McD_Nutrient_Menu - Linear Regression Model

Aravind

January 11, 2018

McDonalds nutrient menu is an interesting dataset that contain values of calories, fat, cholesterol, carbohydrates, sodium content, protein, fiber and vitamins of different type of food served at McDonalds. Dependent Variable(Target) = Calories

```
data <- read.csv("menu.csv")
str(data)
```

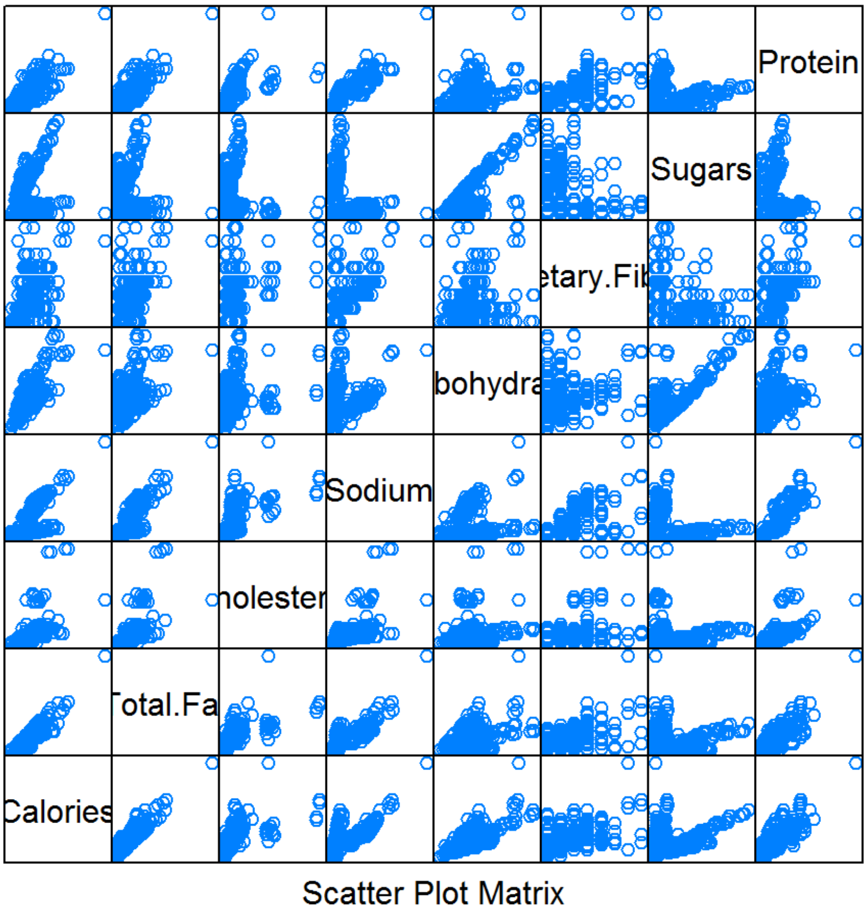
```
## 'data.frame':    260 obs. of  24 variables:
##  $ Category                     : Factor w/ 9 levels "Beef & Pork",..: 3 3 3 3
3 3 3 3 3 3 ...
##  $ Item                         : Factor w/ 260 levels "1% Low Fat Milk Jug",..
: 76 77 228 229 230 245 12 11 14 13 ...
##  $ Serving.Size                 : Factor w/ 107 levels "1 carton (236 ml)",..:
55 54 42 69 69 83 63 72 65 73 ...
##  $ Calories                     : int  300 250 370 450 400 430 460 520 410 470
...
##  $ Calories.from.Fat            : int  120 70 200 250 210 210 230 270 180 220 ..
.
##  $ Total.Fat                    : num  13 8 23 28 23 23 26 30 20 25 ...
##  $ Total.Fat....Daily.Value.    : int  20 12 35 43 35 36 40 47 32 38 ...
##  $ Saturated.Fat                : num  5 3 8 10 8 9 13 14 11 12 ...
##  $ Saturated.Fat....Daily.Value.: int  25 15 42 52 42 46 65 68 56 59 ...
##  $ Trans.Fat                    : num  0 0 0 0 0 1 0 0 0 0 ...
##  $ Cholesterol                  : int  260 25 45 285 50 300 250 250 35 35 ...
##  $ Cholesterol....Daily.Value.  : int  87 8 15 95 16 100 83 83 11 11 ...
##  $ Sodium                       : int  750 770 780 860 880 960 1300 1410 1300 1
420 ...
##  $ Sodium....Daily.Value.       : int  31 32 33 36 37 40 54 59 54 59 ...
##  $ Carbohydrates                : int  31 30 29 30 30 31 38 43 36 42 ...
##  $ Carbohydrates....Daily.Value.: int  10 10 10 10 10 10 13 14 12 14 ...
##  $ Dietary.Fiber                : int  4 4 4 4 4 4 2 3 2 3 ...
##  $ Dietary.Fiber....Daily.Value.: int  17 17 17 17 17 18 7 12 7 12 ...
##  $ Sugars                       : int  3 3 2 2 2 3 3 4 3 4 ...
##  $ Protein                      : int  17 18 14 21 21 26 19 19 20 20 ...
##  $ Vitamin.A....Daily.Value.    : int  10 6 8 15 6 15 10 15 2 6 ...
##  $ Vitamin.C....Daily.Value.    : int  0 0 0 0 0 2 8 8 8 8 ...
##  $ Calcium....Daily.Value.      : int  25 25 25 30 25 30 15 20 15 15 ...
##  $ Iron....Daily.Value.         : int  15 8 10 15 10 20 15 20 10 15 ...
```

we can do scatter plots and corrplots to check relation among variables

```
require(lattice)
```

```
## Loading required package: lattice
```

```
splom(~data[c(4,6,11,13,15,17,19,20)],groups = NULL, data = data, axis.line.tck =
0, axis.text.alpha = 0)
```




```
require(corrplot)
```

```
## Loading required package: corrplot
```

```
## corrplot 0.84 loaded
```

```
cr <- cor(data[c(4,6,11,13,15,17,19,20)])
corrplot(cr,method = "number")
```

Total fat has a high corelation = 0.9, followed by protein = 0.79 and carbohydrates = 0.78

Split the dataset into train and test

```
library(caTools)
set.seed(2) #to get the same split everytime
split <- sample.split(data$Calories,SplitRatio = 0.70)
train <- subset(data,split == "TRUE")
test <- subset(data, split == "FALSE")
```

Let’s first build a Linear regression model between calories and total fat. Independent Variable - Total.Fat

Scatter plot and Conditional expectation(mean) plot

```
require(dplyr)
```

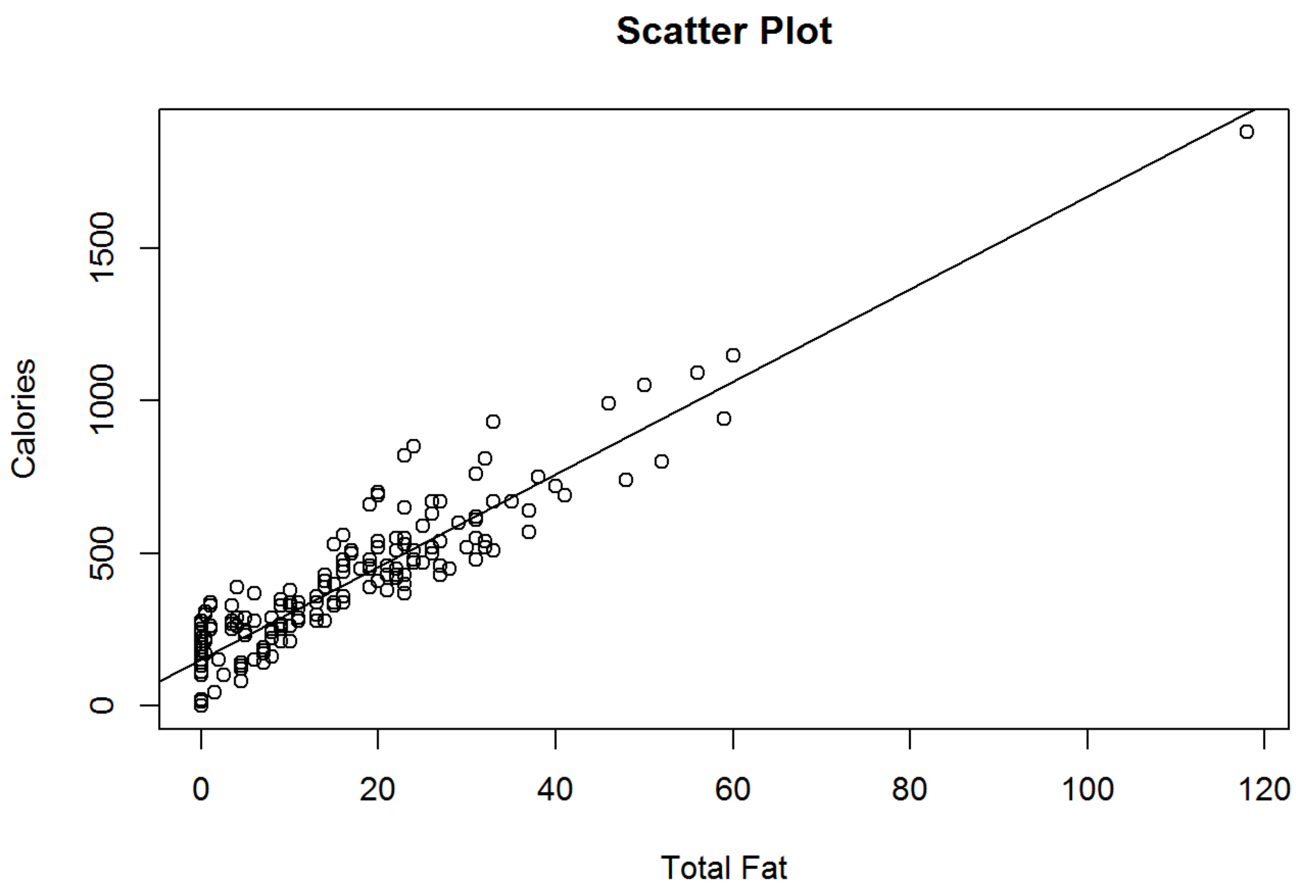
```
## Loading required package: dplyr
```

```
##
## Attaching package: 'dplyr'
```

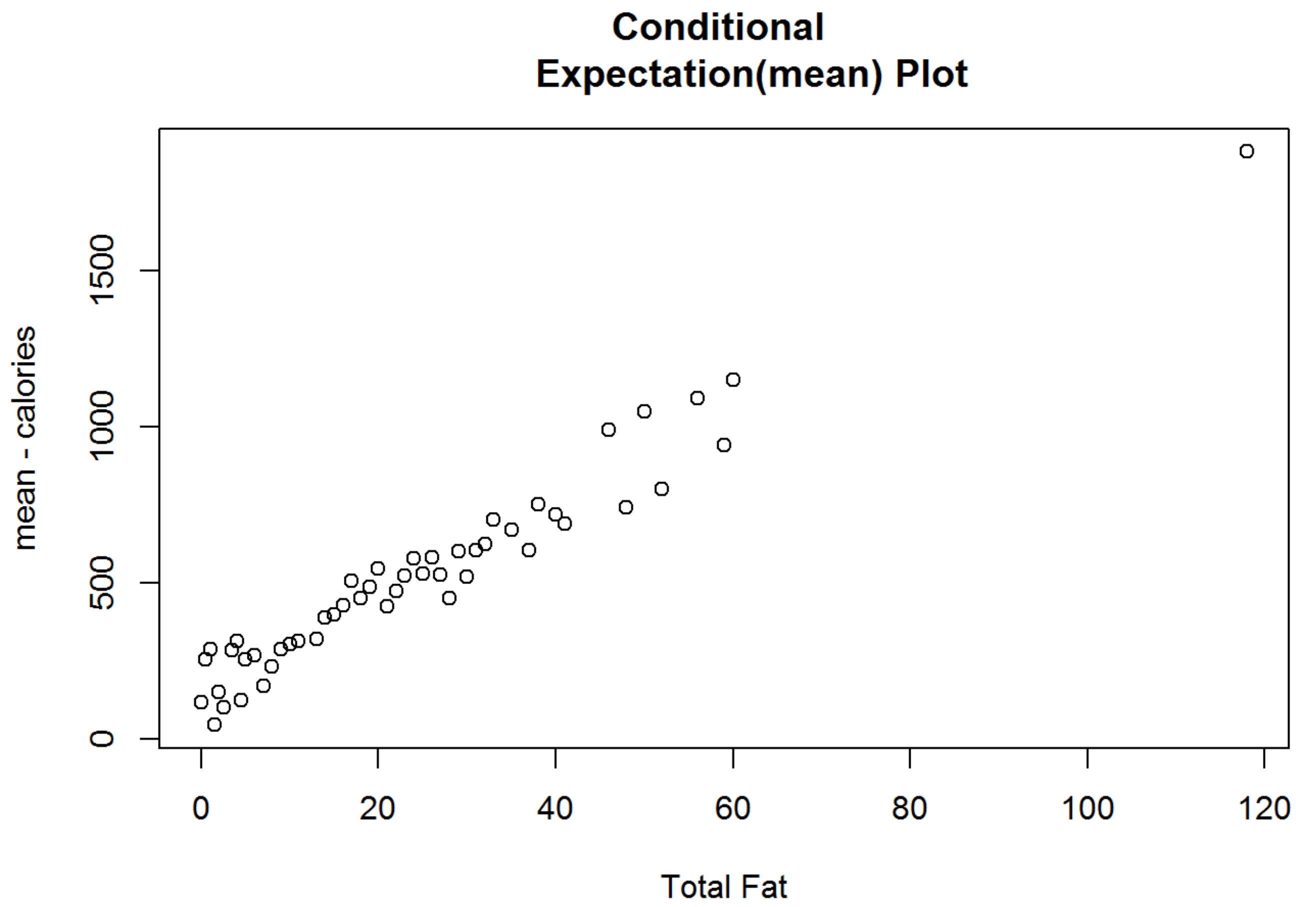
```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
#Scatter PLot
plot(train$Total.Fat,train$Calories,main = "Scatter Plot",xlab = "Total Fat", ylab
= "Calories")
abline(lm(train$Calories~train$Total.Fat))
```




```
#Conditional Expectation Plot
dataexp <- summarise(group_by(train,Total.Fat),calmean = mean(Calories))
plot(dataexp$Total.Fat,dataexp$calmean,xlab = "Total Fat",ylab = "mean - calories",
main = "Conditional
     Expectation(mean) Plot")
```



Linear Regression Model

```
model1 <- lm(Calories~Total.Fat,data = train)
summary(model1)
```

```
##
## Call:
## lm(formula = Calories ~ Total.Fat, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -152.84  -71.97  -10.17   63.21  332.85
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  152.838     10.026   15.24   <2e-16 ***
## Total.Fat     15.180      0.469   32.37   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 99.18 on 187 degrees of freedom
## Multiple R-squared:  0.8485, Adjusted R-squared:  0.8477
## F-statistic:  1048 on 1 and 187 DF,  p-value: < 2.2e-16
```

value of intercept = 151.838 value of slope = 15.180

Both the values are significant(*** refers to high signficance) R-squared = 85% (This means 82% of variance in calories is explained by total fat) The overall p-value is also significant

The linear equation to predict calories : Calories = 151.5882 + 15.2965*total fat

Lets Build a multiple regression model.

From corplots we found out that total fat, protein and carbohydrates are highly corelated.

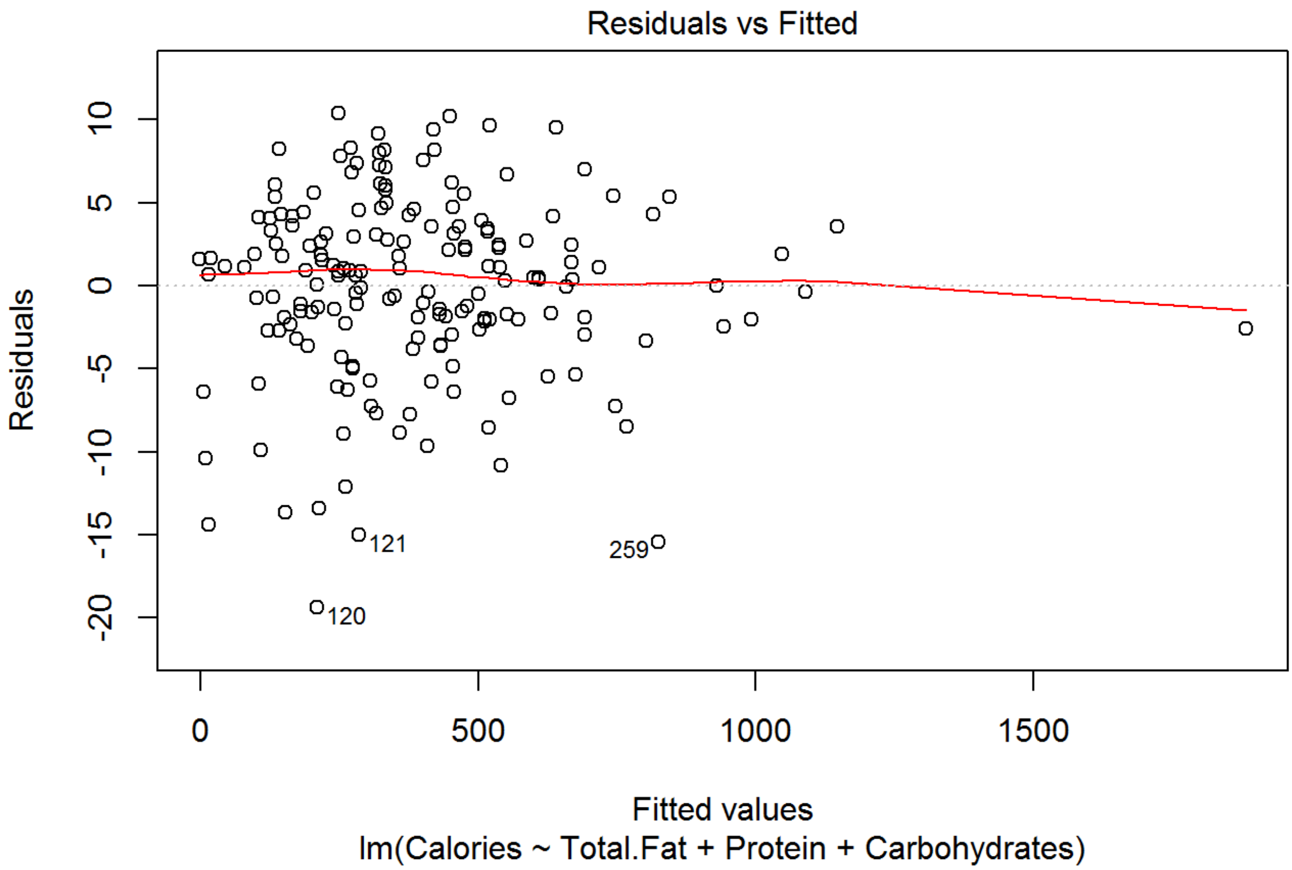
```
model <- lm(Calories~ Total.Fat + Protein + Carbohydrates, data = train)
summary(model)
```

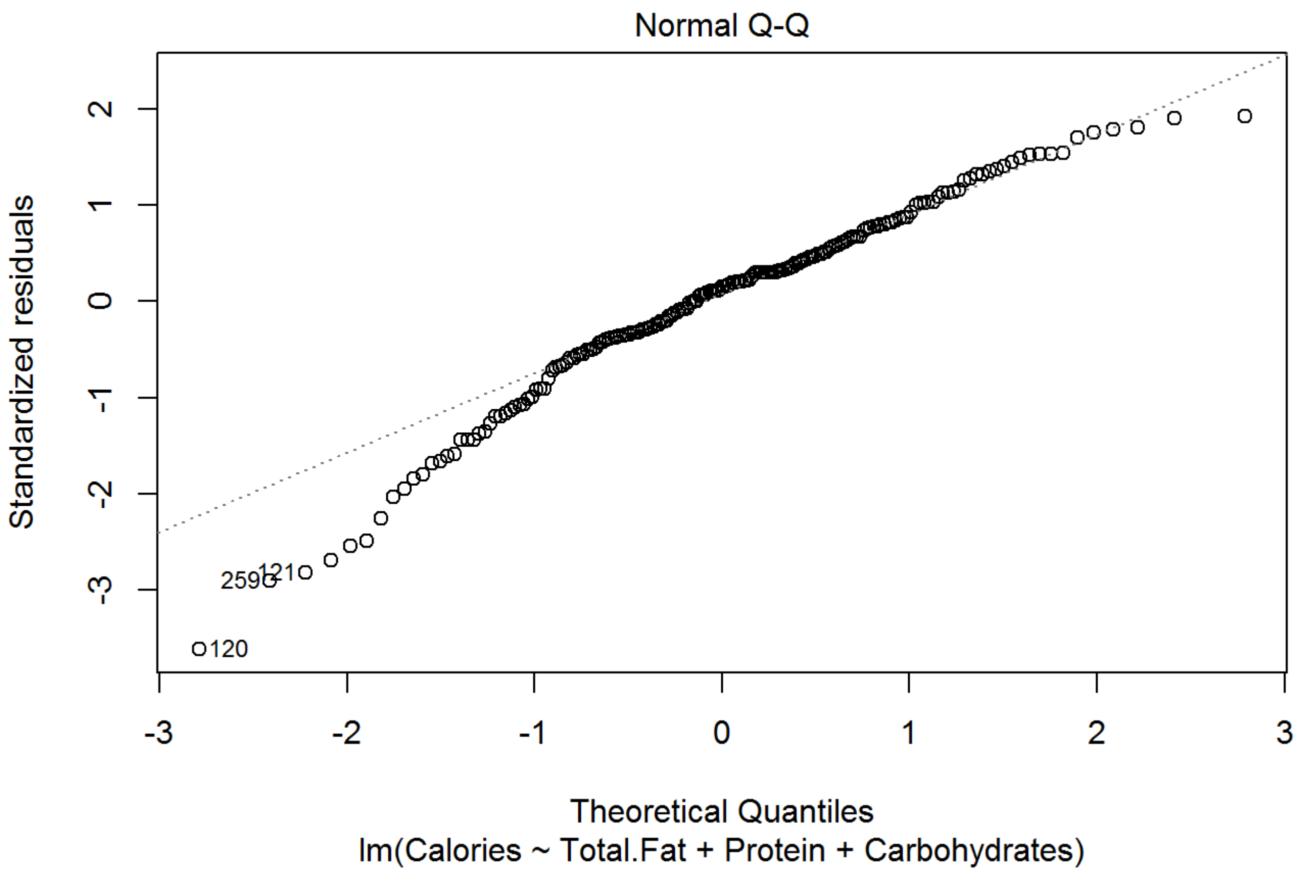
```
##
## Call:
## lm(formula = Calories ~ Total.Fat + Protein + Carbohydrates,
##     data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -19.4031  -2.4901   0.8274   3.4343  10.3781
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -1.58186    0.83800  -1.888   0.0606 .
## Total.Fat      9.03996    0.04810 187.955   <2e-16 ***
## Protein        3.99646    0.06007  66.527   <2e-16 ***
## Carbohydrates  3.98085    0.01621 245.595   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.41 on 185 degrees of freedom
## Multiple R-squared:  0.9996, Adjusted R-squared:  0.9995
## F-statistic: 1.382e+05 on 3 and 185 DF,  p-value: < 2.2e-16
```

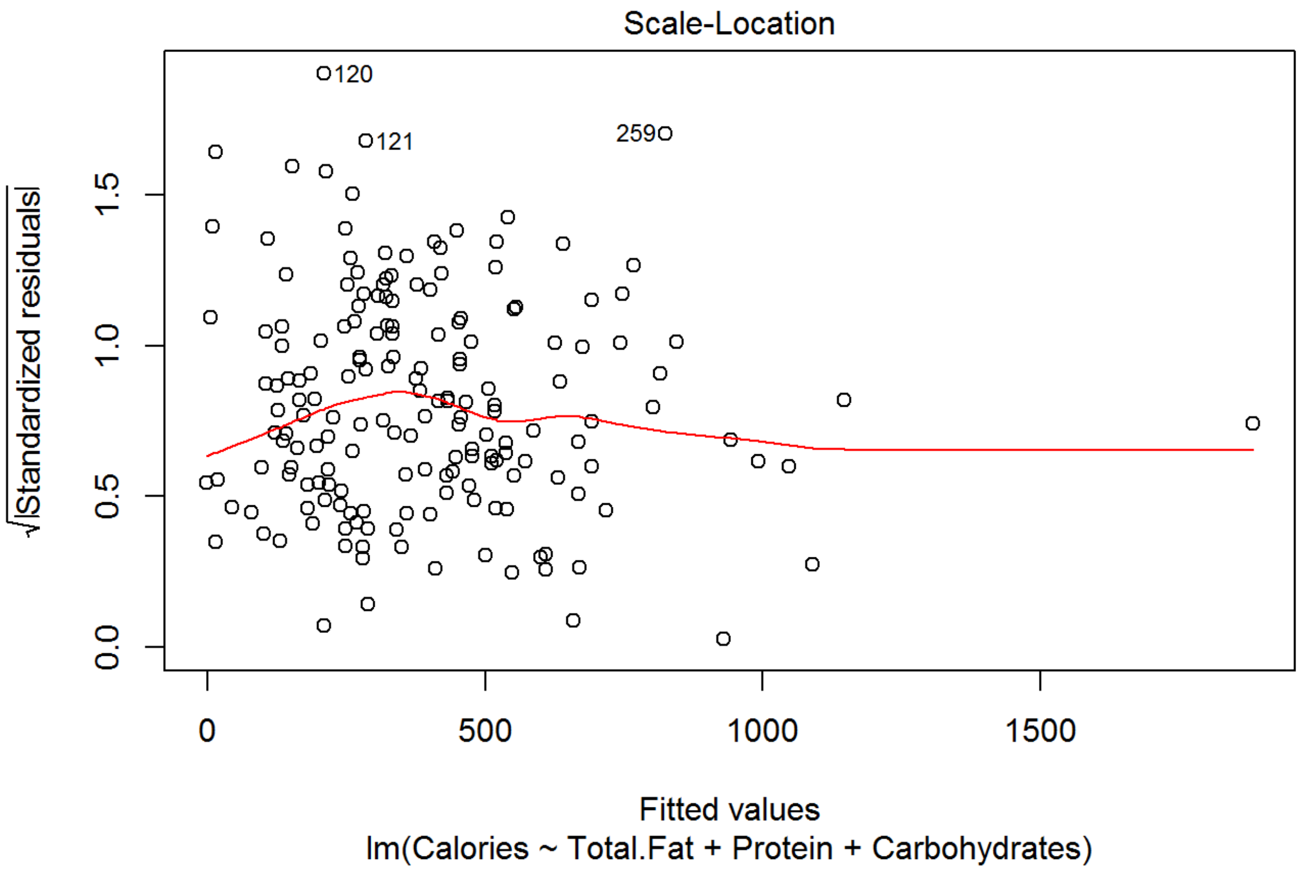
R - sq value of 1. These three variables almost explains 100% of variance in calories

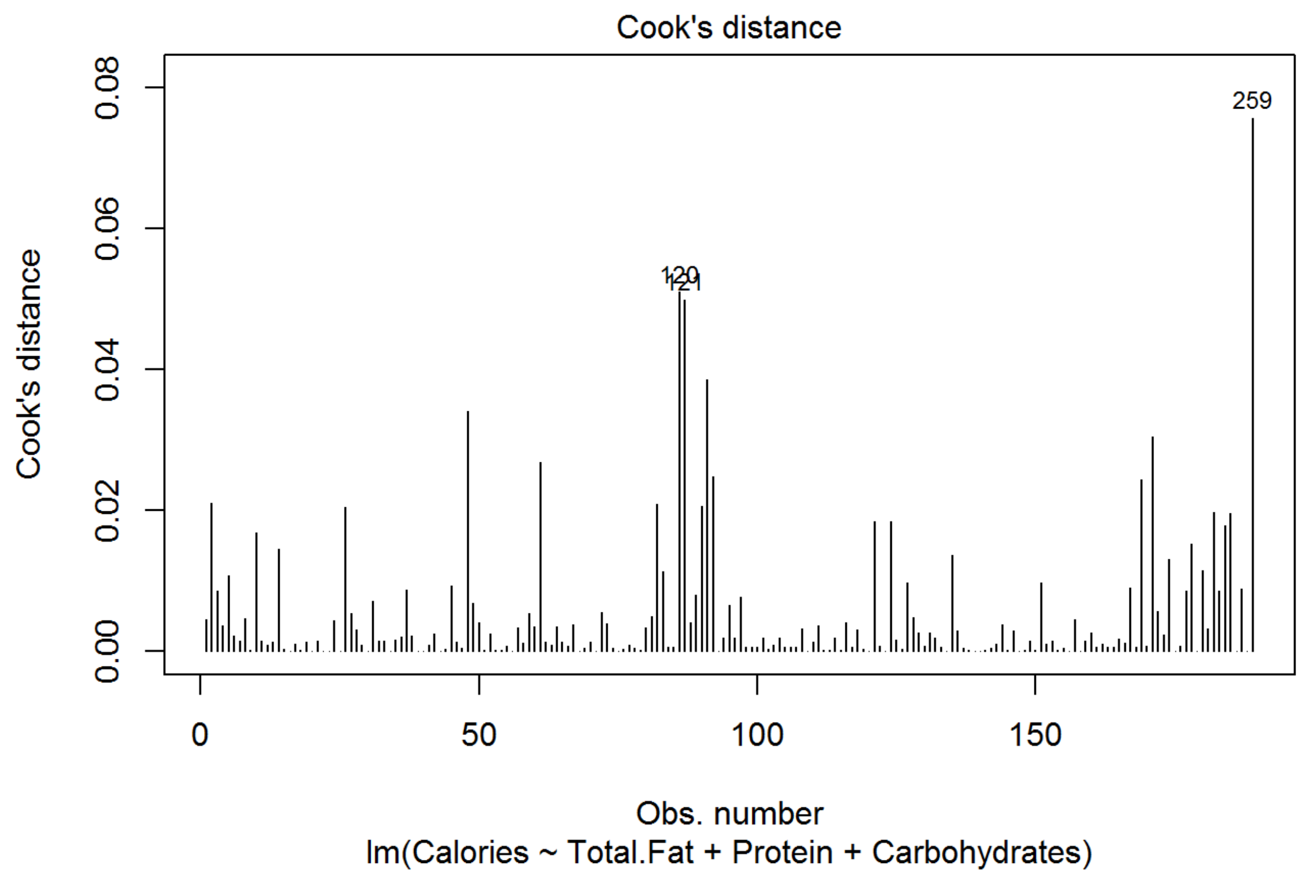
Regression Diagnostics

```
plot(model, which = 1:4)
```


Residuals vs Fitted
Residuals
10
5
0
-5
-10
-15
-20
121
259
120
0
500
1000
1500
Fitted values
lm(Calories ~ Total.Fat + Protein + Carbohydrates)


Normal Q-Q
Standardized residuals
2
1
0
-1
-2
-3
259
121
120
-3
-2
-1
0
1
2
3
Theoretical Quantiles
lm(Calories ~ Total.Fat + Protein + Carbohydrates)





CD > = k/n (k is # of predictors, n is sample size) CD > = 3/189 = 0.016 Rough cut off - 4/n = 4/189 = 0.02. Observations 120,121,259 can be removed and model can be rebuilt.

```r
train <- train[-c(120,121,259),]
model <- lm(Calories~ Total.Fat + Protein + Carbohydrates, data = train)
summary(model)
```

```
##
## Call:
## lm(formula = Calories ~ Total.Fat + Protein + Carbohydrates,
##     data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -19.4695  -2.3911   0.7606   3.3865  10.3096
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -1.63011    0.83846  -1.944   0.0534 .
## Total.Fat      9.02872    0.04915 183.709   <2e-16 ***
## Protein        4.00762    0.06084  65.869   <2e-16 ***
## Carbohydrates  3.98301    0.01631 244.251   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.409 on 183 degrees of freedom
## Multiple R-squared:  0.9996, Adjusted R-squared:  0.9996
## F-statistic: 1.381e+05 on 3 and 183 DF,  p-value: < 2.2e-16
```

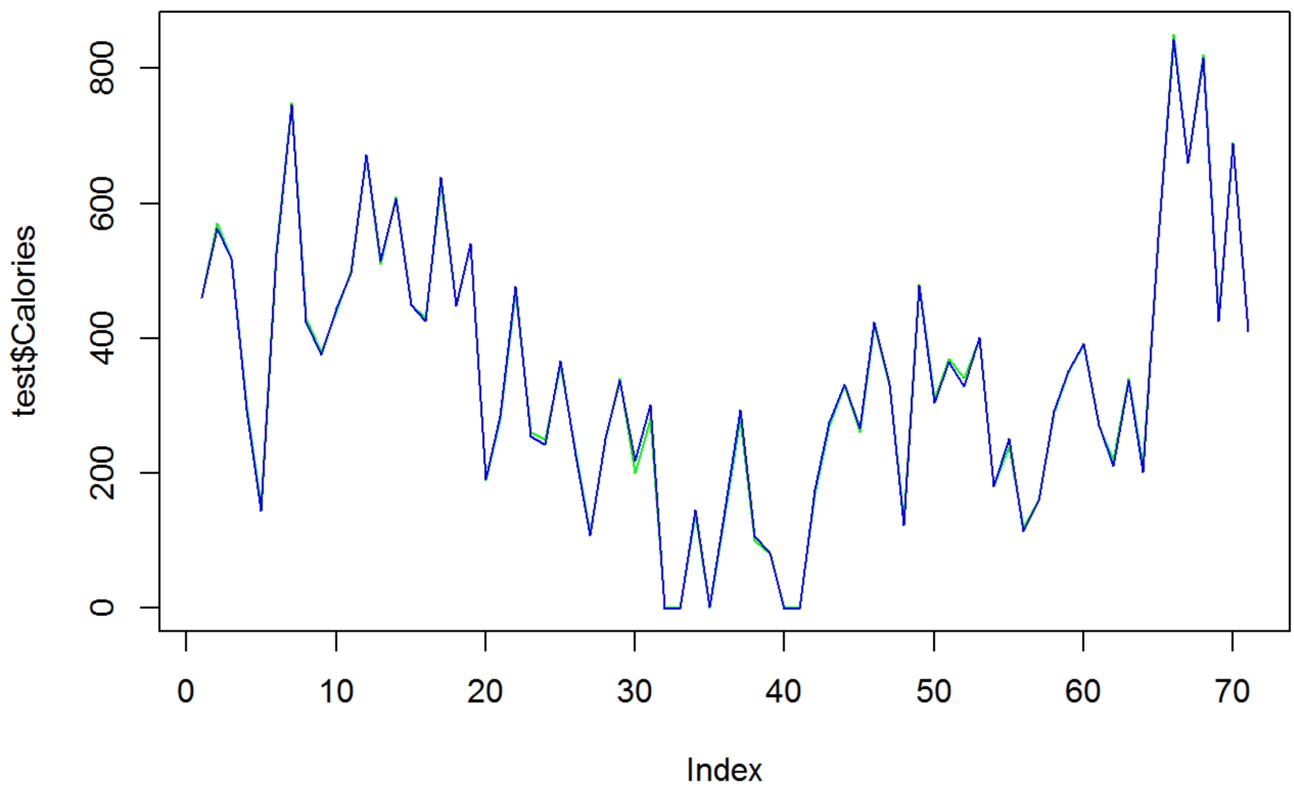
Predict

```r
predictions <- predict(model,test)
predictions
```

```
##          7         26         37         38         39         43 
## 460.615693 563.381981 518.137042 294.479066 143.381119 525.529606 
##         48         51         53         54         57         60 
## 746.256867 423.561177 375.448064 444.612751 496.536618 671.606456 
##         61         62         63         66         71         74 
## 514.501472 606.449389 449.344405 425.344977 638.581241 448.401825 
##         77         79         80         81         92         94 
## 541.169281 190.579188 284.680027 476.889324 254.211862 242.262828 
##         95         97        100        104        109        112 
## 366.514854 225.191338 107.266256 251.214792 337.633624 217.435506 
##        113        115        117        119        126        127 
## 301.078741  -1.630110  -1.630110 145.741304   2.377509 145.741304 
##        129        130        133        138        140        149 
## 293.112719 105.911192  82.013125  -1.630110  -1.630110 175.442074 
##        155        156        158        160        163        165 
## 275.017354 331.891421 267.051332 423.866873 332.257615 122.138535 
##        187        189        190        195        196        201 
## 478.858707 305.118074 364.961675 329.040749 400.857992 181.085844 
##        208        210        211        212        213        217 
## 250.947030 113.374654 159.337772 293.050177 350.962329 391.442651 
##        222        233        235        236        245        250 
## 270.791189 210.050047 338.086968 202.108632 549.391782 843.779699 
##        251        252        255        257        260 
## 660.170381 815.874013 424.781917 688.343830 409.937174
```

Now, lets compare actual values and predicted values

```
plot(test$Calories,type = "l",lty = 1.8, col="green")
lines(predictions,type = "l", col = "blue")
```



almost 100% accurate.(Lines overlap)

Future Predictions

Say, for values of Total fat = 20, Protein = 18 & Carbohydatres = 33

```
predict(model,data.frame(Total.Fat = 20,Protein = 18,Carbohydrates = 33))
```

```
##        1
## 382.5207
```

We get, calories = 382.5207