$$X\hat{\beta} = HY = H_{(J)}Y + (I - H_{(J)})X_J\hat{\beta}_J$$

$$\hat{e}^{(J)} = Y^{(J)} - X^{(J)}b^{(J)} = Y^{(J)} - X^{(J)}\hat{\beta}_J$$

$$= (I - H_J)Y - (I - H_J)X_J\hat{\beta}_J$$

$$\hat{e} = Y - HY = Y - (H_{(J)}Y + (I - H_{(J)})X_J\hat{\beta}_J)$$

$$= (I - H_{(J)})Y - (I - H_{(J)})X_J\hat{\beta}_J$$

$Y^{(J)} = Y - H_{(J)}Y = (I - H_{(J)})Y$, $\quad X^{(J)} = X_J - H_{(J)}X_J = (I - H_{(J)})X_J$

$$\hat{e}^{(J)} = Y^{(J)} - X^{(J)}b^{(J)} = Y^{(J)} - X^{(J)}\hat{\beta}_J = (I - H_{(J)})Y - (I - H_J)X_J\hat{\beta}_J$$

$$\hat{e} = Y - HY = Y - (H_{(J)}Y + (I - H_{(J)})X_J\hat{\beta}_J)$$

$$= (I - H_{(J)})Y - (I - H_{(J)})X_J\hat{\beta}_J.$$

151AHW4

Jiyoon Clover Jeong

10/26/2017

Problem 2 - (a)

```
bodyfat <- read.csv("/Users/cloverjiyoon/2017Fall/Stat 151A/Lab/Lab3/bodyfat.csv")

bodyfat$Density <- NULL

names(bodyfat)
```

```
##  [1] "bodyfat" "Age"     "Weight"  "Height"  "Neck"    "Chest"   "Abdomen"
##  [8] "Hip"     "Thigh"   "Knee"    "Ankle"   "Biceps"  "Forearm" "Wrist"
```

Backward elimination using the individual p-values.

```
dat1 <- bodyfat
n <- ncol(dat1)-1


for(i in 1:n){
  fit <- summary(lm(bodyfat ~., data = dat1))
  maxval <- max(fit$coefficients[-1,4])
  maxindex <- which.max(fit$coefficients[-1,4])
  # deleting intercept, getting pval column from summary
  if(maxval > 0.05){
      dat1 <- dat1[,-(maxindex + 1)]
  }
  else
    break
}

model1 <- dat1

summary(lm(bodyfat ~., data = model1))
```

```
##
## Call:
## lm(formula = bodyfat ~ ., data = model1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.5626  -3.1235  -0.1461   3.1313   9.0867
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -34.85407    7.24500  -4.811 2.62e-06 ***
## Weight       -0.13563    0.02475  -5.480 1.05e-07 ***
## Abdomen       0.99575    0.05607  17.760  < 2e-16 ***
## Forearm       0.47293    0.18166   2.603 0.009790 **
## Wrist        -1.50556    0.44267  -3.401 0.000783 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.343 on 247 degrees of freedom
## Multiple R-squared:  0.735,  Adjusted R-squared:  0.7307
## F-statistic: 171.3 on 4 and 247 DF,  p-value: < 2.2e-16
```

```
# Check with function in R
SignifReg(bodyfat ~., data = bodyfat, alpha = 0.05, direction = "backward", criterion = "p-value")
```

```
##
## Call:
## lm(formula = reg, data = data)
##
## Coefficients:
## (Intercept)       Weight      Abdomen      Forearm        Wrist
##    -34.8541      -0.1356       0.9958       0.4729      -1.5056
```

Forward Selection using p-values.

```
dat1 <- bodyfat
formula <- c("bodyfat ~ 1")
predictors <- c()
plist <- as.vector(names(dat1[,-1]))
plist
```

```
##  [1] "Age"     "Weight"  "Height"  "Neck"    "Chest"   "Abdomen" "Hip"
##  [8] "Thigh"   "Knee"    "Ankle"   "Biceps"  "Forearm" "Wrist"
```

```
for(i in 1:n){

  minindex <- 0
  minval <- 10^3

  for(j in 1:length(plist)){

    fit <- summary(lm( paste(formula, plist[j], sep = "+"),
                       data = dat1))
    if(fit$coefficients[plist[j], 4] < minval){
      minval <- fit$coefficients[plist[j], 4]
      minindex <- j
    }
  }

    # deleting intercept, getting pval column from summary
    if(minval > 0.05){
        break
    }

    else{
      formula <- paste(formula, plist[minindex], sep ="+")
      recovered_minindex <- match(plist[minindex], names(bodyfat))
      plist <- plist[-minindex]
      # recover to original index from plist
      predictors <- append(predictors, recovered_minindex)

    }
}


plist
```

```
## [1] "Age"    "Height" "Neck"   "Chest"  "Hip"    "Thigh"  "Knee"   "Ankle" 
## [9] "Biceps"
```

```
formula
```

```
## [1] "bodyfat ~ 1+Abdomen+Weight+Wrist+Forearm"
```

```
predictors
```

```
## [1]  7  3 14 13
```

```
model2 <- bodyfat[,c(1,predictors)]

summary(lm(bodyfat ~., data = model2))
```

```
## 
## Call:
## lm(formula = bodyfat ~ ., data = model2)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -10.5626  -3.1235  -0.1461   3.1313   9.0867 
## 
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)    
## (Intercept) -34.85407    7.24500  -4.811 2.62e-06 ***
## Abdomen       0.99575    0.05607  17.760  < 2e-16 ***
## Weight       -0.13563    0.02475  -5.480 1.05e-07 ***
## Wrist        -1.50556    0.44267  -3.401 0.000783 ***
## Forearm       0.47293    0.18166   2.603 0.009790 ** 
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 4.343 on 247 degrees of freedom
## Multiple R-squared:  0.735,  Adjusted R-squared:  0.7307 
## F-statistic: 171.3 on 4 and 247 DF,  p-value: < 2.2e-16
```

```
SignifReg(bodyfat ~., data = bodyfat, alpha = 0.05, direction = "forward", criterion = "p-value")
```

```
## 
## Call:
## lm(formula = reg, data = data)
## 
## Coefficients:
## (Intercept)      Abdomen       Weight        Wrist      Forearm  
##    -34.8541       0.9958      -0.1356      -1.5056       0.4729
```

Adjusted R2

```r
formula <- c("bodyfat ~ 1")
predictors <- c()
plist <- as.vector(names(dat1[,-1]))
plist
```

```
##  [1] "Age"     "Weight"  "Height"  "Neck"    "Chest"   "Abdomen" "Hip"
##  [8] "Thigh"   "Knee"    "Ankle"   "Biceps"  "Forearm" "Wrist"
```

```r
for(i in 1:n){

  prevR2 <-  summary(lm( formula, data = dat1))$adj.r.squared
  maxindex <- 0
  maxval <- 0

  for(j in 1:length(plist)){

    fit <- summary(lm( paste(formula, plist[j], sep = "+"),
                       data = dat1))
    fit
    if(fit$adj.r.squared > maxval){
      maxval <- fit$adj.r.squared
      maxindex <- j
    }
  }

    # deleting intercept, getting pval column from summary
    if(maxval < prevR2){
        break
    }

    else{
      formula <- paste(formula, plist[maxindex], sep ="+")

      recovered_maxindex <- match(plist[maxindex], names(bodyfat))
      plist <- plist[-maxindex]
      # recover to original index from plist
      predictors <- append(predictors, recovered_maxindex)

    }
}


plist
```

```
## [1] "Height" "Chest"  "Knee"   "Ankle"
```

```r
formula
```

```
## [1] "bodyfat ~ 1+Abdomen+Weight+Wrist+Forearm+Neck+Age+Thigh+Hip+Biceps"
```

```
predictors
```

```
## [1]  7  3 14 13  5  2  9  8 12
```

```
model3 <- bodyfat[,c(1,predictors)]

summary(lm(bodyfat ~., data = model3))
```

```
##
## Call:
## lm(formula = bodyfat ~ ., data = model3)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.9952  -2.8631  -0.1004   3.0810  10.0148
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -23.30499   11.72660  -1.987  0.04801 *
## Abdomen       0.94926    0.07204  13.177  < 2e-16 ***
## Weight       -0.09843    0.04070  -2.418  0.01634 *
## Wrist        -1.54208    0.50928  -3.028  0.00273 **
## Forearm       0.45150    0.19581   2.306  0.02197 *
## Neck         -0.49330    0.22596  -2.183  0.02999 *
## Age           0.06348    0.03084   2.058  0.04064 *
## Thigh         0.26538    0.13362   1.986  0.04816 *
## Hip          -0.18287    0.13893  -1.316  0.18934
## Biceps        0.17889    0.16827   1.063  0.28878
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.281 on 242 degrees of freedom
## Multiple R-squared:  0.7477, Adjusted R-squared:  0.7384
## F-statistic:  79.7 on 9 and 242 DF,  p-value: < 2.2e-16
```

AIC (pick smallest)

```
AIC <- c()


subsets <- summary(regsubsets(bodyfat~., bodyfat, nvmax = 13,  method ="forward"))

m <- nrow(bodyfat)

for(i in 1:n){

  index <- as.numeric(subsets$which[i,])
  index <- which(index %in% c(1))
  index
  dat1 <- bodyfat[, index]
  head(dat1)
  residual <- residuals(summary(lm(bodyfat ~., data = dat1)))
  RSS_m <- sum((residual)^2)
  RSS_m
  AIC[i] <- m * log(RSS_m / m) + 2 * (1+ (length(index) -1))
  AIC[i]
}

which.min(AIC)
```

```
## [1] 8
```

```
AIC
```

```
##  [1] 800.6453 756.0398 749.8962 745.0747 744.2968 743.6612 741.9088
##  [8] 741.8514 742.6772 743.9212 745.4263 747.3616 749.3574
```

```
index <- as.numeric(subsets$which[which.min(AIC),])
index <- which(index %in% c(1))
index
```

```
## [1]  1  2  3  5  7  8  9 13 14
```

```
model4 <- bodyfat[, index]

summary(lm(bodyfat~., data = model4))
```

```
##
## Call:
## lm(formula = bodyfat ~ ., data = model4)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.9757  -2.9937  -0.1644   2.9766  10.2244
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -22.65637   11.71385  -1.934  0.05426 .
## Age           0.06578    0.03078   2.137  0.03356 *
## Weight       -0.08985    0.03991  -2.252  0.02524 *
## Neck         -0.46656    0.22462  -2.077  0.03884 *
## Abdomen       0.94482    0.07193  13.134  < 2e-16 ***
## Hip          -0.19543    0.13847  -1.411  0.15940
## Thigh         0.30239    0.12904   2.343  0.01992 *
## Forearm       0.51572    0.18631   2.768  0.00607 **
## Wrist        -1.53665    0.50939  -3.017  0.00283 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.282 on 243 degrees of freedom
## Multiple R-squared:  0.7466, Adjusted R-squared:  0.7382
## F-statistic: 89.47 on 8 and 243 DF,  p-value: < 2.2e-16
```

BIC

```
BIC <- c()
```

```
subsets <- summary(regsubsets(bodyfat~., bodyfat, nvmax = 13,  method ="forward"))

m <- nrow(bodyfat)

for(i in 1:n){

  index <- as.numeric(subsets$which[i,])
  index <- which(index %in% c(1))
  index
  dat1 <- bodyfat[, index]
  head(dat1)
  residual <- residuals(summary(lm(bodyfat ~., data = dat1)))
  RSS_m <- sum((residual)^2)
  RSS_m
  BIC[i] <- m * log(RSS_m / m) + log(m) * (1+ (length(index) -1))
  BIC[i]
}

which.min(BIC)
```

```
## [1] 4
```

```
BIC
```

```
##  [1] 807.7042 766.6280 764.0139 762.7218 765.4734 768.3672 770.1442
##  [8] 773.6162 777.9714 782.7450 787.7794 793.2442 798.7694
```

```
index <- as.numeric(subsets$which[which.min(BIC),])
index <- which(index %in% c(1))
index
```

```
## [1]  1  3  7 13 14
```

```
model5 <- bodyfat[, index]

summary(lm(bodyfat~., data = model5))
```

```
## 
## Call:
## lm(formula = bodyfat ~ ., data = model5)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -10.5626  -3.1235  -0.1461   3.1313   9.0867 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) -34.85407    7.24500  -4.811 2.62e-06 ***
## Weight       -0.13563    0.02475  -5.480 1.05e-07 ***
## Abdomen       0.99575    0.05607  17.760  < 2e-16 ***
## Forearm       0.47293    0.18166   2.603 0.009790 ** 
## Wrist        -1.50556    0.44267  -3.401 0.000783 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 4.343 on 247 degrees of freedom
## Multiple R-squared:  0.735,  Adjusted R-squared:  0.7307
## F-statistic: 171.3 on 4 and 247 DF,  p-value: < 2.2e-16
```

Mallow's C_p

```
C_p <- c()

#
# subsets <- summary(regsubsets(bodyfat~., bodyfat, nvmax = 13,  method ="forward"))
#
# m <- nrow(bodyfat)

for(i in 1:n){

  index <- as.numeric(subsets$which[i,])
  index <- which(index %in% c(1))
  index
  dat1 <- bodyfat[, index]
  head(dat1)
  residual <- residuals(summary(lm(bodyfat ~., data = dat1)))
  RSS_m <- sum((residual)^2)
  RSS_m

  residual <- residuals(summary(lm(bodyfat ~., data = bodyfat)))
  RSS <- sum(residual^2)

  sigmasq <- RSS / (m - ncol(bodyfat) - 1 - 1)

  C_p[i] <- ( RSS_m / sigmasq ) - (m - 2 - 2 * (length(index)))
  C_p[i]
}

which.min(C_p)
```

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

```
C_p
```

```
##  [1] 72.172460 20.449648 14.040372  9.202446  8.470539  7.906195  6.301203
##  [8]  6.347254  7.239242  8.528676 10.064564 12.003955 14.000000
```

```
index <- as.numeric(subsets$which[which.min(C_p),])
index <- which(index %in% c(1))
index
```

```
## [1]  1  2  3  5  7  9 13 14
```

```
model6 <- bodyfat[, index]

summary(lm(bodyfat~., data = model6))
```

```
##
## Call:
## lm(formula = bodyfat ~ ., data = model6)
##
```

```
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -10.936  -3.046  -0.112   3.168   9.705 
## 
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)    
## (Intercept) -33.25799    9.00681  -3.693 0.000274 ***
## Age           0.06817    0.03079   2.214 0.027769 *  
## Weight       -0.11944    0.03403  -3.510 0.000533 ***
## Neck         -0.40380    0.22062  -1.830 0.068424 .  
## Abdomen       0.91788    0.06950  13.207  < 2e-16 ***
## Thigh         0.22196    0.11601   1.913 0.056888 .  
## Forearm       0.55314    0.18479   2.993 0.003043 ** 
## Wrist        -1.53240    0.51041  -3.002 0.002958 ** 
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 4.291 on 244 degrees of freedom
## Multiple R-squared:  0.7445, Adjusted R-squared:  0.7371 
## F-statistic: 101.6 on 7 and 244 DF,  p-value: < 2.2e-16
```

Part (b)

```
set.seed(10)

folds <- createFolds(bodyfat[,1], 10)
#folds

models <- list(model1,model2,model3,model4,model5,model6)
MSE <- matrix(0,6,10)

for(i in 1:6){
  for(j in 1:10){
    train <- models[[i]][-folds[[j]], ]
    test <- models[[i]][folds[[j]], ]
    fit <- lm(bodyfat ~., data  = train)
    predicted <- predict(fit, newdata = test, type = "response")
    mse <- mean((test[,1] - predicted)^2)
    mse
    MSE[i,j] <- mse

  }
}

MSE
```

```
##          [,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7]
## [1,] 16.89527 20.95376 22.41013 20.52916 28.20769 14.97150 16.69632
## [2,] 16.89527 20.95376 22.41013 20.52916 28.20769 14.97150 16.69632
## [3,] 17.33394 21.41520 20.47559 20.10821 25.96780 14.80751 15.62805
## [4,] 17.65496 21.16361 20.79752 19.83873 26.34355 13.60248 16.21875
## [5,] 16.89527 20.95376 22.41013 20.52916 28.20769 14.97150 16.69632
## [6,] 17.23644 20.99117 21.59482 19.77463 27.35397 13.92887 16.98683
##          [,8]     [,9]    [,10]
## [1,] 21.35559 18.07839 18.09763
## [2,] 21.35559 18.07839 18.09763
## [3,] 22.62127 19.66936 17.04951
## [4,] 21.97116 20.12167 16.50109
## [5,] 21.35559 18.07839 18.09763
## [6,] 22.11158 18.91631 16.82921
```

```
rowSums(MSE)
```

```
## [1] 198.1954 198.1954 195.0764 194.2135 198.1954 195.7238
```

```
finalmodel <- which.min(rowSums(MSE))

cat("Choose model", finalmodel, "\n")
```

```
## Choose model 4
```

Part (c)

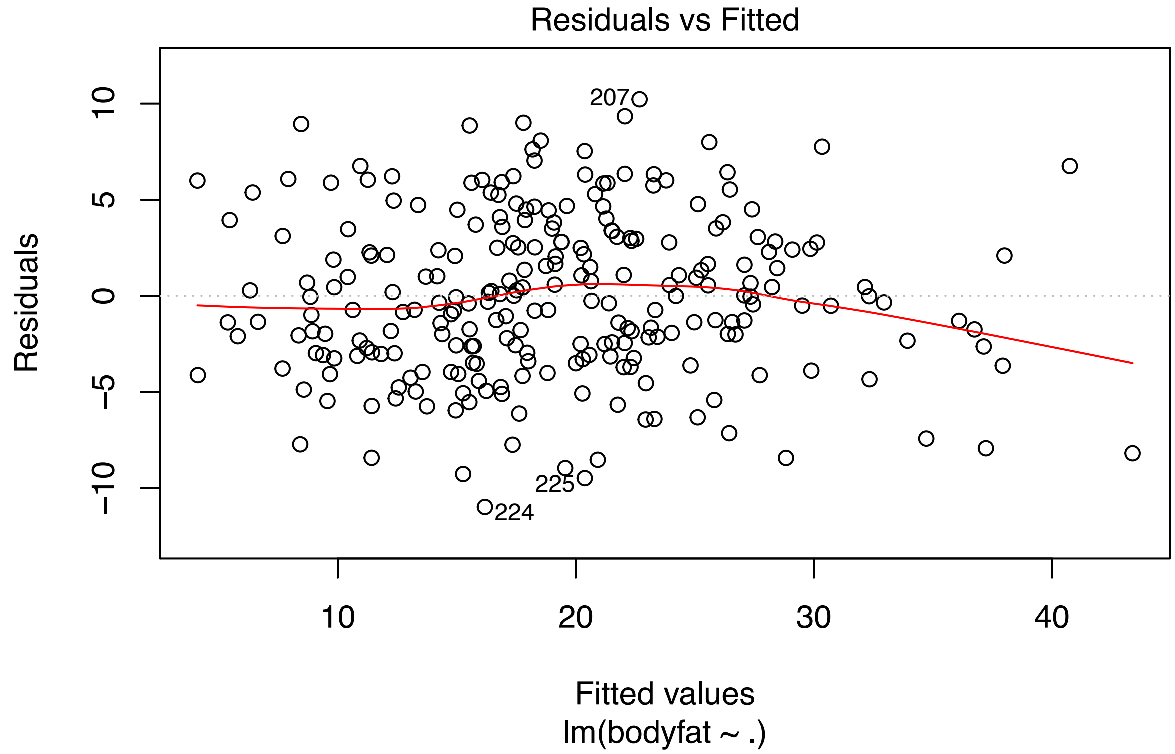
Fit this model to the data.

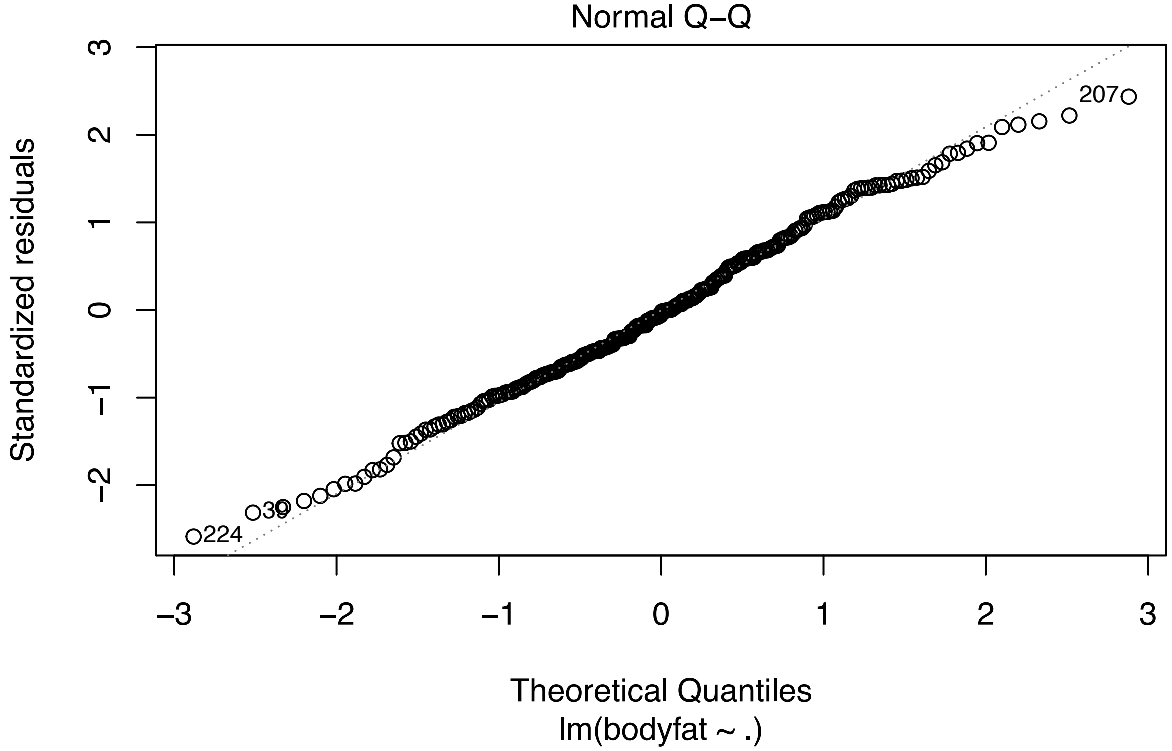
```
X <- models[[finalmodel]]

fit <- lm(bodyfat ~., data = X)
summary(fit)
```

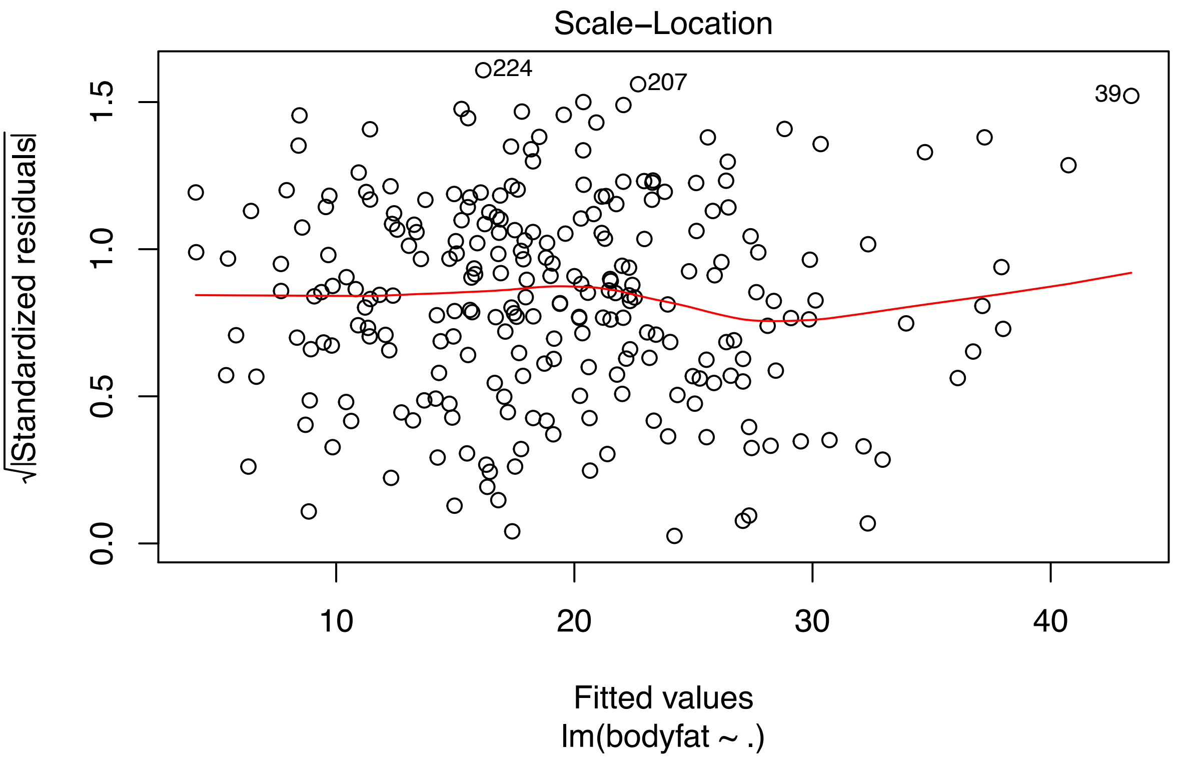
```
##
## Call:
## lm(formula = bodyfat ~ ., data = X)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.9757  -2.9937  -0.1644   2.9766  10.2244
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -22.65637   11.71385  -1.934  0.05426 .
## Age           0.06578    0.03078   2.137  0.03356 *
## Weight       -0.08985    0.03991  -2.252  0.02524 *
## Neck         -0.46656    0.22462  -2.077  0.03884 *
## Abdomen       0.94482    0.07193  13.134  < 2e-16 ***
## Hip          -0.19543    0.13847  -1.411  0.15940
## Thigh         0.30239    0.12904   2.343  0.01992 *
## Forearm       0.51572    0.18631   2.768  0.00607 **
## Wrist        -1.53665    0.50939  -3.017  0.00283 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.282 on 243 degrees of freedom
## Multiple R-squared:  0.7466, Adjusted R-squared:  0.7382
## F-statistic: 89.47 on 8 and 243 DF,  p-value: < 2.2e-16
```

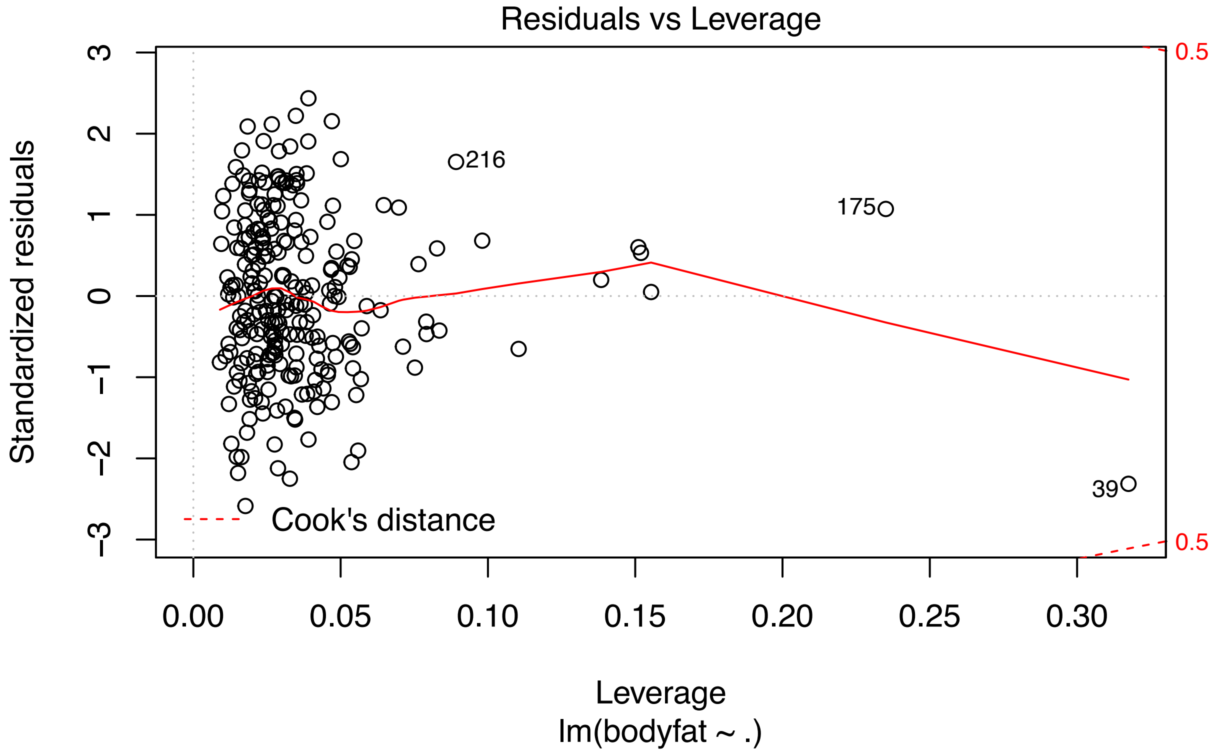
Perform regression diagnostics. <— further

```r
#par(mfrow = c(2, 2))
plot(fit)
```

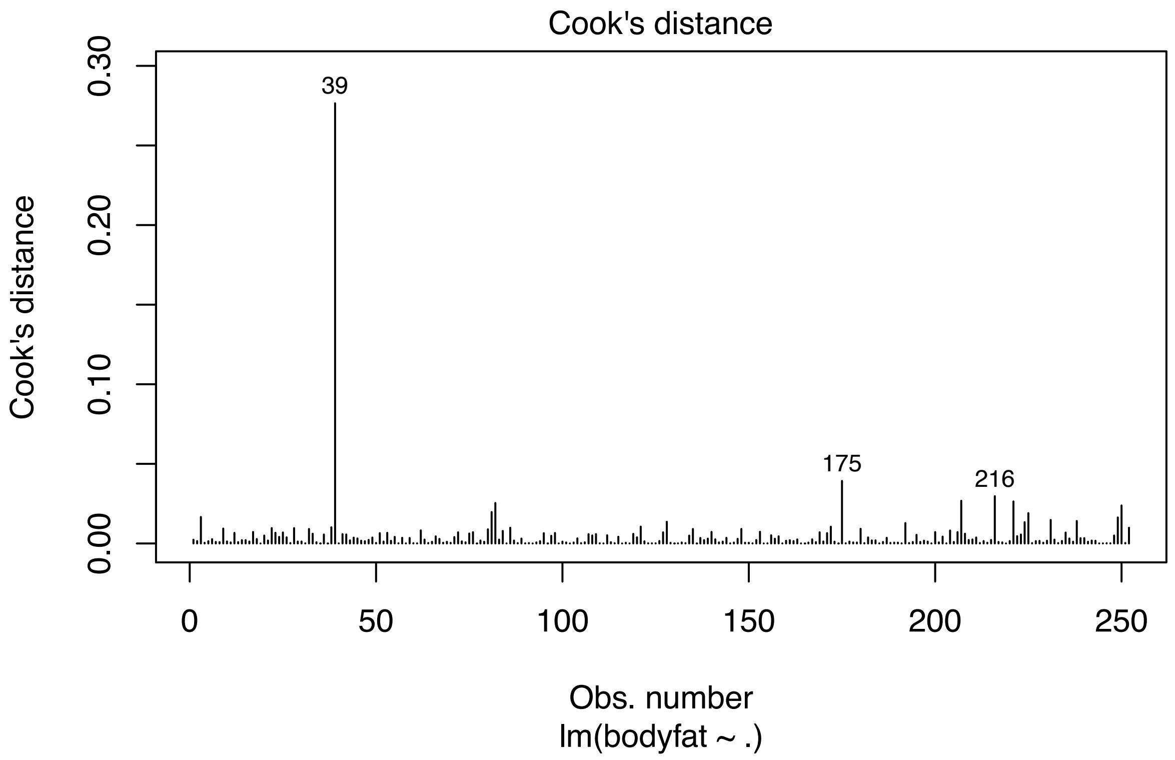




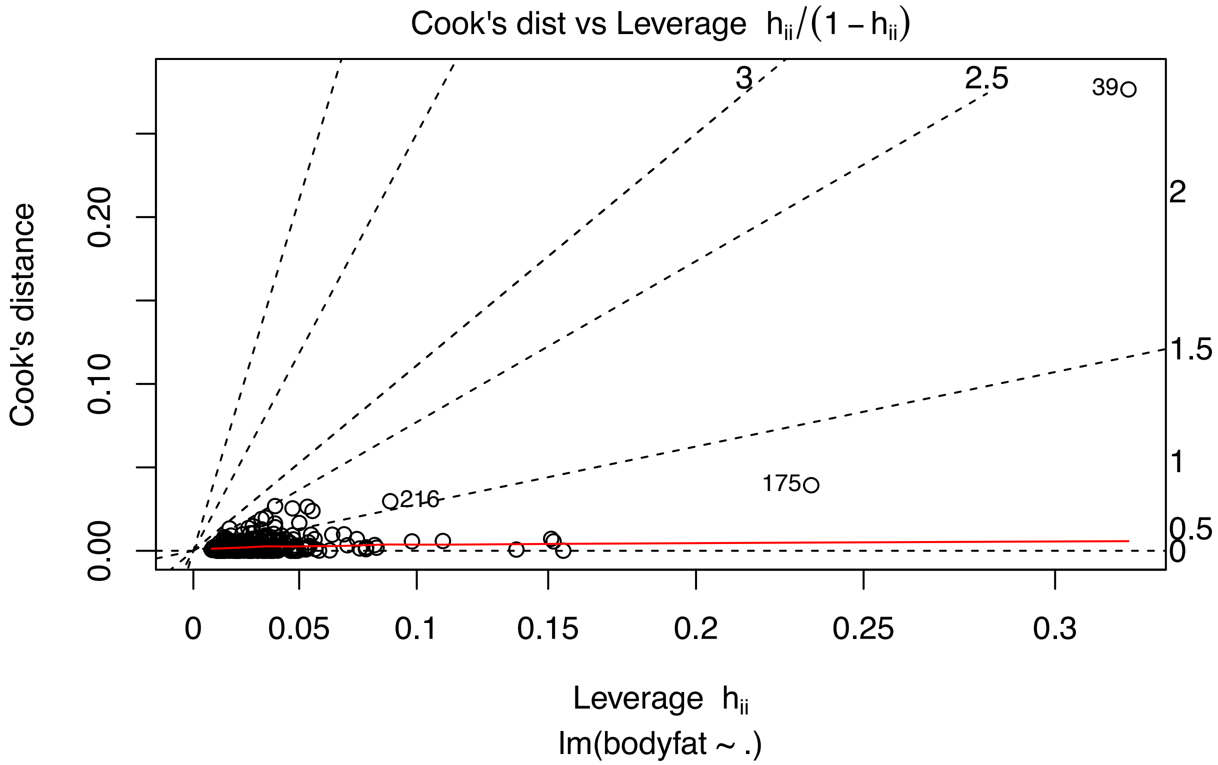




```r
plot(fit, which = 4)
```



```r
plot(fit, which = 6)
```



Comment on the validity of the assumptions of the linear model.

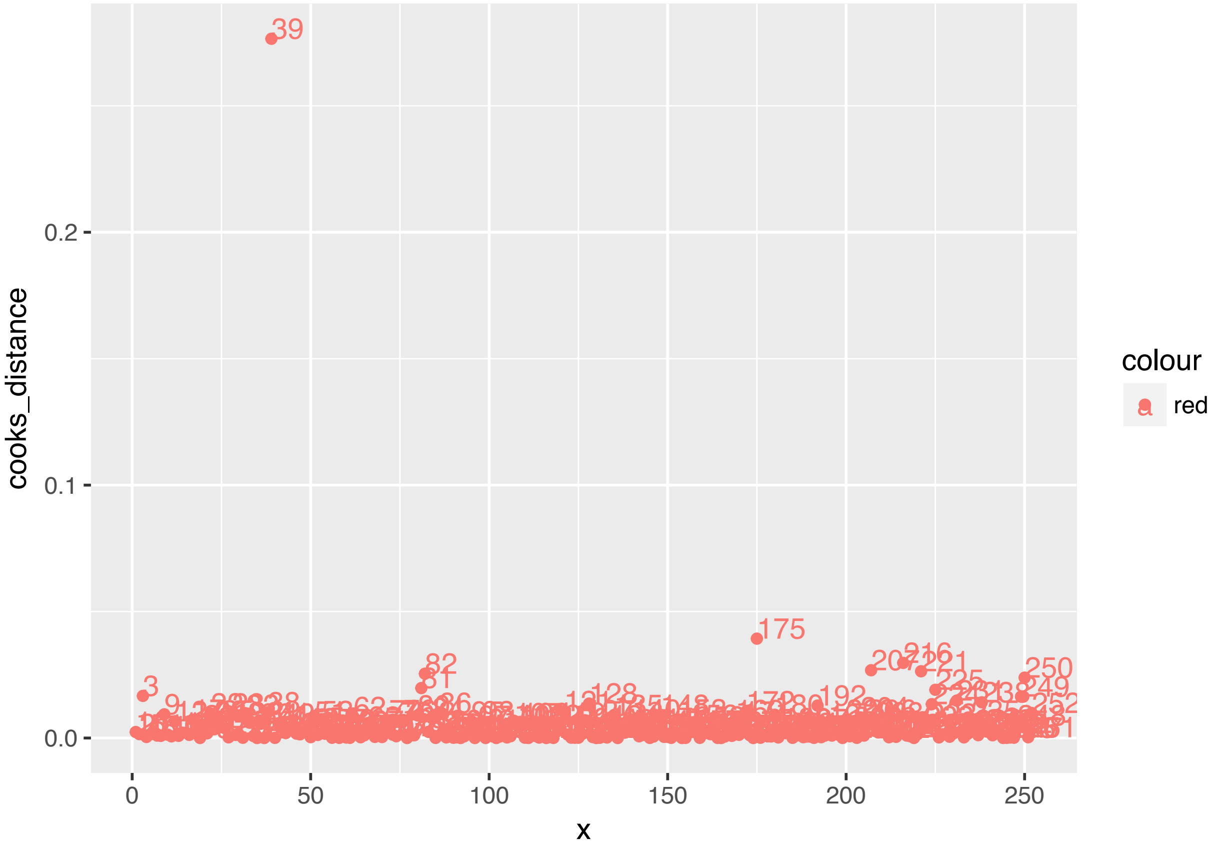
1. The Residual VS Fitted plot does not suggests any violations of the assumptions of the linear model.
2. The QQ plot of standardized residual shows that the standardized residuals are lighter tailed than normal(right most side).
3. The square root of standardized residual vs fitted plot and standardized residual vs leverage plot both suggests 39th observation as a potential outlier. From each plots, potential influential points are 39, 216, 175 207.

Identify influential observations and outliers.

```r
cooks <- data.frame(x = 1:nrow(bodyfat), cooks_distance = cooks.distance(fit))
cooks
```

```
##       x cooks_distance
## 1     1   2.364167e-03
## 2     2   1.526729e-03
## 3     3   1.664947e-02
## 4     4   4.083481e-04
## 5     5   1.421667e-03
## 6     6   2.693506e-03
## 7     7   9.661468e-04
## 8     8   8.377381e-04
## 9     9   9.372911e-03
## 10   10   1.292120e-03
## 11   11   6.513376e-04
## 12   12   6.633579e-03
## 13   13   8.515479e-04
## 14   14   2.102149e-03
## 15   15   2.096031e-03
## 16   16   1.239024e-03
## 17   17   7.235584e-03
## 18   18   2.789331e-03
## 19   19   1.274555e-05
## 20   20   4.978619e-03
## 21   21   1.851076e-03
## 22   22   9.670456e-03
## 23   23   6.724229e-03
## 24   24   4.113606e-03
## 25   25   6.900896e-03
## 26   26   3.863250e-03
## 27   27   2.627048e-04
## 28   28   9.628546e-03
## 29   29   1.224318e-03
## 30   30   1.238006e-03
## 31   31   1.006166e-04
## 32   32   9.109520e-03
## 33   33   6.113971e-03
## 34   34   7.628641e-04
## 35   35   6.770216e-08
## 36   36   5.628227e-03
## 37   37   7.449653e-10
## 38   38   1.015154e-02
## 39   39   2.765129e-01
## 40   40   5.087549e-05
## 41   41   5.855958e-03
## 42   42   5.622067e-03
## 43   43   1.936906e-03
## 44   44   3.673381e-03
## 45   45   3.151626e-03
## 46   46   1.740152e-03
## 47   47   1.441752e-03
## 48   48   2.239302e-03
## 49   49   3.722960e-03
## 50   50   3.389810e-04
## 51   51   6.501882e-03
## 52   52   1.213992e-03
## 53   53   6.666854e-03
## 54   54   1.692122e-03
## 55   55   4.114266e-03
## 56   56   9.948848e-05
## 57   57   3.552351e-03
## 58   58   5.529763e-05
## 59   59   3.449879e-03
## 60   60   1.843322e-04
## 61   61   7.947343e-05
## 62   62   8.228248e-03
## 63   63   2.442216e-03
## 64   64   2.998416e-04
## 65   65   1.060079e-03
## 66   66   4.429492e-03
## 67   67   2.860205e-03
## 68   68   4.475396e-04
## 69   69   1.054798e-03
## 70   70   3.765811e-04
## 71   71   4.016979e-03
## 72   72   7.017881e-03
## 73   73   1.315259e-03
## 74   74   7.037372e-04
## 75   75   6.256613e-03
## 76   76   7.100921e-03
## 77   77   8.054837e-07
## 78   78   1.880014e-03
## 79   79   1.065853e-03
## 80   80   8.896035e-03
## 81   81   1.974411e-02
## 82   82   2.542917e-02
## 83   83   2.440340e-03
## 84   84   7.828187e-03
## 85   85   4.842789e-05
## 86   86   9.892586e-03
## 87   87   1.860218e-03
## 88   88   2.306735e-04
## 89   89   3.082626e-03
## 90   90   9.397929e-05
## 91   91   2.587132e-04
## 92   92   1.503222e-05
## 93   93   6.708160e-04
## 94   94   1.464967e-03
## 95   95   6.464396e-03
## 96   96   1.601746e-08
## 97   97   4.808909e-03
## 98   98   6.586119e-03
## 99   99   1.091283e-05
## 100 100   1.175537e-03
## 101 101   6.362355e-04
## 102 102   1.141613e-05
## 103 103   5.908642e-04
## 104 104   3.180033e-03
## 105 105   2.275734e-04
## 106 106   7.073354e-04
## 107 107   5.843691e-03
## 108 108   5.045636e-03
## 109 109   5.861628e-03
## 110 110   1.257370e-04
## 111 111   3.062853e-05
## 112 112   5.129220e-03
## 113 113   8.035874e-04
## 114 114   1.747324e-04
## 115 115   4.211552e-03
## 116 116   8.342084e-06
## 117 117   4.391209e-04
## 118 118   2.797109e-05
## 119 119   6.008280e-03
## 120 120   3.881678e-03
## 121 121   1.059735e-02
## 122 122   1.360455e-03
## 123 123   1.230340e-04
## 124 124   1.108878e-04
## 125 125   1.179337e-04
## 126 126   1.553175e-03
## 127 127   6.962576e-03
## 128 128   1.361423e-02
## 129 129   3.854700e-04
## 130 130   4.128409e-07
## 131 131   9.463733e-05
## 132 132   6.333659e-04
## 133 133   2.024519e-04
## 134 134   4.955740e-03
## 135 135   9.100646e-03
## 136 136   8.739156e-08
## 137 137   3.439208e-03
## 138 138   2.085187e-03
## 139 139   3.058756e-03
## 140 140   7.249692e-03
## 141 141   2.578300e-03
## 142 142   6.723485e-04
## 143 143   1.192140e-03
## 144 144   3.520538e-03
## 145 145   4.569892e-05
## 146 146   5.642731e-04
## 147 147   2.895374e-03
## 148 148   9.114998e-03
## 149 149   4.581916e-04
## 150 150   4.395181e-04
## 151 151   6.794464e-05
## 152 152   2.163108e-03
## 153 153   7.340747e-03
## 154 154   6.049962e-06
## 155 155   4.605740e-05
## 156 156   5.138691e-03
## 157 157   2.959020e-03
## 158 158   4.497585e-03
## 159 159   5.043830e-05
## 160 160   1.692355e-03
## 161 161   2.024505e-03
## 162 162   1.507229e-03
## 163 163   2.522643e-03
## 164 164   3.681169e-05
## 165 165   2.593806e-07
## 166 166   5.726458e-04
## 167 167   2.569393e-03
## 168 168   8.556699e-04
## 169 169   7.029681e-03
## 170 170   1.255838e-03
## 171 171   6.515301e-03
## 172 172   1.053902e-02
## 173 173   1.113881e-03
## 174 174   6.243808e-07
## 175 175   3.929327e-02
## 176 176   1.083367e-04
## 177 177   1.207213e-03
## 178 178   6.508058e-04
## 179 179   6.093266e-04
## 180 180   9.198226e-03
## 181 181   2.261096e-04
## 182 182   3.800863e-03
## 183 183   1.915410e-03
## 184 184   1.934518e-03
## 185 185   6.571781e-05
## 186 186   9.572006e-04
## 187 187   3.557137e-03
## 188 188   3.300668e-04
## 189 189   5.718055e-04
## 190 190   5.374476e-04
## 191 191   6.935471e-05
## 192 192   1.281732e-02
## 193 193   3.135170e-04
## 194 194   9.201691e-04
## 195 195   5.405657e-03
## 196 196   9.646240e-04
## 197 197   1.735367e-03
## 198 198   1.208176e-03
## 199 199   2.505092e-05
## 200 200   7.176647e-03
## 201 201   6.746528e-04
## 202 202   4.306502e-03
## 203 203   6.814874e-05
## 204 204   8.137408e-03
## 205 205   9.527199e-04
## 206 206   7.152569e-03
## 207 207   2.679949e-02
## 208 208   6.122391e-03
## 209 209   2.187127e-03
## 210 210   2.515552e-03
## 211 211   3.764951e-03
## 212 212   3.012379e-04
## 213 213   1.512792e-03
## 214 214   4.660647e-04
## 215 215   2.218045e-03
## 216 216   2.970340e-02
## 217 217   9.520330e-04
## 218 218   6.745687e-04
## 219 219   2.619282e-05
## 220 220   1.524586e-03
## 221 221   2.638681e-02
## 222 222   4.603787e-03
## 223 223   5.620190e-03
## 224 224   1.337889e-02
## 225 225   1.906400e-02
## 226 226   2.304379e-04
## 227 227   1.484936e-03
## 228 228   1.628613e-03
## 229 229   5.020696e-04
## 230 230   1.738382e-03
## 231 231   1.479903e-02
## 232 232   2.403779e-03
## 233 233   2.562853e-04
## 234 234   1.583933e-03
## 235 235   6.849540e-03
## 236 236   3.110148e-03
## 237 237   1.118509e-03
## 238 238   1.412350e-02
## 239 239   3.310033e-03
## 240 240   3.370111e-03
## 241 241   1.081911e-03
## 242 242   1.832047e-03
## 243 243   1.697986e-03
## 244 244   2.332689e-05
## 245 245   5.831284e-05
## 246 246   2.033929e-04
## 247 247   1.058370e-04
## 248 248   5.005359e-03
## 249 249   1.636252e-02
## 250 250   2.388011e-02
## 251 251   2.952064e-04
## 252 252   9.890195e-03
```

```
ggplot(cooks, aes(x = x, y= cooks_distance, colour="red"))+ geom_point()+ geom_text(aes(label=x),hjust=
```



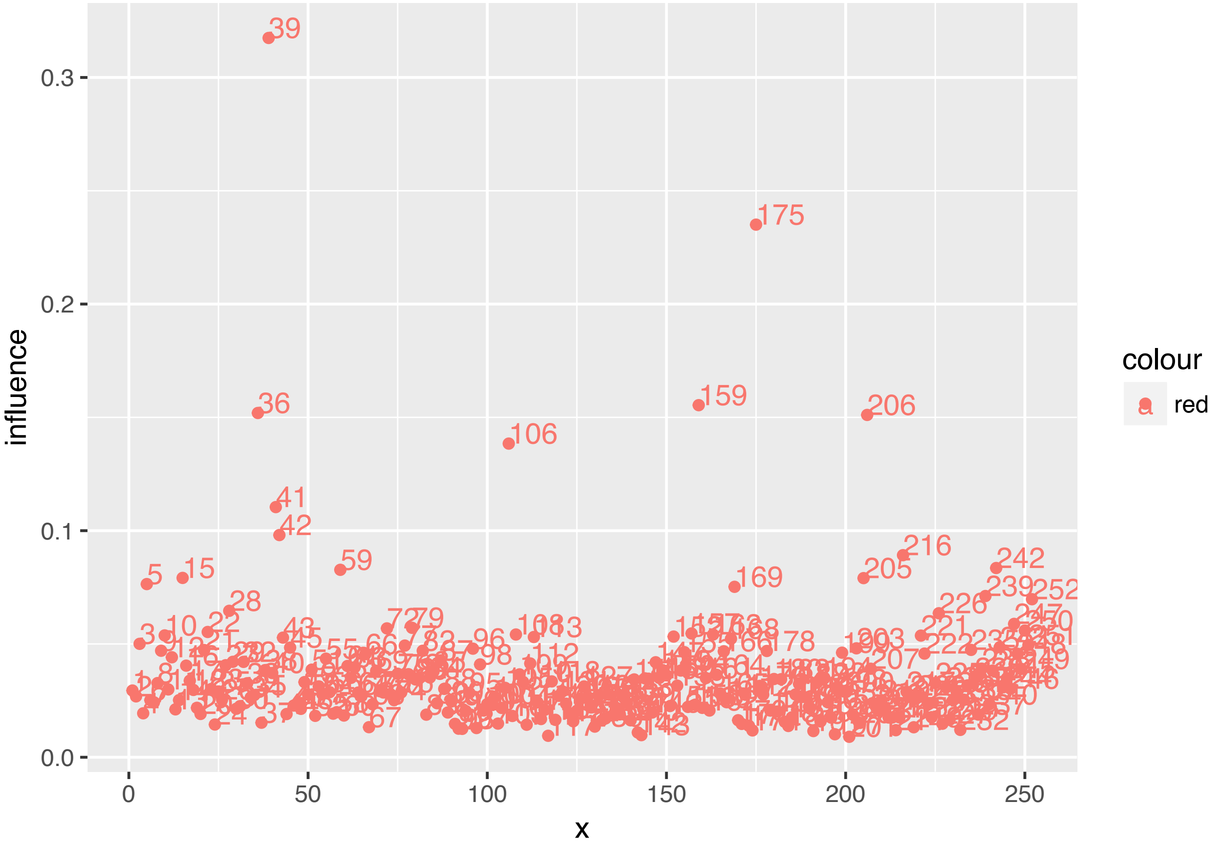
```
remove1 <- which(cooks[,2] %in% sort(cooks[,2], decreasing = T)[1:2])
remove1
```

```
## [1]  39 175
```

```
influ <- data.frame(x = 1:nrow(bodyfat), influence = influence(fit)$hat)
influ
```

```
##       x   influence
## 1     1 0.029454639
## 2     2 0.026879224
## 3     3 0.050062694
## 4     4 0.019406704
## 5     5 0.076429628
## 6     6 0.025216224
## 7     7 0.024254862
## 8     8 0.032729309
## 9     9 0.047014329
## 10   10 0.053772621
## 11   11 0.029936080
## 12   12 0.044121092
## 13   13 0.021137714
## 14   14 0.025210303
## 15   15 0.079144108
## 16   16 0.040429048
## 17   17 0.033804362
## 18   18 0.029727474
## 19   19 0.021822961
## 20   20 0.019113629
## 21   21 0.047341057
## 22   22 0.055291174
## 23   23 0.030114103
## 24   24 0.014453626
## 25   25 0.029045589
## 26   26 0.025495134
## 27   27 0.040622329
## 28   28 0.064626998
## 29   29 0.042115835
## 30   30 0.021408896
## 31   31 0.022542641
## 32   32 0.042087756
## 33   33 0.032669346
## 34   34 0.026367723
## 35   35 0.027462117
## 36   36 0.151934896
## 37   37 0.015252847
## 38   38 0.038462465
## 39   39 0.317434392
## 40   40 0.037184126
## 41   41 0.110410353
## 42   42 0.098063258
## 43   43 0.052789525
## 44   44 0.019068151
## 45   45 0.048390910
## 46   46 0.022818422
## 47   47 0.023389031
## 48   48 0.021361345
## 49   49 0.033164733
## 50   50 0.027764012
## 51   51 0.038654610
## 52   52 0.018286955
## 53   53 0.031273466
## 54   54 0.027826352
## 55   55 0.043503581
## 56   56 0.028726474
## 57   57 0.019236572
## 58   58 0.019885209
## 59   59 0.082729113
## 60   60 0.018415110
## 61   61 0.040283473
## 62   62 0.035050062
## 63   63 0.039718562
## 64   64 0.028561475
## 65   65 0.027594536
## 66   66 0.045507942
## 67   67 0.013276119
## 68   68 0.023370275
## 69   69 0.038144133
## 70   70 0.029158111
## 71   71 0.028581451
## 72   72 0.056872472
## 73   73 0.027843761
## 74   74 0.025188641
## 75   75 0.036864303
## 76   76 0.028598700
## 77   77 0.049295600
## 78   78 0.036668031
## 79   79 0.057130449
## 80   80 0.034329859
## 81   81 0.034796966
## 82   82 0.047012766
## 83   83 0.018803100
## 84   84 0.035207484
## 85   85 0.037850801
## 86   86 0.023850963
## 87   87 0.042759577
## 88   88 0.030183573
## 89   89 0.019758262
## 90   90 0.024630779
## 91   91 0.014742218
## 92   92 0.012612315
## 93   93 0.012523062
## 94   94 0.020407354
## 95   95 0.028411901
## 96   96 0.047921790
## 97   97 0.012905477
## 98   98 0.040977813
## 99   99 0.020610822
## 100 100 0.023475914
## 101 101 0.023817750
## 102 102 0.026638329
## 103 103 0.014863389
## 104 104 0.021781970
## 105 105 0.030630437
## 106 106 0.138427765
## 107 107 0.018226983
## 108 108 0.054177219
## 109 109 0.036578701
## 110 110 0.033239386
## 111 111 0.014364736
## 112 112 0.041392281
## 113 113 0.053067017
## 114 114 0.024210942
## 115 115 0.016874086
## 116 116 0.020912441
## 117 117 0.009477912
## 118 118 0.033472443
## 119 119 0.016519718
## 120 120 0.027136990
## 121 121 0.029077554
## 122 122 0.023627633
## 123 123 0.019439117
## 124 124 0.015892326
## 125 125 0.020499016
## 126 126 0.025834738
## 127 127 0.031255663
## 128 128 0.026635149
## 129 129 0.021962000
## 130 130 0.013534380
## 131 131 0.027359592
## 132 132 0.016243087
## 133 133 0.017145005
## 134 134 0.027613595
## 135 135 0.018429881
## 136 136 0.021853956
## 137 137 0.018922641
## 138 138 0.026482481
## 139 139 0.023892966
## 140 140 0.016304118
## 141 141 0.034352735
## 142 142 0.010940258
## 143 143 0.009764184
## 144 144 0.034832260
## 145 145 0.034659686
## 146 146 0.019727085
## 147 147 0.041931325
## 148 148 0.035021931
## 149 149 0.038436575
## 150 150 0.036117609
## 151 151 0.022609397
## 152 152 0.053259855
## 153 153 0.031595620
## 154 154 0.038269806
## 155 155 0.046389216
## 156 156 0.022227324
## 157 157 0.054630403
## 158 158 0.023192216
## 159 159 0.155385928
## 160 160 0.021807498
## 161 161 0.034942012
## 162 162 0.020628070
## 163 163 0.054163172
## 164 164 0.036385869
## 165 165 0.028158269
## 166 166 0.046761871
## 167 167 0.025893531
## 168 168 0.052286774
## 169 169 0.075213605
## 170 170 0.016342191
## 171 171 0.014721185
## 172 172 0.027594204
## 173 173 0.013881780
## 174 174 0.011847943
## 175 175 0.235057344
## 176 176 0.031503406
## 177 177 0.027678579
## 178 178 0.046899809
## 179 179 0.020601756
## 180 180 0.034539872
## 181 181 0.020182340
## 182 182 0.034436988
## 183 183 0.015643920
## 184 184 0.013812833
## 185 185 0.017605330
## 186 186 0.027705237
## 187 187 0.032342384
## 188 188 0.026658156
## 189 189 0.026432912
## 190 190 0.021597235
## 191 191 0.011567318
## 192 192 0.032862736
## 193 193 0.015822295
## 194 194 0.035162283
## 195 195 0.024285197
## 196 196 0.017470992
## 197 197 0.010166609
## 198 198 0.030027058
## 199 199 0.046146828
## 200 200 0.028815947
## 201 201 0.009032029
## 202 202 0.018800535
## 203 203 0.048084200
## 204 204 0.015176521
## 205 205 0.079060601
## 206 206 0.151075637
## 207 207 0.039062278
## 208 208 0.023308842
## 209 209 0.022042021
## 210 210 0.025188043
## 211 211 0.020949481
## 212 212 0.017535447
## 213 213 0.017541226
## 214 214 0.011951493
## 215 215 0.017617606
## 216 216 0.089186221
## 217 217 0.028890481
## 218 218 0.027151569
## 219 219 0.013252223
## 220 220 0.027220718
## 221 221 0.053691212
## 222 222 0.045754232
## 223 223 0.023606784
## 224 224 0.017683821
## 225 225 0.032770617
## 226 226 0.063556120
## 227 227 0.014802557
## 228 228 0.030799956
## 229 229 0.016531886
## 230 230 0.025200180
## 231 231 0.028731663
## 232 232 0.012078981
## 233 233 0.025409379
## 234 234 0.031717666
## 235 235 0.047391801
## 236 236 0.035003380
## 237 237 0.018836837
## 238 238 0.039084760
## 239 239 0.071157425
## 240 240 0.023295214
## 241 241 0.038227369
## 242 242 0.083520233
## 243 243 0.048565626
## 244 244 0.030910380
## 245 245 0.034898755
## 246 246 0.030237934
## 247 247 0.058854976
## 248 248 0.045656211
## 249 249 0.039011165
## 250 250 0.055934497
## 251 251 0.049575670
## 252 252 0.069745263
```

```
ggplot(influ, aes(x = x, y= influence, colour="red"))+ geom_point()+ geom_text(aes(label=x),hjust=0, vju
```



```
remove2 <- which(influ[,2] %in% sort(influ[,2], decreasing = T)[1:2])
remove2
```

```
## [1]  39 175
```

The potential influential observations and outliers can be 39th and 175th observation as we can see in the plots above.

Delete them if necessary and re-fit the model.

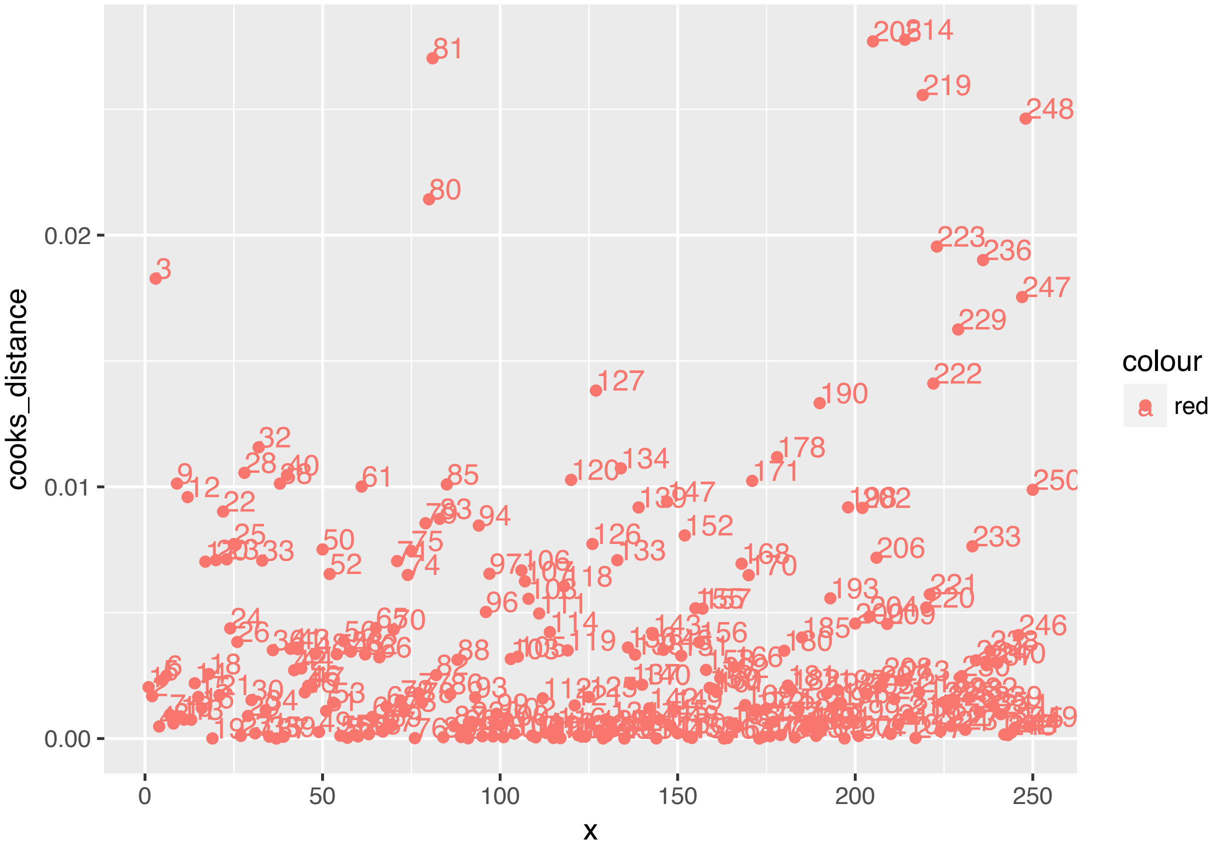
```
X_refit <- X[-unique(c(remove1, remove2)), ]

refit <- lm(bodyfat ~., data = X_refit)
summary(refit)
```

```
##
## Call:
## lm(formula = bodyfat ~ ., data = X_refit)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -10.765  -2.907  -0.280   2.902  10.185
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -23.61973   11.66629  -2.025 0.044010 *
## Age           0.07367    0.03090   2.384 0.017882 *
## Weight       -0.07655    0.04000  -1.914 0.056867 .
## Neck         -0.38378    0.22883  -1.677 0.094809 .
## Abdomen       0.91029    0.07296  12.476  < 2e-16 ***
## Hip          -0.13611    0.14051  -0.969 0.333664
## Thigh         0.27670    0.12854   2.153 0.032340 *
## Forearm       0.43052    0.22477   1.915 0.056631 .
## Wrist        -1.73658    0.51979  -3.341 0.000968 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.249 on 241 degrees of freedom
## Multiple R-squared:  0.7482, Adjusted R-squared:  0.7398
## F-statistic: 89.51 on 8 and 241 DF,  p-value: < 2.2e-16
```

```
cooks <- data.frame(x = 1:(nrow(bodyfat)-2), cooks_distance = cooks.distance(refit))
head(cooks)
```

```
##   x cooks_distance
## 1 1   0.0020412680
## 2 2   0.0016826679
## 3 3   0.0182736939
## 4 4   0.0004802707
## 5 5   0.0023168159
## 6 6   0.0024924283
```

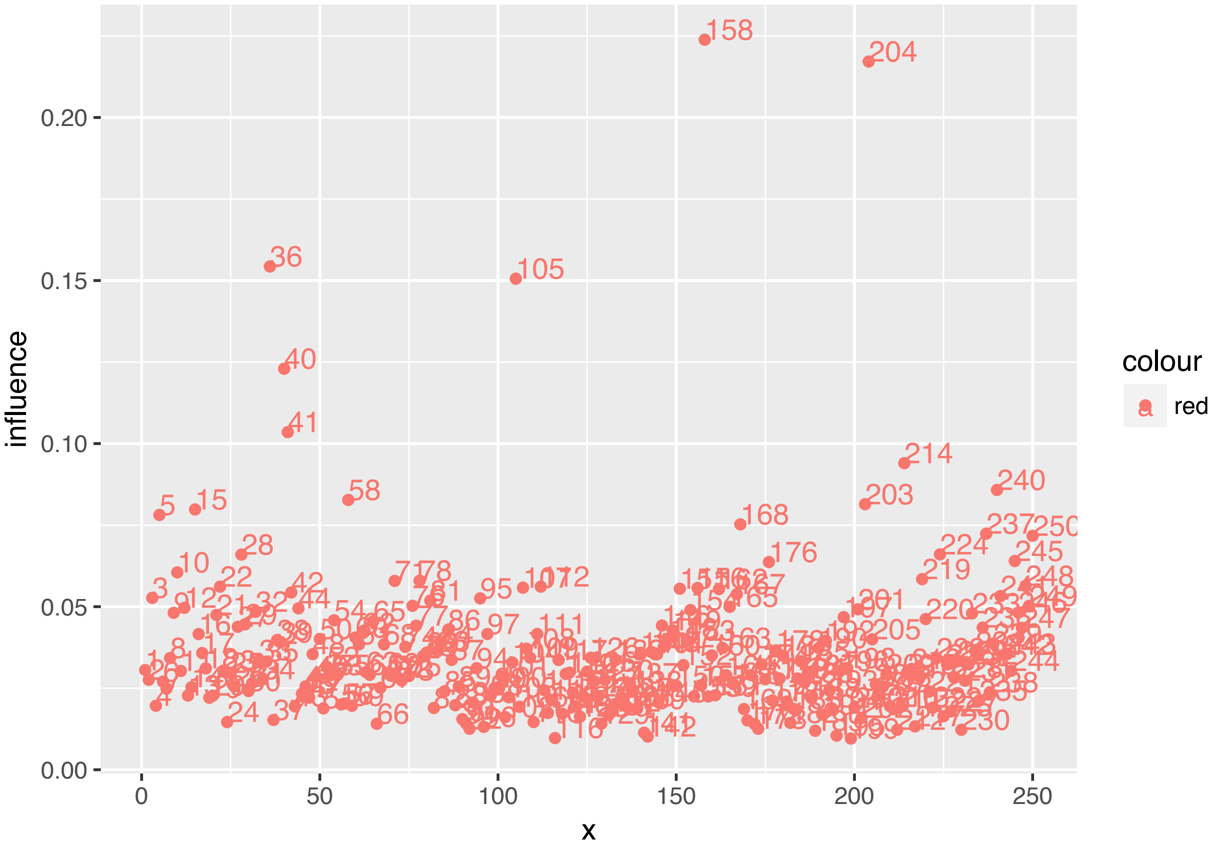
```
ggplot(cooks, aes(x = x, y= cooks_distance, colour="red"))+ geom_point()+ geom_text(aes(label=x),hjust=
```



```r
influ <- data.frame(x = 1:(nrow(bodyfat)-2), influence = influence(refit)$hat)
head(influ)
```

```
##   x  influence
## 1 1 0.03055939
## 2 2 0.02758775
## 3 3 0.05269947
## 4 4 0.01962064
## 5 5 0.07813447
## 6 6 0.02702276
```

```r
ggplot(influ, aes(x = x, y= influence, colour="red"))+ geom_point()+ geom_text(aes(label=x),hjust=0, vj
```


influence
x
0.00
0.05
0.10
0.15
0.20
0
50
100
150
200
250
colour
red
158
204
36
105
40
41
214
240
58
5
15
203
168
237
250
28
176
224
245
219
10
22
248
42
95
3
12
44
201
220
97
111
205
86
65
54
24
37
66
116
142
230
244