STAT 530 HW 4

Md Muhtasim Billah

4/6/2020

Question 1.

Use the data set you used in HW 3 and answer the following questions.

a. Do two iterations of weighted least squares.
b. Run robust regression (M with Huber weights) to see if influential observations were an issue.
c. Use Ridge Regression to see with c between .01 and .1 and provide the ridge plot.

Answer:

a.

Weighted least square (WLS) is performed as a remedy when the assumption of constant variance of the residuals are violated. The data set used in HW3 has been "cleaned" and then used for two iterations of WLS.

```
#setting working directory
setwd("/Users/muhtasim/Desktop/STAT530/HWs/HW4")
#reading the "cleaned" HW3 data
mydata=read.csv("HW3.csv",header=TRUE)
#fit the regression model using unweighted/ordinary least squares (OLS)
fit = lm(Calories ~ Weight.lbs + Height.Inches + Protein + Carbohydrates + Calcium
         + gender, data = mydata)
#find the residuals
r=resid(fit)
#take the square of the residuals
esq=r*r
#regress squared residuals on the relevant predictors
efit=lm(esq ~ Weight.lbs + Height.Inches + Protein + Carbohydrates + Calcium
        + gender, data = mydata)
#find the predicted values from the above regression
p=predict(efit)
#use the predicted values to estimate the weight function
w=1/abs(p)
#now fit the regression model for weighted least square (WLS) using the weight w
fit2=lm(Calories ~ Weight.lbs + Height.Inches + Protein + Carbohydrates + Calcium
        + gender, data = mydata, weight=w)
#iteration 1 ends

#iteration 2
r1=resid(fit2)
esq1=r1*r1
efit1=lm(esq1 ~ Weight.lbs + Height.Inches + Protein + Carbohydrates + Calcium
         + gender, data = mydata)
```

```
p1=predict(efit1)
w1=1/abs(p1)
fit3=lm(Calories ~ Weight.lbs + Height.Inches + Protein + Carbohydrates + Calcium
        + gender, data = mydata, weight=w1)
#find coefficient for OLS regression
coef(fit)
```

```
##   (Intercept)    Weight.lbs Height.Inches       Protein Carbohydrates
##  -366.7064556     1.2523614     3.0139814    13.4288870     4.5916254
##       Calcium        gender
##    -0.2365993   -70.0333150
```

```
#find coefficient for WLS regression for iteration 1
coef(fit2)
```

```
##   (Intercept)    Weight.lbs Height.Inches       Protein Carbohydrates
##  -436.3399636     0.4407559     5.9774742    12.4000489     4.6966715
##       Calcium        gender
##    -0.1598715   -66.5405544
```

```
#find coefficient for WLS regression for iteration 2
coef(fit3)
```

```
##   (Intercept)    Weight.lbs Height.Inches       Protein Carbohydrates
## -345.64586158    0.08568511    5.54363979   12.57723396    4.48917678
##       Calcium        gender
##   -0.14825879  -27.13968482
```

It is evident that the WLS estimates from the first and second iterations are very different from the ordinary least squares (OLS). To reach stability, several more iterations might be required.

b.

Robust regression is performed when there are potential ourliers or influencial points in the dataset. Iterated re-weighted least squares (IRLS) is one of the most frequently used method for robust regression. There are several weighting functions that can be used for IRLS such as Huber weights and bi-weights. Different functions have advantages and drawbacks. Huber weights can have difficulties with severe outliers, and bisquare weights can have difficulties converging or may yield multiple solutions. For this problem, Huber weights will be used with M-estimation where M stands for "maximum likelihood type". The M-estimation is robust to outliers in the response variable, but not resistant to outliers in the explanatory variables (leverage points).

```
#robust regression (M with Huber weights)
#rlm function fits a linear model by robust regression using an M estimator
#fitting is done by iterated re-weighted least squares (IRLS).
library(MASS)
fit_robust = rlm(Calories ~ Weight.lbs + Height.Inches + Protein + Carbohydrates
                 + Calcium + gender, data = mydata, method = c("M"), psi = psi.huber)
r = rstudent(fit_robust)
absr = abs(r)
```

```
hweights = data.frame(residuals= fit_robust$residuals, stud_resids = r,
                      abs_stdres = absr, huber_weights = fit_robust$w)
hweights2 = hweights[order(fit_robust$w), ]
hweights2[1:20, ]
```

```
##    residuals stud_resids abs_stdres huber_weights
## 62  436.6387   2.4306127  2.4306127     0.6028740
## 11 -405.0035  -2.3105093  2.3105093     0.6500027
## 18 -393.3374  -2.1657726  2.1657726     0.6692609
## 7   337.5718   1.8733875  1.8733875     0.7798157
## 67 -331.0123  -1.8345616  1.8345616     0.7953019
## 25  316.8166   1.7400024  1.7400024     0.8308703
## 69  312.6388   1.7124010  1.7124010     0.8419975
## 56 -307.9414  -1.9556458  1.9556458     0.8547840
## 68 -295.0241  -1.6036622  1.6036622     0.8922759
## 28  294.9482   1.7049632  1.7049632     0.8925431
## 93  282.3502   1.5221983  1.5221983     0.9323198
## 92  281.2242   1.5548486  1.5548486     0.9360184
## 21 -280.4642  -1.5196355  1.5196355     0.9386141
## 49  279.6024   1.5340898  1.5340898     0.9414752
## 33 -273.3810  -1.4958917  1.4958917     0.9629257
## 50 -272.2843  -1.5009667  1.5009667     0.9668012
## 9  -268.7135  -1.5651137  1.5651137     0.9795897
## 1   212.6706   1.1654306  1.1654306     1.0000000
## 2  -211.2062  -1.1389890  1.1389890     1.0000000
## 3   135.1525   0.7415853  0.7415853     1.0000000
```

We can see that roughly, as the absolute studentized residuals go down, the huber weights goes up. In other words, cases with a large residual tend to be down-weighted. In OLS regression, all cases have a weight of 1. Hence, the more cases in the robust regression that have a weight close to one, the closer the results of the OLS and robust regressions. Here, we see that apart from the first 17 observation listed in the table above, rest have weights of 1. So, there is an issue of influencial points.

When comparing the results of a regular OLS regression and a robust regression, if the results are very different, it would be better to use the results from the robust regression. Large differences suggest that the model parameters are being highly influenced by outliers.

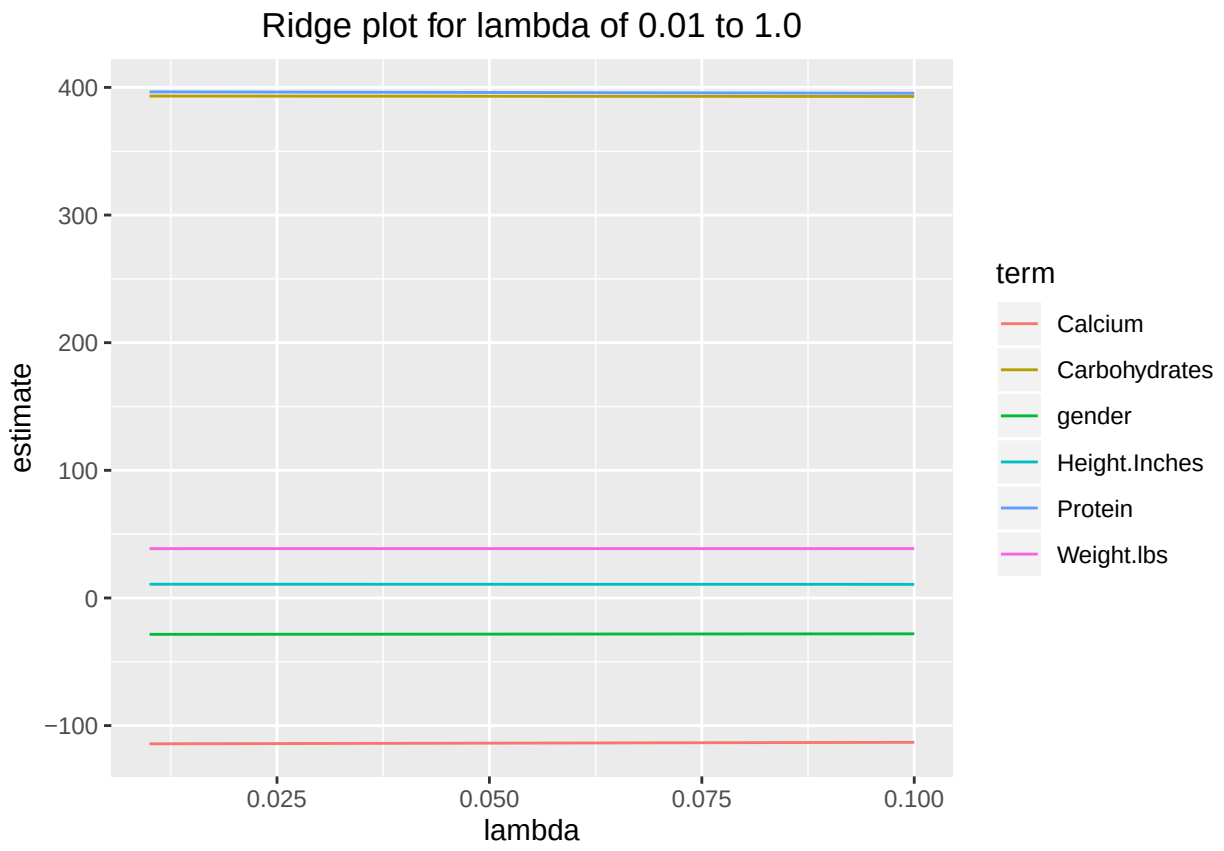
c.

```
#package needed to generate correlated predictors
library(MASS)
#first ridge using library (MASS)
ridgereg = lm.ridge(Calories ~ Weight.lbs + Height.Inches + Protein + Carbohydrates +
                      Calcium + gender, data = mydata, lambda = seq(0.01,0.1,0.01))
#other data output method
library(broom)
td = tidy(ridgereg)
g = glance(ridgereg)
g
```

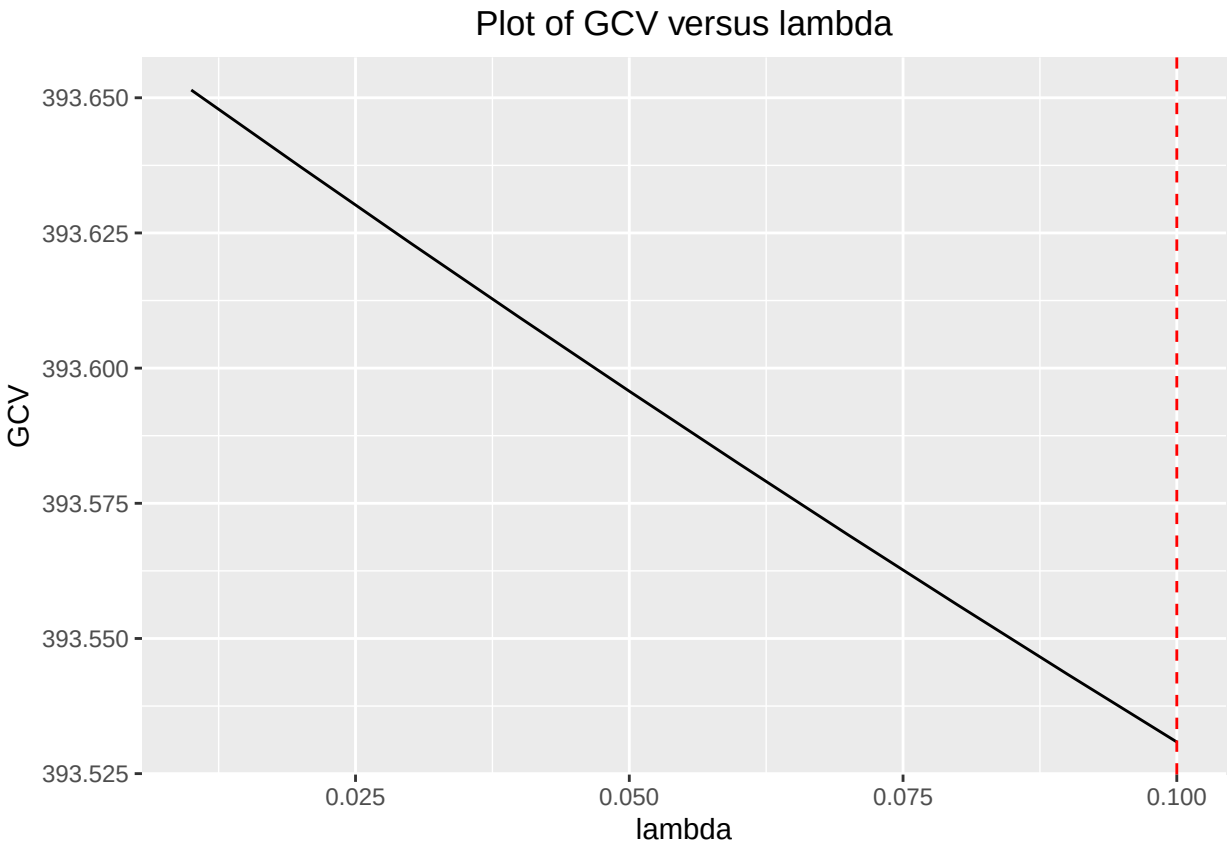
```
## # A tibble: 1 x 3
```

```
##    kHKB   kLW lambdaGCV
##   <dbl> <dbl>     <dbl>
## 1 0.433 0.312      0.10
```

```
#fancy plotting
library(ggplot2)
#plot of the estimates
ggplot(td, aes(lambda, estimate, color = term)) + geom_line() +
  ggtitle("Ridge plot for lambda of 0.01 to 1.0") +
  theme(plot.title = element_text(hjust = 0.5))
```




```
# plot of GCV versus lambda
ggplot(td, aes(lambda, GCV)) + geom_line() +
  geom_vline(xintercept = g$lambdaGCV, col = "red", lty = 2) +
  ggtitle("Plot of GCV versus lambda") + theme(plot.title = element_text(hjust = 0.5))
```



Looking at the ridge plot, it is visible that the lines for each predictors are straight lines which indicates that there is no multicollinearity among the predictors of the dataset. Also, from the GCV vs lambda plot, we can see that the minimum GCV value occurs for the value of lambda = 0.1.

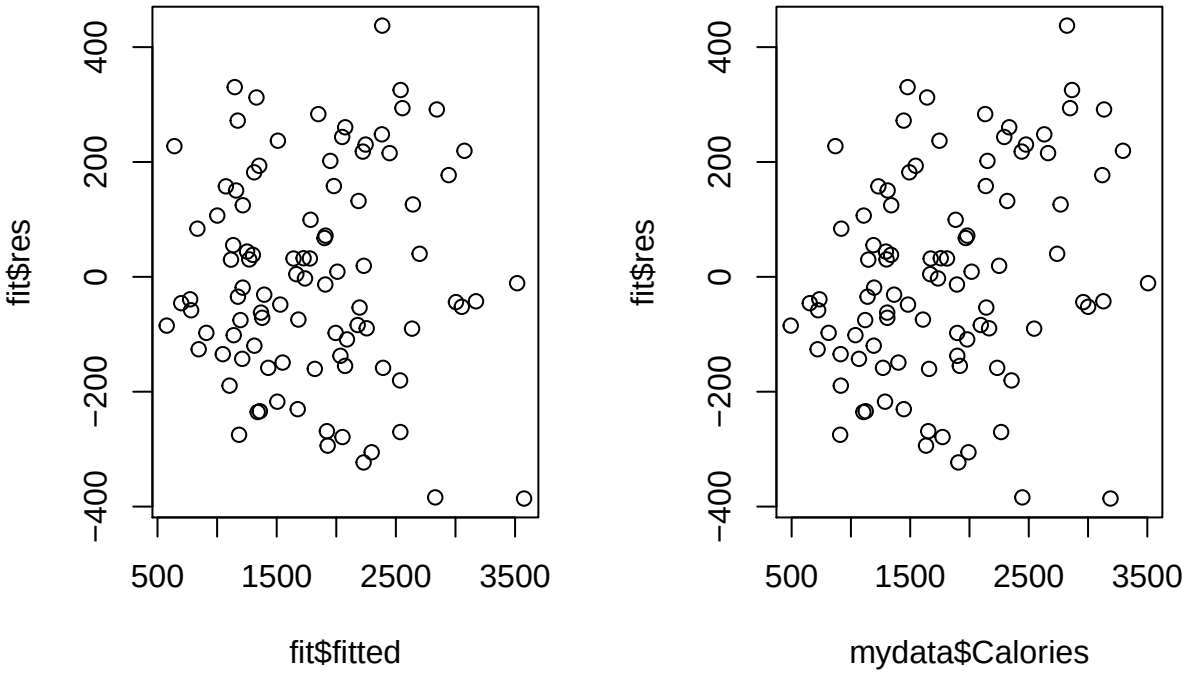
Question 2.

Do plots of residual versus predicted and residual versus observed. Are they the same plot? Comment on your findings.

Answer:

The plots of residual versus predicted and residual versus observed are given below.

```
#plot of residual vs fitted and observed values
par(mfrow=c(1,2))
plot(fit$fitted, fit$res)
plot(mydata$Calories, fit$res)
```



Comment:

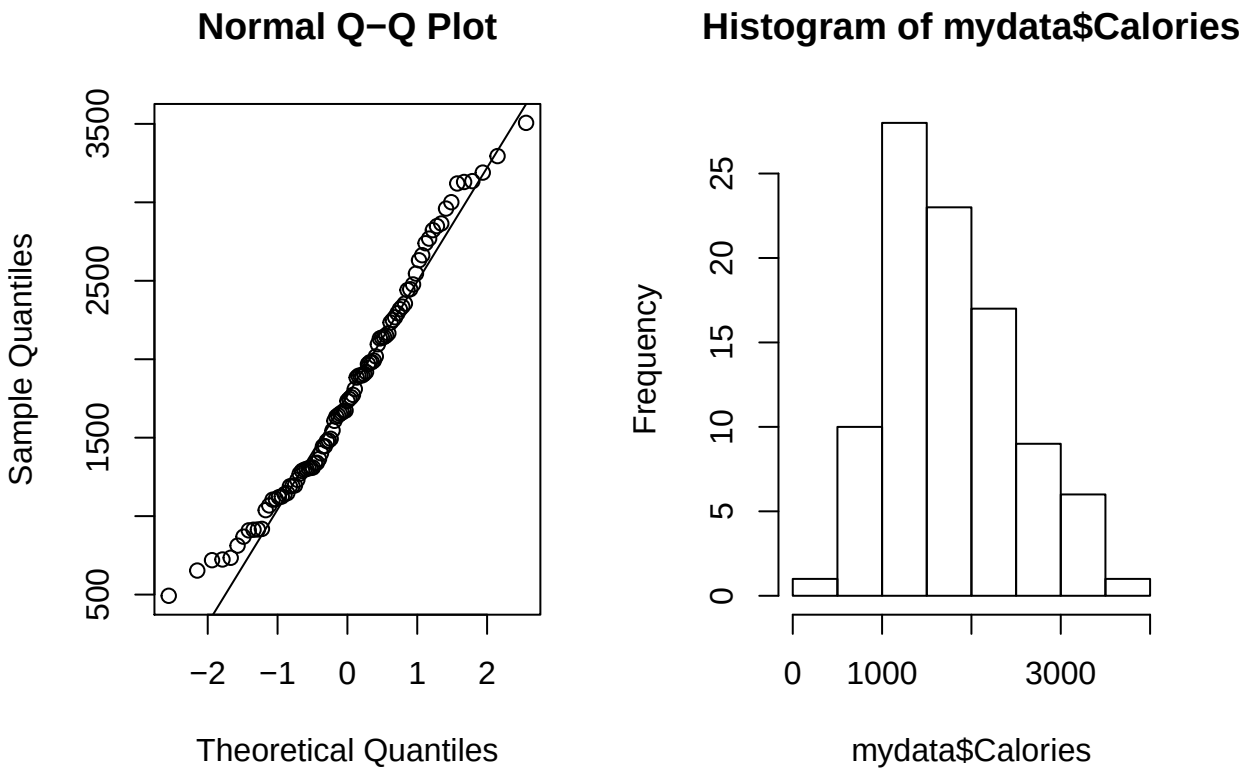
From the plots, it is visible that the two plots are similar but not exactly the same. Since predicted/fitted values are estimation of the observed value, they are mostly similar but not exactly the same.

Question 3.

Instead of doing residual analysis with residuals (for the example above) use the observed Y's. Look at plots and tests. What do you get?

Answer:

```
par(mfrow=c(1,2))
# Normal probability plot of the observed Y's
qqnorm(mydata$Calories)
qqline(mydata$Calories)
hist(mydata$Calories)
```




```
#test for normality
library(stats)
shapiro.test(mydata$Calories)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  mydata$Calories
## W = 0.97169, p-value = 0.03722
```

Comment:

From the diagnostic plots, it seems like the observed Y's don't belong to a normal distribution.