Homework 3

PSTAT Winter 2023

Zejie Gao

Due date: March 10th, 2023 at 23:59 PT

1. This question uses the *cereal* data set available in the Homework Assignment 3 on Canvas.
   The data set *cereal* contains measurements for a set of 77 cereal brands. For this assignment only consider the following variables:

- Rating: Quality rating
- Protein: Amount of protein.
- Fat: Amount of fat.
- Fiber: Amount of fiber.
- Carbo: Amount of carbohydrates.
- Sugars: Amount of sugar.
- Potass: Amount of potassium.
- Vitamins: Amount of vitamins.
- Cups: Portion size in cups.

Our goal is to study how *rating* is related to all other 8 variables.

```
Cereal <- read.table("cereal.txt",header=T)
str(Cereal)
```

```
## 'data.frame':    77 obs. of  16 variables:
##  $ name    : chr  "100%_Bran" "100%_Natural_Bran" "All-Bran" "All-Bran_with_Extra_Fiber" ...
##  $ mfr     : chr  "N" "Q" "K" "K" ...
##  $ type    : chr  "C" "C" "C" "C" ...
##  $ calories: int  70 120 70 50 110 110 110 130 90 90 ...
##  $ protein : int  4 3 4 4 2 2 2 3 2 3 ...
##  $ fat     : int  1 5 1 0 2 2 0 2 1 0 ...
##  $ sodium  : int  130 15 260 140 200 180 125 210 200 210 ...
##  $ fiber   : num  10 2 9 14 1 1.5 1 2 4 5 ...
##  $ carbo   : num  5 8 7 8 14 10.5 11 18 15 13 ...
##  $ sugars  : int  6 8 5 0 8 10 14 8 6 5 ...
##  $ potass  : int  280 135 320 330 -1 70 30 100 125 190 ...
##  $ vitamins: int  25 0 25 25 25 25 25 25 25 25 ...
##  $ shelf   : int  3 3 3 3 3 1 2 3 1 3 ...
##  $ weight  : num  1 1 1 1 1 1 1 1.33 1 1 ...
##  $ cups    : num  0.33 1 0.33 0.5 0.75 0.75 1 0.75 0.67 0.67 ...
##  $ rating  : num  68.4 34 59.4 93.7 34.4 ...
```

```r
Cereal <- as.data.frame(Cereal)
head(Cereal)
```

```
##                         name mfr type calories protein fat sodium fiber carbo
## 1                  100%_Bran   N    C       70       4   1    130  10.0   5.0
## 2          100%_Natural_Bran   Q    C      120       3   5     15   2.0   8.0
## 3                   All-Bran   K    C       70       4   1    260   9.0   7.0
## 4 All-Bran_with_Extra_Fiber   K    C       50       4   0    140  14.0   8.0
## 5             Almond_Delight   R    C      110       2   2    200   1.0  14.0
## 6    Apple_Cinnamon_Cheerios   G    C      110       2   2    180   1.5  10.5
##   sugars potass vitamins shelf weight cups   rating
## 1      6    280       25     3      1 0.33 68.40297
## 2      8    135        0     3      1 1.00 33.98368
## 3      5    320       25     3      1 0.33 59.42551
## 4      0    330       25     3      1 0.50 93.70491
## 5      8     -1       25     3      1 0.75 34.38484
## 6     10     70       25     1      1 0.75 29.50954
```

(a) **(2 pts)** Run a multiple linear regression model after removing observations 5,21 and 58. Calculate the fitted response values and the residuals from the linear model mentioned above. Use *head* function to show the first 5 entries of the fitted response values and the first 5 entries of the residuals.

```r
Cereal_a <- Cereal[c(-5,-21, -58),c("name","rating","protein", "fat", "fiber",
                                    "carbo", "sugars", "potass", "vitamins",
                                    "cups")]
MLR_a <- lm(rating ~ protein + fat + fiber + carbo + sugars + potass + vitamins
            + cups ,data = Cereal_a); summary(MLR_a)
```

```
##
## Call:
## lm(formula = rating ~ protein + fat + fiber + carbo + sugars +
##     potass + vitamins + cups, data = Cereal_a)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.1868 -3.0482 -0.4195  1.9820  9.3381
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 58.23622    4.45056  13.085  < 2e-16 ***
## protein      2.26615    0.64782   3.498 0.000851 ***
## fat         -4.12831    0.61233  -6.742 4.94e-09 ***
## fiber        2.63691    0.66389   3.972 0.000181 ***
## carbo       -0.39997    0.19262  -2.077 0.041805 *
## sugars      -1.90439    0.16915 -11.258  < 2e-16 ***
## potass      -0.01515    0.02233  -0.678 0.500020
## vitamins    -0.09156    0.02475  -3.699 0.000448 ***
## cups         0.50040    2.60766   0.192 0.848421
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 4.352 on 65 degrees of freedom
## Multiple R-squared:  0.9144, Adjusted R-squared:  0.9038
## F-statistic: 86.77 on 8 and 65 DF,  p-value: < 2.2e-16
```

```
y_hat <- MLR_a$fitted.values; head(y_hat,5)
```

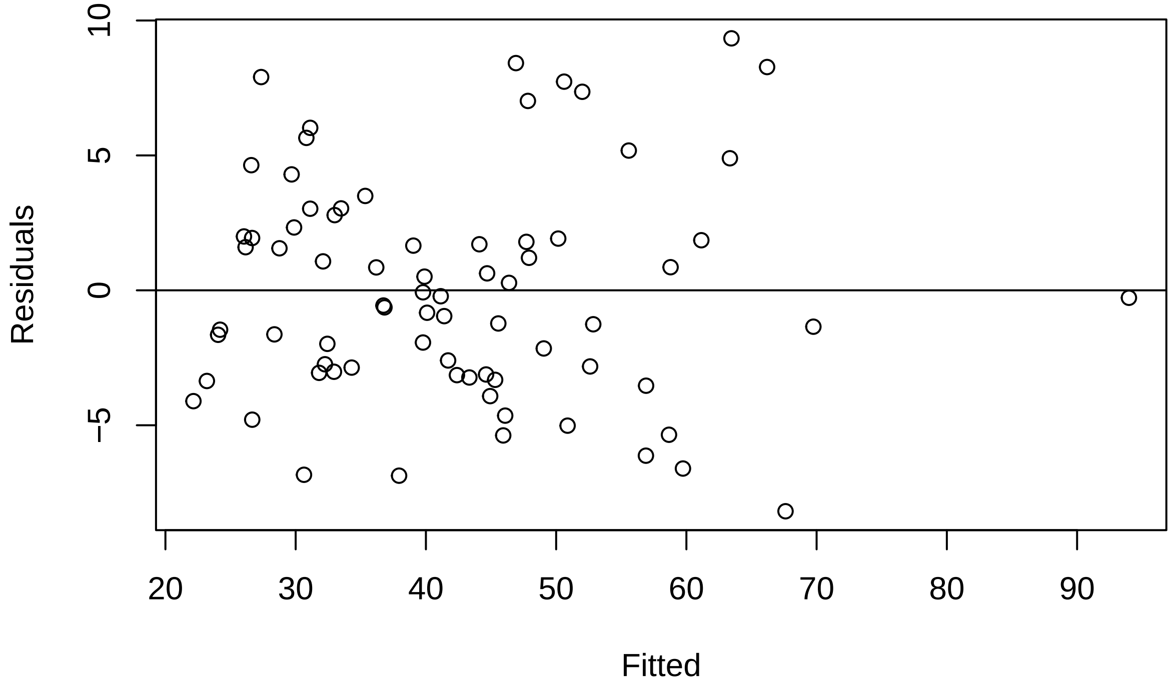
```
##        1        2        3        4        6
## 69.75066 29.68772 67.61235 93.98080 32.24978
```

```
e_hat <- MLR_a$residuals; head(e_hat,5)
```

```
##          1          2          3          4          6
## -1.3476910  4.2959597 -8.1868456 -0.2758917 -2.7402368
```

(b) **(2 pts)** Use a graphical diagnostic approach to check if the random errors have constant variance. Briefly explain what diagnostics method you used and what is your conclusion. Conclusion: To check the homoscedasticity or constant variance, I plot residuals verse fitted response value on the graph. If the random errors have constant variance, the plot should show no trend in the spread of the residuals as the predicted values increase. In this graph we could see a slight decreasing trend in the range (30,45), so the random error could have non-constant variance. To be more specific, I make use of ncvTest which suffice for providing the results of the non-constant variance test. Due to small p-value(0.049959), less than 0.05, we successfully reject the null hypothesis which random error have constant error.

```
plot(y_hat,e_hat,xlab='Fitted',ylab='Residuals')
abline(h=0)
```

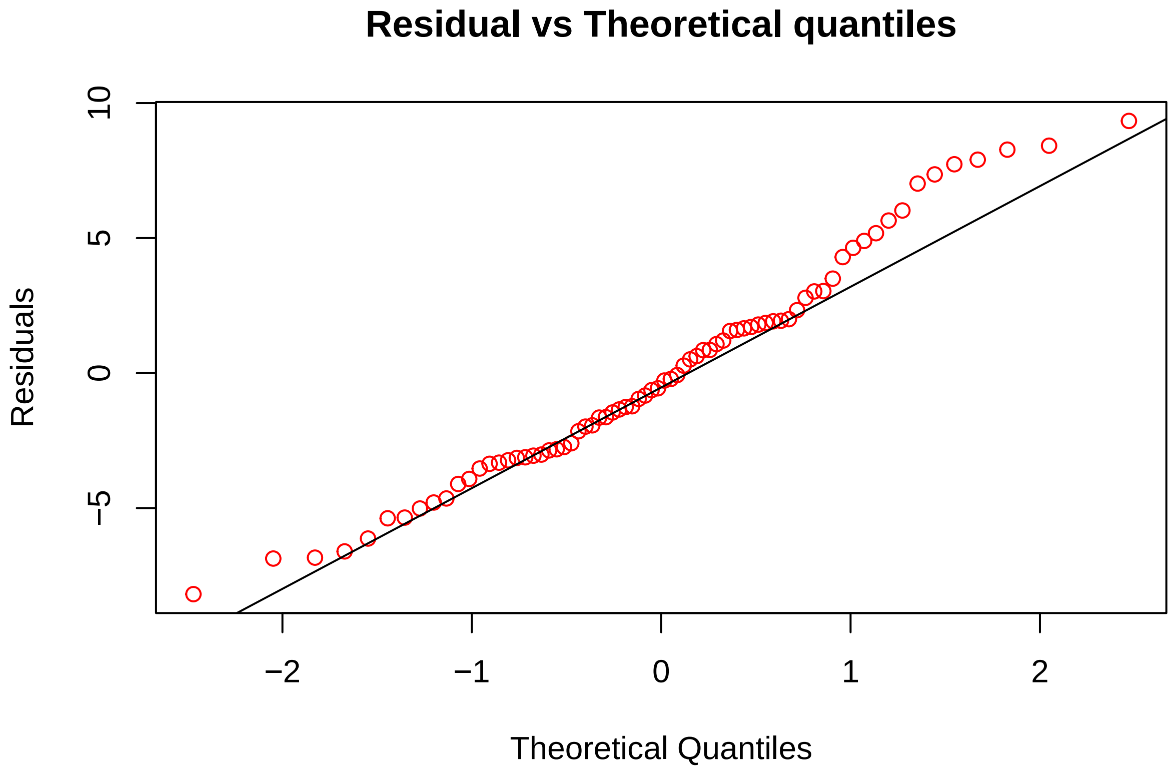


```
car::ncvTest(MLR_a) # Null hypothesis = constant error variance
```

```
## Non-constant Variance Score Test
## Variance formula: ~ fitted.values
## Chisquare = 3.842836, Df = 1, p = 0.049959
```

(c) **(2 pts)** Use a graphical method to check if the random errors follow a normal distribution. What do you conclude? Using QQ plot to check normality. The sample is plotted against the theoretical quantiles, and if the points on the plot form a straight line, then the sample can be assumed to follow the theoretical distribution, which is normal distribution. Most of the points fall on the normality line; only the latter part slightly higher than the line. We need more information to check normality.

```
qqnorm(residuals(MLR_a),ylab='Residuals',
       main='Residual vs Theoretical quantiles', col = "red")
qqline(residuals(MLR_a))
```



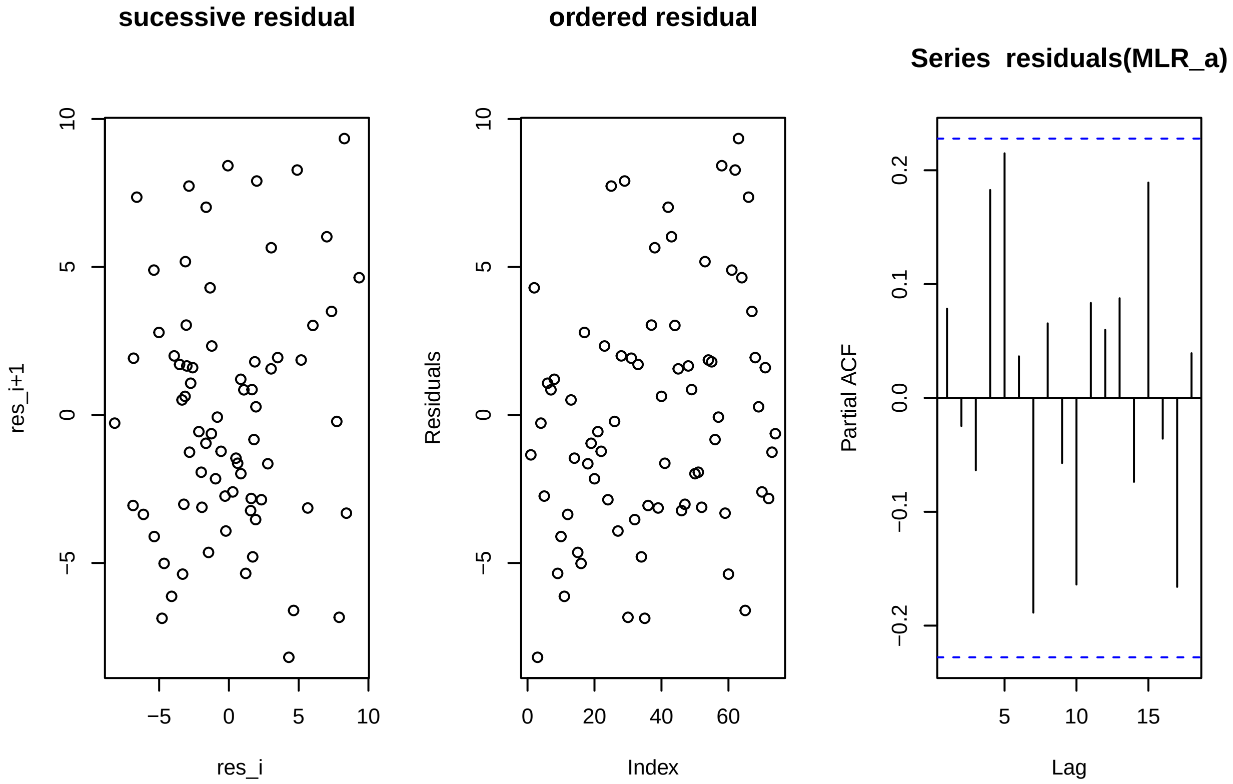
(d) **(3 pts)** Run a *Shapiro-Wilk* test to check if the random errors follow a normal distribution. What is the null hypothesis in this test? What is the p-value associated with the test? What is your conclusion? The Shapiro-Wilk test is a statistical test used to determine if a sample of data comes from a normal distribution. The null hypothesis of the Shapiro-Wilk test is that residuals are normal. The p-value is a measure of the evidence against the null hypothesis of normality. Since the p-value(0.1728) is greater than 0.05, we fail to reject the null hypothesis of normality, meaning that there is no significant evidence that the residuals do not follow a normal distribution.

```
shapiro.test(residuals(MLR_a))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(MLR_a)
## W = 0.97607, p-value = 0.1728
```

(e) **(3 pts)** Plot successive pairs of residuals. Do you find serial correlation among observations? From the successive residual and ordered residual plot, it seems like no trend in the graph. In addition, none of the Lag on Series residuals(MLR_a) plot exceed the upper and lower bound. Thus, we conclude that there is no evidence of serial correlation among the errors.

```
par(mfrow = c(1, 3), mar = c(4,4,8,2))
n <- dim(Cereal_a)[1]
plot(MLR_a$residuals[1:(n-1)], MLR_a$residuals[2:n],
     xlab = " res_i",
     ylab = "res_i+1",
     main = "sucessive residual")
plot(seq(1, dim(Cereal_a)[1],1), e_hat,
     xlab="Index",
     ylab="Residuals",
     main = "ordered residual")
acf(residuals(MLR_a), type="partial")
```



(f) **(3 pts)** Run a *Durvin-Watson* test to check if the random errors are uncorrelated. What is the null hypothesis in this test? What is the p-value associated with the test? What is your conclusion? The null hypothesis in this test is uncorrelated errors. The p-value is a measure of the evidence against the null hypothesis of non-correlation or independence. Conclusion: Since the p-value(0.2041) is greater than 0.05, we fail to reject the null hypothesis of uncorrelated errors, meaning that there is no significant evidence that true autocorrelation is greater than 0.

```
dwtest(rating ~ protein + fat + fiber + carbo + sugars + potass
                + vitamins + cups, data = Cereal_a)
```

```
##
##  Durbin-Watson test
##
## data:  rating ~ protein + fat + fiber + carbo + sugars + potass + vitamins +     cups
## DW = 1.8414, p-value = 0.2041
## alternative hypothesis: true autocorrelation is greater than 0
```

(g) **(2 pts)** Compute the hat matrix $\boldsymbol{H}$ in this data set (you don't need to show the entire matrix). Verify numerically that $\sum_{i=1}^{n} H_{ii} = p^* = p + 1$.

```
X <- model.matrix(MLR_a)
H <- X %*% solve(t(X) %*% X) %*% t(X)
print(H[1:5, 1:5])
```

```
##           1            2           3            4           6
## 1 0.20277172  0.018550985 0.148441142  0.260506267 0.032550237
## 2 0.01855098  0.327918098 0.028175391 -0.005937945 0.075234252
## 3 0.14844114  0.028175391 0.212561379  0.140353214 0.001462151
## 4 0.26050627 -0.005937945 0.140353214  0.475660077 0.011394112
## 6 0.03255024  0.075234252 0.001462151  0.011394112 0.053485279
```

```
sum_diag <-sum(diag(H)); sum_diag
```

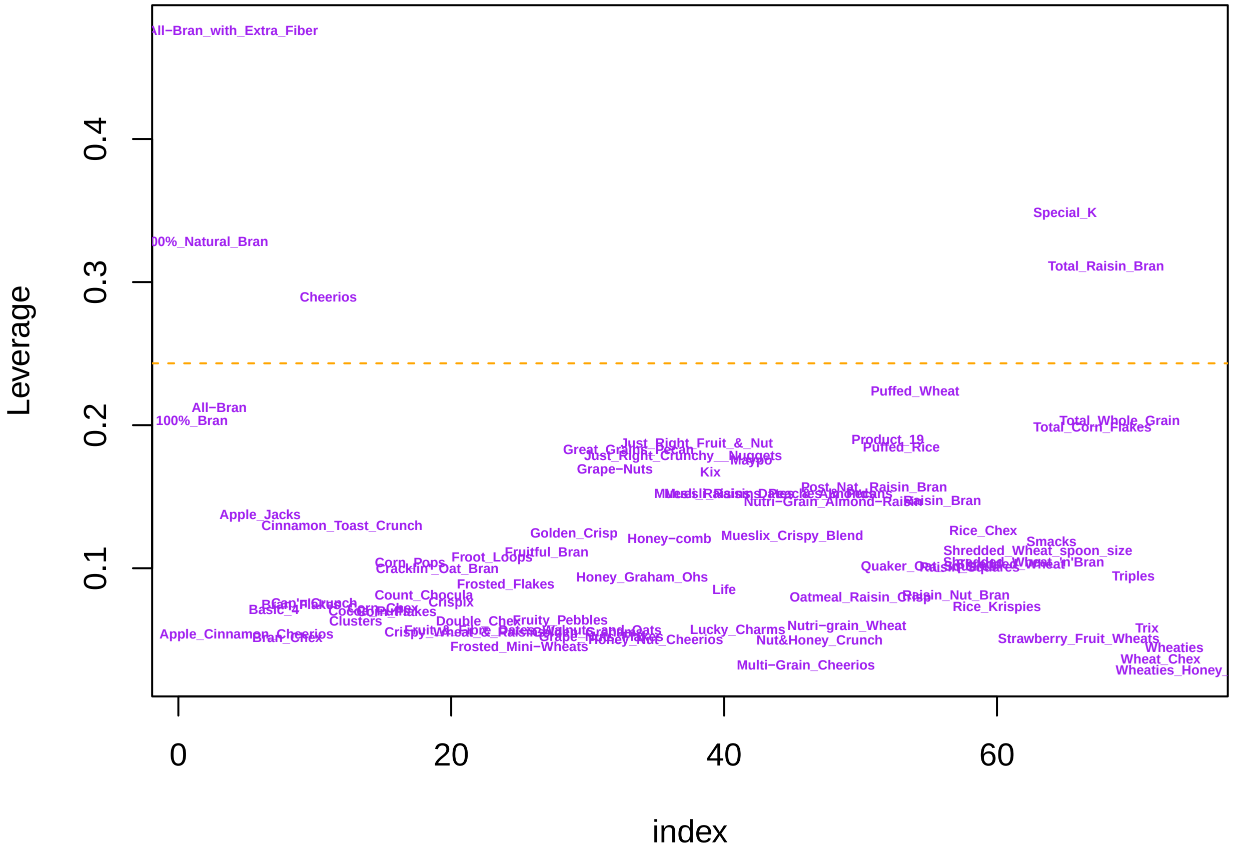
```
## [1] 9
```

```
p_star <- ncol(X); p_star
```

```
## [1] 9
```

(h) **(2 pts)** Check graphically if there is any high-leverage point. What is the criterion you used? I make use of the rule of thumb for identifying high-leverage points in a regression model, based on the number of predictor variables and the sample size. A commonly used threshold is a leverage value greater than three times the average leverage for the model,and here we have 5 observations have high-leverage point including 100%_Natural_Bran, All-Bran_with_Extra_Fiber, Cheerios, Special_K, and Total_Raisin_Bran.

```
hatv <- hatvalues(MLR_a)
Cereal_a_lev <- data.frame(index = seq(length(hatv)),
                           Leverage = hatv, namesC = Cereal_a$name)
par(mar = c(4,4,0.5,0.5))
plot(Leverage ~ index, data = Cereal_a_lev, col = "white", pch = NULL)
text(Leverage ~index, labels = namesC, data = Cereal_a_lev , cex = 0.4, font = 2, col = "purple")
abline(h =2*sum(hatv)/dim(Cereal_a_lev)[1], col = "orange", lty = 2)
```



```r
sum(hatv > 2*sum(hatv)/dim(Cereal_a_lev)[1])
```

```
## [1] 5
```

```r
high_lev <- Cereal_a|>
  filter(hatv > 2*sum(hatv)/dim(Cereal_a_lev)[1])
high_lev
```

```
##                        name   rating protein fat fiber carbo sugars potass
## 1          100%_Natural_Bran 33.98368       3   5     2     8      8    135
## 2 All-Bran_with_Extra_Fiber 93.70491       4   0    14     8      0    330
## 3                  Cheerios 50.76500       6   2     2    17      1    105
## 4                 Special_K 53.13132       6   0     1    16      3     55
## 5          Total_Raisin_Bran 28.59278       3   1     4    15     14    230
##   vitamins cups
## 1        0 1.00
## 2       25 0.50
## 3       25 1.25
## 4       25 1.00
## 5      100 1.00
```

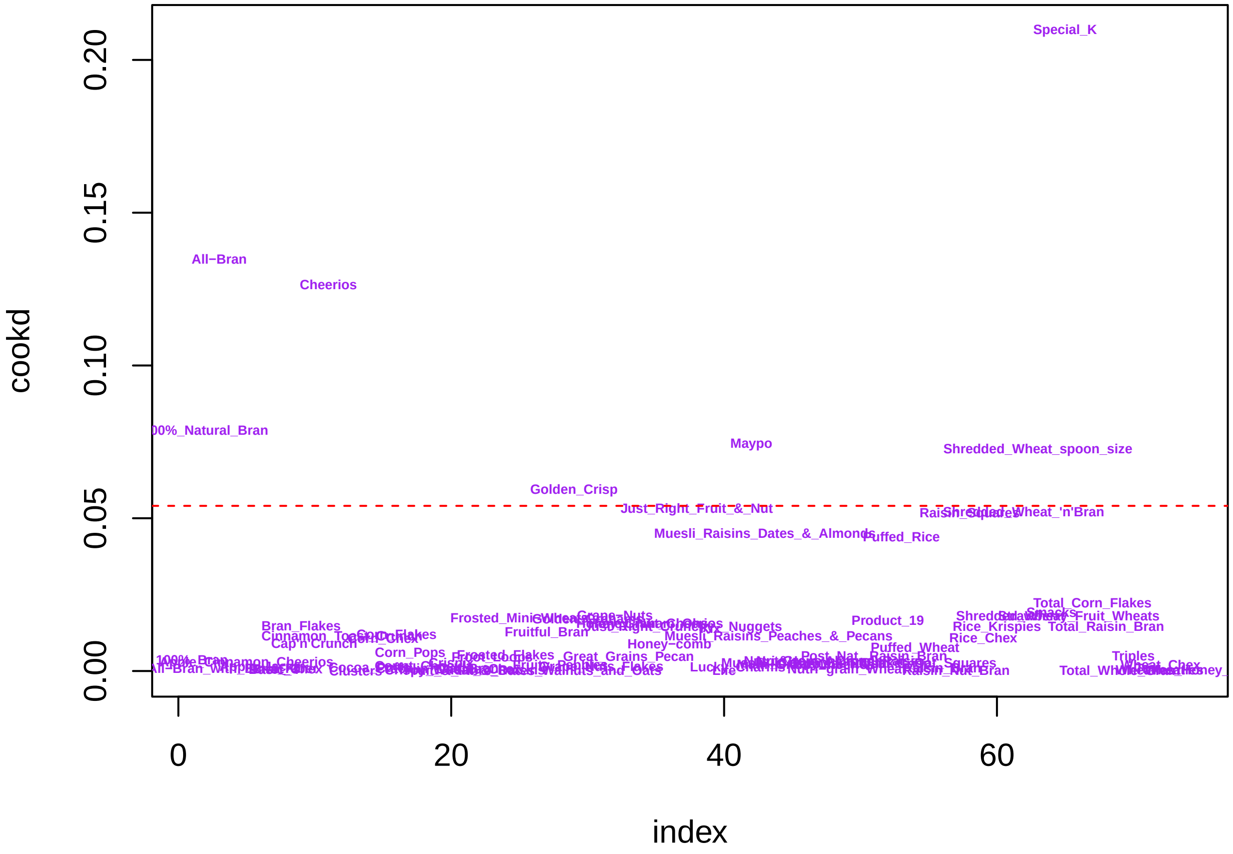
(i) **(2 pts)** Compute the standardized residuals. Without drawing a plot, is there any outlier? What is the criterion you used? To check for outliers based on the standardized residuals, a common criterion is to look for values that are greater than 3 in absolute value. Values greater than 3 or less than -3 indicate that the residual is more than three standard deviations away from the expected residual, which may suggest that the observation is an outlier.In this case, we do not have outlier.

```
r <- rstandard(MLR_a)
outliers <- sum(r > 3 | r< -3)
outliers
```

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

(j) **(2 pts)** Calculate the Cook's distance. How many observations in this data set have a Cook's distance that is greater than $4/n$? There are seven observations in this data set have a Cook's distance that is greater than $4/n$, including 100%_Natural_Bran, All-Bran, Cheerios, Golden_Crisp, Maypo, Shredded_Wheat_spoon_size and Special_K.

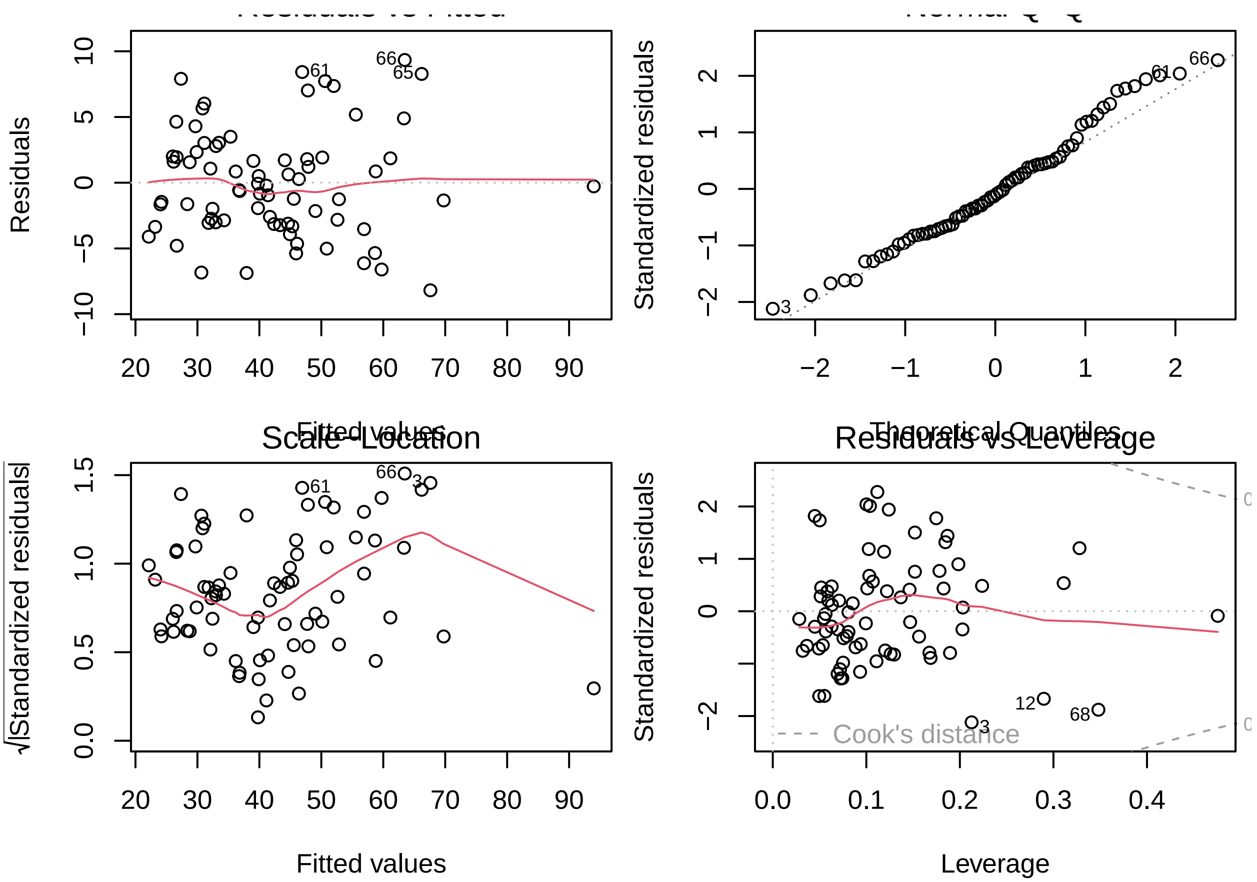
```
cook <- cooks.distance(MLR_a)
Cereal_a_cook <- data.frame(index = seq(length(cook)),
                            cookd = abs(cook), namesC = Cereal_a$name)
par(mar = c(4,4,0.5,0.5))
plot(cookd ~ index, data = Cereal_a_cook, col = "white", pch = NULL)
text(cookd ~index, labels = namesC, data = Cereal_a_cook , cex = 0.4,
     font = 2, col = "purple")
abline(h = 4/dim(X)[1], col = "red", lty = 2)
```



```
CD_greater_than <- Cereal_a|>
  filter(cook > 4/dim(X)[1]); CD_greater_than
```

```
##                         name   rating protein fat fiber carbo sugars potass
## 1          100%_Natural_Bran 33.98368       3   5     2     8      8    135
## 2                   All-Bran 59.42551       4   1     9     7      5    320
## 3                   Cheerios 50.76500       6   2     2    17      1    105
## 4               Golden_Crisp 35.25244       2   0     0    11     15     40
## 5                      Maypo 54.85092       4   1     0    16      3     95
## 6 Shredded_Wheat_spoon_size 72.80179       3   0     3    20      0    120
## 7                  Special_K 53.13132       6   0     1    16      3     55
##   vitamins cups
## 1        0 1.00
## 2       25 0.33
## 3       25 1.25
## 4       25 0.88
## 5       25 1.00
## 6        0 0.67
## 7       25 1.00
```

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

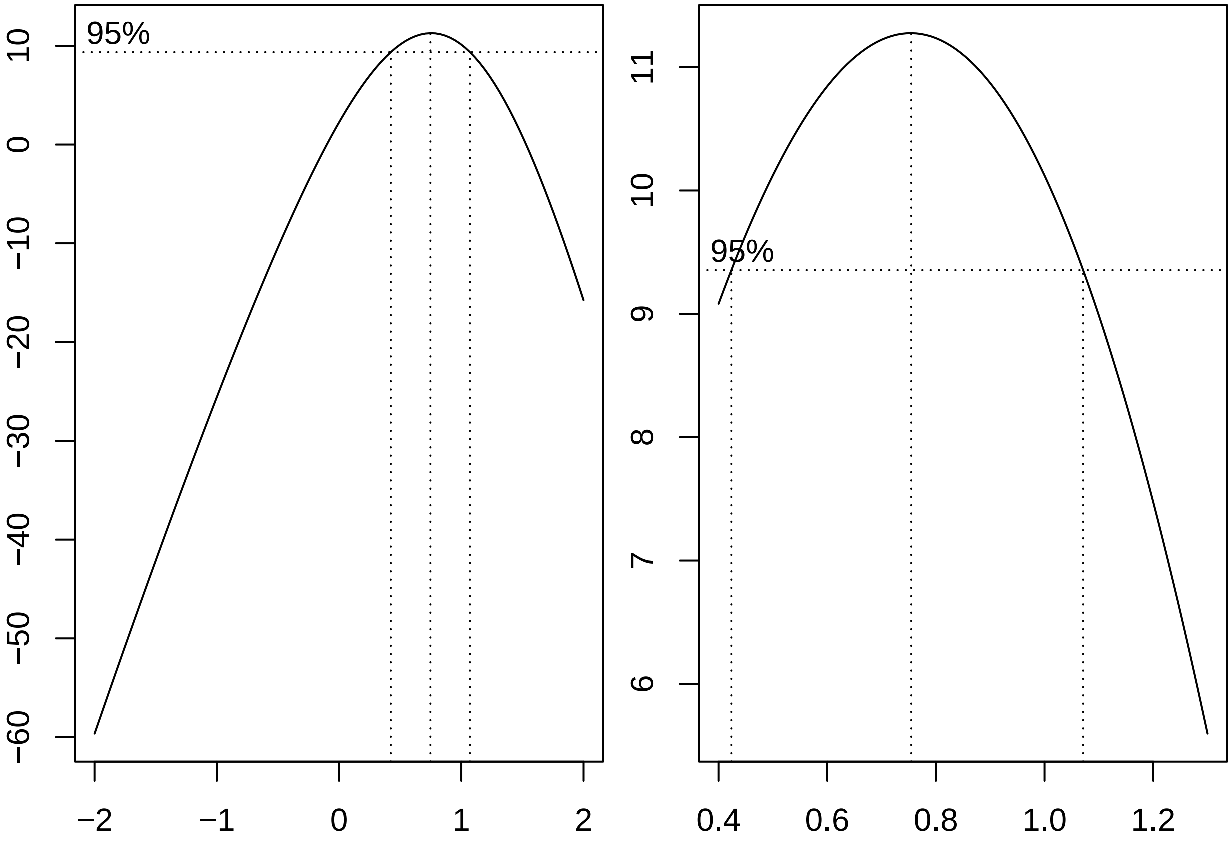


```
sum(cook >= 4/dim(X)[1])
```

```
## [1] 7
```

(k) **(2 pts)** Check whether the response needs a Box-Cox transformation. If a Box-Cox transformation is necessary, what would be the form of the transformation? Since the confidence interval contains lambda = 1, no transformation is necessary.

```
par(mfrow = c(1, 2), mar = c(2, 2, 0.8, 0.5))
boxcox(MLR_a, plotit = TRUE)
boxcox(MLR_a, plotit = TRUE, lambda = seq(0.4, 1.3, by = 0.1))
```


95%
10
0
−10
−20
−30
−40
−50
−60
−2
−1
0
1
2
11
10
9
8
7
6
95%
0.4
0.6
0.8
1.0
1.2