## [66,]  0.331080923
## [67,] -0.041919925
## [68,] -0.125914642
## [69,]  1.776415367
## [70,]  0.606869569
## [71,]  0.164911711
## [72,]  0.695625673
## [73,]  0.675150896
## [74,]  0.244910314
## [75,] -0.389441739
## [76,] -0.429226360
## [77,]  0.439054797
## [78,]  0.102914138
## [79,]  0.045616400
## [80,]  0.006434757
## [81,]  0.952840522
## [82,]  0.215338613
## [83,]  0.178413725
## [84,]  0.137596938
## [85,]  0.956554910
## [86,] -0.252113950
## [87,]  0.716749206
## [88,]  0.959437393
## [89,]  0.457483305
## [90,]  1.362474910
## [91,]  0.187245710
## [92,]  0.800284074
## [93,]  0.841983441
## [94,]  0.428617227
## [95,]  1.544436600
## [96,]  1.896726018
## [97,]  1.578176642
##
## $n
## [1] 97
##
## $q
## [1] 2
```

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

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

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

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

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

```
## [1] 0.5394319
rse2 <- sqrt(sum((fit2[[2]] - fit2[[3]])^2) / (nrow(data) - fit2[[6]]))
rse2
```

```
## [1] 0.7874994
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

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

fit3 <- ols_fit(predictor, response)
fit3
```

```
## $coefficients
##             [,1]
## [1,] -0.3026179
## [2,]  0.6775253
## [3,]  0.5109495
##
## $y_values
##             [,1]
##  [1,] -0.43078
##  [2,] -0.16252
##  [3,] -0.16252
##  [4,] -0.16252
##  [5,]  0.37156
##  [6,]  0.76547
##  [7,]  0.76547
##  [8,]  0.85442
##  [9,]  1.04732
## [10,]  1.04732
## [11,]  1.26695
## [12,]  1.26695
## [13,]  1.26695
## [14,]  1.34807
## [15,]  1.39872
## [16,]  1.44692
## [17,]  1.47018
```

```
## [18,]   1.49290
## [19,]   1.55814
## [20,]   1.59939
## [21,]   1.63900
## [22,]   1.65823
## [23,]   1.69562
## [24,]   1.71380
## [25,]   1.73166
## [26,]   1.76644
## [27,]   1.80006
## [28,]   1.81645
## [29,]   1.84845
## [30,]   1.89462
## [31,]   1.92425
## [32,]   2.00821
## [33,]   2.00821
## [34,]   2.02155
## [35,]   2.04769
## [36,]   2.08567
## [37,]   2.15756
## [38,]   2.19165
## [39,]   2.21375
## [40,]   2.27727
## [41,]   2.29757
## [42,]   2.30757
## [43,]   2.32728
## [44,]   2.37491
## [45,]   2.52172
## [46,]   2.55334
## [47,]   2.56879
## [48,]   2.56879
## [49,]   2.59152
## [50,]   2.59152
## [51,]   2.65676
## [52,]   2.67759
## [53,]   2.68444
## [54,]   2.69124
## [55,]   2.70471
## [56,]   2.71800
## [57,]   2.78809
## [58,]   2.79423
## [59,]   2.80639
## [60,]   2.81241
## [61,]   2.84200
## [62,]   2.85359
## [63,]   2.85359
## [64,]   2.88200
## [65,]   2.88200
## [66,]   2.88759
## [67,]   2.92047
## [68,]   2.96269
## [69,]   2.96269
## [70,]   2.97298
## [71,]   3.01308
```

```
## [72,]   3.03735
## [73,]   3.05636
## [74,]   3.07501
## [75,]   3.27526
## [76,]   3.33755
## [77,]   3.39283
## [78,]   3.43560
## [79,]   3.45789
## [80,]   3.51304
## [81,]   3.51601
## [82,]   3.53076
## [83,]   3.56530
## [84,]   3.57094
## [85,]   3.58768
## [86,]   3.63099
## [87,]   3.68009
## [88,]   3.71235
## [89,]   3.98434
## [90,]   3.99360
## [91,]   4.02981
## [92,]   4.12955
## [93,]   4.38515
## [94,]   4.68444
## [95,]   5.14312
## [96,]   5.47751
## [97,]   5.58293
##
## $fitted_values
##                [,1]
##  [1,] 0.7196150
##  [2,] 0.7198989
##  [3,] 0.7263521
##  [4,] 0.5590050
##  [5,] 1.9602684
##  [6,] 0.6358548
##  [7,] 1.9716124
##  [8,] 1.9755125
##  [9,] 0.9797699
## [10,] 1.5063431
## [11,] 1.7114216
## [12,] 0.6234596
## [13,] 2.3350708
## [14,] 2.2300486
## [15,] 2.2731458
## [16,] 2.3056237
## [17,] 1.2123583
## [18,] 3.1125483
## [19,] 0.9861619
## [20,] 1.7754957
## [21,] 2.2219169
## [22,] 2.8814025
## [23,] 1.0532300
## [24,] 2.6681282
## [25,] 1.8322632
```

```
## [26,] 2.2742190
## [27,] 1.9454118
## [28,] 1.4013791
## [29,] 2.0009567
## [30,] 3.0548912
## [31,] 1.9804835
## [32,] 2.9415845
## [33,] 2.1134305
## [34,] 1.3737533
## [35,] 1.3342461
## [36,] 2.6888202
## [37,] 2.5301672
## [38,] 1.2207529
## [39,] 3.5875284
## [40,] 1.7772552
## [41,] 1.7232416
## [42,] 2.5561329
## [43,] 2.0671785
## [44,] 2.8887755
## [45,] 2.4463615
## [46,] 2.5582834
## [47,] 3.5870188
## [48,] 2.5471783
## [49,] 2.6674497
## [50,] 2.3476440
## [51,] 2.4777155
## [52,] 2.9859316
## [53,] 1.9014190
## [54,] 3.2443840
## [55,] 3.6305176
## [56,] 2.7426859
## [57,] 1.8215922
## [58,] 1.9351451
## [59,] 2.1996696
## [60,] 2.3841788
## [61,] 2.3190898
## [62,] 2.9512618
## [63,] 3.3790407
## [64,] 3.0773356
## [65,] 2.9531783
## [66,] 2.6457349
## [67,] 3.0496459
## [68,] 3.2566148
## [69,] 1.6475320
## [70,] 2.9488376
## [71,] 2.7962871
## [72,] 2.1904588
## [73,] 2.4751022
## [74,] 2.5971148
## [75,] 3.6961292
## [76,] 3.4932142
## [77,] 3.3252620
## [78,] 3.6417917
## [79,] 3.3219445
```

```
## [80,] 3.5339850
## [81,] 2.2607233
## [82,] 3.1752306
## [83,] 3.4547404
## [84,] 3.4727461
## [85,] 2.6514826
## [86,] 3.7331771
## [87,] 2.9755342
## [88,] 2.5920112
## [89,] 4.0103085
## [90,] 2.6439206
## [91,] 3.9927743
## [92,] 3.2925555
## [93,] 3.5956045
## [94,] 4.2773275
## [95,] 3.4025377
## [96,] 3.5786639
## [97,] 4.0807512
##
## $residuals
##                 [,1]
##  [1,] -1.150394988
##  [2,] -0.882418917
##  [3,] -0.888872077
##  [4,] -0.721525042
##  [5,] -1.588708441
##  [6,]  0.129615236
##  [7,] -1.206142374
##  [8,] -1.121092476
##  [9,]  0.067550128
## [10,] -0.459023079
## [11,] -0.444471600
## [12,]  0.643490431
## [13,] -1.068120757
## [14,] -0.881978599
## [15,] -0.874425844
## [16,] -0.858703734
## [17,]  0.257821745
## [18,] -1.619648282
## [19,]  0.571978068
## [20,] -0.176105681
## [21,] -0.582916884
## [22,] -1.223172480
## [23,]  0.642390021
## [24,] -0.954328196
## [25,] -0.100603184
## [26,] -0.507778977
## [27,] -0.145351775
## [28,]  0.415070947
## [29,] -0.152506727
## [30,] -1.160271210
## [31,] -0.056233515
## [32,] -0.933374527
## [33,] -0.105220527
```

```
## [34,]   0.647796738
## [35,]   0.713443882
## [36,]  -0.603150224
## [37,]  -0.372607211
## [38,]   0.970897138
## [39,]  -1.373778416
## [40,]   0.500014781
## [41,]   0.574328414
## [42,]  -0.248562886
## [43,]   0.260101487
## [44,]  -0.513865549
## [45,]   0.075358508
## [46,]  -0.004943411
## [47,]  -1.018228760
## [48,]   0.021611666
## [49,]  -0.075929678
## [50,]   0.243876014
## [51,]   0.179044498
## [52,]  -0.308341576
## [53,]   0.783020973
## [54,]  -0.553143980
## [55,]  -0.925807567
## [56,]  -0.024685904
## [57,]   0.966497828
## [58,]   0.859084943
## [59,]   0.606720430
## [60,]   0.428231232
## [61,]   0.522910190
## [62,]  -0.097671769
## [63,]  -0.525450690
## [64,]  -0.195335590
## [65,]  -0.071178340
## [66,]   0.241855055
## [67,]  -0.129175932
## [68,]  -0.293924841
## [69,]   1.315157960
## [70,]   0.024142433
## [71,]   0.216792928
## [72,]   0.846891174
## [73,]   0.581257772
## [74,]   0.477895166
## [75,]  -0.420869186
## [76,]  -0.155664217
## [77,]   0.067568014
## [78,]  -0.206191672
## [79,]   0.135945545
## [80,]  -0.020945030
## [81,]   1.255286738
## [82,]   0.355529434
## [83,]   0.110559585
## [84,]   0.098193943
## [85,]   0.936197396
## [86,]  -0.102187089
## [87,]   0.704555828
```

```
## [88,]   1.120338807
## [89,]  -0.025968472
## [90,]   1.349679448
## [91,]   0.037035740
## [92,]   0.836994525
## [93,]   0.789545476
## [94,]   0.407112489
## [95,]   1.740582276
## [96,]   1.898846070
## [97,]   1.502178786
##
## $n
## [1] 97
##
## $q
## [1] 3
```

```r
#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

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

```
fit4 <- ols_fit(predictor, response)
fit4
```

```
## $coefficients
##              [,1]
## [1,] -0.2680926
## [2,]  0.5516380
## [3,]  0.5085413
## [4,]  0.6661584
##
## $y_values
##              [,1]
##  [1,] -0.43078
##  [2,] -0.16252
##  [3,] -0.16252
##  [4,] -0.16252
##  [5,]  0.37156
##  [6,]  0.76547
##  [7,]  0.76547
##  [8,]  0.85442
##  [9,]  1.04732
## [10,]  1.04732
## [11,]  1.26695
## [12,]  1.26695
## [13,]  1.26695
## [14,]  1.34807
## [15,]  1.39872
## [16,]  1.44692
## [17,]  1.47018
## [18,]  1.49290
## [19,]  1.55814
## [20,]  1.59939
## [21,]  1.63900
## [22,]  1.65823
## [23,]  1.69562
## [24,]  1.71380
## [25,]  1.73166
## [26,]  1.76644
## [27,]  1.80006
## [28,]  1.81645
## [29,]  1.84845
## [30,]  1.89462
## [31,]  1.92425
## [32,]  2.00821
## [33,]  2.00821
## [34,]  2.02155
## [35,]  2.04769
## [36,]  2.08567
## [37,]  2.15756
## [38,]  2.19165
## [39,]  2.21375
## [40,]  2.27727
## [41,]  2.29757
## [42,]  2.30757
```

```
## [43,]   2.32728
## [44,]   2.37491
## [45,]   2.52172
## [46,]   2.55334
## [47,]   2.56879
## [48,]   2.56879
## [49,]   2.59152
## [50,]   2.59152
## [51,]   2.65676
## [52,]   2.67759
## [53,]   2.68444
## [54,]   2.69124
## [55,]   2.70471
## [56,]   2.71800
## [57,]   2.78809
## [58,]   2.79423
## [59,]   2.80639
## [60,]   2.81241
## [61,]   2.84200
## [62,]   2.85359
## [63,]   2.85359
## [64,]   2.88200
## [65,]   2.88200
## [66,]   2.88759
## [67,]   2.92047
## [68,]   2.96269
## [69,]   2.96269
## [70,]   2.97298
## [71,]   3.01308
## [72,]   3.03735
## [73,]   3.05636
## [74,]   3.07501
## [75,]   3.27526
## [76,]   3.33755
## [77,]   3.39283
## [78,]   3.43560
## [79,]   3.45789
## [80,]   3.51304
## [81,]   3.51601
## [82,]   3.53076
## [83,]   3.56530
## [84,]   3.57094
## [85,]   3.58768
## [86,]   3.63099
## [87,]   3.68009
## [88,]   3.71235
## [89,]   3.98434
## [90,]   3.99360
## [91,]   4.02981
## [92,]   4.12955
## [93,]   4.38515
## [94,]   4.68444
## [95,]   5.14312
## [96,]   5.47751
```

```
## [97,]  5.58293
##
## $fitted_values
##              [,1]
##  [1,] 0.8204627
##  [2,] 0.8715938
##  [3,] 0.8187030
##  [4,] 0.7371897
##  [5,] 1.8919343
##  [6,] 0.7947638
##  [7,] 1.9049735
##  [8,] 1.9142558
##  [9,] 1.1035266
## [10,] 1.5049642
## [11,] 1.7052115
## [12,] 0.8188981
## [13,] 2.1592063
## [14,] 2.0714122
## [15,] 2.1475660
## [16,] 2.1387652
## [17,] 1.2907246
## [18,] 2.8501941
## [19,] 1.0835817
## [20,] 1.7778569
## [21,] 2.1037645
## [22,] 2.6482650
## [23,] 1.1481998
## [24,] 2.4700471
## [25,] 1.8094573
## [26,] 2.1190712
## [27,] 1.9064216
## [28,] 1.4770095
## [29,] 1.8969894
## [30,] 2.7779434
## [31,] 1.9692587
## [32,] 2.9384499
## [33,] 1.9800894
## [34,] 1.3991569
## [35,] 1.3622899
## [36,] 2.5487221
## [37,] 2.3767345
## [38,] 1.1919751
## [39,] 3.9433939
## [40,] 1.7041286
## [41,] 1.6720778
## [42,] 2.4002352
## [43,] 2.0191004
## [44,] 2.6909002
## [45,] 2.2855902
## [46,] 2.3751713
## [47,] 3.9346791
## [48,] 2.4255607
## [49,] 2.4737880
## [50,] 2.2198899
```

```
## [51,] 2.3651645
## [52,] 2.8012696
## [53,] 1.8626362
## [54,] 3.0012169
## [55,] 3.2595791
## [56,] 2.6074116
## [57,] 1.7265333
## [58,] 1.9022262
## [59,] 2.1558615
## [60,] 2.2758312
## [61,] 2.2851355
## [62,] 3.3915385
## [63,] 3.0556513
## [64,] 3.5124431
## [65,] 2.7179931
## [66,] 2.4874011
## [67,] 2.8201776
## [68,] 3.0046427
## [69,] 1.7276229
## [70,] 2.8215514
## [71,] 3.2536539
## [72,] 2.0709065
## [73,] 3.0136318
## [74,] 3.0585023
## [75,] 4.0099794
## [76,] 3.7906100
## [77,] 3.0959641
## [78,] 3.3463714
## [79,] 3.6806147
## [80,] 3.2094075
## [81,] 2.1030680
## [82,] 2.8849540
## [83,] 3.8171151
## [84,] 3.1609534
## [85,] 2.4803945
## [86,] 4.0096000
## [87,] 2.7462530
## [88,] 3.0665885
## [89,] 4.3461901
## [90,] 3.1390264
## [91,] 3.6087301
## [92,] 3.6655228
## [93,] 3.9306587
## [94,] 4.4876113
## [95,] 3.7290324
## [96,] 3.9073816
## [97,] 4.3347863
##
## $residuals
##              [,1]
## [1,] -1.25124267
## [2,] -1.03411381
## [3,] -0.98122300
## [4,] -0.89970969
```

```
##  [5,] -1.52037435
##  [6,] -0.02929385
##  [7,] -1.13950345
##  [8,] -1.05983578
##  [9,] -0.05620661
## [10,] -0.45764423
## [11,] -0.43826152
## [12,]  0.44805195
## [13,] -0.89225627
## [14,] -0.72334224
## [15,] -0.74884600
## [16,] -0.69184522
## [17,]  0.17945539
## [18,] -1.35729413
## [19,]  0.47455826
## [20,] -0.17846692
## [21,] -0.46476447
## [22,] -0.99003502
## [23,]  0.54742017
## [24,] -0.75624714
## [25,] -0.07779725
## [26,] -0.35263117
## [27,] -0.10636157
## [28,]  0.33944050
## [29,] -0.04853940
## [30,] -0.88332344
## [31,] -0.04500866
## [32,] -0.93023993
## [33,]  0.02812055
## [34,]  0.62239314
## [35,]  0.68540013
## [36,] -0.46305208
## [37,] -0.21917454
## [38,]  0.99967488
## [39,] -1.72964388
## [40,]  0.57314143
## [41,]  0.62549215
## [42,] -0.09266516
## [43,]  0.30817956
## [44,] -0.31599020
## [45,]  0.23612977
## [46,]  0.17816867
## [47,] -1.36588911
## [48,]  0.14322928
## [49,]  0.11773198
## [50,]  0.37163009
## [51,]  0.29159554
## [52,] -0.12367960
## [53,]  0.82180383
## [54,] -0.30997690
## [55,] -0.55486909
## [56,]  0.11058839
## [57,]  1.06155671
## [58,]  0.89200376
```

```
## [59,]   0.65052852
## [60,]   0.53657880
## [61,]   0.55686445
## [62,]  -0.53794849
## [63,]  -0.20206125
## [64,]  -0.63044312
## [65,]   0.16400691
## [66,]   0.40018886
## [67,]   0.10029238
## [68,]  -0.04195270
## [69,]   1.23506710
## [70,]   0.15142859
## [71,]  -0.24057392
## [72,]   0.96644353
## [73,]   0.04272821
## [74,]   0.01650766
## [75,]  -0.73471940
## [76,]  -0.45305997
## [77,]   0.29686591
## [78,]   0.08922862
## [79,]  -0.22272474
## [80,]   0.30363249
## [81,]   1.41294202
## [82,]   0.64580601
## [83,]  -0.25181513
## [84,]   0.40998655
## [85,]   1.10728550
## [86,]  -0.37860996
## [87,]   0.93383704
## [88,]   0.64576149
## [89,]  -0.36185009
## [90,]   0.85457356
## [91,]   0.42107989
## [92,]   0.46402716
## [93,]   0.45449130
## [94,]   0.19682866
## [95,]   1.41408764
## [96,]   1.57012843
## [97,]   1.24814372
##
## $n
## [1] 97
##
## $q
## [1] 4
```

```r
#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
```

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

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

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

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

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

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

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

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

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

```r
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

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

fit5 <- ols_fit(predictor, response)
fit5
```

```
## $coefficients
##            [,1]
## [1,] 0.1455407
## [2,] 0.5496031
## [3,] 0.3908759
## [4,] 0.7117370
## [5,] 0.0900933
##
## $y_values
##            [,1]
##  [1,] -0.43078
##  [2,] -0.16252
##  [3,] -0.16252
##  [4,] -0.16252
##  [5,]  0.37156
##  [6,]  0.76547
##  [7,]  0.76547
##  [8,]  0.85442
##  [9,]  1.04732
## [10,]  1.04732
## [11,]  1.26695
## [12,]  1.26695
```

```
## [13,]   1.26695
## [14,]   1.34807
## [15,]   1.39872
## [16,]   1.44692
## [17,]   1.47018
## [18,]   1.49290
## [19,]   1.55814
## [20,]   1.59939
## [21,]   1.63900
## [22,]   1.65823
## [23,]   1.69562
## [24,]   1.71380
## [25,]   1.73166
## [26,]   1.76644
## [27,]   1.80006
## [28,]   1.81645
## [29,]   1.84845
## [30,]   1.89462
## [31,]   1.92425
## [32,]   2.00821
## [33,]   2.00821
## [34,]   2.02155
## [35,]   2.04769
## [36,]   2.08567
## [37,]   2.15756
## [38,]   2.19165
## [39,]   2.21375
## [40,]   2.27727
## [41,]   2.29757
## [42,]   2.30757
## [43,]   2.32728
## [44,]   2.37491
## [45,]   2.52172
## [46,]   2.55334
## [47,]   2.56879
## [48,]   2.56879
## [49,]   2.59152
## [50,]   2.59152
## [51,]   2.65676
## [52,]   2.67759
## [53,]   2.68444
## [54,]   2.69124
## [55,]   2.70471
## [56,]   2.71800
## [57,]   2.78809
## [58,]   2.79423
## [59,]   2.80639
## [60,]   2.81241
## [61,]   2.84200
## [62,]   2.85359
## [63,]   2.85359
## [64,]   2.88200
## [65,]   2.88200
## [66,]   2.88759
```

```
## [67,]   2.92047
## [68,]   2.96269
## [69,]   2.96269
## [70,]   2.97298
## [71,]   3.01308
## [72,]   3.03735
## [73,]   3.05636
## [74,]   3.07501
## [75,]   3.27526
## [76,]   3.33755
## [77,]   3.39283
## [78,]   3.43560
## [79,]   3.45789
## [80,]   3.51304
## [81,]   3.51601
## [82,]   3.53076
## [83,]   3.56530
## [84,]   3.57094
## [85,]   3.58768
## [86,]   3.63099
## [87,]   3.68009
## [88,]   3.71235
## [89,]   3.98434
## [90,]   3.99360
## [91,]   4.02981
## [92,]   4.12955
## [93,]   4.38515
## [94,]   4.68444
## [95,]   5.14312
## [96,]   5.47751
## [97,]   5.58293
##
## $fitted_values
##              [,1]
##  [1,] 0.7845057
##  [2,] 0.7717524
##  [3,] 0.7918188
##  [4,] 0.6421051
##  [5,] 1.7752680
##  [6,] 0.7057195
##  [7,] 1.9638200
##  [8,] 2.0484633
##  [9,] 0.9773675
## [10,] 1.4114822
## [11,] 1.5693529
## [12,] 0.9259735
## [13,] 2.0889699
## [14,] 2.0043597
## [15,] 2.0288451
## [16,] 2.0641811
## [17,] 1.4035819
## [18,] 2.7048667
## [19,] 0.9889678
## [20,] 1.8903972
```

```
## [21,] 1.9878220
## [22,] 2.7786277
## [23,] 1.0937747
## [24,] 2.5134048
## [25,] 1.9348746
## [26,] 2.1891403
## [27,] 1.7564355
## [28,] 1.6002134
## [29,] 1.9604346
## [30,] 2.6645509
## [31,] 2.0778815
## [32,] 2.7866453
## [33,] 2.1478742
## [34,] 1.3033788
## [35,] 1.2725300
## [36,] 2.6705580
## [37,] 2.3049201
## [38,] 1.2003382
## [39,] 4.0402787
## [40,] 1.8459371
## [41,] 1.5898478
## [42,] 2.2527233
## [43,] 2.1310562
## [44,] 2.5175024
## [45,] 2.4526104
## [46,] 2.4416277
## [47,] 4.0875470
## [48,] 2.5164372
## [49,] 2.3473796
## [50,] 2.3349596
## [51,] 2.1817714
## [52,] 2.9000142
## [53,] 1.9821776
## [54,] 3.0847085
## [55,] 3.1281878
## [56,] 2.7060487
## [57,] 1.6761645
## [58,] 2.0001534
## [59,] 2.1162454
## [60,] 2.3507982
## [61,] 2.3750451
## [62,] 3.5549019
## [63,] 2.9239817
## [64,] 3.6875459
## [65,] 2.5762101
## [66,] 2.5657596
## [67,] 2.9340009
## [68,] 3.1450487
## [69,] 1.4985406
## [70,] 2.8798517
## [71,] 3.1613815
## [72,] 2.2466386
## [73,] 2.8953584
## [74,] 3.1726084
```

```
## [75,] 3.8852865
## [76,] 3.8547725
## [77,] 3.1750021
## [78,] 3.4520157
## [79,] 3.5880526
## [80,] 3.0426308
## [81,] 2.2028521
## [82,] 2.9242453
## [83,] 3.7658964
## [84,] 3.2179581
## [85,] 2.6070853
## [86,] 3.9231306
## [87,] 2.8643379
## [88,] 3.0009661
## [89,] 4.1196359
## [90,] 3.2446094
## [91,] 3.4082214
## [92,] 3.8083066
## [93,] 3.8030973
## [94,] 4.3556218
## [95,] 3.6578169
## [96,] 4.0570487
## [97,] 4.3586970
##
## $residuals
##              [,1]
## [1,] -1.21528569
## [2,] -0.93427241
## [3,] -0.95433882
## [4,] -0.80462513
## [5,] -1.40370804
## [6,]  0.05975046
## [7,] -1.19835005
## [8,] -1.19404332
## [9,]  0.06995246
## [10,] -0.36416224
## [11,] -0.30240294
## [12,]  0.34097648
## [13,] -0.82201985
## [14,] -0.65628968
## [15,] -0.63012514
## [16,] -0.61726105
## [17,]  0.06659810
## [18,] -1.21196667
## [19,]  0.56917218
## [20,] -0.29100720
## [21,] -0.34882203
## [22,] -1.12039767
## [23,]  0.60184529
## [24,] -0.79960478
## [25,] -0.20321462
## [26,] -0.42270028
## [27,]  0.04362450
## [28,]  0.21623659
```

```
## [29,] -0.11198457
## [30,] -0.76993093
## [31,] -0.15363146
## [32,] -0.77843533
## [33,] -0.13966415
## [34,]  0.71817115
## [35,]  0.77516004
## [36,] -0.58488804
## [37,] -0.14736009
## [38,]  0.99131177
## [39,] -1.82652869
## [40,]  0.43133289
## [41,]  0.70772217
## [42,]  0.05484670
## [43,]  0.19622376
## [44,] -0.14259243
## [45,]  0.06910960
## [46,]  0.11171233
## [47,] -1.51875698
## [48,]  0.05235283
## [49,]  0.24414042
## [50,]  0.25656038
## [51,]  0.47498857
## [52,] -0.22242416
## [53,]  0.70226245
## [54,] -0.39346853
## [55,] -0.42347781
## [56,]  0.01195132
## [57,]  1.11192549
## [58,]  0.79407657
## [59,]  0.69014457
## [60,]  0.46161183
## [61,]  0.46695488
## [62,] -0.70131187
## [63,] -0.07039169
## [64,] -0.80554585
## [65,]  0.30578987
## [66,]  0.32183036
## [67,] -0.01353090
## [68,] -0.18235870
## [69,]  1.46414942
## [70,]  0.09312829
## [71,] -0.14830148
## [72,]  0.79071143
## [73,]  0.16100157
## [74,] -0.09759839
## [75,] -0.61002649
## [76,] -0.51722252
## [77,]  0.21782786
## [78,] -0.01641574
## [79,] -0.13016259
## [80,]  0.47040924
## [81,]  1.31315790
## [82,]  0.60651467
```

```
## [83,] -0.20059642
## [84,]  0.35298193
## [85,]  0.98059465
## [86,] -0.29214056
## [87,]  0.81575210
## [88,]  0.71138387
## [89,] -0.13529588
## [90,]  0.74899059
## [91,]  0.62158862
## [92,]  0.32124338
## [93,]  0.58205267
## [94,]  0.32881820
## [95,]  1.48530313
## [96,]  1.42046127
## [97,]  1.22423303
##
## $n
## [1] 97
##
## $q
## [1] 5
```

```r
#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
```

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

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

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

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

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

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

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

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

```r
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

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

fit6 <- ols_fit(predictor, response)
fit6
```

```
## $coefficients
##               [,1]
## [1,]   0.95099742
## [2,]   0.56560801
## [3,]   0.42369200
## [4,]   0.72095499
## [5,]   0.11183992
## [6,]  -0.01489225
##
## $y_values
##               [,1]
##   [1,]  -0.43078
##   [2,]  -0.16252
##   [3,]  -0.16252
##   [4,]  -0.16252
##   [5,]   0.37156
##   [6,]   0.76547
##   [7,]   0.76547
##   [8,]   0.85442
##   [9,]   1.04732
## [10,]   1.04732
## [11,]   1.26695
## [12,]   1.26695
## [13,]   1.26695
## [14,]   1.34807
## [15,]   1.39872
## [16,]   1.44692
## [17,]   1.47018
## [18,]   1.49290
## [19,]   1.55814
## [20,]   1.59939
## [21,]   1.63900
## [22,]   1.65823
## [23,]   1.69562
## [24,]   1.71380
## [25,]   1.73166
## [26,]   1.76644
## [27,]   1.80006
## [28,]   1.81645
## [29,]   1.84845
## [30,]   1.89462
## [31,]   1.92425
## [32,]   2.00821
## [33,]   2.00821
## [34,]   2.02155
## [35,]   2.04769
## [36,]   2.08567
```

```
## [37,]   2.15756
## [38,]   2.19165
## [39,]   2.21375
## [40,]   2.27727
## [41,]   2.29757
## [42,]   2.30757
## [43,]   2.32728
## [44,]   2.37491
## [45,]   2.52172
## [46,]   2.55334
## [47,]   2.56879
## [48,]   2.56879
## [49,]   2.59152
## [50,]   2.59152
## [51,]   2.65676
## [52,]   2.67759
## [53,]   2.68444
## [54,]   2.69124
## [55,]   2.70471
## [56,]   2.71800
## [57,]   2.78809
## [58,]   2.79423
## [59,]   2.80639
## [60,]   2.81241
## [61,]   2.84200
## [62,]   2.85359
## [63,]   2.85359
## [64,]   2.88200
## [65,]   2.88200
## [66,]   2.88759
## [67,]   2.92047
## [68,]   2.96269
## [69,]   2.96269
## [70,]   2.97298
## [71,]   3.01308
## [72,]   3.03735
## [73,]   3.05636
## [74,]   3.07501
## [75,]   3.27526
## [76,]   3.33755
## [77,]   3.39283
## [78,]   3.43560
## [79,]   3.45789
## [80,]   3.51304
## [81,]   3.51601
## [82,]   3.53076
## [83,]   3.56530
## [84,]   3.57094
## [85,]   3.58768
## [86,]   3.63099
## [87,]   3.68009
## [88,]   3.71235
## [89,]   3.98434
## [90,]   3.99360
```

```
## [91,]  4.02981
## [92,]  4.12955
## [93,]  4.38515
## [94,]  4.68444
## [95,]  5.14312
## [96,]  5.47751
## [97,]  5.58293
##
## $fitted_values
##              [,1]
##  [1,] 0.8968067
##  [2,] 0.7763345
##  [3,] 0.5452404
##  [4,] 0.6421231
##  [5,] 1.7519220
##  [6,] 0.8255706
##  [7,] 1.9553355
##  [8,] 2.1508372
##  [9,] 1.1564654
## [10,] 1.3586229
## [11,] 1.4990139
## [12,] 0.9173055
## [13,] 2.0510898
## [14,] 1.9039173
## [15,] 2.0875505
## [16,] 1.9817211
## [17,] 1.3023680
## [18,] 2.6536733
## [19,] 1.2519314
## [20,] 1.8179096
## [21,] 2.0150638
## [22,] 2.8704672
## [23,] 1.1052891
## [24,] 2.5319640
## [25,] 1.8740628
## [26,] 2.1141448
## [27,] 1.6940221
## [28,] 1.5678481
## [29,] 1.8922939
## [30,] 2.6212137
## [31,] 2.0968168
## [32,] 2.8645236
## [33,] 2.0436202
## [34,] 1.3818990
## [35,] 1.2150326
## [36,] 2.7262651
## [37,] 2.1534215
## [38,] 1.1077994
## [39,] 4.0487991
## [40,] 1.9494265
## [41,] 1.5846624
## [42,] 2.1592902
## [43,] 2.1866477
## [44,] 2.5406189
```

```
## [45,] 2.4488894
## [46,] 2.4900044
## [47,] 3.9413779
## [48,] 2.4975221
## [49,] 2.6250527
## [50,] 2.2644625
## [51,] 2.0929380
## [52,] 2.9629908
## [53,] 1.9944438
## [54,] 3.0852006
## [55,] 3.1907084
## [56,] 2.7355021
## [57,] 1.8611571
## [58,] 2.2378022
## [59,] 2.0345669
## [60,] 2.4193499
## [61,] 2.2997538
## [62,] 3.6206152
## [63,] 2.7871471
## [64,] 3.7241094
## [65,] 2.5504889
## [66,] 2.6407665
## [67,] 2.9252200
## [68,] 3.0965637
## [69,] 1.3838115
## [70,] 2.8396379
## [71,] 3.2001228
## [72,] 2.0716386
## [73,] 2.7972996
## [74,] 3.2389266
## [75,] 3.8165631
## [76,] 3.8130370
## [77,] 3.1320526
## [78,] 3.3299881
## [79,] 3.5049507
## [80,] 3.0496753
## [81,] 2.1618414
## [82,] 3.0445915
## [83,] 3.5918293
## [84,] 3.2484845
## [85,] 2.7026302
## [86,] 3.9228953
## [87,] 2.9965432
## [88,] 3.0004462
## [89,] 4.1359316
## [90,] 3.0940935
## [91,] 3.3574223
## [92,] 3.9050931
## [93,] 3.7474579
## [94,] 4.6739258
## [95,] 3.8259305
## [96,] 4.0629142
## [97,] 4.3562413
##
```

```
## $residuals
##                [,1]
##  [1,] -1.327586672
##  [2,] -0.938854526
##  [3,] -0.707760406
##  [4,] -0.804643065
##  [5,] -1.380361998
##  [6,] -0.060100608
##  [7,] -1.189865549
##  [8,] -1.296417170
##  [9,] -0.109145359
## [10,] -0.311302857
## [11,] -0.232063852
## [12,]  0.349644490
## [13,] -0.784139780
## [14,] -0.555847268
## [15,] -0.688830497
## [16,] -0.534801106
## [17,]  0.167811989
## [18,] -1.160773308
## [19,]  0.306208641
## [20,] -0.218519606
## [21,] -0.376063848
## [22,] -1.212237166
## [23,]  0.590330931
## [24,] -0.818164022
## [25,] -0.142402806
## [26,] -0.347704813
## [27,]  0.106037883
## [28,]  0.248601871
## [29,] -0.043843858
## [30,] -0.726593735
## [31,] -0.172566824
## [32,] -0.856313567
## [33,] -0.035410189
## [34,]  0.639651032
## [35,]  0.832657381
## [36,] -0.640595089
## [37,]  0.004138511
## [38,]  1.083850641
## [39,] -1.835049071
## [40,]  0.327843456
## [41,]  0.712907576
## [42,]  0.148279752
## [43,]  0.140632263
## [44,] -0.165708943
## [45,]  0.072830613
## [46,]  0.063335618
## [47,] -1.372587917
## [48,]  0.071267871
## [49,] -0.033532746
## [50,]  0.327057530
## [51,]  0.563821978
## [52,] -0.285400785
```

```
## [53,]   0.689996234
## [54,] -0.393960556
## [55,] -0.485998447
## [56,] -0.017502063
## [57,]   0.926932899
## [58,]   0.556427838
## [59,]   0.771823126
## [60,]   0.393060146
## [61,]   0.542246161
## [62,] -0.767025175
## [63,]   0.066442870
## [64,] -0.842109446
## [65,]   0.331511144
## [66,]   0.246823454
## [67,] -0.004749983
## [68,] -0.133873726
## [69,]   1.578878548
## [70,]   0.133342122
## [71,] -0.187042849
## [72,]   0.965711357
## [73,]   0.259060445
## [74,] -0.163916558
## [75,] -0.541303068
## [76,] -0.475487008
## [77,]   0.260777396
## [78,]   0.105611856
## [79,] -0.047060745
## [80,]   0.463364741
## [81,]   1.354168608
## [82,]   0.486168506
## [83,] -0.026529335
## [84,]   0.322455452
## [85,]   0.885049771
## [86,] -0.291905270
## [87,]   0.683546790
## [88,]   0.711903823
## [89,] -0.151591619
## [90,]   0.899506492
## [91,]   0.672387748
## [92,]   0.224456934
## [93,]   0.637692110
## [94,]   0.010514198
## [95,]   1.317189531
## [96,]   1.414595793
## [97,]   1.226688737
##
## $n
## [1] 97
##
## $q
## [1] 6
```

```r
#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
```

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

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

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

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

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

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

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

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

```r
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

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

fit7 <- ols_fit(predictor, response)
fit7
```

```
## $coefficients
##               [,1]
## [1,]   0.93486843
## [2,]   0.58764668
## [3,]   0.41808376
## [4,]   0.78256448
## [5,]   0.11381222
## [6,]  -0.01511242
## [7,]  -0.04118380
##
## $y_values
##             [,1]
##  [1,] -0.43078
##  [2,] -0.16252
##  [3,] -0.16252
```

52

```
##  [4,] -0.16252
##  [5,]  0.37156
##  [6,]  0.76547
##  [7,]  0.76547
##  [8,]  0.85442
##  [9,]  1.04732
## [10,]  1.04732
## [11,]  1.26695
## [12,]  1.26695
## [13,]  1.26695
## [14,]  1.34807
## [15,]  1.39872
## [16,]  1.44692
## [17,]  1.47018
## [18,]  1.49290
## [19,]  1.55814
## [20,]  1.59939
## [21,]  1.63900
## [22,]  1.65823
## [23,]  1.69562
## [24,]  1.71380
## [25,]  1.73166
## [26,]  1.76644
## [27,]  1.80006
## [28,]  1.81645
## [29,]  1.84845
## [30,]  1.89462
## [31,]  1.92425
## [32,]  2.00821
## [33,]  2.00821
## [34,]  2.02155
## [35,]  2.04769
## [36,]  2.08567
## [37,]  2.15756
## [38,]  2.19165
## [39,]  2.21375
## [40,]  2.27727
## [41,]  2.29757
## [42,]  2.30757
## [43,]  2.32728
## [44,]  2.37491
## [45,]  2.52172
## [46,]  2.55334
## [47,]  2.56879
## [48,]  2.56879
## [49,]  2.59152
## [50,]  2.59152
## [51,]  2.65676
## [52,]  2.67759
## [53,]  2.68444
## [54,]  2.69124
## [55,]  2.70471
## [56,]  2.71800
## [57,]  2.78809
```

```
## [58,]   2.79423
## [59,]   2.80639
## [60,]   2.81241
## [61,]   2.84200
## [62,]   2.85359
## [63,]   2.85359
## [64,]   2.88200
## [65,]   2.88200
## [66,]   2.88759
## [67,]   2.92047
## [68,]   2.96269
## [69,]   2.96269
## [70,]   2.97298
## [71,]   3.01308
## [72,]   3.03735
## [73,]   3.05636
## [74,]   3.07501
## [75,]   3.27526
## [76,]   3.33755
## [77,]   3.39283
## [78,]   3.43560
## [79,]   3.45789
## [80,]   3.51304
## [81,]   3.51601
## [82,]   3.53076
## [83,]   3.56530
## [84,]   3.57094
## [85,]   3.58768
## [86,]   3.63099
## [87,]   3.68009
## [88,]   3.71235
## [89,]   3.98434
## [90,]   3.99360
## [91,]   4.02981
## [92,]   4.12955
## [93,]   4.38515
## [94,]   4.68444
## [95,]   5.14312
## [96,]   5.47751
## [97,]   5.58293
##
## $fitted_values
##              [,1]
##  [1,] 0.8957174
##  [2,] 0.7612652
##  [3,] 0.5408267
##  [4,] 0.6226382
##  [5,] 1.7738116
##  [6,] 0.8115472
##  [7,] 1.9801878
##  [8,] 2.1774880
##  [9,] 1.1473830
## [10,] 1.3697037
## [11,] 1.5083318
```

```
## [12,] 0.8970274
## [13,] 2.0615821
## [14,] 1.9431331
## [15,] 2.0811534
## [16,] 2.0222173
## [17,] 1.2690265
## [18,] 2.6349453
## [19,] 1.2504196
## [20,] 1.8292965
## [21,] 2.0464139
## [22,] 2.8142666
## [23,] 1.1007501
## [24,] 2.4741967
## [25,] 1.8909124
## [26,] 2.1550937
## [27,] 1.6841747
## [28,] 1.5671357
## [29,] 1.8652225
## [30,] 2.5555195
## [31,] 2.0864802
## [32,] 2.8643039
## [33,] 2.0825236
## [34,] 1.3901327
## [35,] 1.1969219
## [36,] 2.7631497
## [37,] 2.0626380
## [38,] 1.1287002
## [39,] 4.0422787
## [40,] 1.9301854
## [41,] 1.6057375
## [42,] 2.1936797
## [43,] 2.1691394
## [44,] 2.4921164
## [45,] 2.4530517
## [46,] 2.5363943
## [47,] 3.9014677
## [48,] 2.4905475
## [49,] 2.6726707
## [50,] 2.2754115
## [51,] 2.1178636
## [52,] 3.0067900
## [53,] 1.9560830
## [54,] 3.0217572
## [55,] 3.2657304
## [56,] 2.7700385
## [57,] 1.8167318
## [58,] 2.2598912
## [59,] 2.0495030
## [60,] 2.4512272
## [61,] 2.3139400
## [62,] 3.5999364
## [63,] 2.7296662
## [64,] 3.6949879
## [65,] 2.5999989
```

```
## [66,] 2.6421870
## [67,] 2.8660317
## [68,] 3.1126050
## [69,] 1.3722899
## [70,] 2.8446335
## [71,] 3.1961543
## [72,] 2.1059270
## [73,] 2.8209858
## [74,] 3.2458962
## [75,] 3.8099880
## [76,] 3.7946871
## [77,] 3.1030849
## [78,] 3.3898708
## [79,] 3.4643623
## [80,] 3.0414530
## [81,] 2.1382258
## [82,] 2.9568485
## [83,] 3.6320469
## [84,] 3.1856892
## [85,] 2.6568643
## [86,] 3.9087596
## [87,] 3.0516520
## [88,] 3.0364521
## [89,] 4.0983117
## [90,] 3.1049994
## [91,] 3.4292248
## [92,] 4.0320918
## [93,] 3.7614333
## [94,] 4.6800101
## [95,] 3.8007869
## [96,] 4.0746892
## [97,] 4.3222347
##
## $residuals
##                [,1]
## [1,] -1.326497353
## [2,] -0.923785202
## [3,] -0.703346702
## [4,] -0.785158163
## [5,] -1.402251606
## [6,] -0.046077168
## [7,] -1.214717754
## [8,] -1.323067997
## [9,] -0.100062962
## [10,] -0.322383662
## [11,] -0.241381785
## [12,]  0.369922626
## [13,] -0.794632065
## [14,] -0.595063114
## [15,] -0.682433350
## [16,] -0.575297269
## [17,]  0.201153514
## [18,] -1.142045293
## [19,]  0.307720403
```

```
## [20,] -0.229906469
## [21,] -0.407413879
## [22,] -1.156036591
## [23,]  0.594869939
## [24,] -0.760396657
## [25,] -0.159252434
## [26,] -0.388653683
## [27,]  0.115885303
## [28,]  0.249314254
## [29,] -0.016772522
## [30,] -0.660899521
## [31,] -0.162230207
## [32,] -0.856093888
## [33,] -0.074313631
## [34,]  0.631417328
## [35,]  0.850768087
## [36,] -0.677479688
## [37,]  0.094922007
## [38,]  1.062949830
## [39,] -1.828528712
## [40,]  0.347084568
## [41,]  0.691832531
## [42,]  0.113890329
## [43,]  0.158140555
## [44,] -0.117206362
## [45,]  0.068668303
## [46,]  0.016945723
## [47,] -1.332677659
## [48,]  0.078242483
## [49,] -0.081150673
## [50,]  0.316108483
## [51,]  0.538896396
## [52,] -0.329199989
## [53,]  0.728356998
## [54,] -0.330517246
## [55,] -0.561020436
## [56,] -0.052038460
## [57,]  0.971358153
## [58,]  0.534338808
## [59,]  0.756887028
## [60,]  0.361182845
## [61,]  0.528059969
## [62,] -0.746346380
## [63,]  0.123923801
## [64,] -0.812987948
## [65,]  0.282001062
## [66,]  0.245403000
## [67,]  0.054438283
## [68,] -0.149915025
## [69,]  1.590400105
## [70,]  0.128346496
## [71,] -0.183074314
## [72,]  0.931422977
## [73,]  0.235374161
```

```
## [74,] -0.170886150
## [75,] -0.534727972
## [76,] -0.457137072
## [77,]  0.289745095
## [78,]  0.045729226
## [79,] -0.006472281
## [80,]  0.471586992
## [81,]  1.377784221
## [82,]  0.573911497
## [83,] -0.066746864
## [84,]  0.385250765
## [85,]  0.930815702
## [86,] -0.277769621
## [87,]  0.628437953
## [88,]  0.675897945
## [89,] -0.113971714
## [90,]  0.888600626
## [91,]  0.600585170
## [92,]  0.097458213
## [93,]  0.623716747
## [94,]  0.004429905
## [95,]  1.342333081
## [96,]  1.402820754
## [97,]  1.260695254
##
## $n
## [1] 97
##
## $q
## [1] 7
```

```r
#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
```

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

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

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

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

```r
#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)
fit8
```

```
## $coefficients
##               [,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
##
## $y_values
##              [,1]
##  [1,] -0.43078
##  [2,] -0.16252
##  [3,] -0.16252
##  [4,] -0.16252
##  [5,]  0.37156
##  [6,]  0.76547
##  [7,]  0.76547
##  [8,]  0.85442
##  [9,]  1.04732
## [10,]  1.04732
## [11,]  1.26695
## [12,]  1.26695
## [13,]  1.26695
## [14,]  1.34807
## [15,]  1.39872
## [16,]  1.44692
## [17,]  1.47018
## [18,]  1.49290
## [19,]  1.55814
## [20,]  1.59939
```

```
## [21,]   1.63900
## [22,]   1.65823
## [23,]   1.69562
## [24,]   1.71380
## [25,]   1.73166
## [26,]   1.76644
## [27,]   1.80006
## [28,]   1.81645
## [29,]   1.84845
## [30,]   1.89462
## [31,]   1.92425
## [32,]   2.00821
## [33,]   2.00821
## [34,]   2.02155
## [35,]   2.04769
## [36,]   2.08567
## [37,]   2.15756
## [38,]   2.19165
## [39,]   2.21375
## [40,]   2.27727
## [41,]   2.29757
## [42,]   2.30757
## [43,]   2.32728
## [44,]   2.37491
## [45,]   2.52172
## [46,]   2.55334
## [47,]   2.56879
## [48,]   2.56879
## [49,]   2.59152
## [50,]   2.59152
## [51,]   2.65676
## [52,]   2.67759
## [53,]   2.68444
## [54,]   2.69124
## [55,]   2.70471
## [56,]   2.71800
## [57,]   2.78809
## [58,]   2.79423
## [59,]   2.80639
## [60,]   2.81241
## [61,]   2.84200
## [62,]   2.85359
## [63,]   2.85359
## [64,]   2.88200
## [65,]   2.88200
## [66,]   2.88759
## [67,]   2.92047
## [68,]   2.96269
## [69,]   2.96269
## [70,]   2.97298
## [71,]   3.01308
## [72,]   3.03735
## [73,]   3.05636
## [74,]   3.07501
```

```
## [75,]   3.27526
## [76,]   3.33755
## [77,]   3.39283
## [78,]   3.43560
## [79,]   3.45789
## [80,]   3.51304
## [81,]   3.51601
## [82,]   3.53076
## [83,]   3.56530
## [84,]   3.57094
## [85,]   3.58768
## [86,]   3.63099
## [87,]   3.68009
## [88,]   3.71235
## [89,]   3.98434
## [90,]   3.99360
## [91,]   4.02981
## [92,]   4.12955
## [93,]   4.38515
## [94,]   4.68444
## [95,]   5.14312
## [96,]   5.47751
## [97,]   5.58293
##
## $fitted_values
##             [,1]
##  [1,] 0.8811992
##  [2,] 0.7279282
##  [3,] 0.5291943
##  [4,] 0.5873573
##  [5,] 1.7339126
##  [6,] 0.8090390
##  [7,] 1.9207343
##  [8,] 2.1395377
##  [9,] 1.1680171
## [10,] 1.3178083
## [11,] 1.4589764
## [12,] 0.8333075
## [13,] 2.1181324
## [14,] 1.8984651
## [15,] 2.0305743
## [16,] 1.9573392
## [17,] 1.2866999
## [18,] 2.4795351
## [19,] 1.2890379
## [20,] 1.7465223
## [21,] 2.0203655
## [22,] 2.7057803
## [23,] 1.0630637
## [24,] 2.5802423
## [25,] 1.8087559
## [26,] 2.0729555
## [27,] 1.9743696
## [28,] 1.6013300
```

```
## [29,] 2.1208784
## [30,] 2.3175663
## [31,] 1.9941569
## [32,] 2.8713064
## [33,] 1.9784609
## [34,] 1.3761085
## [35,] 1.1060619
## [36,] 2.7425184
## [37,] 1.8678271
## [38,] 1.1270065
## [39,] 3.9449190
## [40,] 1.8380106
## [41,] 1.9904111
## [42,] 2.1919127
## [43,] 2.0621471
## [44,] 2.3673467
## [45,] 2.4250963
## [46,] 2.5706179
## [47,] 4.0455749
## [48,] 2.5783325
## [49,] 2.7189552
## [50,] 2.1535269
## [51,] 2.3368098
## [52,] 2.9650429
## [53,] 2.1765866
## [54,] 2.9967980
## [55,] 3.2771490
## [56,] 2.7991158
## [57,] 1.7257635
## [58,] 2.2665485
## [59,] 2.0926609
## [60,] 2.6256766
## [61,] 2.2366014
## [62,] 3.5302341
## [63,] 2.9771573
## [64,] 3.7099859
## [65,] 2.5626542
## [66,] 2.6507576
## [67,] 3.0006679
## [68,] 3.0249011
## [69,] 1.3275570
## [70,] 2.7715546
## [71,] 3.2967917
## [72,] 2.1152179
## [73,] 2.7448784
## [74,] 3.4930494
## [75,] 3.6349298
## [76,] 3.7257328
## [77,] 3.2342610
## [78,] 3.3477167
## [79,] 3.5030389
## [80,] 3.1717984
## [81,] 2.1722215
## [82,] 3.0190977
```

```
## [83,] 3.5562150
## [84,] 3.3114316
## [85,] 2.6385379
## [86,] 3.9323204
## [87,] 3.0241620
## [88,] 3.0264815
## [89,] 4.1432619
## [90,] 3.2377715
## [91,] 3.3941034
## [92,] 4.0490351
## [93,] 3.8406672
## [94,] 4.7051849
## [95,] 3.5952276
## [96,] 4.2243436
## [97,] 4.0829163
##
## $residuals
##                 [,1]
##  [1,] -1.311979212
##  [2,] -0.890448195
##  [3,] -0.691714347
##  [4,] -0.749877336
##  [5,] -1.362352587
##  [6,] -0.043568960
##  [7,] -1.155264308
##  [8,] -1.285117666
##  [9,] -0.120697068
## [10,] -0.270488292
## [11,] -0.192026376
## [12,]  0.433642452
## [13,] -0.851182377
## [14,] -0.550395073
## [15,] -0.631854301
## [16,] -0.510419243
## [17,]  0.183480079
## [18,] -0.986635111
## [19,]  0.269102085
## [20,] -0.147132257
## [21,] -0.381365457
## [22,] -1.047550318
## [23,]  0.632556257
## [24,] -0.866442344
## [25,] -0.077095868
## [26,] -0.306515481
## [27,] -0.174309579
## [28,]  0.215119959
## [29,] -0.272428395
## [30,] -0.422946312
## [31,] -0.069906866
## [32,] -0.863096368
## [33,]  0.029749098
## [34,]  0.645441508
## [35,]  0.941628068
## [36,] -0.656848386
```

```
## [37,]   0.289732852
## [38,]   1.064643527
## [39,]  -1.731169025
## [40,]   0.439259393
## [41,]   0.307158883
## [42,]   0.115657330
## [43,]   0.265132913
## [44,]   0.007563347
## [45,]   0.096623713
## [46,]  -0.017277912
## [47,]  -1.476784931
## [48,]  -0.009542516
## [49,]  -0.127435155
## [50,]   0.437993070
## [51,]   0.319950178
## [52,]  -0.287452894
## [53,]   0.507853437
## [54,]  -0.305558032
## [55,]  -0.572438970
## [56,]  -0.081115758
## [57,]   1.062326515
## [58,]   0.527681545
## [59,]   0.713729050
## [60,]   0.186733419
## [61,]   0.605398563
## [62,]  -0.676644119
## [63,]  -0.123567322
## [64,]  -0.827985868
## [65,]   0.319345824
## [66,]   0.236832435
## [67,]  -0.080197950
## [68,]  -0.062211071
## [69,]   1.635132970
## [70,]   0.201425389
## [71,]  -0.283711694
## [72,]   0.922132145
## [73,]   0.311481623
## [74,]  -0.418039414
## [75,]  -0.359669833
## [76,]  -0.388182819
## [77,]   0.158568956
## [78,]   0.087883262
## [79,]  -0.045148942
## [80,]   0.341241556
## [81,]   1.343788491
## [82,]   0.511662311
## [83,]   0.009084955
## [84,]   0.259508449
## [85,]   0.949142095
## [86,]  -0.301330363
## [87,]   0.655927968
## [88,]   0.685868476
## [89,]  -0.158921866
## [90,]   0.755828511
```

```
## [91,]   0.635706572
## [92,]   0.080514914
## [93,]   0.544482812
## [94,] -0.020744883
## [95,]   1.547892400
## [96,]   1.253166357
## [97,]   1.500013710
##
## $n
## [1] 97
##
## $q
## [1] 8
```

```
#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)
fit9
```

```
## $coefficients
##              [,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
##
## $y_values
##              [,1]
##  [1,] -0.43078
##  [2,] -0.16252
##  [3,] -0.16252
##  [4,] -0.16252
##  [5,]  0.37156
##  [6,]  0.76547
##  [7,]  0.76547
##  [8,]  0.85442
##  [9,]  1.04732
## [10,]  1.04732
## [11,]  1.26695
## [12,]  1.26695
## [13,]  1.26695
## [14,]  1.34807
## [15,]  1.39872
## [16,]  1.44692
## [17,]  1.47018
## [18,]  1.49290
## [19,]  1.55814
## [20,]  1.59939
## [21,]  1.63900
## [22,]  1.65823
## [23,]  1.69562
## [24,]  1.71380
## [25,]  1.73166
## [26,]  1.76644
## [27,]  1.80006
## [28,]  1.81645
## [29,]  1.84845
## [30,]  1.89462
## [31,]  1.92425
## [32,]  2.00821
## [33,]  2.00821
## [34,]  2.02155
```

```
## [35,]    2.04769
## [36,]    2.08567
## [37,]    2.15756
## [38,]    2.19165
## [39,]    2.21375
## [40,]    2.27727
## [41,]    2.29757
## [42,]    2.30757
## [43,]    2.32728
## [44,]    2.37491
## [45,]    2.52172
## [46,]    2.55334
## [47,]    2.56879
## [48,]    2.56879
## [49,]    2.59152
## [50,]    2.59152
## [51,]    2.65676
## [52,]    2.67759
## [53,]    2.68444
## [54,]    2.69124
## [55,]    2.70471
## [56,]    2.71800
## [57,]    2.78809
## [58,]    2.79423
## [59,]    2.80639
## [60,]    2.81241
## [61,]    2.84200
## [62,]    2.85359
## [63,]    2.85359
## [64,]    2.88200
## [65,]    2.88200
## [66,]    2.88759
## [67,]    2.92047
## [68,]    2.96269
## [69,]    2.96269
## [70,]    2.97298
## [71,]    3.01308
## [72,]    3.03735
## [73,]    3.05636
## [74,]    3.07501
## [75,]    3.27526
## [76,]    3.33755
## [77,]    3.39283
## [78,]    3.43560
## [79,]    3.45789
## [80,]    3.51304
## [81,]    3.51601
## [82,]    3.53076
## [83,]    3.56530
## [84,]    3.57094
## [85,]    3.58768
## [86,]    3.63099
## [87,]    3.68009
## [88,]    3.71235
```

```
## [89,]   3.98434
## [90,]   3.99360
## [91,]   4.02981
## [92,]   4.12955
## [93,]   4.38515
## [94,]   4.68444
## [95,]   5.14312
## [96,]   5.47751
## [97,]   5.58293
##
## $fitted_values
##              [,1]
##  [1,] 0.8744185
##  [2,] 0.7240419
##  [3,] 0.5436880
##  [4,] 0.5842070
##  [5,] 1.7215026
##  [6,] 0.8072530
##  [7,] 1.9068071
##  [8,] 2.1274558
##  [9,] 1.1677967
## [10,] 1.3063635
## [11,] 1.4490060
## [12,] 0.8296243
## [13,] 2.1195193
## [14,] 1.9197169
## [15,] 2.0578709
## [16,] 1.9378065
## [17,] 1.2967219
## [18,] 2.4584591
## [19,] 1.2879588
## [20,] 1.7348683
## [21,] 2.0069587
## [22,] 2.7132032
## [23,] 1.0567972
## [24,] 2.5568413
## [25,] 1.7955313
## [26,] 2.0526057
## [27,] 1.9530127
## [28,] 1.6227001
## [29,] 2.0830917
## [30,] 2.2933079
## [31,] 1.9843982
## [32,] 2.8752199
## [33,] 1.9568618
## [34,] 1.3684926
## [35,] 1.0949346
## [36,] 2.7701779
## [37,] 1.9252901
## [38,] 1.1420697
## [39,] 3.9468978
## [40,] 1.8631363
## [41,] 2.0494372
## [42,] 2.2132876
```

```
## [43,] 2.0503309
## [44,] 2.3916276
## [45,] 2.4343772
## [46,] 2.5860927
## [47,] 4.0810274
## [48,] 2.5765308
## [49,] 2.7080974
## [50,] 2.1351071
## [51,] 2.3300150
## [52,] 2.9506782
## [53,] 2.1523816
## [54,] 2.9892078
## [55,] 3.2963041
## [56,] 2.8194593
## [57,] 1.7534362
## [58,] 2.2596875
## [59,] 2.1115771
## [60,] 2.6265188
## [61,] 2.2266866
## [62,] 3.5303174
## [63,] 3.0102333
## [64,] 3.6939256
## [65,] 2.5447492
## [66,] 2.6645671
## [67,] 2.9683951
## [68,] 3.0406475
## [69,] 1.3245701
## [70,] 2.8007343
## [71,] 3.2841966
## [72,] 2.1192517
## [73,] 2.7667808
## [74,] 3.5438934
## [75,] 3.6471108
## [76,] 3.7088439
## [77,] 3.2127993
## [78,] 3.3629133
## [79,] 3.4748902
## [80,] 3.1559609
## [81,] 2.1637282
## [82,] 2.9931352
## [83,] 3.5608938
## [84,] 3.3670778
## [85,] 2.6420046
## [86,] 3.9104588
## [87,] 3.0070909
## [88,] 3.0372964
## [89,] 4.1306431
## [90,] 3.2095665
## [91,] 3.3725635
## [92,] 4.0702492
## [93,] 3.8229776
## [94,] 4.7052980
## [95,] 3.6175206
## [96,] 4.1878802
```

```
## [97,]  4.0918918
##
## $residuals
##                  [,1]
##  [1,] -1.305198480
##  [2,] -0.886561920
##  [3,] -0.706208030
##  [4,] -0.746727007
##  [5,] -1.349942595
##  [6,] -0.041782996
##  [7,] -1.141337122
##  [8,] -1.273035810
##  [9,] -0.120476684
## [10,] -0.259043503
## [11,] -0.182056003
## [12,]  0.437325653
## [13,] -0.852569279
## [14,] -0.571646890
## [15,] -0.659150931
## [16,] -0.490886516
## [17,]  0.173458070
## [18,] -0.965559088
## [19,]  0.270181218
## [20,] -0.135478348
## [21,] -0.367958705
## [22,] -1.054973183
## [23,]  0.638822834
## [24,] -0.843041318
## [25,] -0.063871290
## [26,] -0.286165662
## [27,] -0.152952682
## [28,]  0.193749858
## [29,] -0.234641701
## [30,] -0.398687863
## [31,] -0.060148239
## [32,] -0.867009935
## [33,]  0.051348218
## [34,]  0.653057393
## [35,]  0.952755370
## [36,] -0.684507903
## [37,]  0.232269856
## [38,]  1.049580300
## [39,] -1.733147829
## [40,]  0.414133656
## [41,]  0.248132796
## [42,]  0.094282397
## [43,]  0.276949085
## [44,] -0.016717571
## [45,]  0.087342839
## [46,] -0.032752717
## [47,] -1.512237448
## [48,] -0.007740796
## [49,] -0.116577443
## [50,]  0.456412900
```

```
## [51,]   0.326745017
## [52,] -0.273088165
## [53,]   0.532058379
## [54,] -0.297967802
## [55,] -0.591594097
## [56,] -0.101459301
## [57,]   1.034653832
## [58,]   0.534542461
## [59,]   0.694812905
## [60,]   0.185891239
## [61,]   0.615313405
## [62,] -0.676727354
## [63,] -0.156643258
## [64,] -0.811925558
## [65,]   0.337250761
## [66,]   0.223022915
## [67,] -0.047925122
## [68,] -0.077957525
## [69,]   1.638119900
## [70,]   0.172245699
## [71,] -0.271116586
## [72,]   0.918098326
## [73,]   0.289579209
## [74,] -0.468883429
## [75,] -0.371850841
## [76,] -0.371293914
## [77,]   0.180030656
## [78,]   0.072686671
## [79,] -0.017000212
## [80,]   0.357079070
## [81,]   1.352281797
## [82,]   0.537624831
## [83,]   0.004406231
## [84,]   0.203862155
## [85,]   0.945675352
## [86,] -0.279468843
## [87,]   0.672999128
## [88,]   0.675053596
## [89,] -0.146303147
## [90,]   0.784033542
## [91,]   0.657246540
## [92,]   0.059300838
## [93,]   0.562172376
## [94,] -0.020858043
## [95,]   1.525599382
## [96,]   1.289629827
## [97,]   1.491038202
##
## $n
## [1] 97
##
## $q
## [1] 9
```

```r
#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
```

```r
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
```

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

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

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

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

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

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

```r
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

```r
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")
```
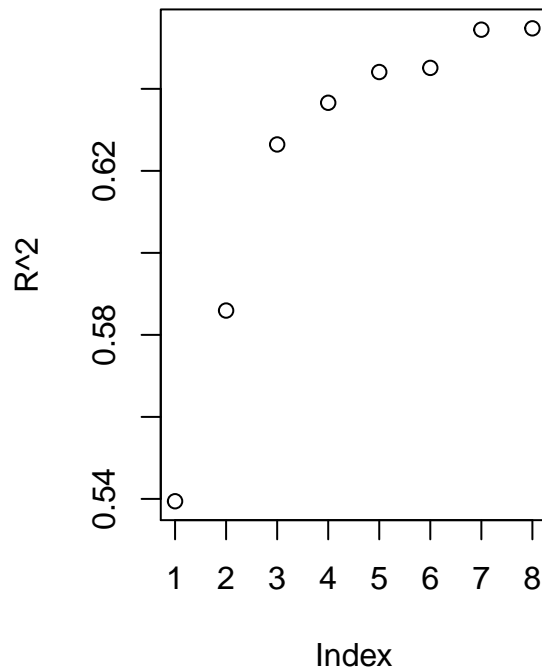
| Coefficient of determination | RSE |
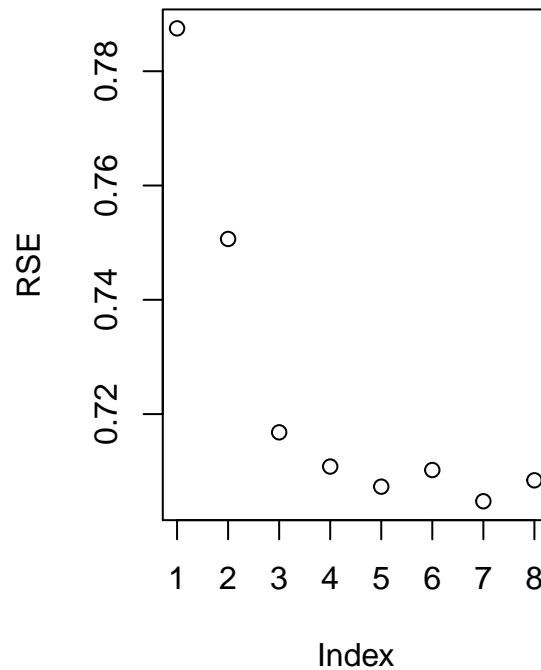
Comment: R^2 strictly goes up, as we add variables. $R^2 = \frac{Regss}{TSS} = \frac{\sum (\hat{y_i} - \bar{y})^2}{\sum (y_i - \bar{y})^2}$. Since RSS strictly goes down as we add variables, we can say R^2 should go up always.

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

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

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

Part a

Scatterplot matrix

```r
plot(data2)
```
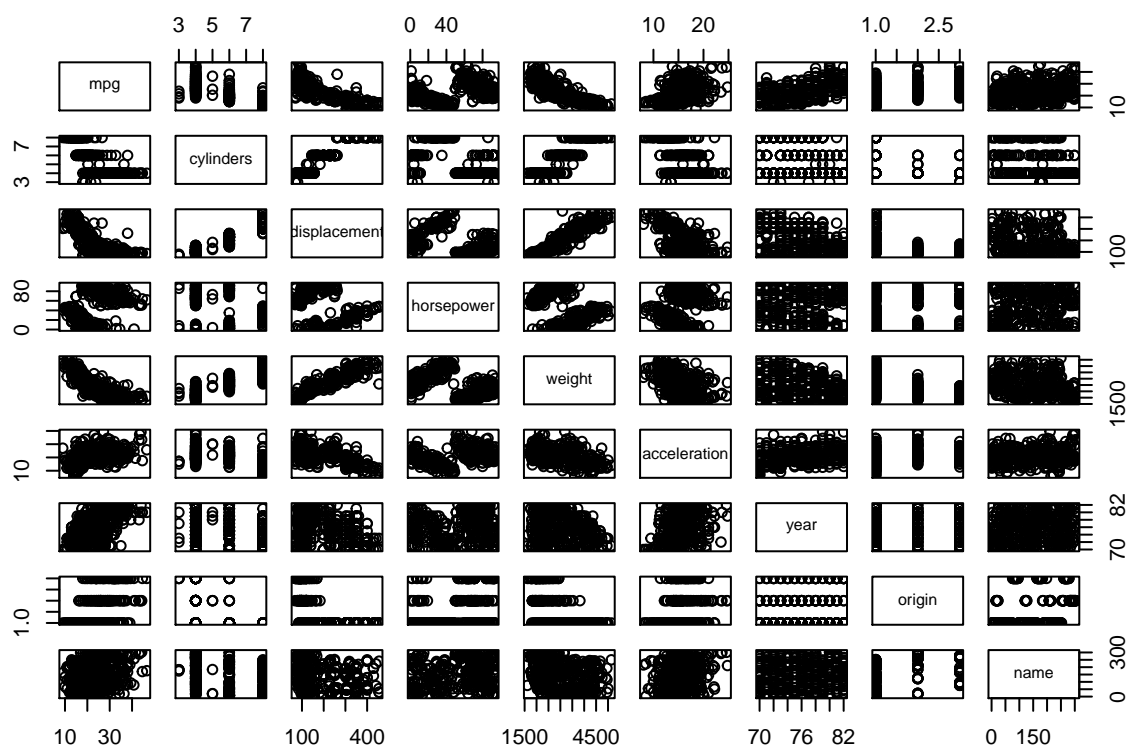

Part b

```r
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 -0.7762599   -0.8044430  0.4228227 -0.8317389
## cylinders    -0.7762599  1.0000000    0.9509199 -0.5466585  0.8970169
## displacement -0.8044430  0.9509199    1.0000000 -0.4820705  0.9331044
## horsepower    0.4228227 -0.5466585   -0.4820705  1.0000000 -0.4821507
## weight       -0.8317389  0.8970169    0.9331044 -0.4821507  1.0000000
## acceleration  0.4222974 -0.5040606   -0.5441618  0.2662877 -0.4195023
## year          0.5814695 -0.3467172   -0.3698041  0.1274167 -0.3079004
## origin        0.5636979 -0.5649716   -0.6106643  0.2973734 -0.5812652
##              acceleration       year     origin
## mpg             0.4222974  0.5814695  0.5636979
## cylinders      -0.5040606 -0.3467172 -0.5649716
```

74

```
## displacement  -0.5441618 -0.3698041 -0.6106643
## horsepower      0.2662877  0.1274167  0.2973734
## weight         -0.4195023 -0.3079004 -0.5812652
## acceleration    1.0000000  0.2829009  0.2100836
## year            0.2829009  1.0000000  0.1843141
## origin          0.2100836  0.1843141  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.

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

```
##
## Call:
## lm(formula = mpg ~ ., data = cor_dat)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -9.629 -2.034 -0.046  1.801 13.010
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -2.128e+01  4.259e+00  -4.998 8.78e-07 ***
## cylinders    -2.927e-01  3.382e-01  -0.865   0.3874
## displacement  1.603e-02  7.284e-03   2.201   0.0283 *
## horsepower    7.942e-03  6.809e-03   1.166   0.2442
## weight       -6.870e-03  5.799e-04 -11.846   < 2e-16 ***
## acceleration  1.539e-01  7.750e-02   1.986   0.0477 *
## year          7.734e-01  4.939e-02  15.661   < 2e-16 ***
## origin        1.346e+00  2.691e-01   5.004 8.52e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.331 on 389 degrees of freedom
## Multiple R-squared:  0.822,  Adjusted R-squared:  0.8188
## F-statistic: 256.7 on 7 and 389 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.

- **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, origin shows quite high significance, but again, since origin should be considered as dummy variable, this might not be correct.

- **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.

Part d

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

```
plot(lms)
```


Residuals vs Fitted

Fitted values
lm(mpg ~ .)

Normal Q–Q

Standardized residuals

Theoretical Quantiles
lm(mpg ~ .)

Scale–Location

√|Standardized residuals|

Fitted values
lm(mpg ~ .)

Residuals vs Leverage
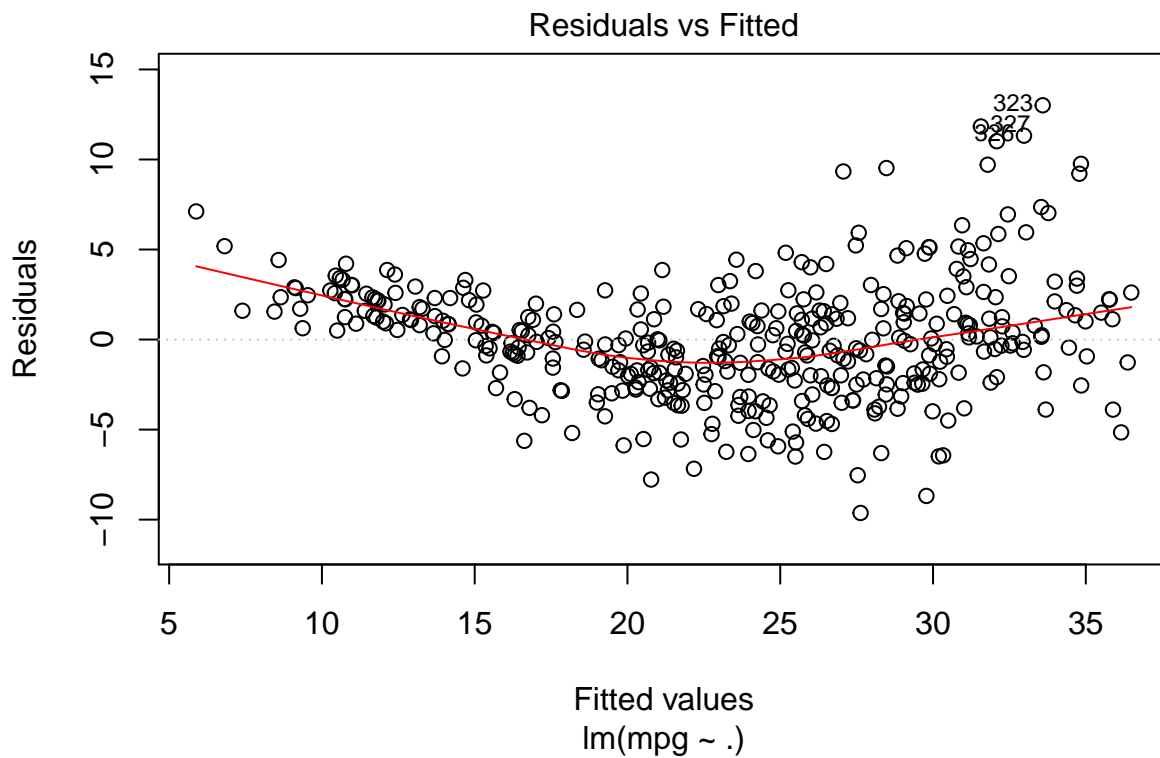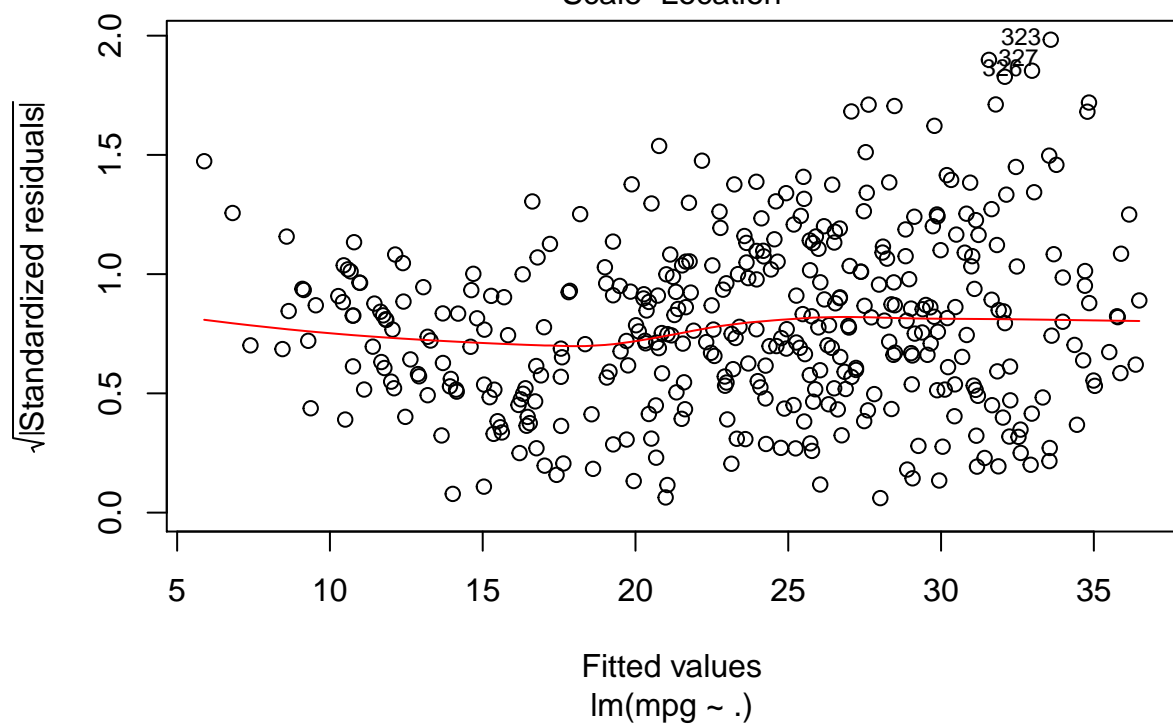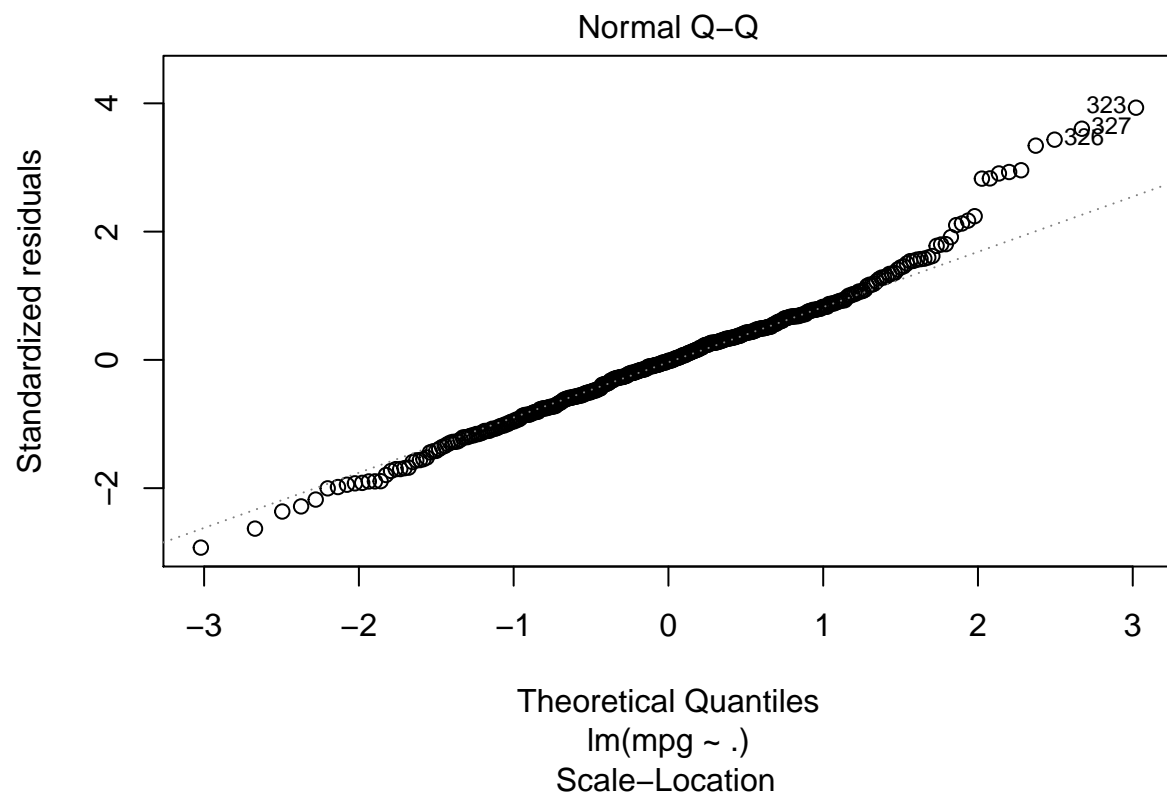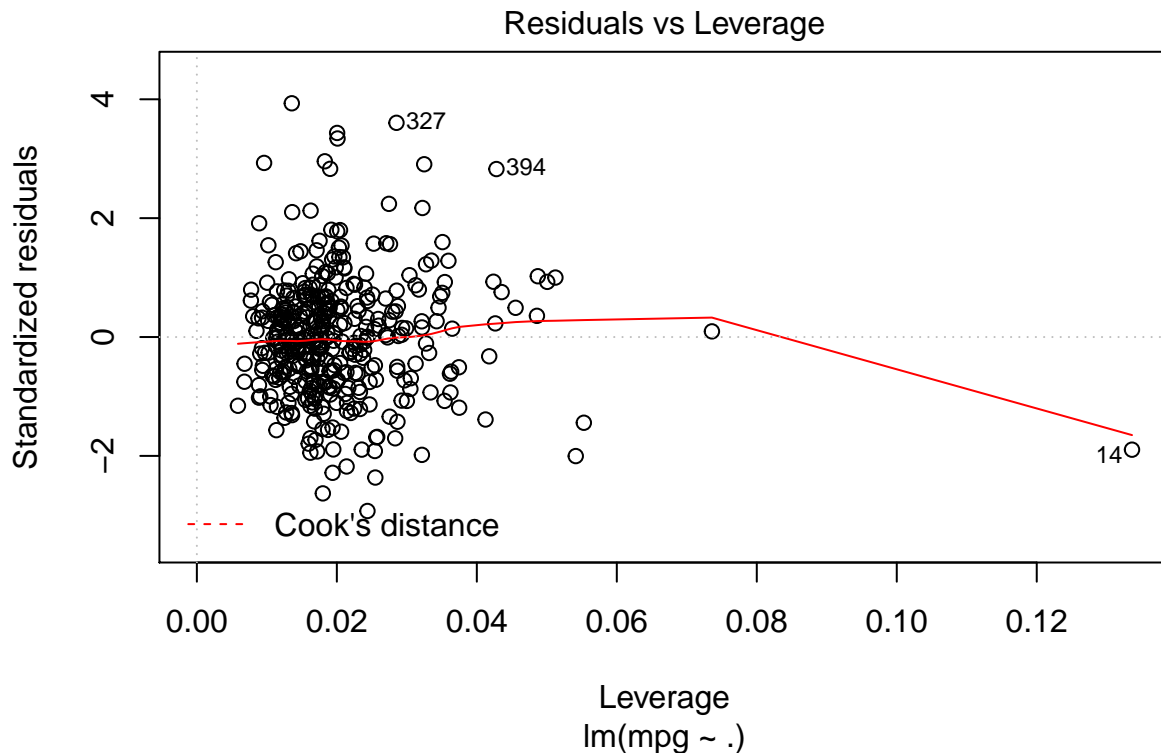
Comment:

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). 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 unusual high leverage.

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 ~ weight*year, data = cor_dat)
summary(lms2)
```

```
##
## Call:
## lm(formula = mpg ~ weight * year, data = cor_dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.0341 -1.9851 -0.0912  1.6987 12.9292
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.124e+02  1.280e+01  -8.781  < 2e-16 ***
## weight       2.821e-02  4.376e-03   6.447 3.34e-10 ***
```

78

```
## year         2.068e+00  1.699e-01  12.171  < 2e-16 ***
## weight:year -4.672e-04  5.857e-05  -7.977 1.66e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.187 on 393 degrees of freedom
## Multiple R-squared:  0.8354, Adjusted R-squared:  0.8341
## F-statistic: 664.9 on 3 and 393 DF,  p-value: < 2.2e-16
```

Comment: Yes! the interaction definitely appears to be statistically significant, so we might need to investigate further. Statisticians always investigate interaction effect first, and if there is some statistical significance, there is no use to investigate single effect.

Part f

```
#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
## -9.7837 -1.8387 -0.0159  1.6038 12.8664
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -85.7854    17.4463  -4.917  1.3e-06 ***
## cylinders      2.1669     1.6926   1.280   0.2012
## displacement  -1.1947     1.5626  -0.765   0.4450
## horsepower     0.3619     0.1547   2.340   0.0198 *
## weight       -18.6398     1.7788 -10.479  < 2e-16 ***
## acceleration   0.4547     1.0968   0.415   0.6787
## year          59.4030     3.4845  17.048  < 2e-16 ***
## origin         1.2770     0.5100   2.504   0.0127 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.121 on 389 degrees of freedom
## Multiple R-squared:  0.8437, Adjusted R-squared:  0.8409
## F-statistic: 300.1 on 7 and 389 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
## -9.6183 -1.8942 -0.0425  1.7479 13.0007
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -65.51455    8.12220  -8.066 9.09e-15 ***
## cylinders      0.96626    1.58377   0.610 0.542152
## displacement   0.09233    0.23005   0.401 0.688388
## horsepower     0.16523    0.07307   2.261 0.024289 *
## weight        -0.72686    0.06555 -11.088  < 2e-16 ***
## acceleration   0.63037    0.59181   1.065 0.287469
## year          13.51140    0.82740  16.330  < 2e-16 ***
## origin         2.82648    0.75116   3.763 0.000194 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.215 on 389 degrees of freedom
## Multiple R-squared:  0.8342, Adjusted R-squared:  0.8312
## F-statistic: 279.6 on 7 and 389 DF,  p-value: < 2.2e-16
```

```
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
## -9.4889 -2.3161  0.0161  1.9375 12.8428
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.272e+00  2.321e+00   0.548 0.584008
## cylinders    -9.429e-02  2.625e-02  -3.593 0.000369 ***
## displacement  5.339e-05  1.154e-05   4.626 5.08e-06 ***
## horsepower   -9.195e-05  7.635e-05  -1.204 0.229195
## weight       -9.674e-07  8.660e-08 -11.170  < 2e-16 ***
## acceleration  7.249e-03  2.455e-03   2.952 0.003348 **
## year          5.051e-03  3.466e-04  14.572  < 2e-16 ***
## origin        4.012e-01  6.770e-02   5.926 6.85e-09 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.545 on 389 degrees of freedom
## Multiple R-squared:  0.7984, Adjusted R-squared:  0.7948
## F-statistic: 220.1 on 7 and 389 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
## -0.42022 -0.06147 -0.00070   0.06967   0.40336
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.841770   0.652065  -1.291   0.1975
## cylinders    -0.068805   0.063263  -1.088   0.2774
## displacement  0.004520   0.058401   0.077   0.9383
## horsepower    0.006771   0.005782   1.171   0.2423
## weight       -0.827905   0.066485 -12.453   <2e-16 ***
## acceleration  0.048446   0.040993   1.182   0.2380
## year          2.414434   0.130235  18.539   <2e-16 ***
## origin        0.032852   0.019060   1.724   0.0856 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1167 on 389 degrees of freedom
## Multiple R-squared:  0.8844, Adjusted R-squared:  0.8823
## F-statistic: 425.2 on 7 and 389 DF,  p-value: < 2.2e-16
```

```r
root_dat2 <- sqrt(cor_dat)

lms_root2 <- lm(mpg ~., data = root_dat2)
summary(lms_root2)
```
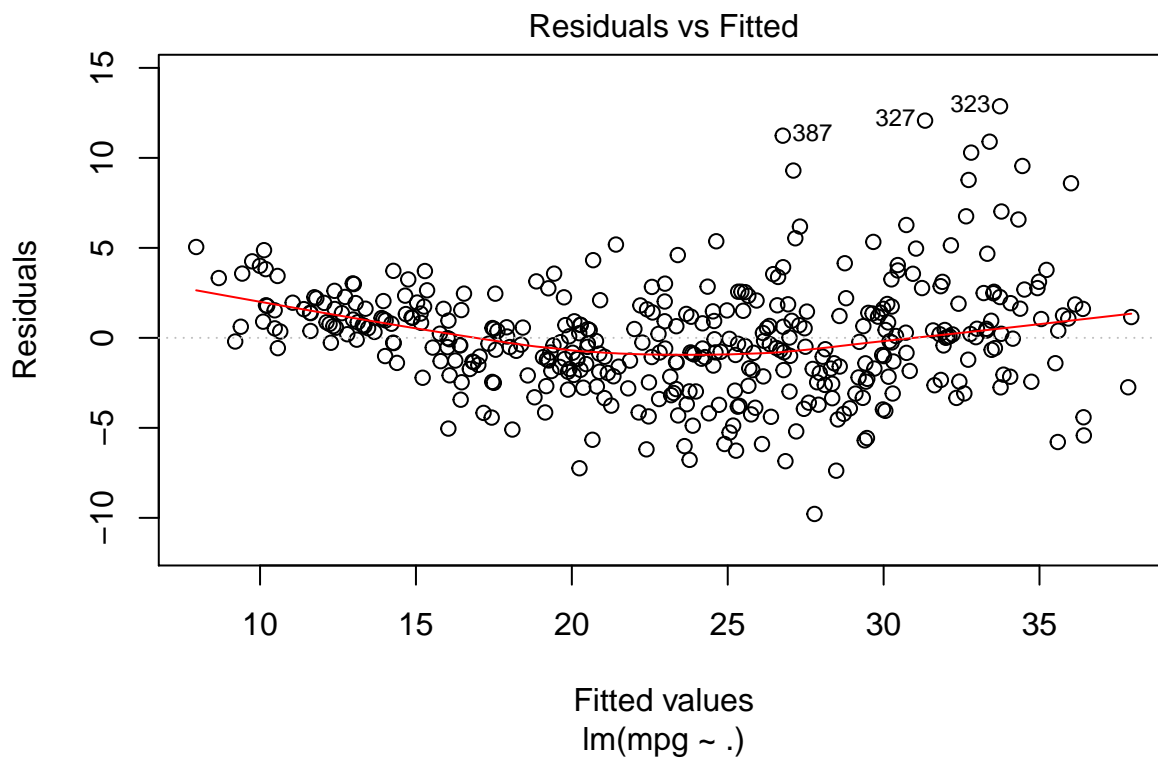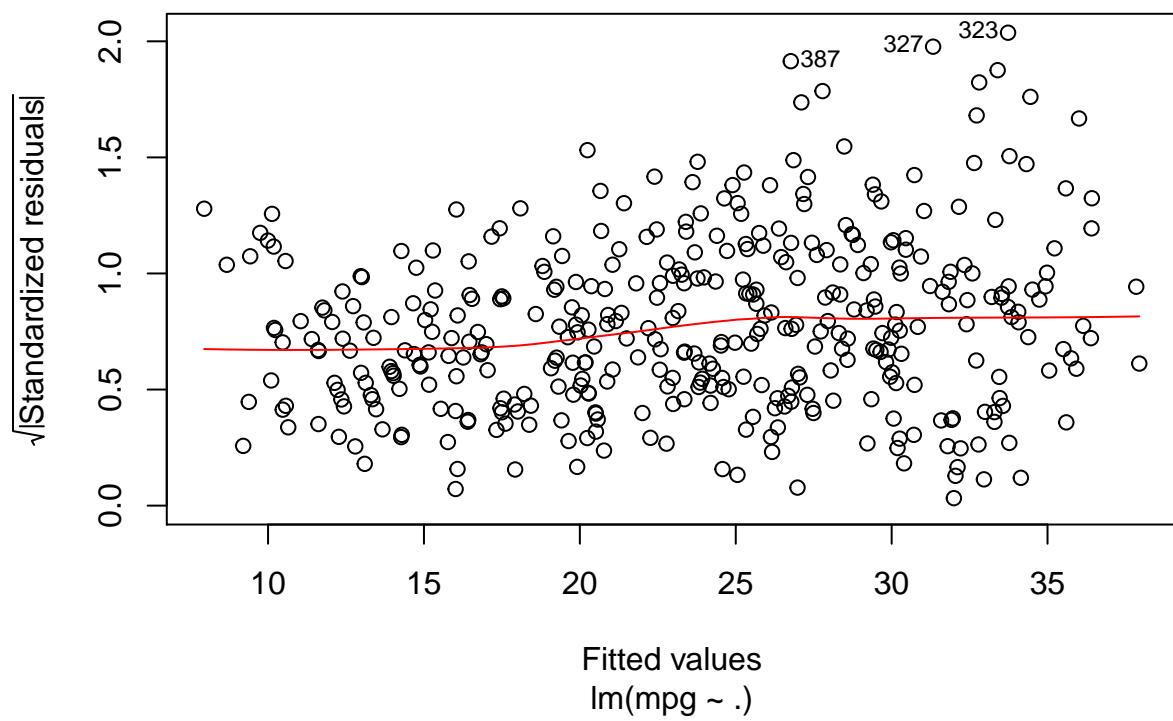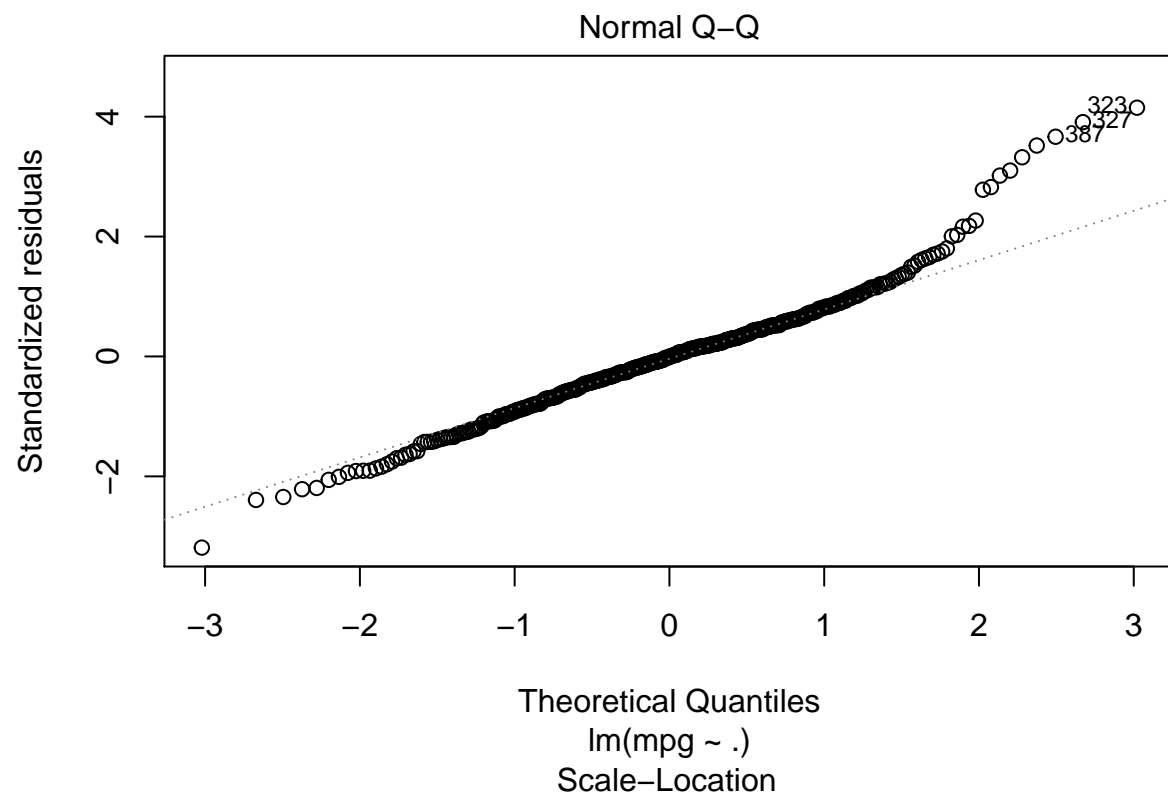
```
##
## Call:
## lm(formula = mpg ~ ., data = root_dat2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.9952 -0.1731  0.0059  0.1659  1.0188
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -3.635703   0.752859  -4.829 1.97e-06 ***
## cylinders    -0.001738   0.146802  -0.012  0.99056
## displacement  0.006314   0.021323   0.296  0.76732
```

```
## horsepower     0.014071    0.006773    2.078  0.03839 *
## weight        -0.073829    0.006076 -12.151  < 2e-16 ***
## acceleration   0.063608    0.054856    1.160  0.24694
## year           1.342996    0.076693   17.511  < 2e-16 ***
## origin         0.223733    0.069627    3.213  0.00142 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.298 on 389 degrees of freedom
## Multiple R-squared:  0.8651, Adjusted R-squared:  0.8627
## F-statistic: 356.3 on 7 and 389 DF,  p-value: < 2.2e-16
```
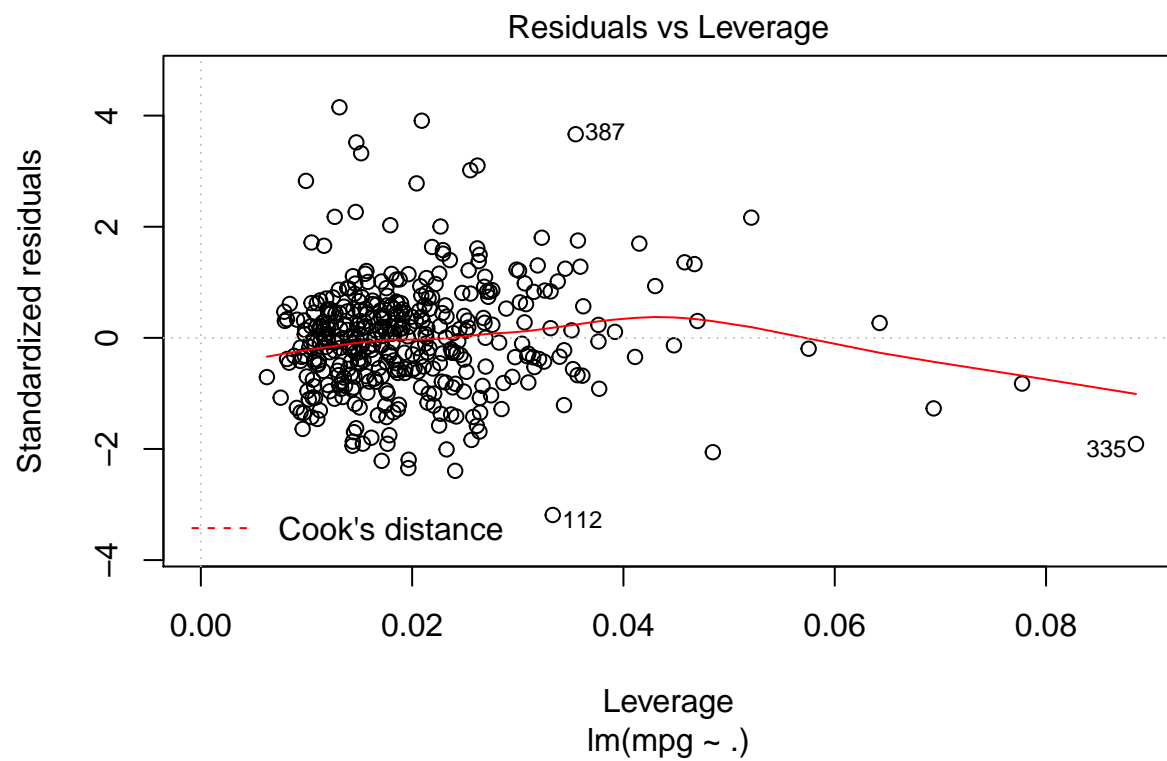
```r
square_dat2 <- as.data.frame((cor_dat)^2)

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

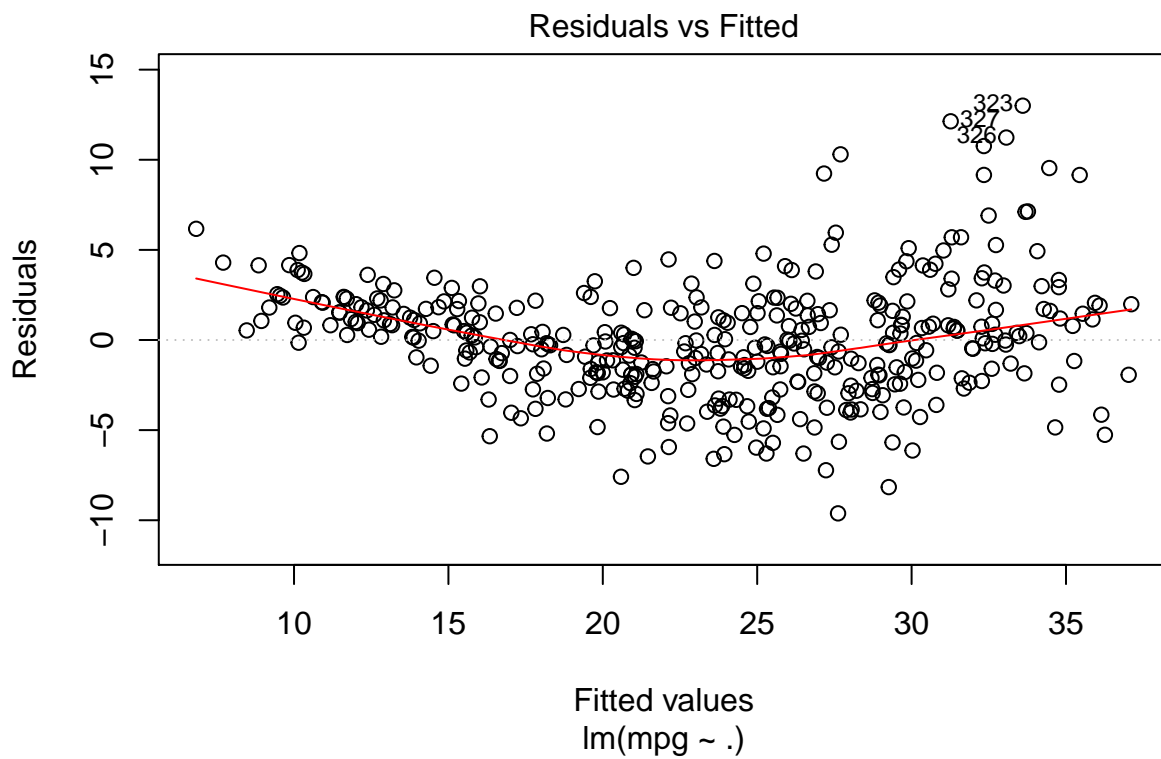```
##
Call:
lm(formula = mpg ~ ., data = square_dat2)
##
Residuals:
Min   1Q Median   3Q   Max
-492.8 -146.7 -18.8 113.9 1021.8
##
Coefficients:
Estimate Std. Error t value Pr(>|t|)
(Intercept) -6.935e+02 1.435e+02 -4.832 1.95e-06 ***
cylinders   -4.659e+00 1.623e+00 -2.870 0.00432 **
displacement 3.554e-03 7.137e-04   4.980 9.57e-07 ***
horsepower   -8.332e-03 4.722e-03 -1.765 0.07841 .
weight   -4.935e-05 5.356e-06 -9.214 < 2e-16 ***
acceleration 4.973e-01 1.519e-01   3.275 0.00115 **
year   2.735e-01 2.144e-02 12.760 < 2e-16 ***
origin   2.592e+01 4.187e+00   6.191 1.52e-09 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
Residual standard error: 219.3 on 389 degrees of freedom
Multiple R-squared: 0.7075, Adjusted R-squared: 0.7022
F-statistic: 134.4 on 7 and 389 DF, p-value: < 2.2e-16
```

```r
#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
```

```
## -0.40961 -0.06879  0.00542  0.06653  0.35192
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    1.481e+00  1.532e-01   9.668  < 2e-16 ***
## cylinders     -1.831e-02  1.217e-02  -1.505 0.133264
## displacement   3.545e-04  2.621e-04   1.353 0.176940
## horsepower     2.909e-04  2.450e-04   1.187 0.235835
## weight        -2.870e-04  2.087e-05 -13.756  < 2e-16 ***
## acceleration   5.116e-03  2.788e-03   1.835 0.067298 .
## year           3.101e-02  1.777e-03  17.450  < 2e-16 ***
## origin         3.361e-02  9.681e-03   3.472 0.000574 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1198 on 389 degrees of freedom
## Multiple R-squared:  0.878,  Adjusted R-squared:  0.8758
## F-statistic: 399.9 on 7 and 389 DF,  p-value: < 2.2e-16
```

```r
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
## -0.99151 -0.17759  0.00876  0.17884  1.02296
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    5.574e-01  3.937e-01   1.416   0.1576
## cylinders     -3.735e-02  3.126e-02  -1.195   0.2329
## displacement   1.230e-03  6.734e-04   1.827   0.0685 .
## horsepower     7.925e-04  6.295e-04   1.259   0.2088
## weight        -6.922e-04  5.361e-05 -12.912  < 2e-16 ***
## acceleration   1.367e-02  7.165e-03   1.908   0.0571 .
## year           7.660e-02  4.566e-03  16.777  < 2e-16 ***
## origin         1.096e-01  2.487e-02   4.407 1.36e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3079 on 389 degrees of freedom
## Multiple R-squared:  0.8559, Adjusted R-squared:  0.8533
## F-statistic: 330.2 on 7 and 389 DF,  p-value: < 2.2e-16
```

```r
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
## -492.47 -142.41  -23.37  102.68 1037.98
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -2.009e+03  2.692e+02  -7.464 5.55e-13 ***
## cylinders    -6.479e+00  2.138e+01  -0.303  0.76199
## displacement  1.232e+00  4.604e-01   2.676  0.00777 **
## horsepower    3.245e-01  4.304e-01   0.754  0.45128
## weight       -3.640e-01  3.666e-02  -9.929  < 2e-16 ***
## acceleration  1.086e+01  4.899e+00   2.216  0.02724 *
## year          4.169e+01  3.122e+00  13.355  < 2e-16 ***
## origin        9.370e+01  1.701e+01   5.510 6.55e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 210.5 on 389 degrees of freedom
## Multiple R-squared:  0.7303, Adjusted R-squared:  0.7254
## F-statistic: 150.5 on 7 and 389 DF,  p-value: < 2.2e-16
```

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


Residuals vs Fitted
lm(mpg ~ .)

Normal Q-Q

Standardized residuals (y-axis)

Theoretical Quantiles

lm(mpg ~ .)

Scale-Location

√|Standardized residuals| (y-axis)

Fitted values
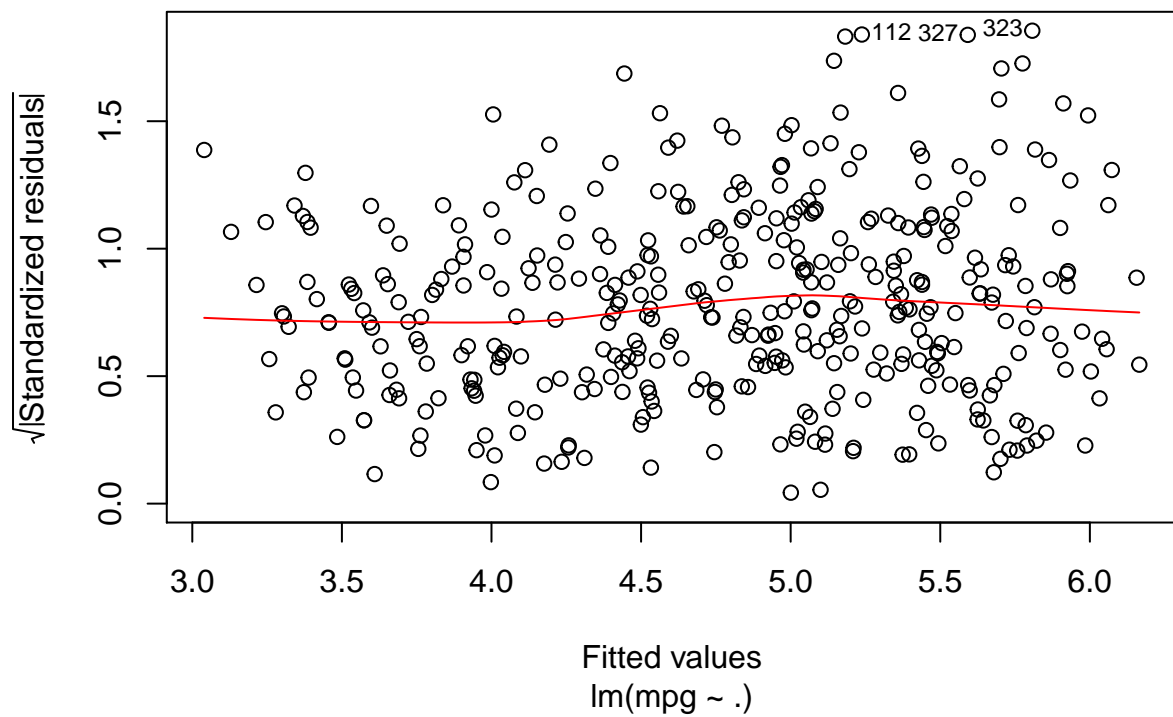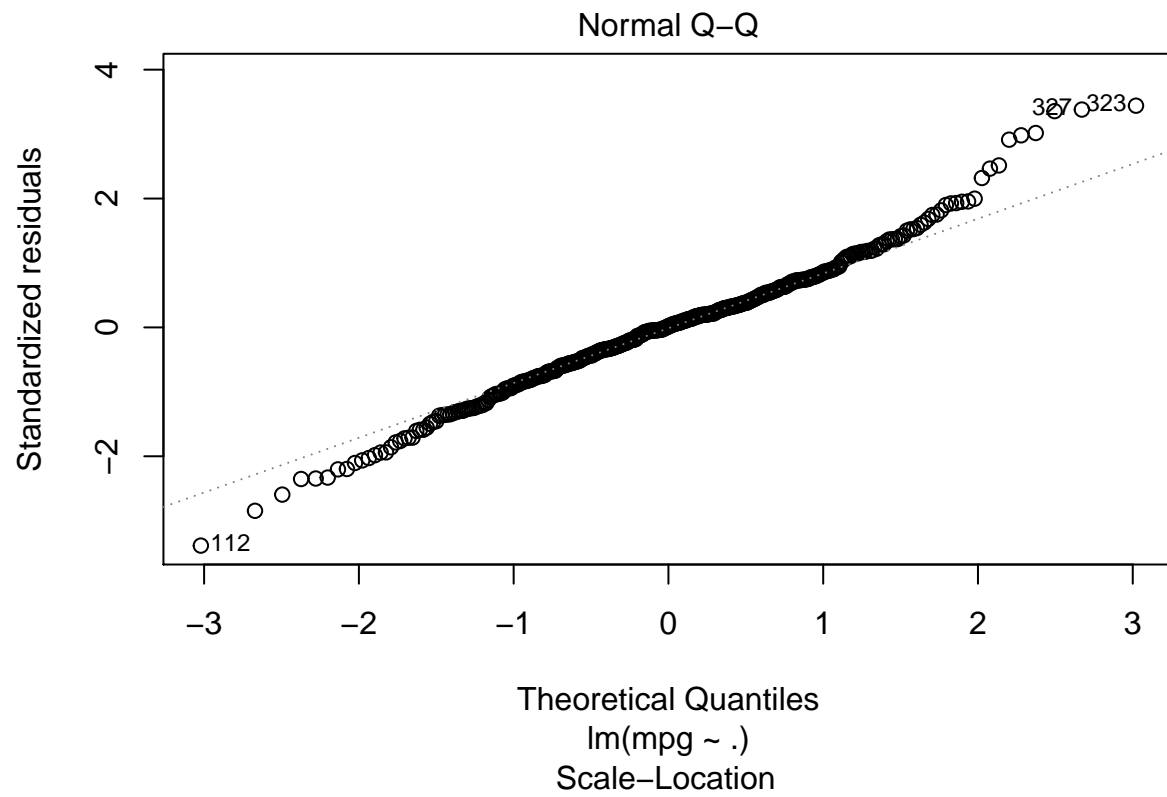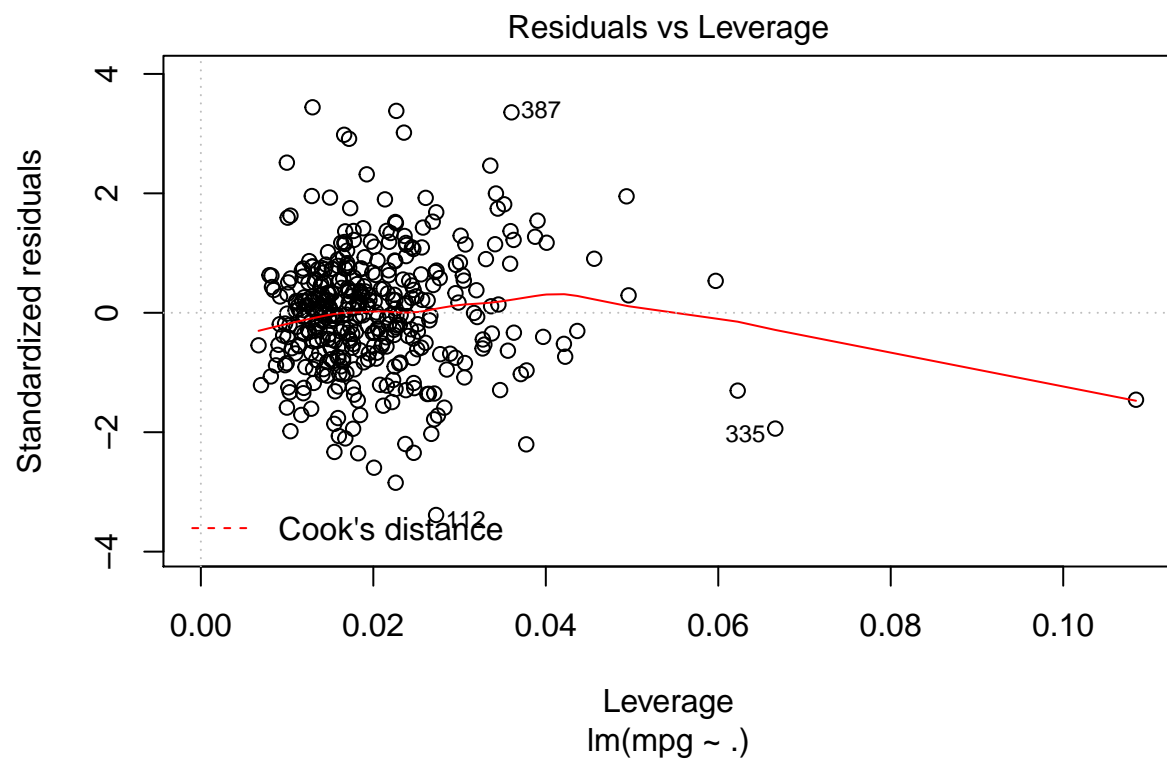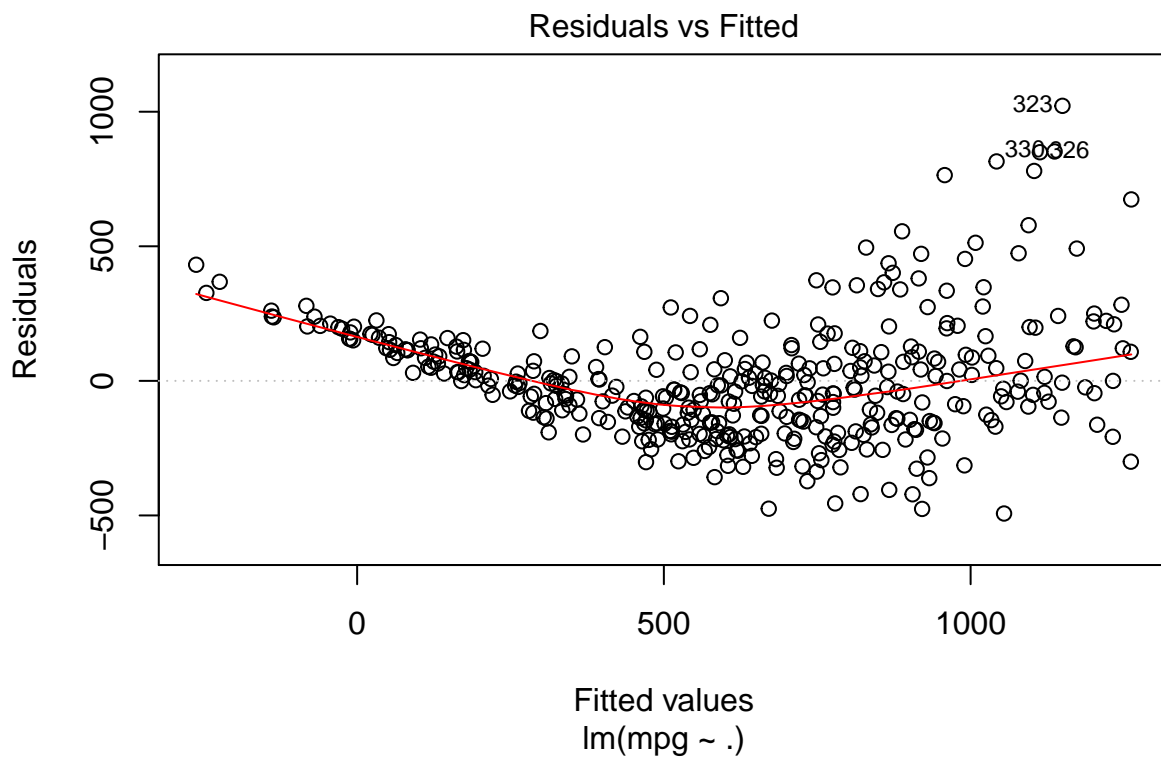
lm(mpg ~ .)

Residuals vs Leverage

Leverage
lm(mpg ~ .)

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

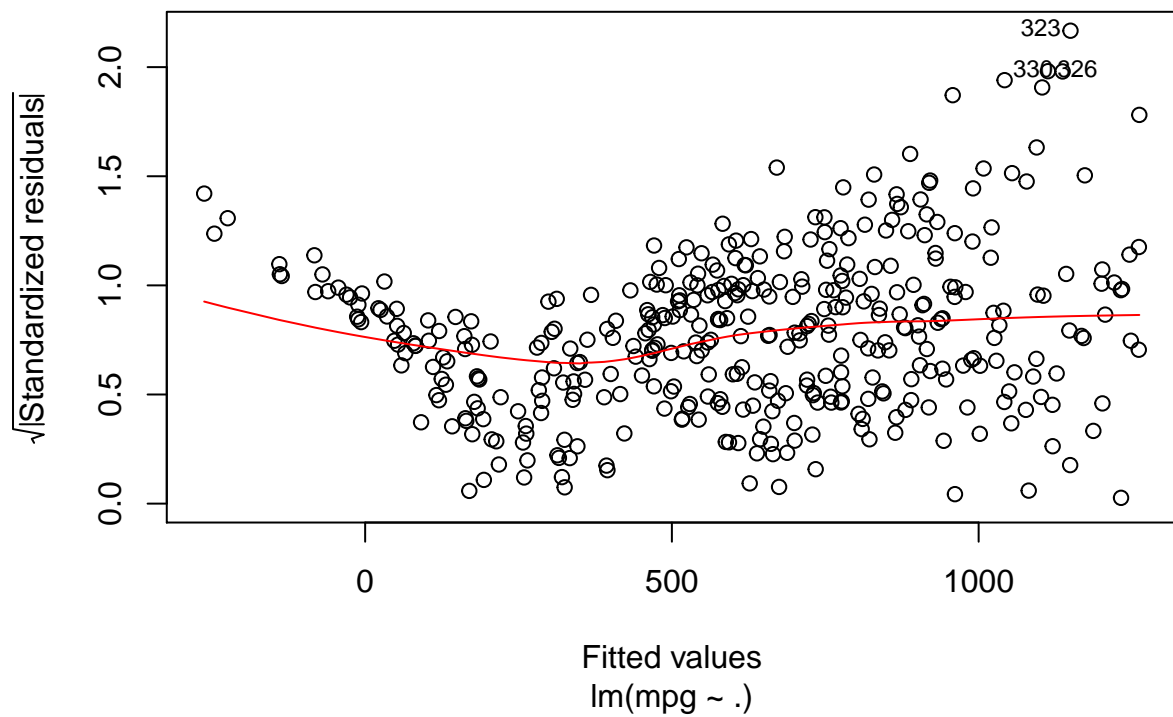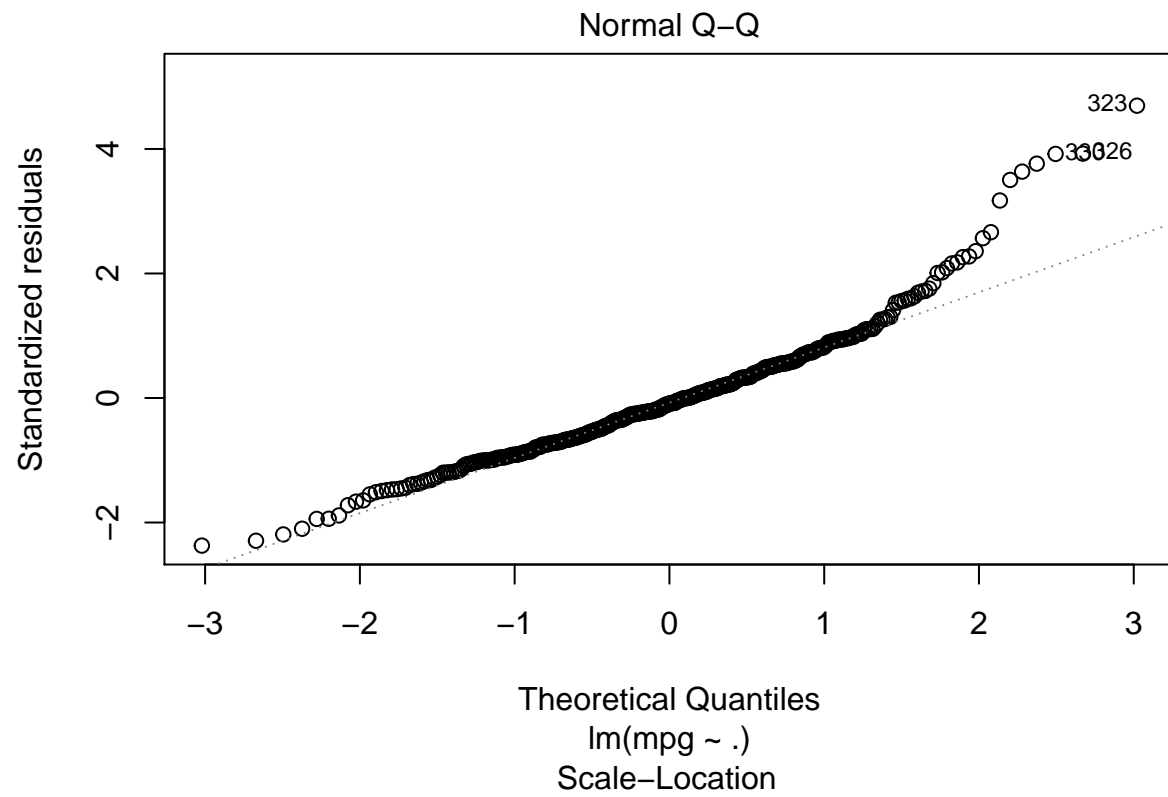## Residuals vs Fitted

Fitted values
lm(mpg ~ .)

Normal Q–Q

Standardized residuals

Theoretical Quantiles
lm(mpg ~ .)

Scale–Location

√|Standardized residuals|

Fitted values
lm(mpg ~ .)

Residuals vs Leverage

Leverage
lm(mpg ~ .)
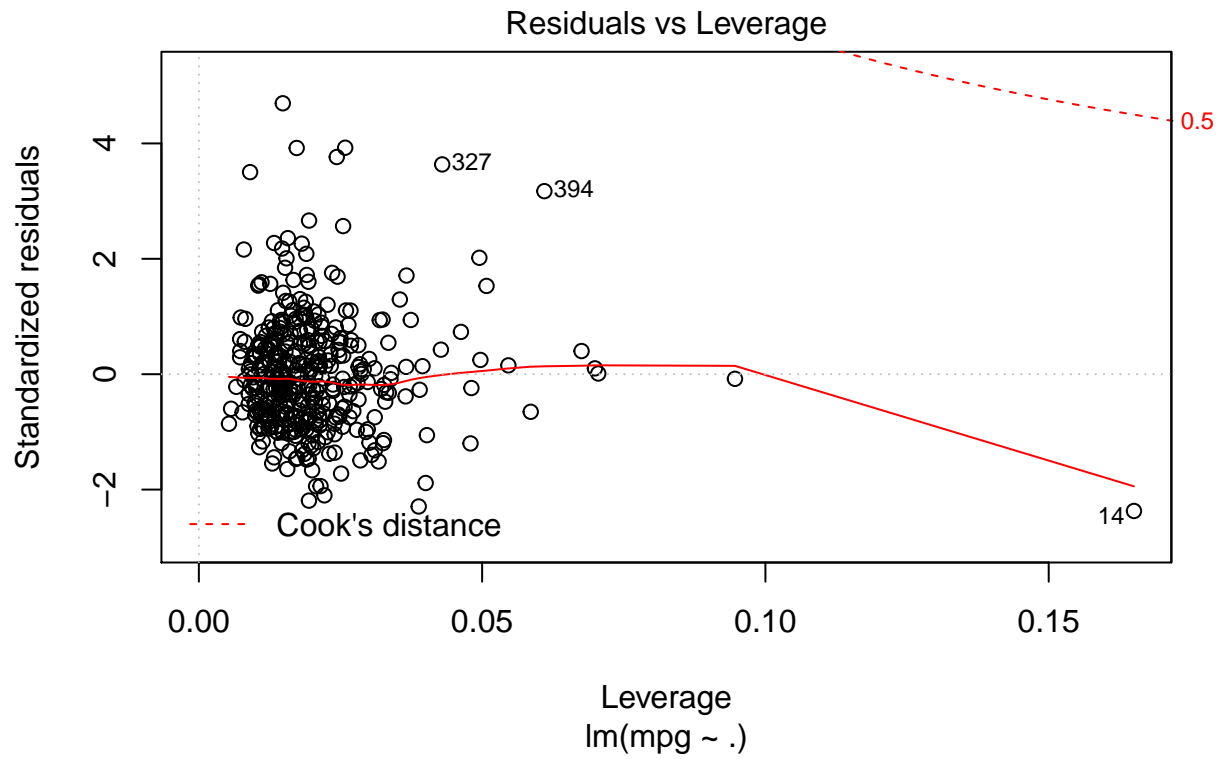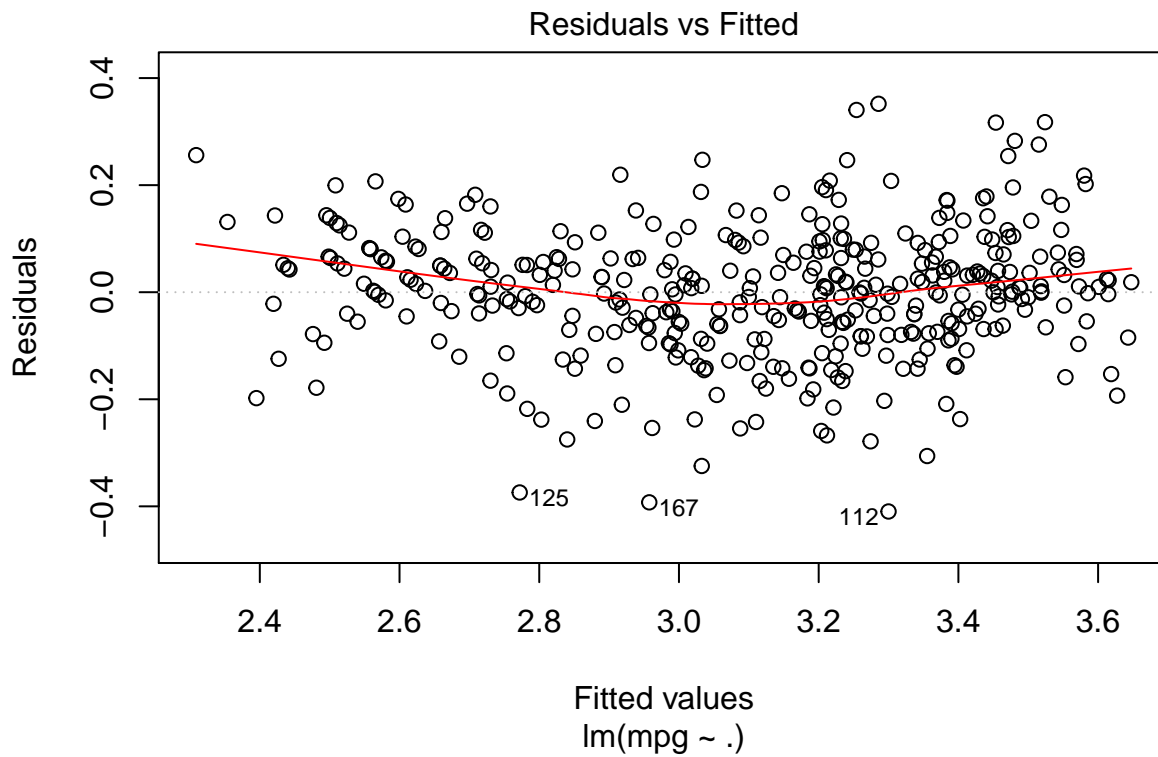
```
#Diagnostic plot for square transformation on x only
plot(lms_square)
```

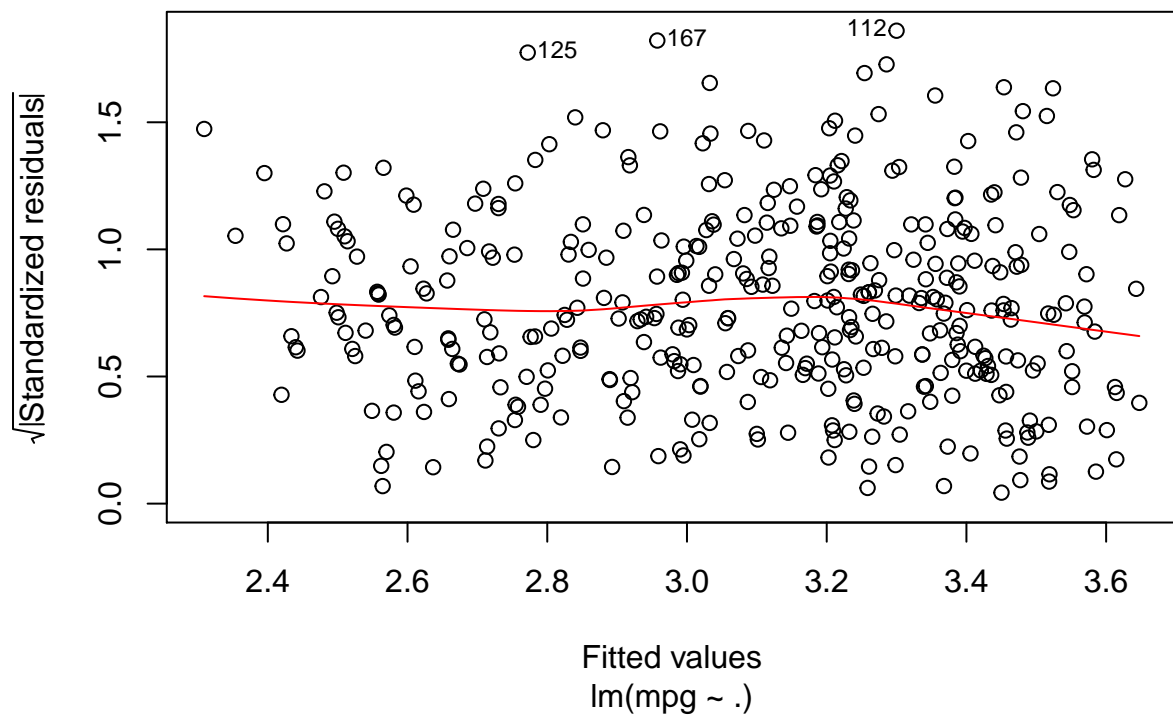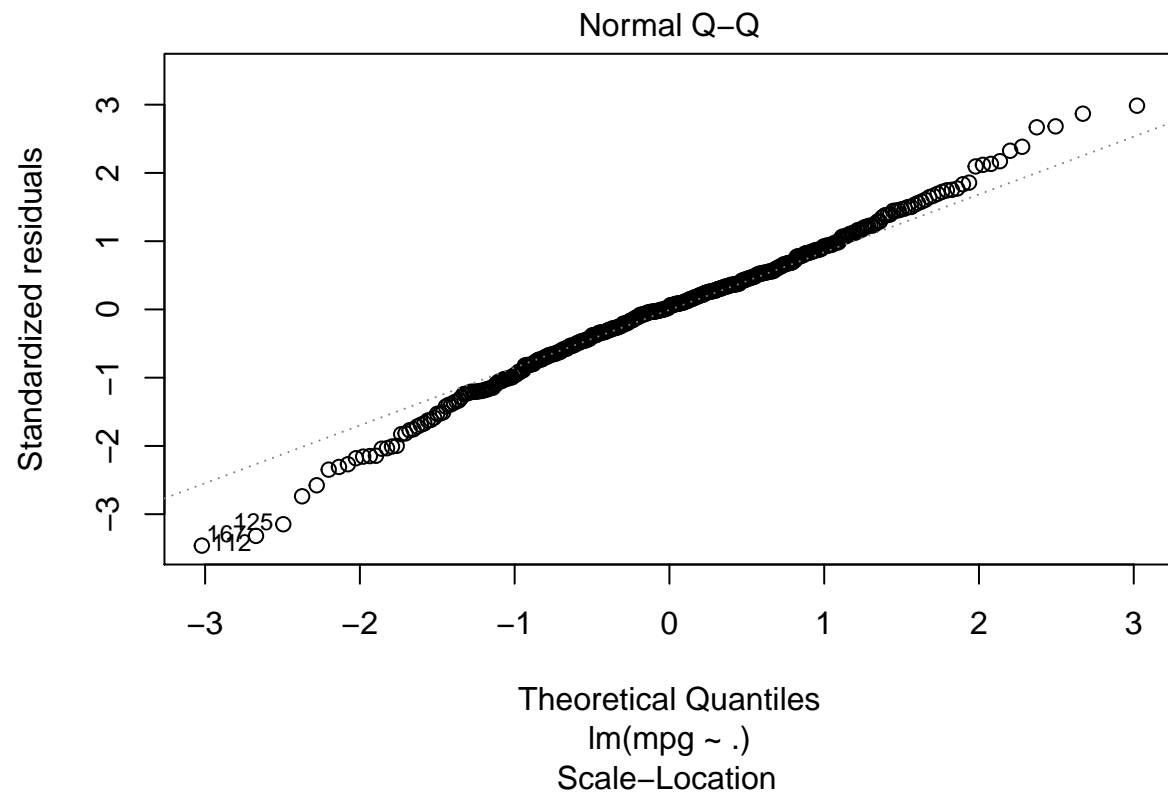## Residuals vs Fitted

Fitted values
lm(mpg ~ .)

Normal Q-Q

lm(mpg ~ .)

Scale-Location

lm(mpg ~ .)

Residuals vs Leverage

Leverage
lm(mpg ~ .)

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

Residuals vs Fitted

Fitted values
lm(mpg ~ .)

Normal Q–Q

387○

○167
○112

Standardized residuals

Theoretical Quantiles
lm(mpg ~ .)

Scale–Location

387○ 112○

○167

√|Standardized residuals|

Fitted values
lm(mpg ~ .)

Residuals vs Leverage

Leverage
lm(mpg ~ .)

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

Residuals vs Fitted

Fitted values
lm(mpg ~ .)

Normal Q–Q

lm(mpg ~ .)

Scale–Location

lm(mpg ~ .)

Residuals vs Leverage

Leverage
lm(mpg ~ .)

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

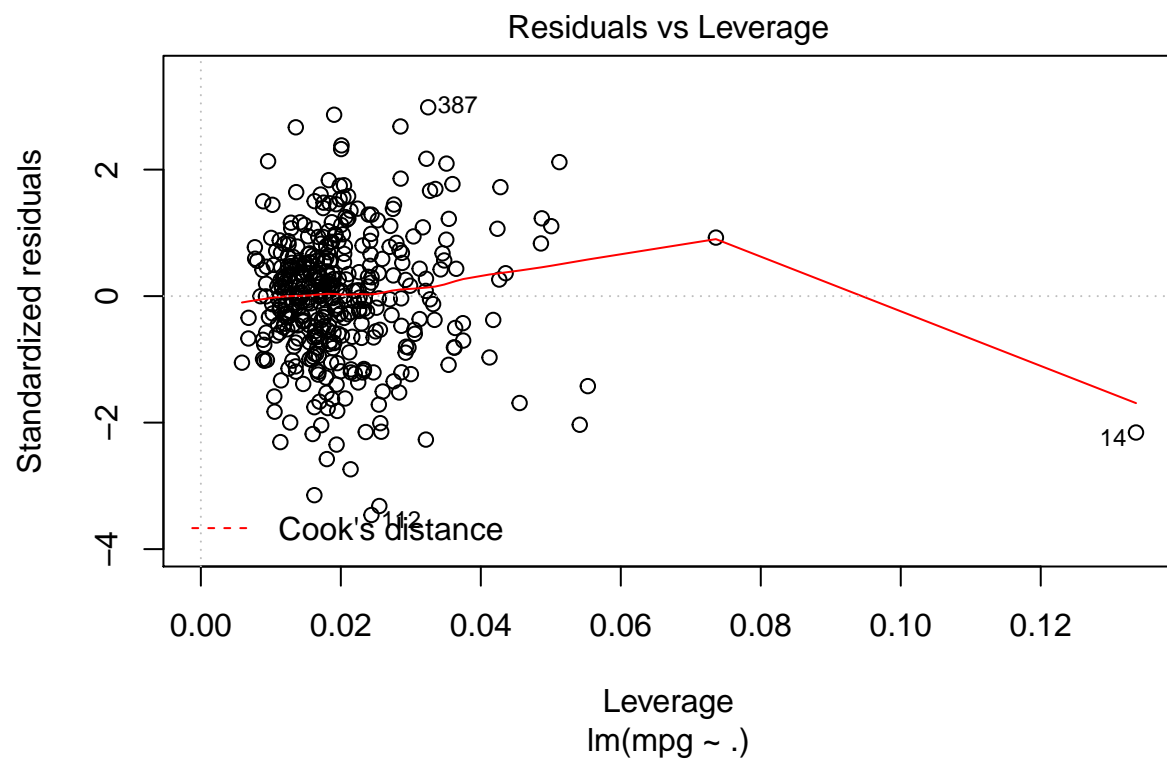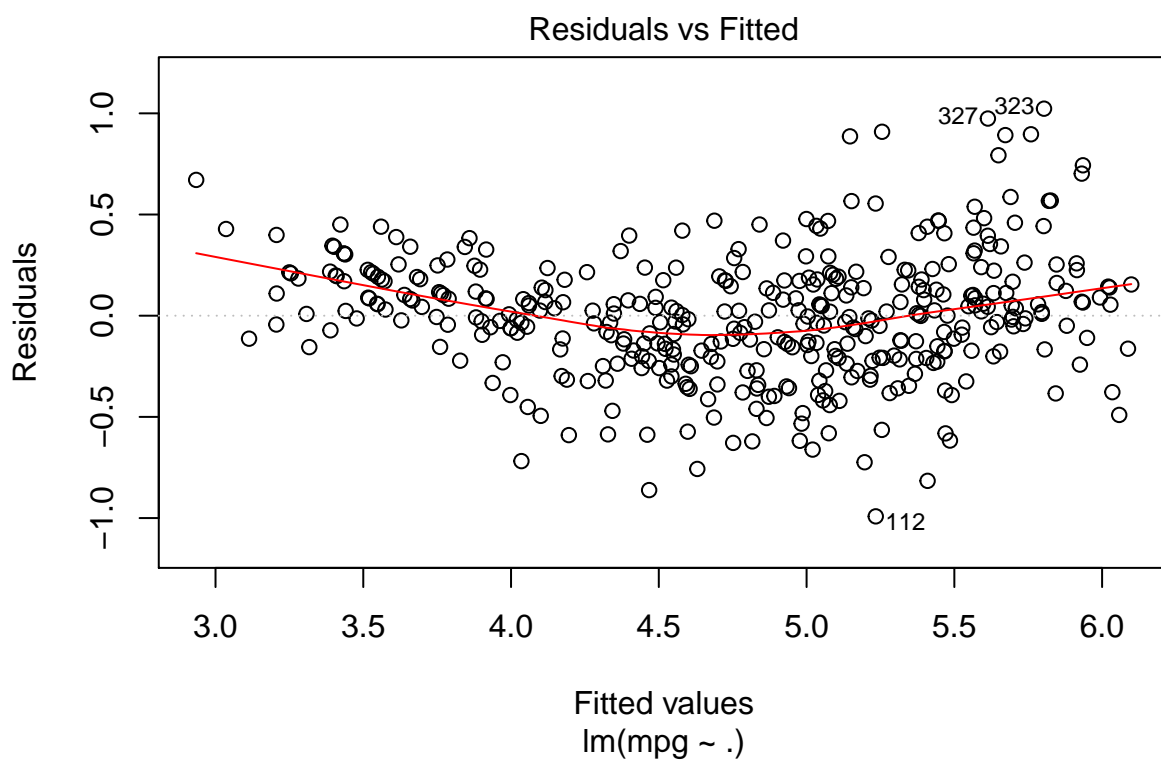## Residuals vs Fitted

Fitted values
lm(mpg ~ .)

Normal Q–Q

Standardized residuals

Theoretical Quantiles
lm(mpg ~ .)

Scale–Location

√|Standardized residuals|

Fitted values
lm(mpg ~ .)

Residuals vs Leverage

Leverage
lm(mpg ~ .)

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

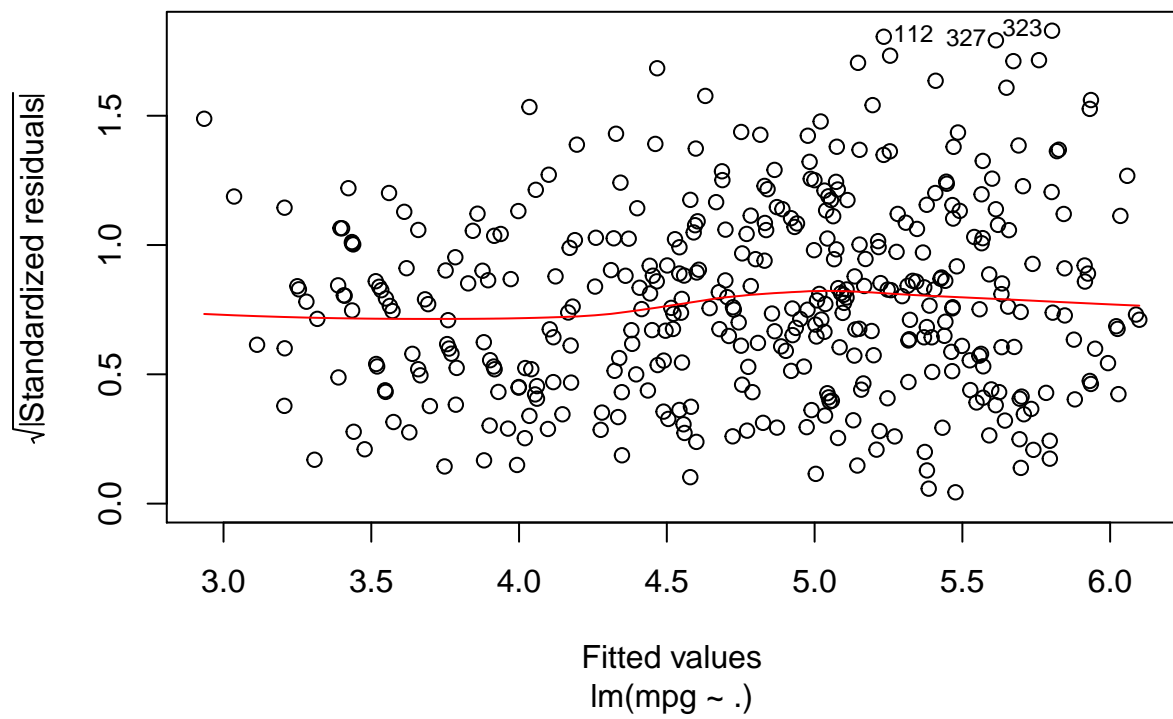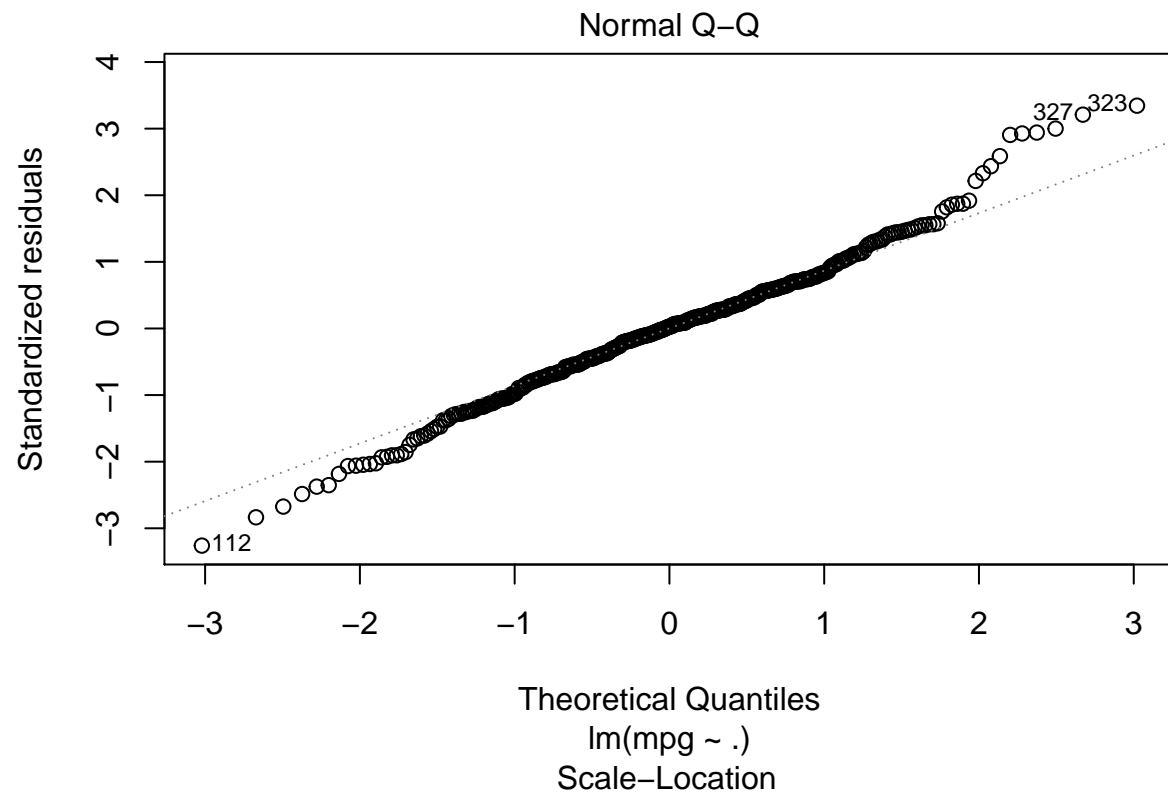## Residuals vs Fitted

Fitted values
lm(mpg ~ .)

Normal Q–Q

Standardized residuals

167 125
112

Theoretical Quantiles
lm(mpg ~ .)

Scale–Location

125   167   112

√|Standardized residuals|

Fitted values
lm(mpg ~ .)

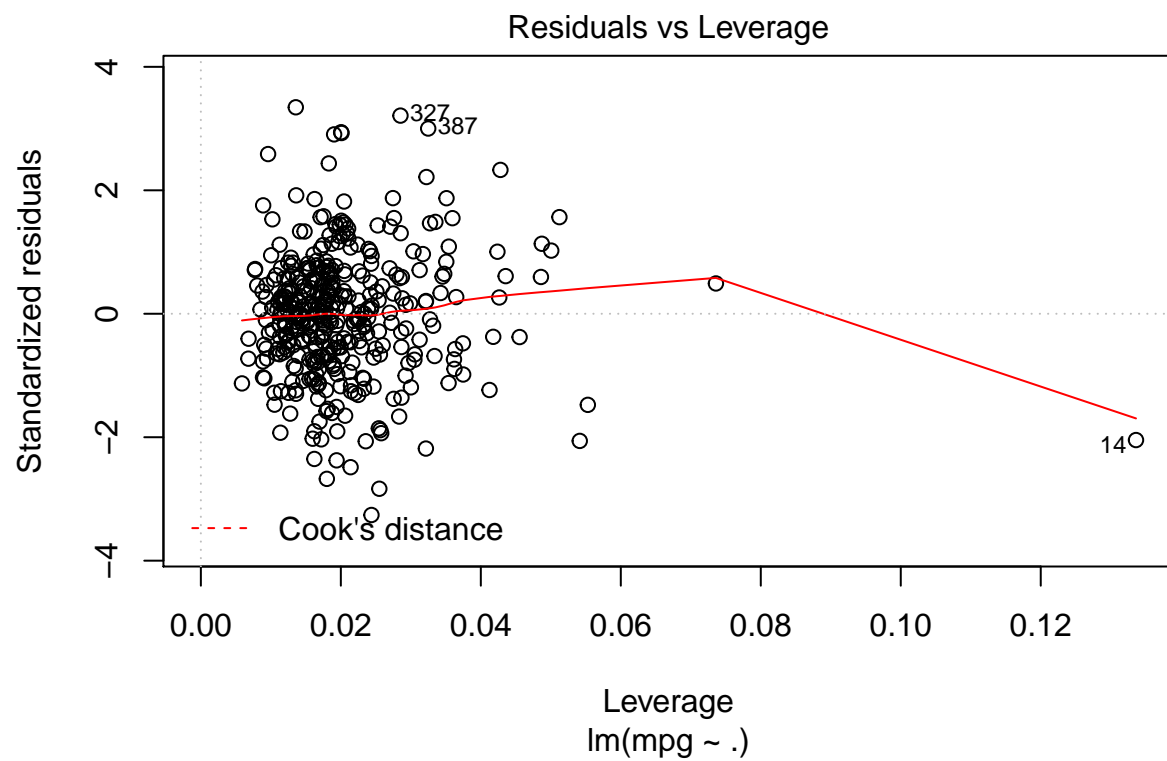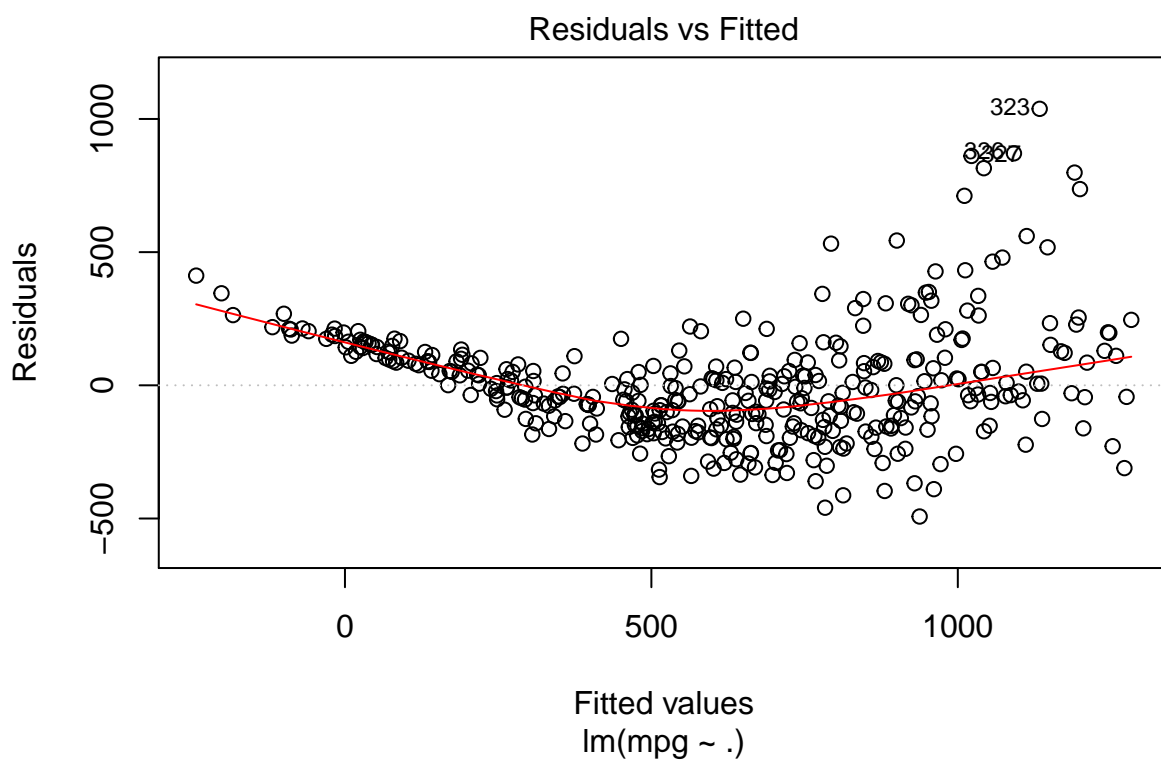97

Residuals vs Leverage

Leverage
lm(mpg ~ .)

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

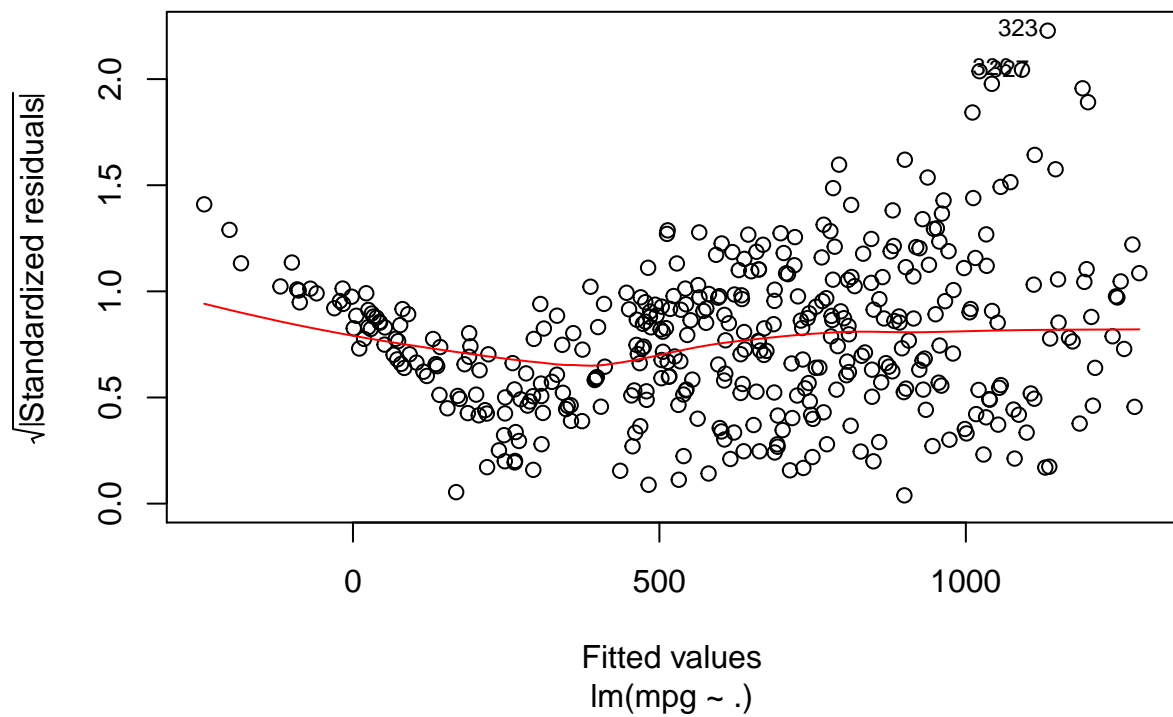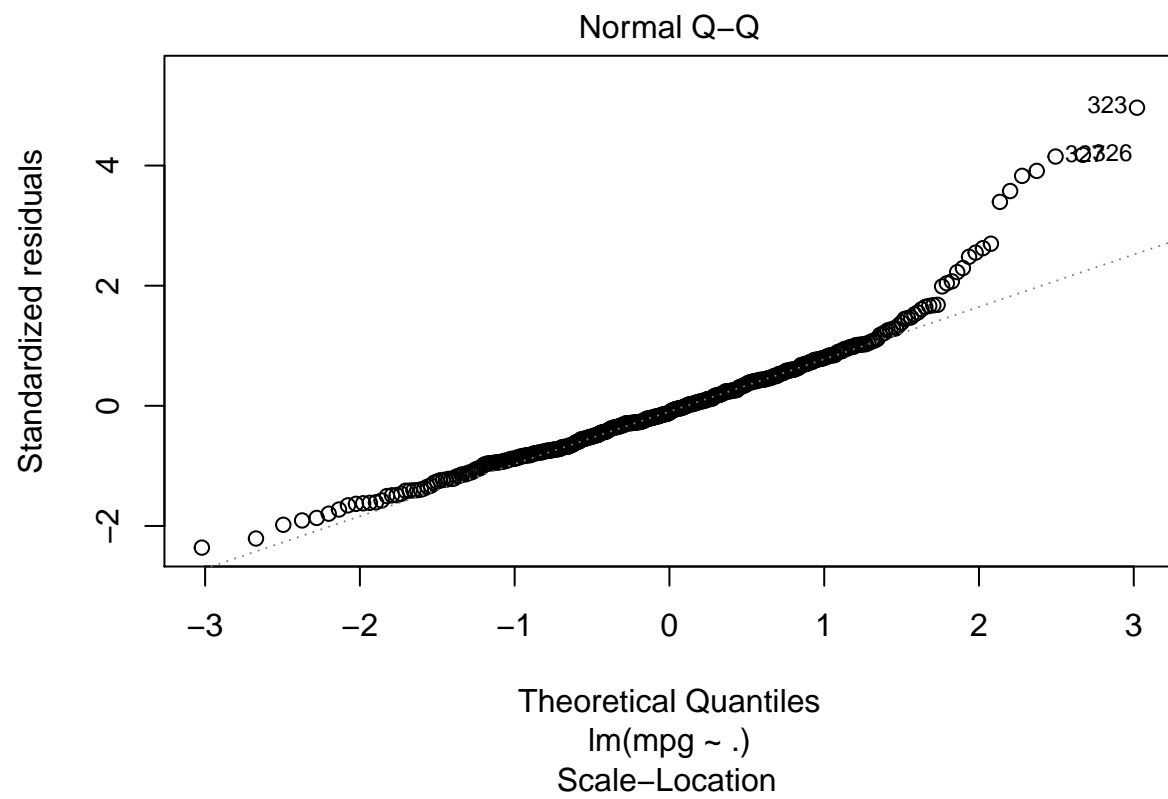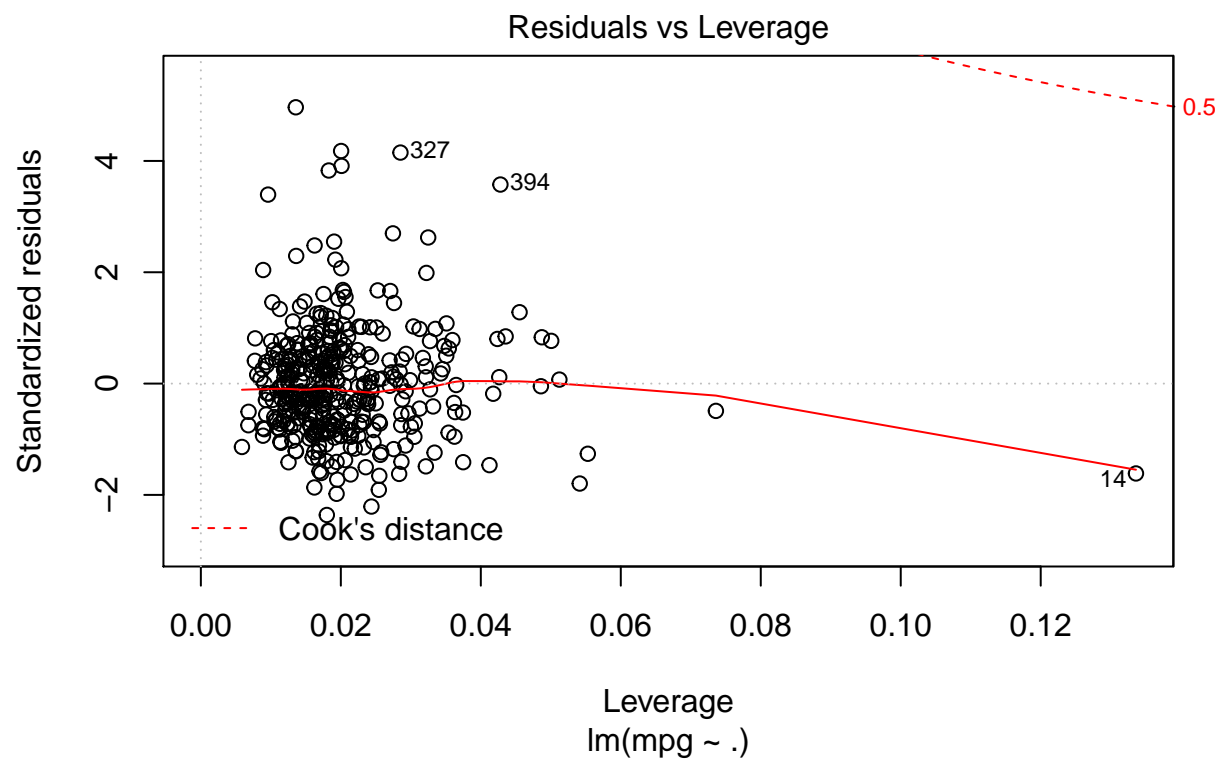## Residuals vs Fitted

Fitted values
lm(mpg ~ .)

Normal Q–Q

lm(mpg ~ .)

Scale–Location

Fitted values

lm(mpg ~ .)

Residuals vs Leverage

Standardized residuals

327
387

14

Cook's distance

Leverage
lm(mpg ~ .)

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

Residuals vs Fitted

Residuals

323
325
327

Fitted values
lm(mpg ~ .)

Normal Q-Q

Standardized residuals

Theoretical Quantiles
lm(mpg ~ .)

Scale-Location

√|Standardized residuals|

Fitted values
lm(mpg ~ .)

Residuals vs Leverage

lm(mpg ~ .)

Comment: I think log transformation makes the linear modelling better when I look at the plot of fitted values v.s. residuals.