#1.)

Let $H_{(-p)}$ be the projection matrix on all columns in X except the pth column, $X_{(p)}$. And, $X^{(p)}$ and $Y^{(p)}$ are n residuals regressing $X_{(p)}$ on all other variables except pth and Y on other variables except $X_{(p)}$.

So, $Y^{(p)} = (I - H_{(-p)})Y$ and $X^{(p)} = (I - H_{(-p)})X_{(p)}$.

Since this is a simple regression, $b^{(p)} = [(X^{(p)})^T X^{(p)}]^{-1} (X^{(p)})^T Y^{(p)}$

And since $X^{(p)}$ is a column vector, $[X^{(p)}]^T X^{(p)}$ is a number, so $b^{(p)} = \frac{(X^{(p)})^T Y^{(p)}}{\|X^{(p)}\|_2^2}$

So, I got $b^{(p)} = \frac{[I - H_{(-p)} X_{(p)}]^T [I - H_{(-p)}] Y}{\|(I - H_{(-p)}) X_{(p)}\|_2^2} = \frac{[I - H_{(-p)}] Y [I - H_{(-p)}) X_{(p)}]^T}{X_{(p)}^T (I - H_{(-p)})^T (I - H_{(-p)}) X_{(p)}}$

$$= \frac{Y^T (I - H_{(-p)})^T (I - H_{(-p)}) X_{(p)}}{X_{(p)}^T (I - H_{(-p)}) X_{(p)}} = \frac{Y^T (I - H_{(-p)}) X_{(p)}}{X_{(p)}^T (I - H_{(-p)}) X_{(p)}}$$

And prove $\hat{\beta}_p$ is equal to $b^{(p)}$

If we say V is a matrix with the first p columns including intercept and α be coefficient for the first p columns and b is the coefficient for $p+1$ th column

$$\min_{\alpha} \|Y - V\alpha - X_{(p)} b\|_2^2 = \min_{\alpha} \|Y - V\alpha\|_2^2 - 2(Y - V\alpha)^T X_{(p)} b + b^2 \|X_{(p)}\|_2^2$$

And take the gradient, and get $0 = -2V^T(Y - V\alpha) + 2V^T X_{(p)} b$

And $0 = -2V^T Y + 2V^T V\alpha + 2V^T X_{(p)} b$, so $\alpha = (V^T V)^{-1} V^T (Y - X_{(p)} b)$

Then I could say $HY = X\hat{\beta} = V\alpha(b) + X_{(p)} \hat{\beta}_p = H_{(-p)} Y + (I - H_{(-p)}) X_{(p)} \hat{\beta}_p$

And, thus, $\hat{\beta}_p = \frac{Y^T (I - H_{(-p)}) X_{(p)}}{X_{(p)}^T (I - H_{(-p)}) X_{(p)}} = b^{(p)}$

So, $e^{(p)} = Y^{(p)} - X^{(p)} b^{(p)} = Y^{(p)} - X^{(p)} \hat{\beta}_p$

$= (I - H_{(-p)}) Y - (I - H_{(-p)}) X_{(p)} \hat{\beta}_p$

$= Y - H_{(-p)} Y - X_{(p)} \hat{\beta}_p + H_{(-p)} X_{(p)} \hat{\beta}_p$

$= Y - [H_{(-p)} Y + X_{(p)} \hat{\beta}_p - H_{(-p)} X_{(p)} \hat{\beta}_p]$

$= Y - [H_{(-p)} Y + (I - H_{(-p)}) X_{(p)} \hat{\beta}_p]$

$= Y - X\hat{\beta} = Y - HY = \hat{e}$

Jin Kweon (3032235207) HW4

Jin Kweon

10/22/2017

Data import

```
data <- read.table("Data-Bodyfat.txt")
colnames(data) <- c("case_#", "Brozek_body_fat_%", "bodyfat", "density",
                    "age", "weight", "height", "adiposity", "fat_free", "neck", "chest",
                    "abdomen", "hip", "thigh", "knee", "ankle", "bicep", "forearm", "wrist")
dim(data)
```

```
## [1] 252  19
```

```
data <- data[, 3:19]
data <- data[,-c(2, 7)]
data <- data[,-5]
```

Part a - i)

Backward elimination using the individual p-values.

```
#Make sure to run the function with the dataset having the response on the first column
backsubset <- function(dataset){
  indicator <- 1 #True
  response <- dataset[,1]
  predictor <- dataset[,-1]
  combine <- as.data.frame(cbind(response, predictor))

  while(indicator){
    setlm <- lm(response ~., data = combine)
    if(ncol(as.matrix((predictor))) == 0){
      indicator <- 0 #False
    }else if(max(summary(setlm)$coefficients[-1,4]) >= 0.05){
      maxpoint <- as.numeric(which.max(summary(setlm)$coefficients[-1,4]))
      predictor <- predictor[, -maxpoint]
      combine <- combine[,-(maxpoint + 1)]
    }else{
      indicator <- 0 #False
    }
  }
```

```
  backlm <- lm(response ~., data = as.data.frame(cbind(response, predictor)))
  model <- as.data.frame(cbind(response, predictor))
  things <- list(backlm, names(model)[-1])
  return(things)
}

backsubset(data)
```

```
## [[1]]
##
## Call:
## lm(formula = response ~ ., data = as.data.frame(cbind(response,
##     predictor)))
##
## Coefficients:
## (Intercept)       weight      abdomen      forearm        wrist
##    -34.8541      -0.1356       0.9958       0.4729      -1.5056
##
##
## [[2]]
## [1] "weight"  "abdomen" "forearm" "wrist"
```

```
#Answer check
SignifReg(bodyfat ~., data = data, alpha = 0.05, direction = "backward", criterion = "p-value",
          correction = "None")
```

```
##
## Call:
## lm(formula = reg, data = data)
##
## Coefficients:
## (Intercept)       weight      abdomen      forearm        wrist
##    -34.8541      -0.1356       0.9958       0.4729      -1.5056
```

Comment:

I will go with the cutoff p-value be 0.1. Cut off being 0.05 tends to pick models that are way smaller than desirable for prediction purposes, as it was mentioned in the lecture note.

Part a - ii)

```
forwardsubset <- function(data){
  indicator <- 1 #True
  response <- data[,1]
  predictor <- as.data.frame(data[,-1])
  newpredictor <- as.data.frame(rep(0, length(response))) #Define a new predictor...

  while(indicator){
    if(ncol(predictor) == 0){
      indicator <- 0
    } #pre-check

    pval <- c()
    for(i in 1:ncol(predictor)){
      setlm2 <- lm(response ~., data = as.data.frame(cbind(response, newpredictor[,-1],
                                                           predictor[,i])))
      pval[i] <- summary(setlm2)$coefficients[-1,4][ncol(as.data.frame(newpredictor[,-1])) + 1]
    } #get p-value for the leftover predictor set
    minpoint <- as.numeric(which.min(pval))

    if(min(pval) < 0.05){
      newpredictor <- as.data.frame(cbind(newpredictor, predictor[,minpoint]))
      colnames(newpredictor)[ncol(as.data.frame(newpredictor[,-1])) + 1] <-
        colnames(predictor)[minpoint] #Assign column names when I add up a variable in newpredictor
      predictor <- predictor[, -minpoint]
    }else{
      indicator <- 0
    } #move qualified predictor variable into newpredictor set.
  }

  newpredictor <- newpredictor[,-1] #take out the column with all zeros...
  forwardlm <- lm(response ~., data = as.data.frame(cbind(response, newpredictor)))
  model2 <- as.data.frame(cbind(response, newpredictor))
  things2 <- list(forwardlm, names(model2)[-1])
  return(things2)
}

forwardsubset(data)
```

```
## [[1]]
##
## Call:
## lm(formula = response ~ ., data = as.data.frame(cbind(response,
##     newpredictor)))
##
## Coefficients:
## (Intercept)      abdomen       weight        wrist      forearm
##    -34.8541       0.9958      -0.1356      -1.5056       0.4729
##
##
## [[2]]
## [1] "abdomen" "weight"  "wrist"   "forearm"
```

```r
#Answer check
SignifReg(bodyfat ~., data = data, alpha = 0.05, direction = "forward", criterion = "p-value",
          correction = "None")
```

```
##
## Call:
## lm(formula = reg, data = data)
##
## Coefficients:
## (Intercept)      abdomen       weight        wrist      forearm
##    -34.8541       0.9958      -0.1356      -1.5056       0.4729
```

Comment:

Forward and backward selections for the sigficance level being 0.1 are the same, by chance. It is usually not the case!!!

Part a - iii)

Good reference: https://rpubs.com/davoodastaraky/subset

I know that AdjR^2 = 1 - $\frac{RSS(m)/(n-p(m)-1)}{TSS/(n-1)}$, so it can penalize when the (unnecessary) explanatory variables increase.

From the summary, I learned that the best subset of each size are below: (based on RSS)

1. abdomen
2. weight, abdomen
3. weight, abdomen, wrist
4. weight, abdomen, forearm, wrist
5. weight, neck, abdomen, forearm, wrist
6. age, weight, abdomen, thigh, forearm, wrist
7. age, weight, neck, abdomen, thigh, forearm, wrist
8. age, weight, neck, abdomen, hip, thigh, forearm, wrist
9. age, weight, neck, abdomen, hip, thigh, bicep, forearm, wrist

10. age, weight, neck, abdomen, hip, thigh, ankle, bicep, forearm, wrist
11. age, weight, height, neck, abdomen, hip, thigh, ankle, bicep, forearm, wrist
12. age, weight, height, neck, chest, abdomen, hip, thigh, ankle, bicep, forearm, wrist
13. age, weight, height, neck, chest, abdomen, hip, thigh, knee, ankle, bicep, forearm, wrist

```
subset <- regsubsets(bodyfat ~., data = data, nbest = 1, nvmax = (ncol(data) - 1))
summary(subset)
```

```
## Subset selection object
## Call: regsubsets.formula(bodyfat ~ ., data = data, nbest = 1, nvmax = (ncol(data) -
##     1))
## 13 Variables  (and intercept)
##         Forced in Forced out
## age         FALSE      FALSE
## weight      FALSE      FALSE
## height      FALSE      FALSE
## neck        FALSE      FALSE
## chest       FALSE      FALSE
## abdomen     FALSE      FALSE
## hip         FALSE      FALSE
## thigh       FALSE      FALSE
## knee        FALSE      FALSE
## ankle       FALSE      FALSE
## bicep       FALSE      FALSE
## forearm     FALSE      FALSE
## wrist       FALSE      FALSE
## 1 subsets of each size up to 13
## Selection Algorithm: exhaustive
##           age weight height neck chest abdomen hip thigh knee ankle bicep
## 1  ( 1 )  " " " "    " "    " "  " "   "*"     " " " "   " "  " "   " "
## 2  ( 1 )  " " "*"    " "    " "  " "   "*"     " " " "   " "  " "   " "
## 3  ( 1 )  " " "*"    " "    " "  " "   "*"     " " " "   " "  " "   " "
## 4  ( 1 )  " " "*"    " "    " "  " "   "*"     " " " "   " "  " "   " "
## 5  ( 1 )  " " "*"    " "    "*"  " "   "*"     " " " "   " "  " "   " "
## 6  ( 1 )  "*" "*"    " "    " "  " "   "*"     " " "*"   " "  " "   " "
## 7  ( 1 )  "*" "*"    " "    "*"  " "   "*"     " " "*"   " "  " "   " "
## 8  ( 1 )  "*" "*"    " "    "*"  " "   "*"     "*" "*"   " "  " "   " "
## 9  ( 1 )  "*" "*"    " "    "*"  " "   "*"     "*" "*"   " "  " "   "*"
## 10  ( 1 ) "*" "*"    " "    "*"  " "   "*"     "*" "*"   " "  "*"   "*"
## 11  ( 1 ) "*" "*"    "*"    "*"  " "   "*"     "*" "*"   " "  "*"   "*"
## 12  ( 1 ) "*" "*"    "*"    "*"  "*"   "*"     "*" "*"   " "  "*"   "*"
## 13  ( 1 ) "*" "*"    "*"    "*"  "*"   "*"     "*" "*"   "*"  "*"   "*"
##           forearm wrist
## 1  ( 1 )  " "     " "
## 2  ( 1 )  " "     " "
## 3  ( 1 )  " "     "*"
## 4  ( 1 )  "*"     "*"
## 5  ( 1 )  "*"     "*"
## 6  ( 1 )  "*"     "*"
## 7  ( 1 )  "*"     "*"
## 8  ( 1 )  "*"     "*"
## 9  ( 1 )  "*"     "*"
## 10  ( 1 ) "*"     "*"
```

```
## 11  ( 1 ) "*"     "*"
## 12  ( 1 ) "*"     "*"
## 13  ( 1 ) "*"     "*"
```

```r
#variable subset (makes it more convenient later)
one <- as.data.frame(data$abdomen); colnames(one) <- "abdomen"
two <- as.data.frame(cbind(weight = data$weight, abdomen = data$abdomen))
three <- as.data.frame(cbind(weight = data$weight, abdomen = data$abdomen, wrist = data$wrist))
four <- as.data.frame(cbind(weight = data$weight, abdomen = data$abdomen, forearm = data$forearm,
             wrist = data$wrist))
five <- as.data.frame(cbind(weight = data$weight, neck = data$neck, abdomen = data$abdomen,
                            forearm = data$forearm, wrist = data$wrist))
six <- as.data.frame(cbind(age = data$age, weight = data$weight, abdomen = data$abdomen,
                           thigh = data$thigh, forearm = data$forearm, wrist = data$wrist))
seven <- as.data.frame(cbind(age = data$age, weight = data$weight, neck = data$neck,
                             abdomen = data$abdomen, thigh = data$thigh, forearm = data$forearm,
                             wrist = data$wrist))
eight <- as.data.frame(cbind(age = data$age, weight = data$weight, neck = data$neck,
                             abdomen = data$abdomen, hip = data$hip, thigh = data$thigh,
                             forearm = data$forearm, wrist = data$wrist))
nine <- as.data.frame(cbind(age = data$age, weight = data$weight, neck = data$neck,
                            abdomen = data$abdomen, hip = data$hip, thigh = data$thigh,
                            bicep = data$bicep, forearm = data$forearm, wrist = data$wrist))
ten <- as.data.frame(cbind(age = data$age, weight = data$weight, neck = data$neck,
                           abdomen = data$abdomen, hip = data$hip, thigh = data$thigh,
                           ankle = data$ankle, bicep = data$bicep, forearm = data$forearm,
                           wrist = data$wrist))
eleven <- as.data.frame(cbind(age = data$age, weight = data$weight, height = data$height,
                              neck = data$neck, abdomen = data$abdomen, hip = data$hip,
                              thigh = data$thigh, ankle = data$ankle, bicep = data$bicep,
                              forearm = data$forearm, wrist = data$wrist))
twelve <- as.data.frame(cbind(age = data$age, weight = data$weight, height = data$height,
                              neck = data$neck, chest = data$chest, abdomen = data$abdomen,
                              hip = data$hip, thigh = data$thigh, ankle = data$ankle,
                              bicep = data$bicep, forearm = data$forearm, wrist = data$wrist))
thirteen <- as.data.frame(cbind(age = data$age, weight = data$weight, height = data$height,
                                neck = data$neck, chest = data$chest,
                  abdomen = data$abdomen, hip = data$hip, thigh = data$thigh,
                  knee = data$knee,
                  ankle = data$ankle, bicep = data$bicep, forearm = data$forearm,
                  wrist = data$wrist))


lists <- list(one, two, three, four, five, six, seven, eight, nine, ten, eleven,
              twelve, thirteen)

adj <- function(y, x){
  combined <- cbind(y, x); colnames(combined)[1] <- "y"
  lim <- lm(y ~ ., data = combined)
  rssm <- sum((lim$residuals)^2)
  tss <- sum((combined$y - mean(combined$y))^2)
  ans <- 1 - ((rssm / (length(combined$y) - (ncol(combined) - 1) - 1)) / (tss / (length(combined$y) - 1
  ans <- as.numeric(ans)
  return(ans)
```

```
}

adjs <- c()

for (i in 1:13){
  adjs[i] <- as.numeric(adj(data$bodyfat, lists[[i]]))#Change to numeric

  a <- paste(i, ":", adjs[i])
  print(a)
}
```

```
## [1] "1 : 0.660318770251815"
## [1] "2 : 0.71653945507665"
## [1] "3 : 0.724446621539021"
## [1] "4 : 0.73071986350693"
## [1] "5 : 0.732589239379987"
## [1] "6 : 0.734624415745718"
## [1] "7 : 0.737145697045321"
## [1] "8 : 0.738210121885681"
## [1] "9 : 0.738350364726977"
## [1] "10 : 0.738051612524622"
## [1] "11 : 0.737476292347814"
## [1] "12 : 0.736445551360969"
## [1] "13 : 0.735342614521921"
```

```
b <- paste("The max adjusted R-square is", which.max(adjs), "th model, so I keep the",
           which.max(adjs), "variables.")
print(b)
```

```
## [1] "The max adjusted R-square is 9 th model, so I keep the 9 variables."
```

```
paste("Variables keeped:", names(lists[[which.max(adjs)]]))
```

```
## [1] "Variables keeped: age"     "Variables keeped: weight"
## [3] "Variables keeped: neck"    "Variables keeped: abdomen"
## [5] "Variables keeped: hip"     "Variables keeped: thigh"
## [7] "Variables keeped: bicep"   "Variables keeped: forearm"
## [9] "Variables keeped: wrist"
```

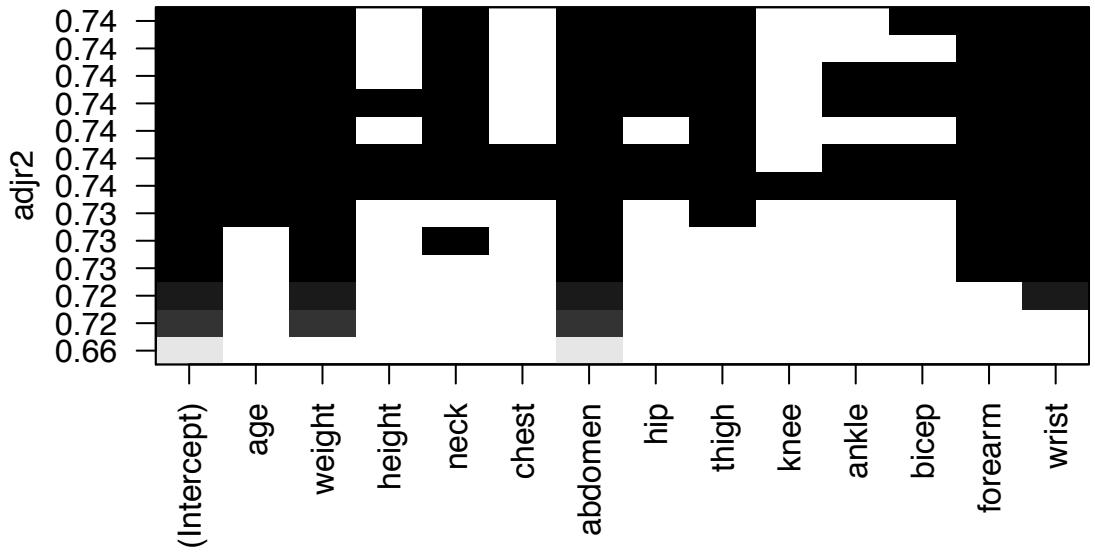
```
#Answer check
summary(subset)$adjr2 #compare answers
```

```
##  [1] 0.6603188 0.7165395 0.7244466 0.7307199 0.7325892 0.7346244 0.7371457
##  [8] 0.7382101 0.7383504 0.7380516 0.7374763 0.7364456 0.7353426
```

```
paste("So, I keep the", which.max(summary(subset)$adjr2), "variables.")
```

```
## [1] "So, I keep the 9 variables."
```

```
plot(subset, scale = "adjr2")
```




```
coef(subset, which.max(summary(subset)$adjr2))
```

```
##  (Intercept)          age       weight         neck      abdomen
## -23.30499184   0.06348330  -0.09842527  -0.49329528   0.94926069
##          hip        thigh        bicep      forearm        wrist
##  -0.18287103   0.26537882   0.17888997   0.45149625  -1.54208372
```

Comment:

So, regsubsets() function tells us what variables to use for each number of variables (they compare RSS of each models), and then, I will calculate what we learned in the class.

The higher Adjusted R^2, the better.

I keep the 9 variables. (which variables are stated before the r code chunk, or from the regsubset() function output)

Part a - iv)

Good reference: https://stats.stackexchange.com/questions/11115/how-to-plot-aic-values-when-using-the-leaps-package

I know that AIC(m) := -2 log(maximum value of likelihood in m) + 2(number of parameters in m) = $nlog(\frac{RSS(m)}{n}) + nlog(2\pi e) + 2(1 + p(m)) \approx nlog(\frac{RSS(m)}{n}) + 2(1 + p(m))$, as the dropped term is the same for all models. So, we do not really need this term when we select the models.

From the summary, I learned that the best subset of each size are below: (based on RSS)

1. abdomen
2. weight, abdomen
3. weight, abdomen, wrist
4. weight, abdomen, forearm, wrist
5. weight, neck, abdomen, forearm, wrist
6. age, weight, abdomen, thigh, forearm, wrist
7. age, weight, neck, abdomen, thigh, forearm, wrist
8. age, weight, neck, abdomen, hip, thigh, forearm, wrist
9. age, weight, neck, abdomen, hip, thigh, bicep, forearm, wrist
10. age, weight, neck, abdomen, hip, thigh, ankle, bicep, forearm, wrist
11. age, weight, height, neck, abdomen, hip, thigh, ankle, bicep, forearm, wrist
12. age, weight, height, neck, chest, abdomen, hip, thigh, ankle, bicep, forearm, wrist
13. age, weight, height, neck, chest, abdomen, hip, thigh, knee, ankle, bicep, forearm, wrist

```
summary(subset)
```

```
## Subset selection object
## Call: regsubsets.formula(bodyfat ~ ., data = data, nbest = 1, nvmax = (ncol(data) -
##     1))
## 13 Variables  (and intercept)
##         Forced in Forced out
## age         FALSE      FALSE
## weight      FALSE      FALSE
## height      FALSE      FALSE
## neck        FALSE      FALSE
## chest       FALSE      FALSE
## abdomen     FALSE      FALSE
## hip         FALSE      FALSE
## thigh       FALSE      FALSE
## knee        FALSE      FALSE
## ankle       FALSE      FALSE
## bicep       FALSE      FALSE
## forearm     FALSE      FALSE
## wrist       FALSE      FALSE
## 1 subsets of each size up to 13
## Selection Algorithm: exhaustive
##           age weight height neck chest abdomen hip thigh knee ankle bicep
## 1  ( 1 )  " " " "    " "    " "   " "   "*"     " " " "   " "   " "   " "
## 2  ( 1 )  " " "*"    " "    " "   " "   "*"     " " " "   " "   " "   " "
## 3  ( 1 )  " " "*"    " "    " "   " "   "*"     " " " "   " "   " "   " "
## 4  ( 1 )  " " "*"    " "    " "   " "   "*"     " " " "   " "   " "   " "
## 5  ( 1 )  " " "*"    " "    "*"   " "   "*"     " " " "   " "   " "   " "
## 6  ( 1 )  "*" "*"    " "    " "   " "   "*"     " " "*"   " "   " "   " "
## 7  ( 1 )  "*" "*"    " "    "*"   " "   "*"     " " "*"   " "   " "   " "
## 8  ( 1 )  "*" "*"    " "    "*"   " "   "*"     "*" "*"   " "   " "   " "
```

```
## 9  ( 1 )  "*" "*"    " "    "*"  " "   "*"     "*" "*"   " "  " "   "*"
## 10  ( 1 ) "*" "*"    " "    "*"  " "   "*"     "*" "*"   " "  "*"   "*"
## 11  ( 1 ) "*" "*"    "*"    "*"  " "   "*"     "*" "*"   " "  "*"   "*"
## 12  ( 1 ) "*" "*"    "*"    "*"  "*"   "*"     "*" "*"   " "  "*"   "*"
## 13  ( 1 ) "*" "*"    "*"    "*"  "*"   "*"     "*" "*"   "*"  "*"   "*"
##           forearm wrist
## 1  ( 1 )  " "     " "
## 2  ( 1 )  " "     " "
## 3  ( 1 )  " "     "*"
## 4  ( 1 )  "*"     "*"
## 5  ( 1 )  "*"     "*"
## 6  ( 1 )  "*"     "*"
## 7  ( 1 )  "*"     "*"
## 8  ( 1 )  "*"     "*"
## 9  ( 1 )  "*"     "*"
## 10  ( 1 ) "*"     "*"
## 11  ( 1 ) "*"     "*"
## 12  ( 1 ) "*"     "*"
## 13  ( 1 ) "*"     "*"
```

```
lists <- list(one, two, three, four, five, six, seven, eight, nine, ten, eleven,
              twelve, thirteen)
aic <- function(y, x){
  combined <- cbind(y, x); colnames(combined)[1] <- "y"
  lim <- lm(y ~ ., data = combined)
  rssm <- sum((lim$residuals)^2)
  ans <- (length(combined$y) * log(rssm / length(combined$y))) + (2 * (1 + (ncol(combined) - 1)))
  ans <- as.numeric(ans)
  return(ans)
}

aics <- c()
for (i in 1:13){
  aics[i] <- as.numeric(aic(data$bodyfat, lists[[i]]))#Change to numeric

  a <- paste(i, ":", aics[i])
  print(a)
}
```

```
## [1] "1 : 800.645316730991"
## [1] "2 : 756.039756955108"
## [1] "3 : 749.896192644591"
## [1] "4 : 745.074670263521"
## [1] "5 : 744.296840873292"
## [1] "6 : 743.345137337846"
## [1] "7 : 741.908812574697"
## [1] "8 : 741.851361245443"
## [1] "9 : 742.677150013707"
## [1] "10 : 743.921240343848"
## [1] "11 : 745.426285300209"
## [1] "12 : 747.361576389139"
## [1] "13 : 749.357353727093"
```

```
b <- paste("The min AIC is", which.min(aics), "th model, so I keep the",
           which.min(aics), "variables.")
```

```r
print(b)
```

```
## [1] "The min AIC is 8 th model, so I keep the 8 variables."
```

```r
paste("Variables keeped:", names(lists[[which.min(aics)]]))
```

```
## [1] "Variables keeped: age"     "Variables keeped: weight"
## [3] "Variables keeped: neck"    "Variables keeped: abdomen"
## [5] "Variables keeped: hip"     "Variables keeped: thigh"
## [7] "Variables keeped: forearm" "Variables keeped: wrist"
```

```r
#Answer check
aics3 <- c()
for(i in 1:13){
  combined <- cbind(data$bodyfat, lists[[i]]); colnames(combined)[1] <- "y"
  lim <- lm(y ~ ., data = combined)
  aics3[i] <- extractAIC(lim)[2]
}
print(aics3)
```

```
##  [1] 800.6453 756.0398 749.8962 745.0747 744.2968 743.3451 741.9088
##  [8] 741.8514 742.6772 743.9212 745.4263 747.3616 749.3574
```

```r
#Not working...........???
aics2 <- c()
for(i in 1:13){
  combined <- cbind(data$bodyfat, lists[[i]]); colnames(combined)[1] <- "y"
  lim <- lm(y ~ ., data = combined)
  aics2[i] <- AIC(lim)
}
print(aics2)
```

```
##  [1] 1517.790 1473.185 1467.041 1462.220 1461.442 1460.490 1459.054
##  [8] 1458.996 1459.822 1461.066 1462.571 1464.507 1466.502
```

Comment:

There was a good explanation why I used BIC... "Although BIC and AIC use different penalizations, the function returns the best model when the number of parameters is fixed. If number of parameters is fixed then penalisation term is the same for all models considered, hence the best model is selected only depending on its log likelihood, which is the same for BIC and AIC."

I keep the 8 variables. (which variables are stated before the r code chunk, or from the regsubset() function output)

Part a - v)

Good reference: https://stats.stackexchange.com/questions/87468/why-do-i-get-different-bic-values-when-i-use-regsubsets-and

I know that AIC(m) := -2 log(maximum value of likelihood in m) + log(n) (number of parameters in m) = $nlog(\frac{RSS(m)}{n}) + nlog(2\pi e) + log(n)(1 + p(m)) \approx nlog(\frac{RSS(m)}{n}) + log(n)(1 + p(m))$, as the dropped term is the same for all models. So, we do not really need this term when we select the models.

From the summary, I learned that the best subset of each size are below: (based on RSS)

1. abdomen
2. weight, abdomen
3. weight, abdomen, wrist
4. weight, abdomen, forearm, wrist
5. weight, neck, abdomen, forearm, wrist
6. age, weight, abdomen, thigh, forearm, wrist
7. age, weight, neck, abdomen, thigh, forearm, wrist
8. age, weight, neck, abdomen, hip, thigh, forearm, wrist
9. age, weight, neck, abdomen, hip, thigh, bicep, forearm, wrist
10. age, weight, neck, abdomen, hip, thigh, ankle, bicep, forearm, wrist
11. age, weight, height, neck, abdomen, hip, thigh, ankle, bicep, forearm, wrist
12. age, weight, height, neck, chest, abdomen, hip, thigh, ankle, bicep, forearm, wrist
13. age, weight, height, neck, chest, abdomen, hip, thigh, knee, ankle, bicep, forearm, wrist

```
summary(subset)
```

```
## Subset selection object
## Call: regsubsets.formula(bodyfat ~ ., data = data, nbest = 1, nvmax = (ncol(data) -
##     1))
## 13 Variables  (and intercept)
##         Forced in Forced out
## age         FALSE      FALSE
## weight      FALSE      FALSE
## height      FALSE      FALSE
## neck        FALSE      FALSE
## chest       FALSE      FALSE
## abdomen     FALSE      FALSE
## hip         FALSE      FALSE
## thigh       FALSE      FALSE
## knee        FALSE      FALSE
## ankle       FALSE      FALSE
## bicep       FALSE      FALSE
## forearm     FALSE      FALSE
## wrist       FALSE      FALSE
## 1 subsets of each size up to 13
## Selection Algorithm: exhaustive
##           age weight height neck chest abdomen hip thigh knee ankle bicep
```

```
## 1  ( 1 )  " " " " "    " " "    " " " " "   "*"     " " " " "   " " " " "   " " "
## 2  ( 1 )  " " " "*"    " " "    " " " " "   "*"     " " " " "   " " " " "   " " "
## 3  ( 1 )  " " " "*"    " " "    " " " " "   "*"     " " " " "   " " " " "   " " "
## 4  ( 1 )  " " " "*"    " " "    " " " " "   "*"     " " " " "   " " " " "   " " "
## 5  ( 1 )  " " " "*"    " " "    "*" " " "   "*"     " " " " "   " " " " "   " " "
## 6  ( 1 )  "*" "*"      " " "    " " " " "   "*"     " " "*"     " " " " "   " " "
## 7  ( 1 )  "*" "*"      " " "    "*" " " "   "*"     " " "*"     " " " " "   " " "
## 8  ( 1 )  "*" "*"      " " "    "*" " " "   "*"     "*" "*"     " " " " "   " " "
## 9  ( 1 )  "*" "*"      " " "    "*" " " "   "*"     "*" "*"     " " " " "   "*"
## 10  ( 1 ) "*" "*"      " " "    "*" " " "   "*"     "*" "*"     " " "*"   "   "*"
## 11  ( 1 ) "*" "*"      "*"    "*" " " "   "*"     "*" "*"     " " "*"   "   "*"
## 12  ( 1 ) "*" "*"      "*"    "*" "*"   "*"     "*" "*"     " " "*"   "   "*"
## 13  ( 1 ) "*" "*"      "*"    "*" "*"   "*"     "*" "*"     "*" "*"   "   "*"
##           forearm wrist
## 1  ( 1 )  " "     " "
## 2  ( 1 )  " "     " "
## 3  ( 1 )  " "     "*"
## 4  ( 1 )  "*"     "*"
## 5  ( 1 )  "*"     "*"
## 6  ( 1 )  "*"     "*"
## 7  ( 1 )  "*"     "*"
## 8  ( 1 )  "*"     "*"
## 9  ( 1 )  "*"     "*"
## 10  ( 1 ) "*"     "*"
## 11  ( 1 ) "*"     "*"
## 12  ( 1 ) "*"     "*"
## 13  ( 1 ) "*"     "*"
```

```r
lists <- list(one, two, three, four, five, six, seven, eight, nine, ten, eleven,
              twelve, thirteen)

bic <- function(y, x){
  combined <- cbind(y, x); colnames(combined)[1] <- "y"
  lim <- lm(y ~ ., data = combined)
  rssm <- sum((lim$residuals)^2)
  ans <- (length(combined$y) * log(rssm / length(combined$y))) + (log(length(combined$y)) *
                                                                    (1 + (ncol(combined) - 1)))
  ans <- as.numeric(ans)
  return(ans)
}

bics <- c()
for (i in 1:13){
  bics[i] <- as.numeric(bic(data$bodyfat, lists[[i]]))#Change to numeric

  a <- paste(i, ":", bics[i])
  print(a)
}
```

```
## [1] "1 : 807.704174906014"
## [1] "2 : 766.628044217643"
## [1] "3 : 764.013908994636"
## [1] "4 : 762.721815701078"
## [1] "5 : 765.473415398361"
## [1] "6 : 768.051140950426"
```

```
## [1] "7 : 770.144245274788"
## [1] "8 : 773.616223033046"
## [1] "9 : 777.971440888822"
## [1] "10 : 782.744960306473"
## [1] "11 : 787.779434350346"
## [1] "12 : 793.244154526788"
## [1] "13 : 798.769360952253"
```

```
b <- paste("The min BIC is", which.min(bics), "th model, so I keep the",
           which.min(bics), "variables.")
print(b)
```

```
## [1] "The min BIC is 4 th model, so I keep the 4 variables."
```

```
paste("Variables keeped:", names(lists[[which.min(bics)]]))
```

```
## [1] "Variables keeped: weight"  "Variables keeped: abdomen"
## [3] "Variables keeped: forearm" "Variables keeped: wrist"
```

```
#Use different algorithm...
bics2 <- c()
for(i in 1:13){
  combined <- cbind(data$bodyfat, lists[[i]]); colnames(combined)[1] <- "y"
  lim <- lm(y ~ ., data = combined)
  bics2[i] <- BIC(lim)
}
print(bics2)
```

```
##  [1] 1528.379 1487.302 1484.688 1483.396 1486.148 1488.726 1490.819
##  [8] 1494.291 1498.646 1503.419 1508.454 1513.919 1519.444
```

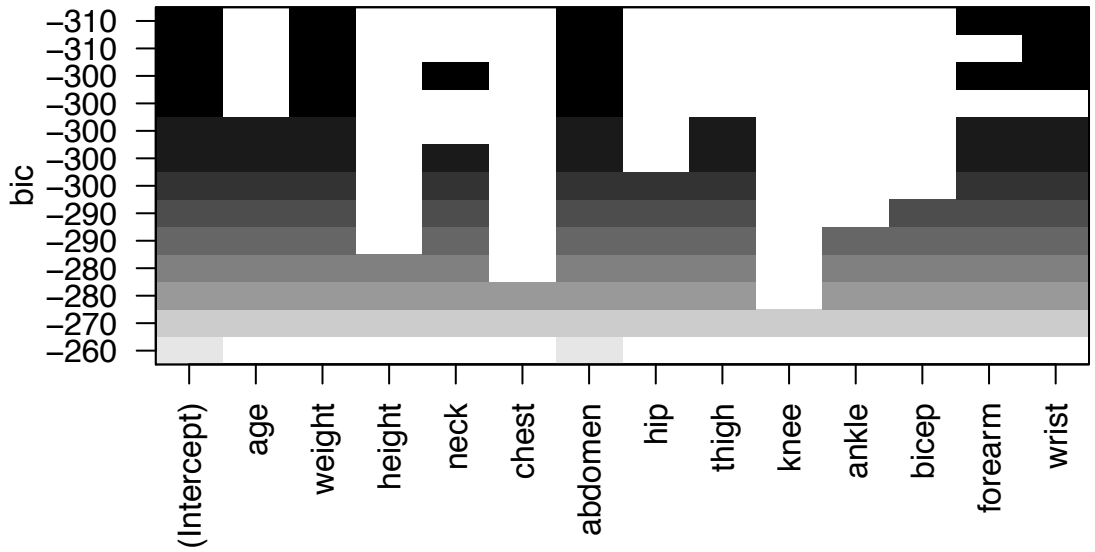
```
#Answer check
summary(subset)$bic
```

```
##  [1] -262.0435 -303.1197 -305.7338 -307.0259 -304.2743 -301.6966 -299.6035
##  [8] -296.1315 -291.7763 -287.0028 -281.9683 -276.5036 -270.9784
```

```
paste("So, I keep the", which.min(summary(subset)$bic), "variables.")
```

```
## [1] "So, I keep the 4 variables."
```

```
plot(subset, scale = "bic")
```




```
coef(subset, which.min(summary(subset)$bic))
```

```
## (Intercept)      weight     abdomen     forearm       wrist
## -34.8540743  -0.1356315   0.9957513   0.4729284  -1.5055620
```

Comment:

BIC from the regsubsets() function is differed by the additive constant, so this does not really matter.

Part a - vi)

Mallow's $C_p = \frac{Rss(m)}{\hat{\sigma}^2} - (n - 2 - 2p(m))$

From the summary, I learned that the best subset of each size are below: (based on RSS)

1. abdomen
2. weight, abdomen
3. weight, abdomen, wrist
4. weight, abdomen, forearm, wrist
5. weight, neck, abdomen, forearm, wrist

6. age, weight, abdomen, thigh, forearm, wrist
7. age, weight, neck, abdomen, thigh, forearm, wrist
8. age, weight, neck, abdomen, hip, thigh, forearm, wrist
9. age, weight, neck, abdomen, hip, thigh, bicep, forearm, wrist
10. age, weight, neck, abdomen, hip, thigh, ankle, bicep, forearm, wrist
11. age, weight, height, neck, abdomen, hip, thigh, ankle, bicep, forearm, wrist
12. age, weight, height, neck, chest, abdomen, hip, thigh, ankle, bicep, forearm, wrist
13. age, weight, height, neck, chest, abdomen, hip, thigh, knee, ankle, bicep, forearm, wrist

```
summary(subset)
```

```
## Subset selection object
## Call: regsubsets.formula(bodyfat ~ ., data = data, nbest = 1, nvmax = (ncol(data) -
##     1))
## 13 Variables  (and intercept)
##         Forced in Forced out
## age         FALSE      FALSE
## weight      FALSE      FALSE
## height      FALSE      FALSE
## neck        FALSE      FALSE
## chest       FALSE      FALSE
## abdomen     FALSE      FALSE
## hip         FALSE      FALSE
## thigh       FALSE      FALSE
## knee        FALSE      FALSE
## ankle       FALSE      FALSE
## bicep       FALSE      FALSE
## forearm     FALSE      FALSE
## wrist       FALSE      FALSE
## 1 subsets of each size up to 13
## Selection Algorithm: exhaustive
##           age weight height neck chest abdomen hip thigh knee ankle bicep
## 1  ( 1 )  " " " "    " "    " "   " "   "*"     " " " "   " "  " "   " "
## 2  ( 1 )  " " "*"    " "    " "   " "   "*"     " " " "   " "  " "   " "
## 3  ( 1 )  " " "*"    " "    " "   " "   "*"     " " " "   " "  " "   " "
## 4  ( 1 )  " " "*"    " "    " "   " "   "*"     " " " "   " "  " "   " "
## 5  ( 1 )  " " "*"    " "    "*"   " "   "*"     " " " "   " "  " "   " "
## 6  ( 1 )  "*" "*"    " "    " "   " "   "*"     " " "*"   " "  " "   " "
## 7  ( 1 )  "*" "*"    " "    "*"   " "   "*"     " " "*"   " "  " "   " "
## 8  ( 1 )  "*" "*"    " "    "*"   " "   "*"     "*" "*"   " "  " "   " "
## 9  ( 1 )  "*" "*"    " "    "*"   " "   "*"     "*" "*"   " "  " "   "*"
## 10  ( 1 ) "*" "*"    " "    "*"   " "   "*"     "*" "*"   " "  "*"   "*"
## 11  ( 1 ) "*" "*"    "*"    "*"   " "   "*"     "*" "*"   " "  "*"   "*"
## 12  ( 1 ) "*" "*"    "*"    "*"   "*"   "*"     "*" "*"   " "  "*"   "*"
## 13  ( 1 ) "*" "*"    "*"    "*"   "*"   "*"     "*" "*"   "*"  "*"   "*"
##           forearm wrist
## 1  ( 1 )  " "     " "
## 2  ( 1 )  " "     " "
## 3  ( 1 )  " "     "*"
## 4  ( 1 )  "*"     "*"
## 5  ( 1 )  "*"     "*"
```

```
## 6  ( 1 )  "*"     "*"
## 7  ( 1 )  "*"     "*"
## 8  ( 1 )  "*"     "*"
## 9  ( 1 )  "*"     "*"
## 10  ( 1 ) "*"     "*"
## 11  ( 1 ) "*"     "*"
## 12  ( 1 ) "*"     "*"
## 13  ( 1 ) "*"     "*"
```

```
lists <- list(one, two, three, four, five, six, seven, eight, nine, ten, eleven,
              twelve, thirteen)
cp <- function(y, x){
  limbig <- lm(bodyfat ~., data = data)
  rssM <- sum((limbig$residuals)^2)
  sig <- rssM / (nrow(data) - (ncol(data) - 1) - 1)

  combined <- cbind(y, x); colnames(combined)[1] <- "y"
  lim <- lm(y ~ ., data = combined)
  rssm <- sum((lim$residuals)^2)

  ans <- (rssm / sig) - (length(combined$y) - 2 - (2 * (ncol(combined) - 1)))

  ans <- as.numeric(ans)
  return(ans)
}

cps <- c()
for (i in 1:13){
  cps[i] <- as.numeric(cp(data$bodyfat, lists[[i]]))#Change to numeric

  a <- paste(i, ":", cps[i])
  print(a)
}
```

```
## [1] "1 : 72.8688368308552"
## [1] "2 : 20.6907464472037"
## [1] "3 : 14.2102053750663"
## [1] "4 : 9.31433076632388"
## [1] "5 : 8.55927218389246"
## [1] "6 : 7.66485467561012"
## [1] "7 : 6.33765403933555"
## [1] "8 : 6.36714587376875"
## [1] "9 : 7.24974404810757"
## [1] "10 : 8.5331558668897"
## [1] "11 : 10.0651109460289"
## [1] "12 : 12.0039881031235"
## [1] "13 : 14"
```

```
b <- paste("The min CP is", which.min(cps), "th model, so I keep the",
           which.min(cps), "variables.")
print(b)
```

```
## [1] "The min CP is 7 th model, so I keep the 7 variables."
```

```
paste("Variables keeped:", names(lists[[which.min(cps)]]))
```

```
## [1] "Variables keeped: age"     "Variables keeped: weight"
## [3] "Variables keeped: neck"    "Variables keeped: abdomen"
## [5] "Variables keeped: thigh"   "Variables keeped: forearm"
## [7] "Variables keeped: wrist"
```

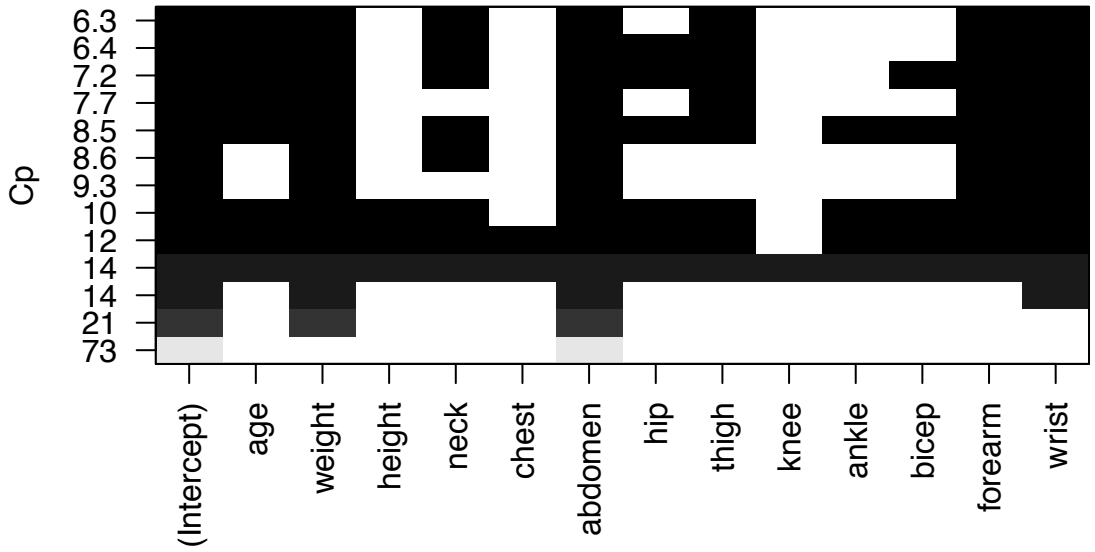
```
#Answer check
summary(subset)$cp
```

```
##  [1] 72.868837 20.690746 14.210205  9.314331  8.559272  7.664855  6.337654
##  [8]  6.367146  7.249744  8.533156 10.065111 12.003988 14.000000
```

```
paste("So, I keep the", which.min(summary(subset)$cp), "variables.")
```

```
## [1] "So, I keep the 7 variables."
```

```
plot(subset, scale = "Cp")
```




```
coef(subset, which.min(summary(subset)$cp))
```

```
## (Intercept)         age      weight        neck     abdomen       thigh
## -33.2579912   0.0681658  -0.1194405  -0.4038021   0.9178850   0.2219598
##     forearm       wrist
##   0.5531394  -1.5324011
```

Comment:

I keep the 7 variables.

Part b)

So, before we get into this question, I want to make a summary, below:

1. M1 (backward elimination): keep four variables (weight, abdomen, forearm, wrist)

2. M2 (forward selection): keep four variables (weight, abdomen, forearm, wrist)

3. M3 (adjusted R^2): keep nine variables (age, weight, neck, abdomen, hip, thigh, bicep, forearm, wrist)

4. M4 (AIC): keep eight variables (age, weight, neck, abdomen, hip, thigh, forearm, wrist)

5. M5 (BIC): keep four variables (weight, abdomen, forearm, wrist)

6. M6 (MAlllow's Cp): keep seven variables (age, weight, neck, abdomen, thigh, forearm, wrist)

So, basically, the models: M1, M2, and M5 are the same, so we can test one of these three.

```
set.seed(10)
# #manually create the folds...
# datafold <- 1:nrow(data)
# foldlist <- list()
#
# pick <- c()
# picking <- function(pick, datanumber, foldlist){
#   for(i in 1:10){
#     pick[i] <- floor(runif(1, (floor(nrow(data) / 10) - 4), (ceiling(nrow(data) / 10) + 4)))
#   }
#
#   if(sum(pick) == nrow(data)){
#     pick <- pick
#   }else if(sum(pick) < nrow(data)){
#     dif <- nrow(data) - sum(pick)
#     pick[which.min(pick)] <- pick[which.min(pick)] + dif
#   }else{
#     dif <- sum(pick) - nrow(data)
#     pick[which.max(pick)] <- pick[which.max(pick)] - dif
#   }
#
#   for(i in 1:10){
#     foldlist[[i]] <- sample(datanumber[datanumber != 0], size = pick[i], replace = F)
#     if(datanumber[datanumber %in% foldlist[[i]]]){
#       datanumber[datanumber %in% foldlist[[i]]] <- 0
#     }
```

```
#   }
#   return(foldlist)
# }
#
# numberpick <- picking(pick, datafold, foldlist)
#
# #Check whether my function works fine...
# duplicated(numberpick)
#
# sum <- 0
# for (i in 1:10){
#   sum <- sum + length(numberpick[[i]])
# }
# sum



#-------------------------------------------------------------------------------------------
folds <- createFolds(1:nrow(data), k = 10)


lists <- list(one, two, three, four, five, six, seven, eight, nine, ten, eleven,
              twelve, thirteen)

xnumber <- c(4, 4, 9, 8, 4, 7)

mat <- matrix(0, 6, 10) #MSE matrix initiator

for(i in 1:10){
  ytest <- as.data.frame(data$bodyfat[folds[[i]]])
  ytrain <- as.data.frame(data$bodyfat[-folds[[i]]])
  for(j in 1:6){
    xtest <- lists[[xnumber[j]]][folds[[i]], ]
    xtrain <- lists[[xnumber[j]]][-folds[[i]], ]
    combined <- cbind(ytrain, xtrain); colnames(combined)[1] <- "y"
    coef <- as.matrix(lm(y ~., data = combined)$coefficients)
    yhat <- as.matrix(cbind(1, xtest)) %*% coef
    mat[j, i] <- mean((ytest - yhat)^2)
  }
}
rownames(mat) <- c("Backward", "Forward", "Adj R2", "AIC", "BIC", "Cp")
mat
```

```
##              [,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7]
## Backward 21.09559 33.19354 20.37970 19.70993 17.94818 14.69865 17.58010
## Forward  21.09559 33.19354 20.37970 19.70993 17.94818 14.69865 17.58010
## Adj R2   24.33017 30.56757 18.67189 19.98861 17.08435 13.34519 20.33148
## AIC      22.22739 30.90945 18.89282 19.87606 17.01640 13.68691 20.74537
## BIC      21.09559 33.19354 20.37970 19.70993 17.94818 14.69865 17.58010
## Cp       22.26475 31.85520 19.38565 19.23429 17.35549 13.36128 20.34595
##              [,8]     [,9]    [,10]
## Backward 15.67184 15.35730 20.40212
## Forward  15.67184 15.35730 20.40212
## Adj R2   17.77015 14.45168 18.99605
```

```
## AIC      17.16968 14.67760 18.93815
## BIC      15.67184 15.35730 20.40212
## Cp       17.31809 14.98892 19.24962
```

```
mse <- apply(mat, 1, mean)
conc <- paste0("The smallest average MSE is ", which.min(mse), "th model, so I keep the M",
           which.min(mse), " model, which has ", xnumber[which.min(mse)], " variables.")
print(conc)
```

```
## [1] "The smallest average MSE is 4th model, so I keep the M4 model, which has 8 variables."
```

```
paste("Variables keeped:", names(lists[[xnumber[which.min(mse)]]]))
```

```
## [1] "Variables keeped: age"     "Variables keeped: weight"
## [3] "Variables keeped: neck"    "Variables keeped: abdomen"
## [5] "Variables keeped: hip"     "Variables keeped: thigh"
## [7] "Variables keeped: forearm" "Variables keeped: wrist"
```

Comment:

I set seeded 10, just to get the same answer for all the times. I found out that the algorithm gies either AIC or Mallow's Cp for the best models. Again Cross validation is randomly performed, so the outperformed models might be different, but not switched all the times. So, I will say Mallow's Cp or AIC are the best!

What I tried to do is get MSE 10 times for each model, and get average for each model, and compare which model has the smallest average MSE.

As you can see above, 10-fold CV told us to keep **AIC** (sometimes Mallow's Cp if set seed 100) model.

Part c - i)

M is the model selected by AIC.

```
x <- lists[[xnumber[which.min(mse)]]]
combined <- cbind(data$bodyfat, x); colnames(combined)[1] <- "y"
fit <- lm(y ~., data = combined)
fit
```

```
##
## Call:
## lm(formula = y ~ ., data = combined)
##
## Coefficients:
## (Intercept)          age       weight         neck      abdomen
```

```
##   -22.65637      0.06578     -0.08985     -0.46656      0.94482
##         hip        thigh      forearm        wrist
##    -0.19543      0.30239      0.51572     -1.53665
```

Part c - ii)

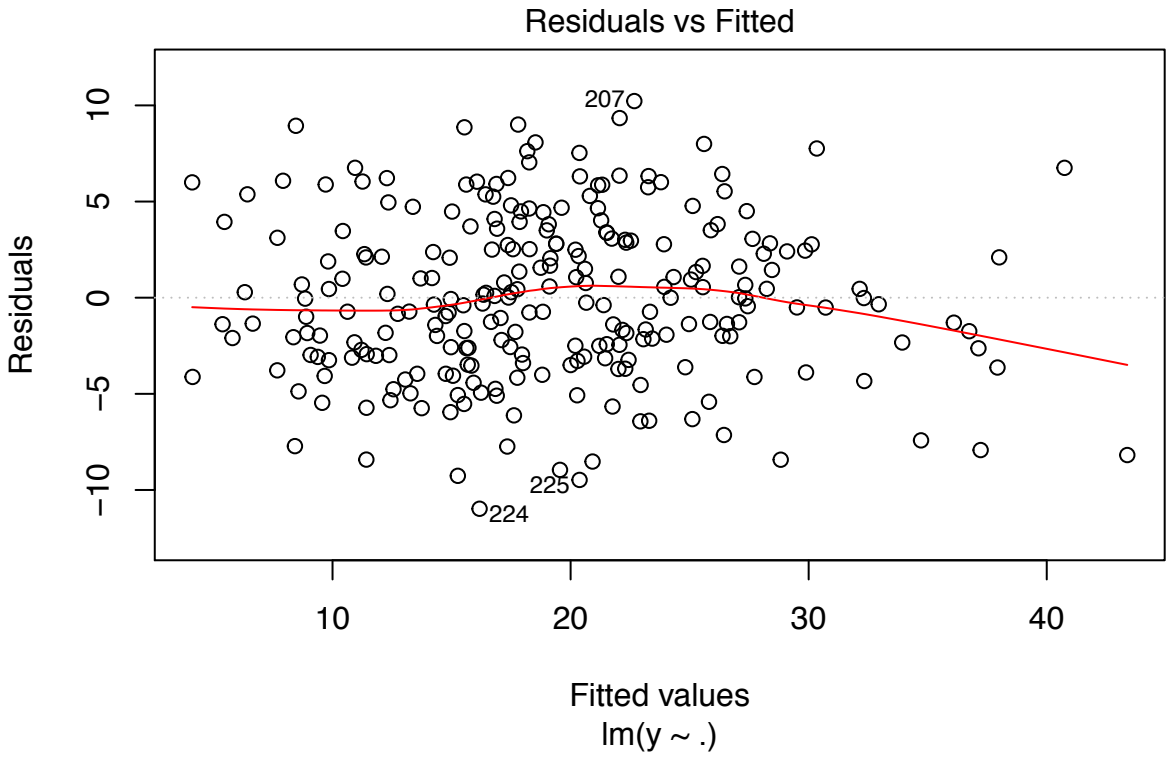
Here are the game plans, below:

1. Draw residuals v.s. fitted
2. Draw standardized residual v.s. fitted
3. Draw residual v.s. standardized residual
4. Draw predicted residual v.s. fitted
5. Draw residual v.s. predicted residual
6. Draw residual v.s. leverage
7. Draw residual v.s. standardized predicted residual
8. Cook's distance

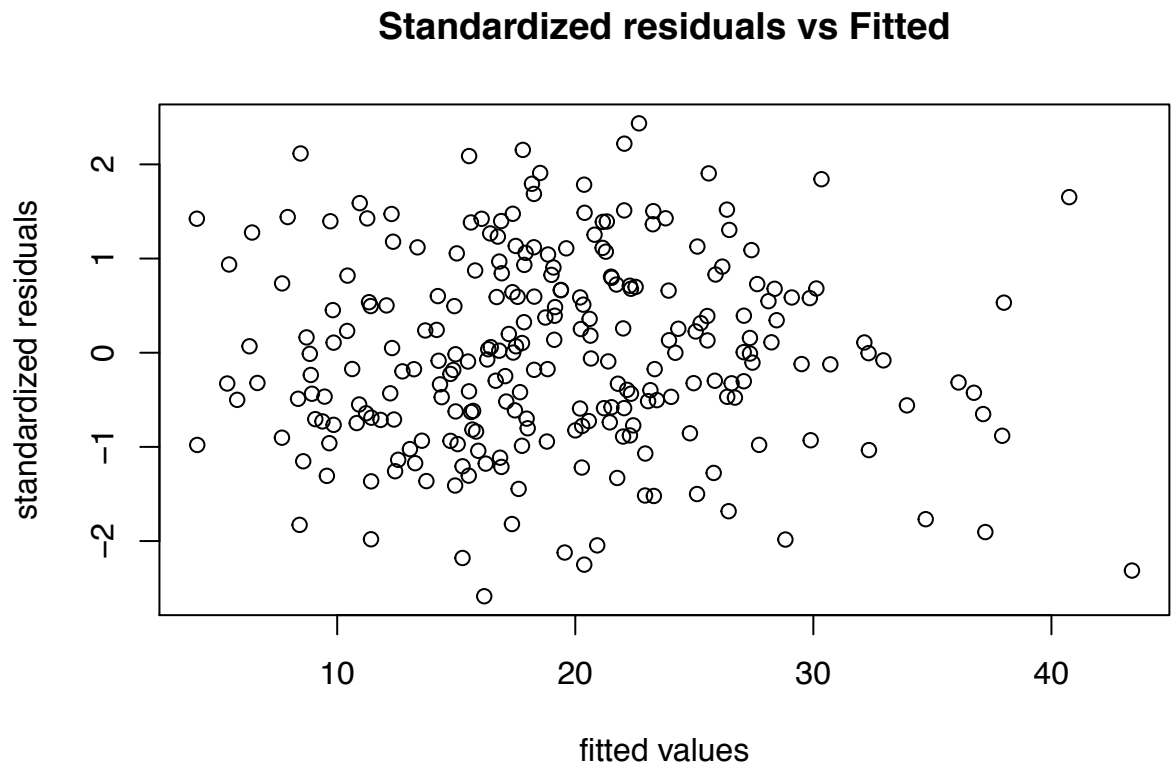
```
multiplot <- function(..., plotlist=NULL, file, cols=1, layout=NULL) {
  require(grid)

  # Make a list from the ... arguments and plotlist
  plots <- c(list(...), plotlist)

  numPlots = length(plots)

  # If layout is NULL, then use 'cols' to determine layout
  if (is.null(layout)) {
    # Make the panel
    # ncol: Number of columns of plots
    # nrow: Number of rows needed, calculated from # of cols
    layout <- matrix(seq(1, cols * ceiling(numPlots/cols)),
                     ncol = cols, nrow = ceiling(numPlots/cols))
  }

 if (numPlots==1) {
    print(plots[[1]])
```

```
  } else {
    # Set up the page
    grid.newpage()
    pushViewport(viewport(layout = grid.layout(nrow(layout), ncol(layout))))

    # Make each plot, in the correct location
    for (i in 1:numPlots) {
      # Get the i,j matrix positions of the regions that contain this subplot
      matchidx <- as.data.frame(which(layout == i, arr.ind = TRUE))

      print(plots[[i]], vp = viewport(layout.pos.row = matchidx$row,
                                      layout.pos.col = matchidx$col))
    }
  }
}
```

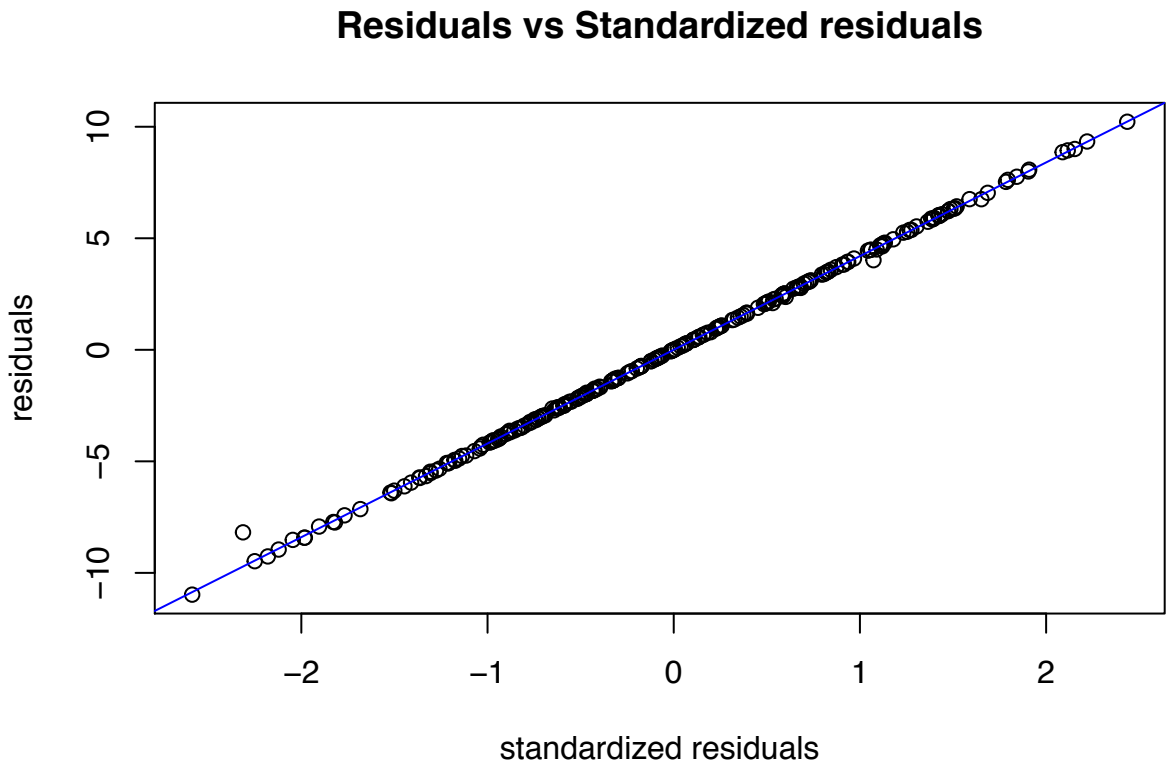
```
y <- data$bodyfat
X <- model.matrix(fit)
H <- X %*% solve(t(X) %*% X) %*% t(X)
yhat <- H %*% y
residual <- y - yhat
lev <- diag(H)
pre_res <- fit$residuals / (1 - lev)
cook <- rstandard(fit)^2 * (lev / ((1 - lev) * (ncol(X))))


plot(fit, which = 1)
```



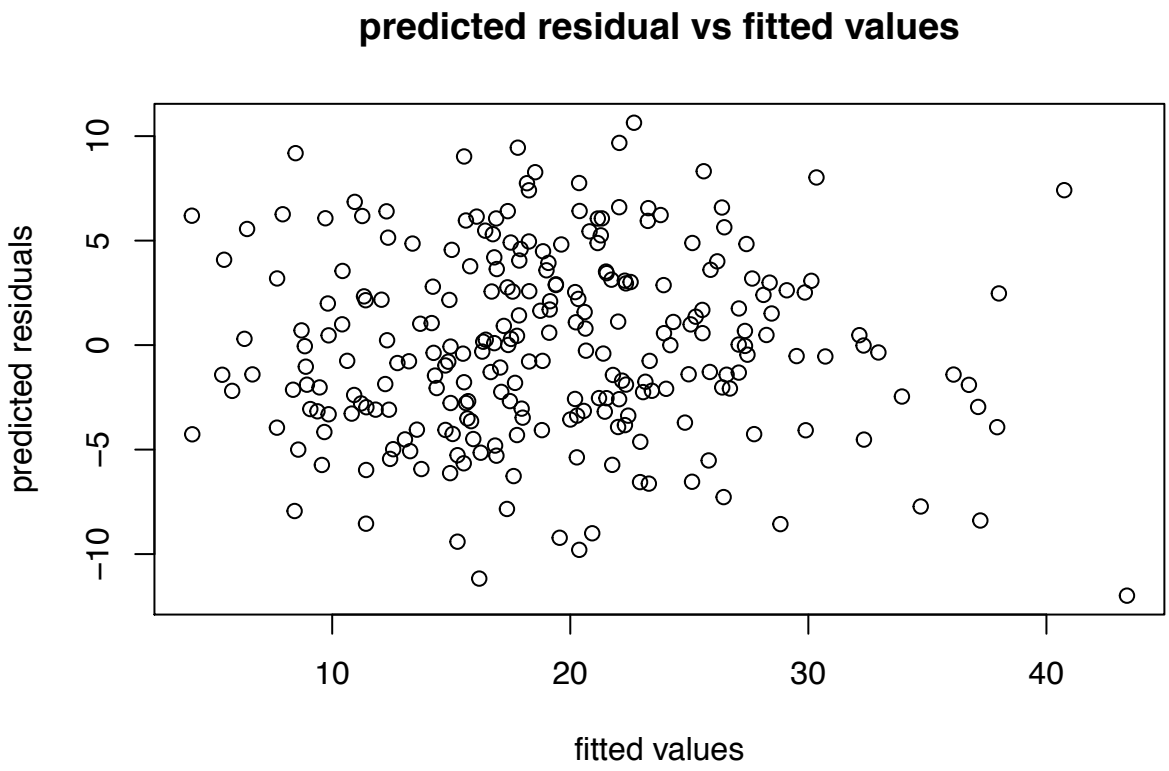

```
plot(fit$fitted.values, rstandard(fit), main = "Standardized residuals vs Fitted",
     xlab = "fitted values", ylab = "standardized residuals")
```



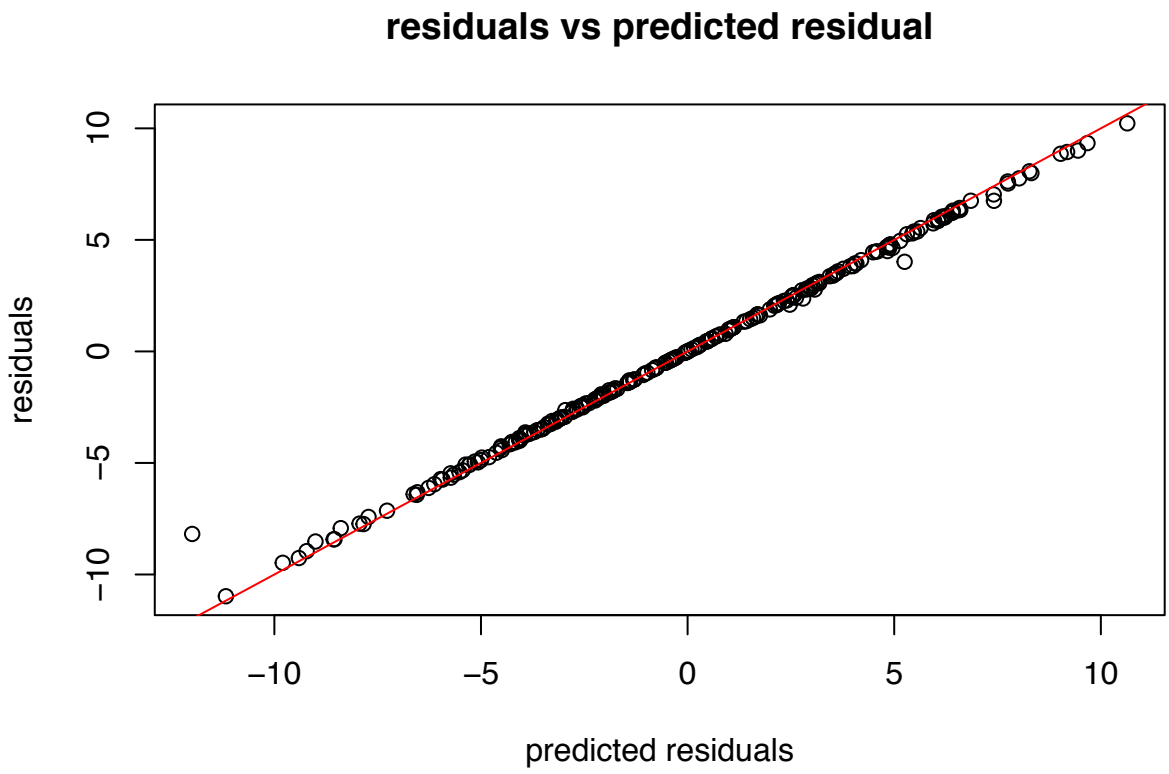

```
plot(rstandard(fit), fit$residuals, main = "Residuals vs Standardized residuals",
     xlab = "standardized residuals", ylab = "residuals")
abline(0, 4.2, col = "blue")
```



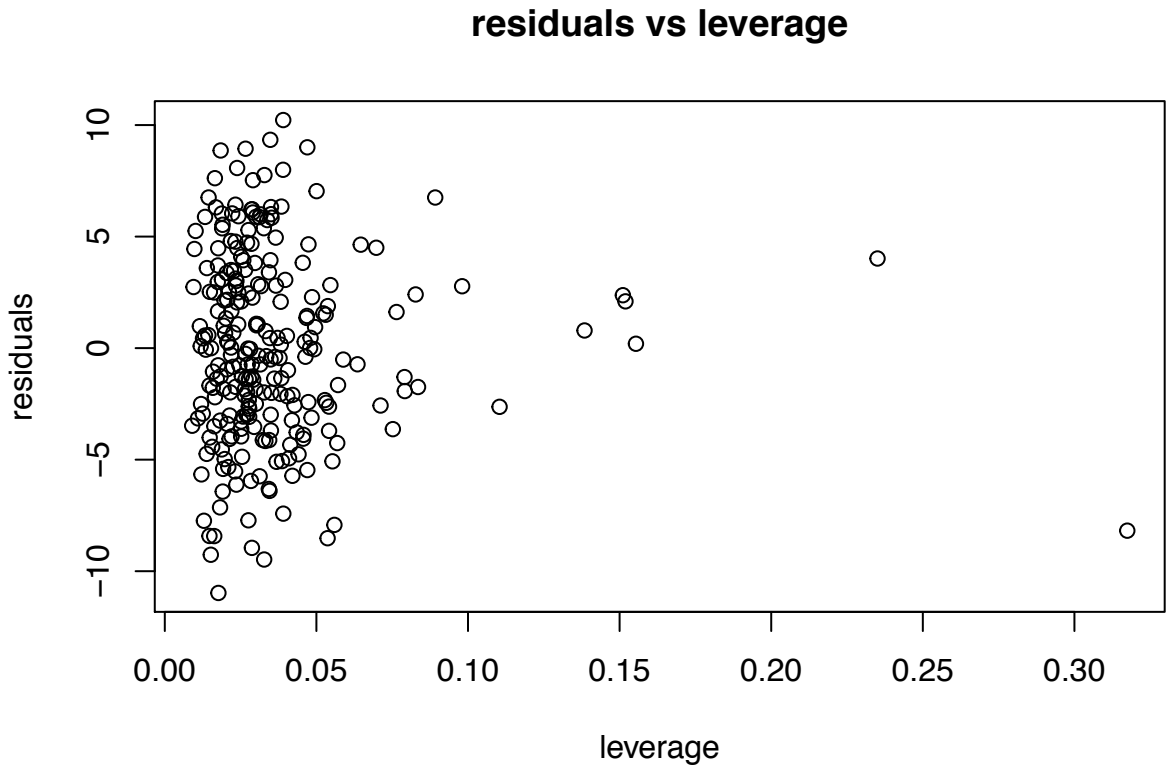
```r
plot(fit$fitted.values, pre_res, main = "predicted residual vs fitted values",
     xlab = "fitted values", ylab = "predicted residuals")
```



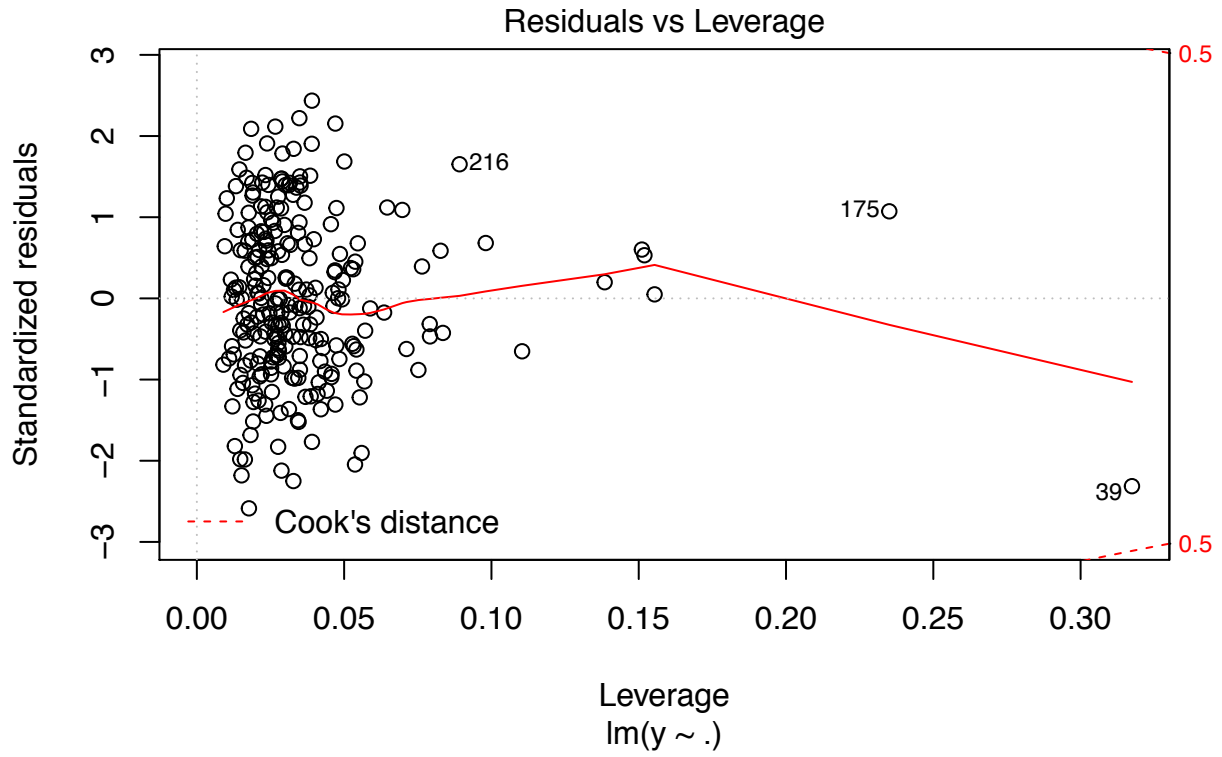

```r
plot(pre_res, residual, main = "residuals vs predicted residual",
     xlab = "predicted residuals", ylab = "residuals")
abline(0, 1, col = "red")
```



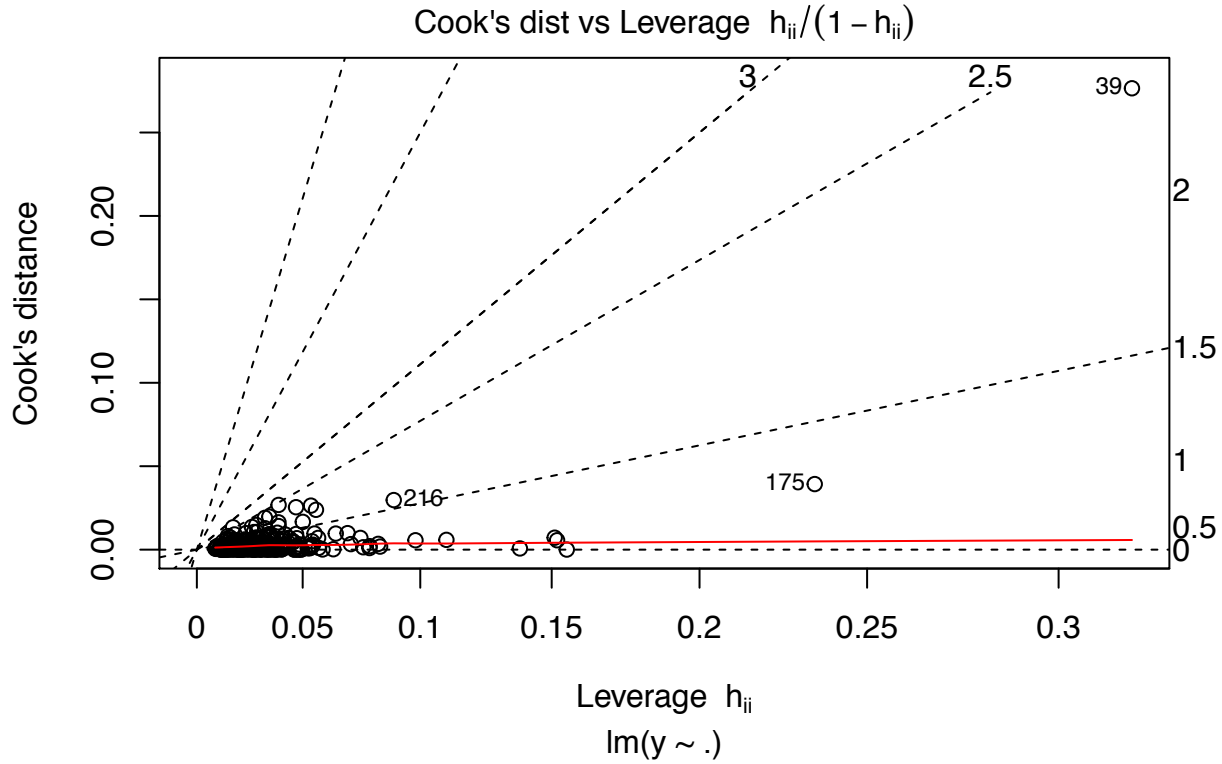

```
plot(lev, fit$residuals, main = "residuals vs leverage",
     xlab = "leverage", ylab = "residuals")
```



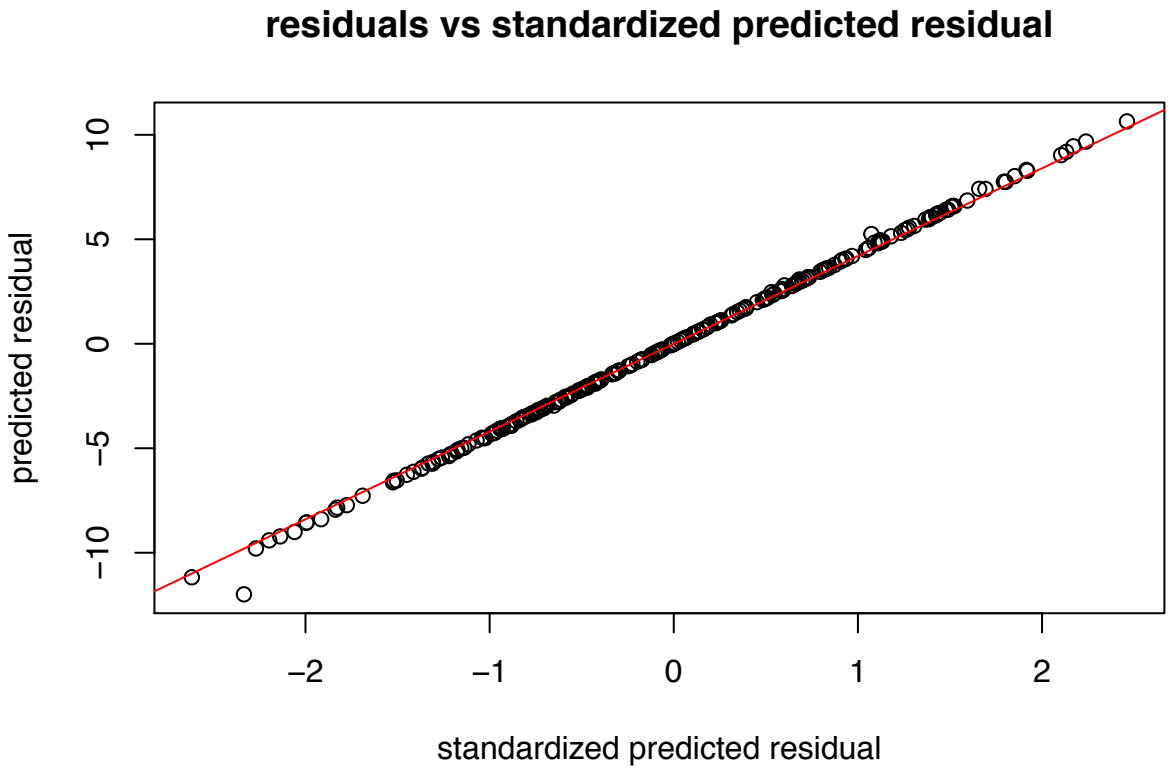
```r
plot(fit, which = 5)
```



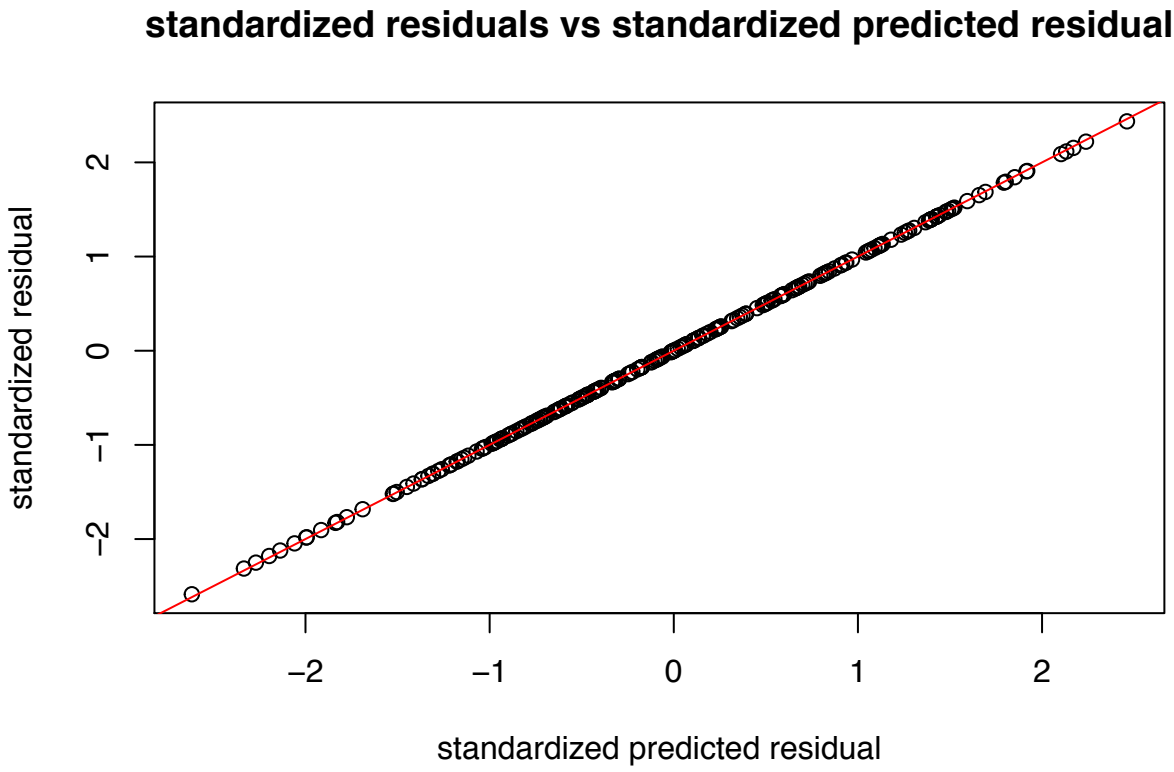

```r
plot(fit, which = 6)
```



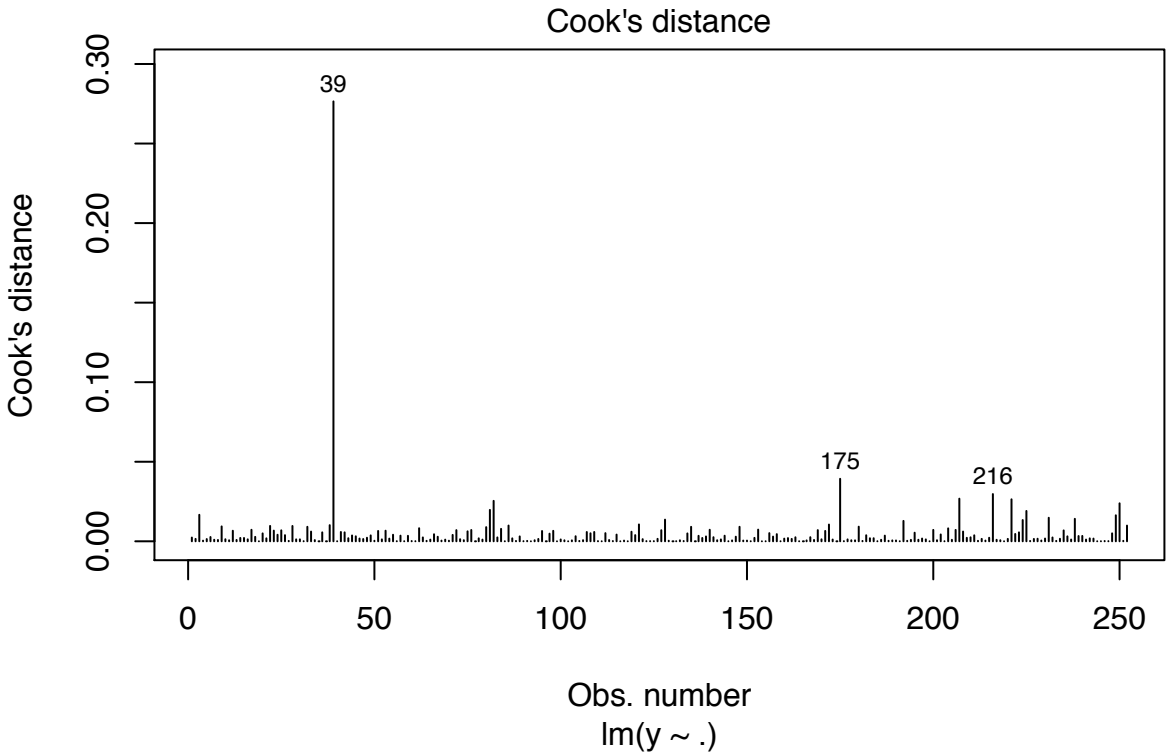
```r
plot(rstudent(fit), pre_res, main = "residuals vs standardized predicted residual",
     xlab = "standardized predicted residual", ylab = "predicted residual")
abline(0, 4.2, col = "red")
```




```r
plot(rstudent(fit), rstandard(fit),
     main = "standardized residuals vs standardized predicted residual",
     xlab = "standardized predicted residual", ylab = "standardized residual")
abline(0, 1, col = "red")
```

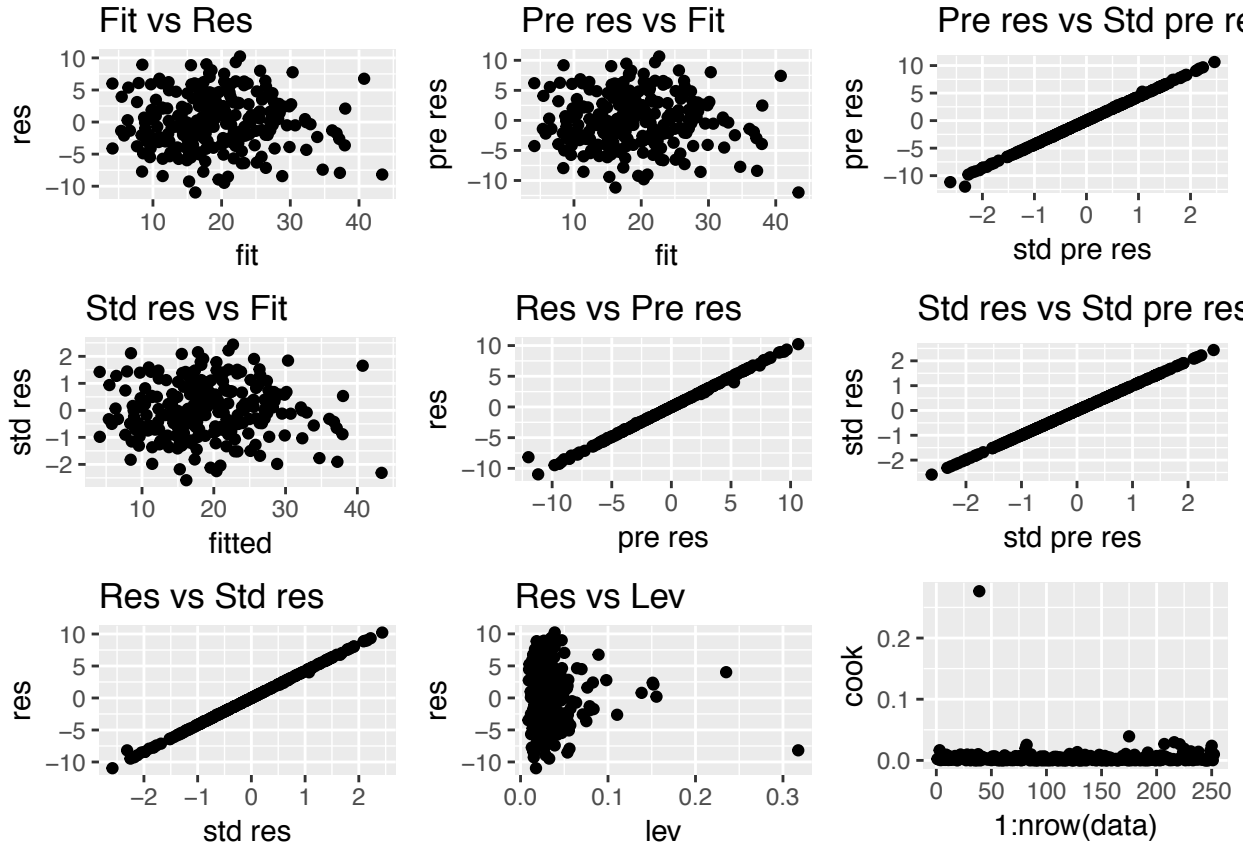



```
plot(fit, which = 4)
```




```
pic <- list()
pic[[1]] <- qplot(fit$fitted.values, fit$residuals, main = "Fit vs Res",
     xlab = "fit", ylab = "res")
```

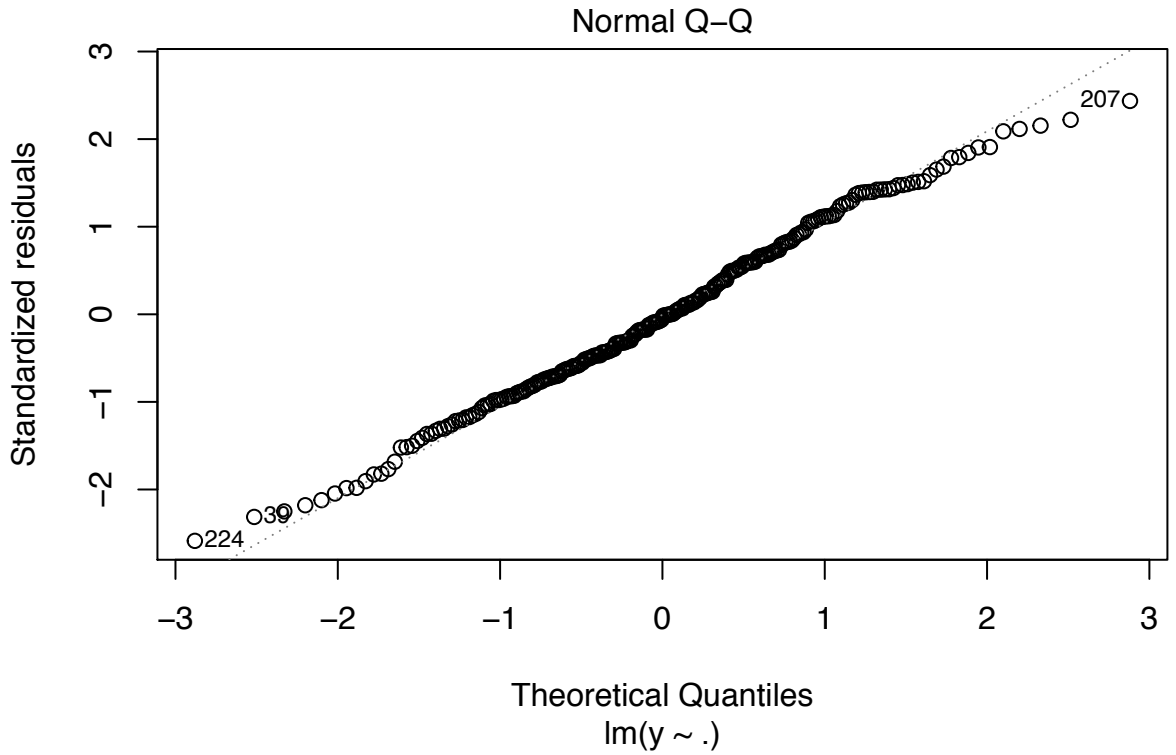
```
pic[[2]] <- qplot(fit$fitted.values, rstandard(fit), main = "Std res vs Fit",
     xlab = "fitted", ylab = "std res")
pic[[3]] <- qplot(rstandard(fit), fit$residuals, main = "Res vs Std res",
     xlab = "std res", ylab = "res")
pic[[4]] <- qplot(fit$fitted.values, pre_res, main = "Pre res vs Fit",
     xlab = "fit", ylab = "pre res")
pic[[5]] <- qplot(pre_res, fit$residuals, main = "Res vs Pre res",
     xlab = "pre res", ylab = "res")
pic[[6]] <- qplot(lev, fit$residuals, main = "Res vs Lev",
     xlab = "lev", ylab = "res")
pic[[7]] <- qplot(rstudent(fit), pre_res, main = "Pre res vs Std pre res",
     xlab = "std pre res", ylab = "pre res")
pic[[8]] <- qplot(rstudent(fit), rstandard(fit),
     main = "Std res vs Std pre res",
     xlab = "std pre res", ylab = "std res")
pic[[9]] <- qplot(1:nrow(data), cook)

multiplot(plotlist = pic, cols = 3)
```



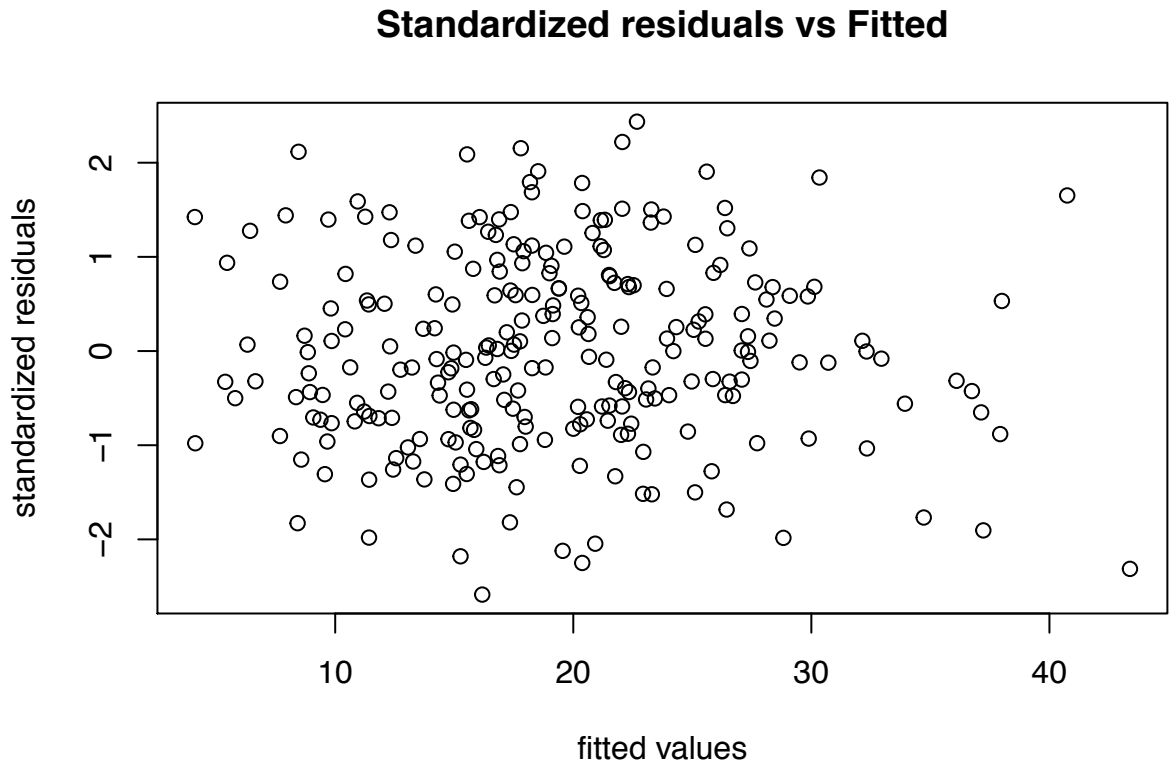
9. Added variable plot/partial regression plot
10. Component plus residual plot/partial residual plot
11. Transform x (if necessary, from the result of 10)
12. QQ plot

13. Transform y (if necessary)

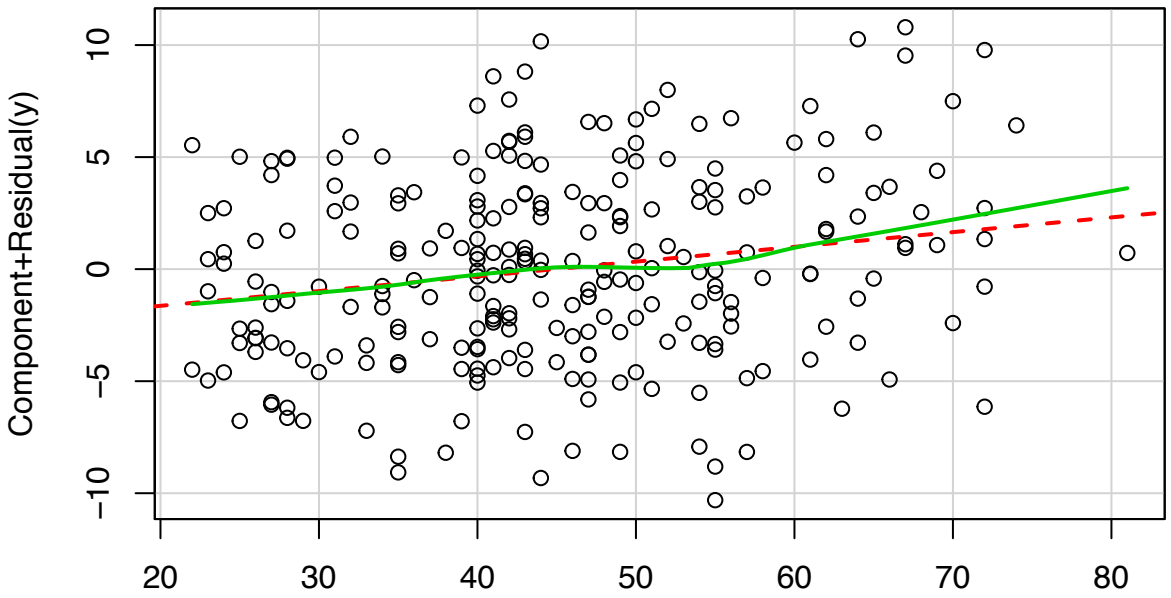
```
plot(fit, which = 2)
```

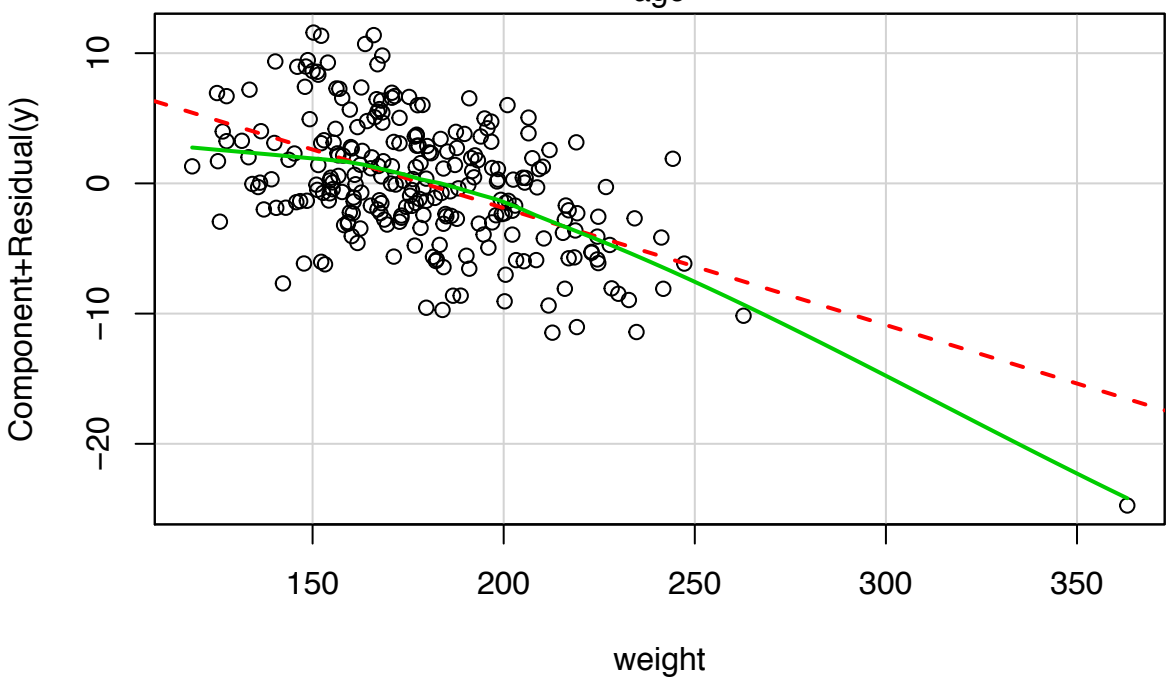


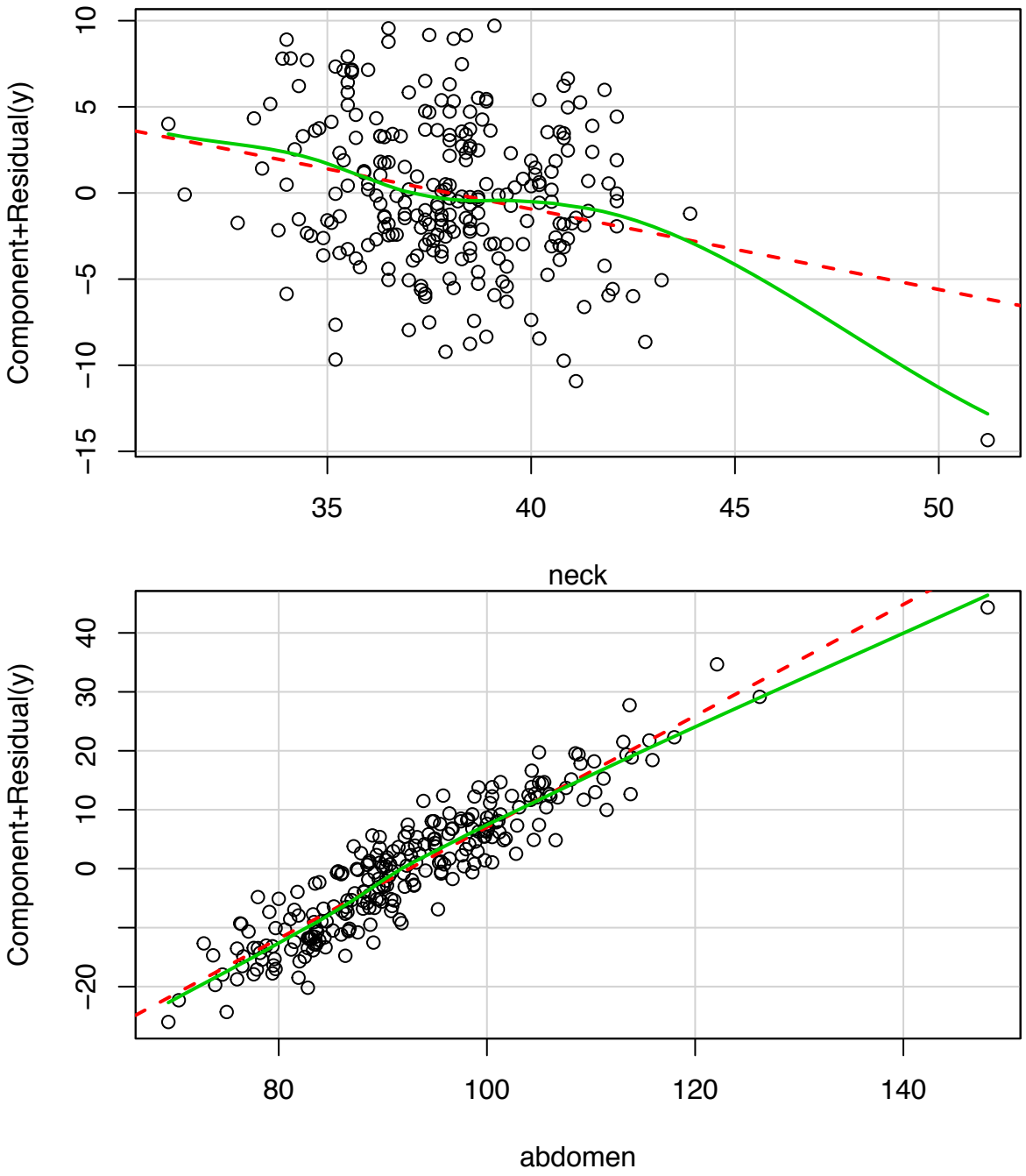

```
plot(fit$fitted.values, rstandard(fit), main = "Standardized residuals vs Fitted",
     xlab = "fitted values", ylab = "standardized residuals")
```

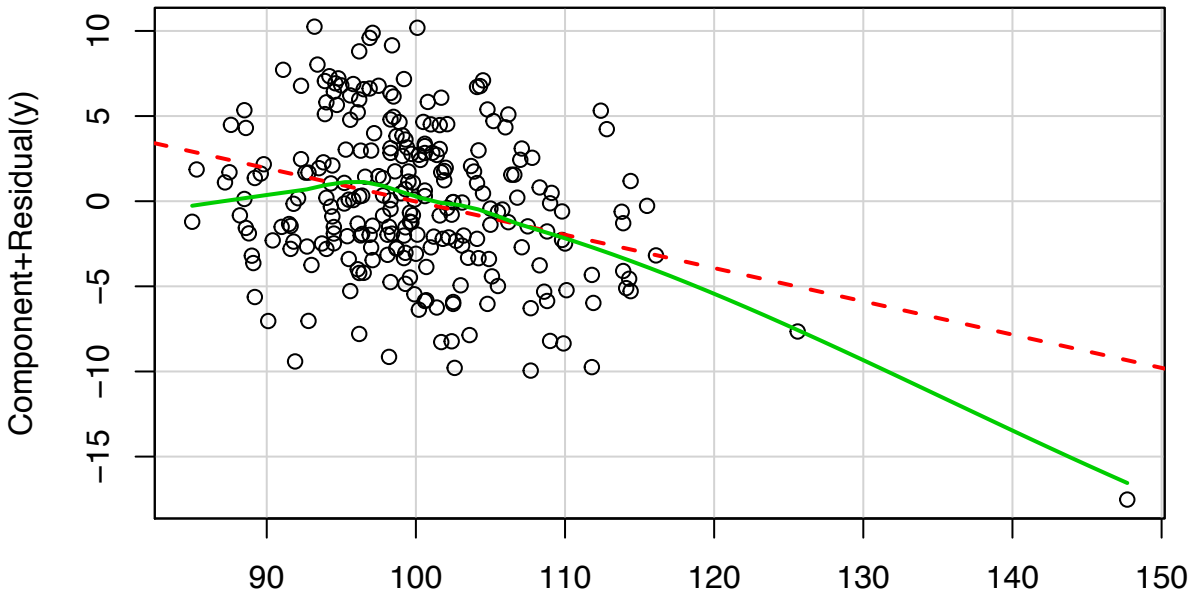


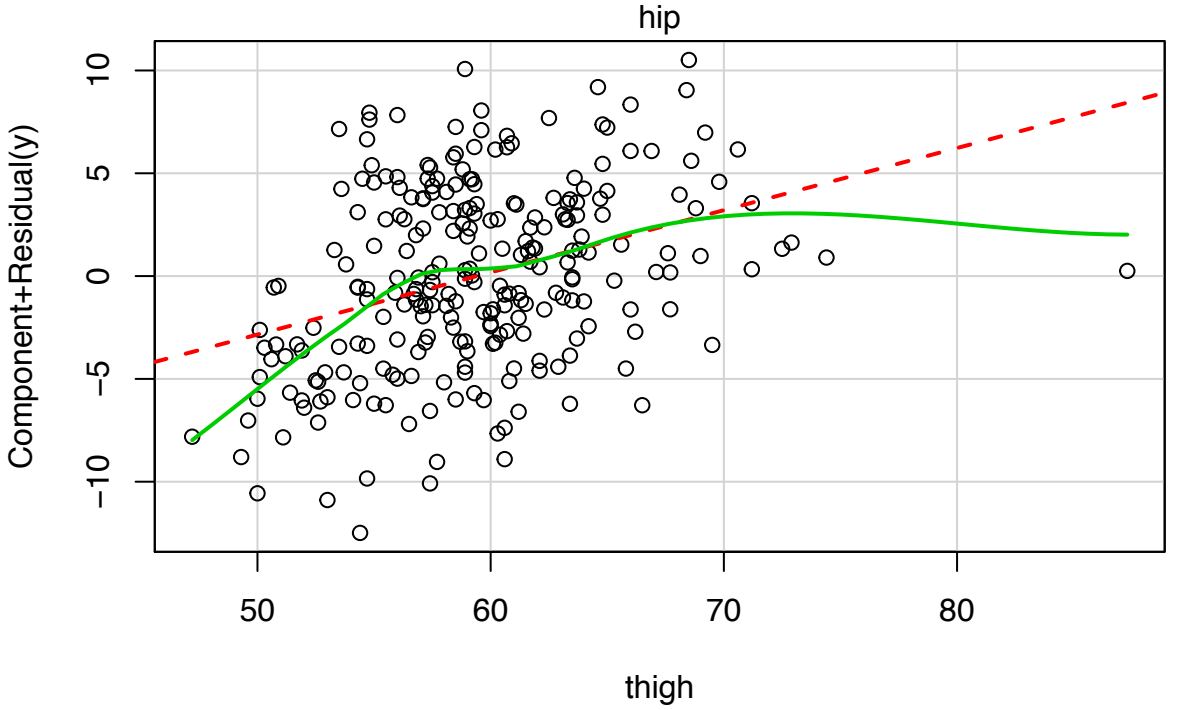
```
for(i in 1:ncol(x)){
  crPlot(fit, colnames(x)[i])
}
```

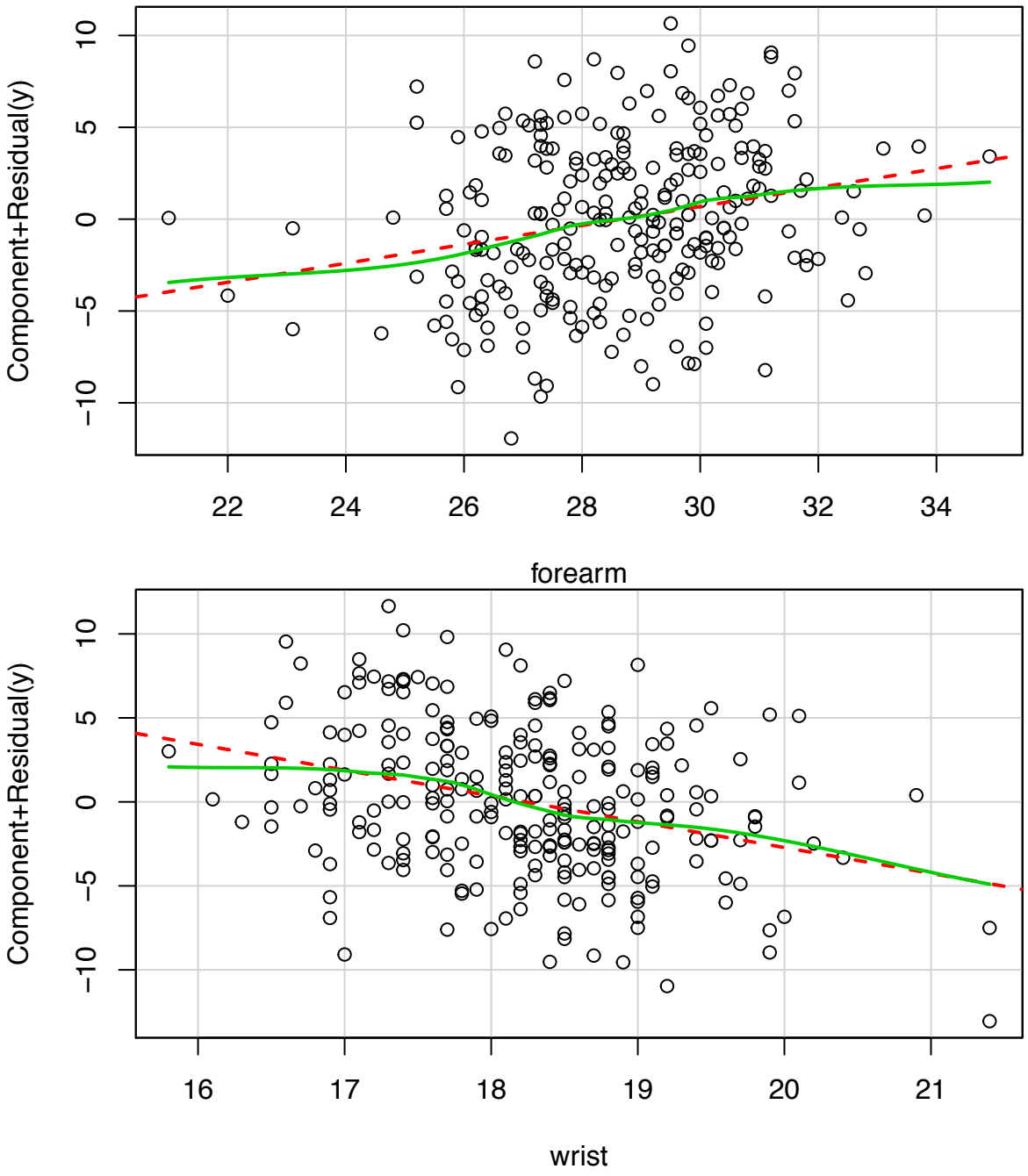





Component+Residual(y)
10
5
0
−5
−10
−15
35
40
45
50
neck
Component+Residual(y)
40
30
20
10
0
−20
80
100
120
140
abdomen


Component+Residual(y)
10
5
0
−5
−10
−15
90
100
110
120
130
140
150
hip


Component+Residual(y)
10
5
0
−5
−10
50
60
70
80
thigh



Comment:

Please see my comments below. . .

Part c - iii)

Comment:

As I can see from normal QQ plot, my model is slightly off - light tailed...

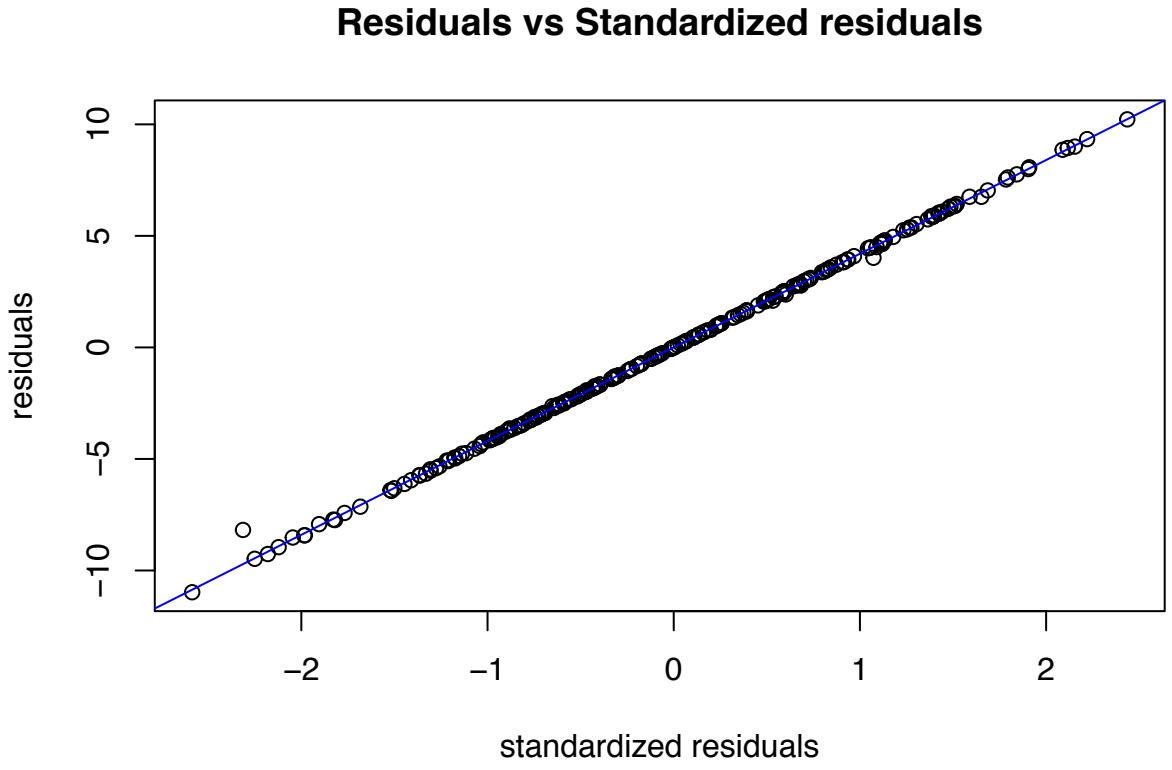
Also, from standardized residual v.s. fitted values, I can say that I do not really need to transform y to stabilize the variance. (they are quite closed enough to be homoscedastic)

From the component-plus residual plots, I can tell that the variables age, abdomen, forearm, and wrist satisfy linearity pretty well. And, by saying that, I would love to transform the rest variables: weight, neck, hip and thigh, to get better linearity.

Overall, linearity model works pretty well here.

Part c - iv)

```
plot(rstandard(fit), fit$residuals, main = "Residuals vs Standardized residuals",
     xlab = "standardized residuals", ylab = "residuals")
abline(0, 4.2, col = "blue")
```



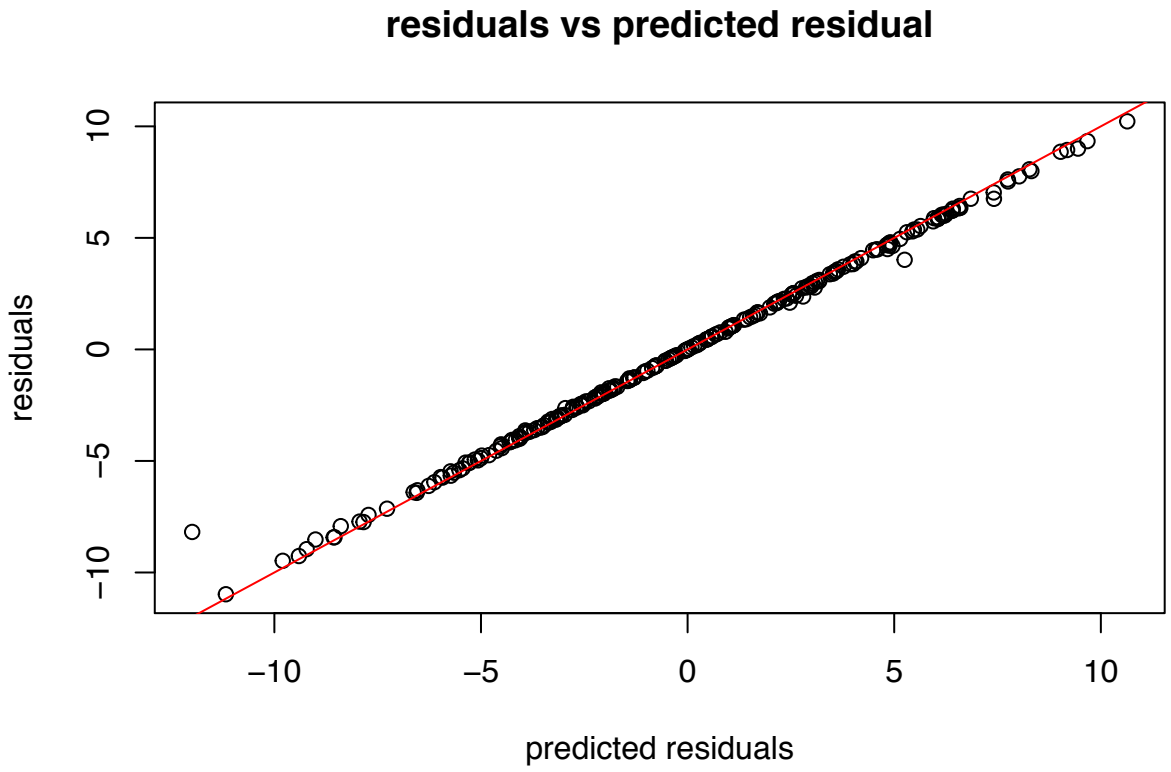

```
sort(rstandard(fit))[1:5]
```

```
##       224        39       225       204       231
## -2.586240 -2.313259 -2.250353 -2.180004 -2.121911
```

```
a <- rstandard(fit)
b <- fit$residuals
c <- as.data.frame(cbind(a, b))
d <- c[order(c$a),]
head(d)
```

```
##             a          b
## 224 -2.586240 -10.975670
## 39  -2.313259  -8.183384
## 225 -2.250353  -9.476586
## 204 -2.180004  -9.263457
## 231 -2.121911  -8.954333
## 221 -2.045877  -8.521823
```

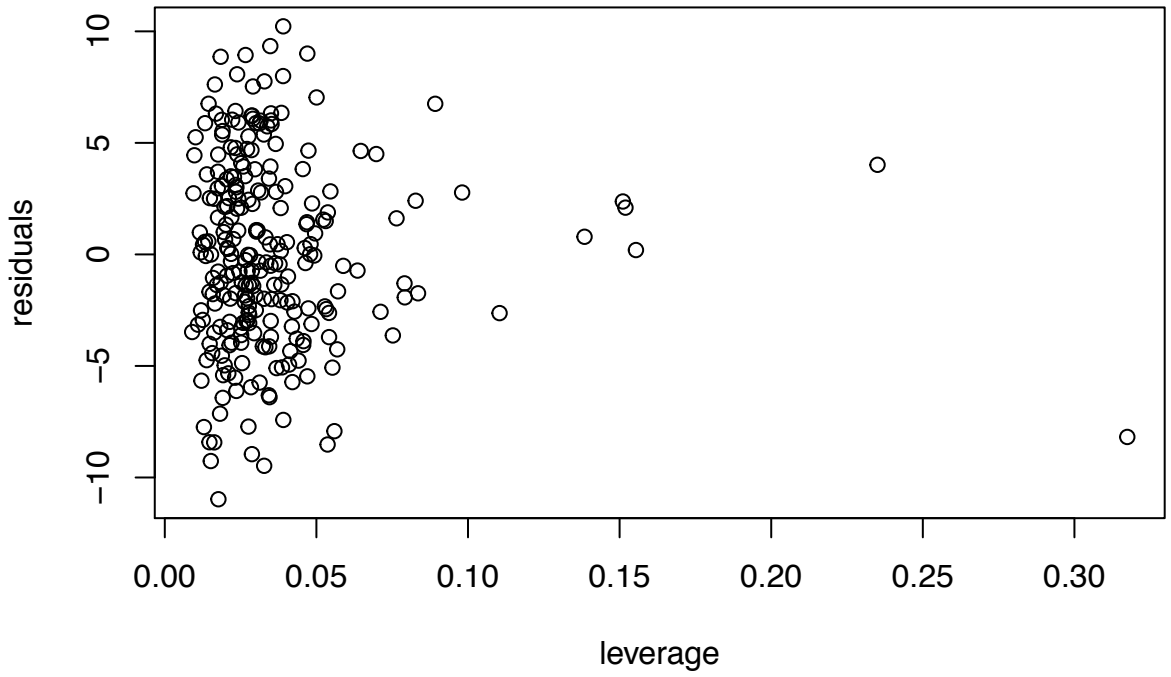
```
plot(pre_res, residual, main = "residuals vs predicted residual",
     xlab = "predicted residuals", ylab = "residuals")
abline(0, 1, col = "red")
```




```
sort(pre_res)[1:5]
```

```
##         39        224        225        204        231
## -11.989153 -11.173256  -9.797662  -9.406211  -9.219217
```

```
plot(lev, fit$residuals, main = "residuals vs leverage",
     xlab = "leverage", ylab = "residuals")
```




```
print("unusual residual....")
```

```
## [1] "unusual residual...."
```

```
head(sort(fit$residuals))[1:5]
```

```
##        224        225        204        231        221
## -10.975670  -9.476586  -9.263457  -8.954333  -8.521823
```

```
tail(sort(fit$residuals))[1:5]
```

```
##       86      135      128       82       81
## 8.075620 8.860413 8.939279 9.003328 9.339520
```

```
print("Unusual Leverage....")
```

```
## [1] "Unusual Leverage...."
```

```
tail(sort(lev), 10)
```

```
##        242        216         42         41        106        206
## 0.08352023 0.08918622 0.09806326 0.11041035 0.13842776 0.15107564
##         36        159        175         39
## 0.15193490 0.15538593 0.23505734 0.31743439
```

```
print("Cook's distance....")
```

```
## [1] "Cook's distance...."
```

```
tail(sort(cook), 10)
```

```
##          3        225         81        250         82        221
## 0.01664947 0.01906400 0.01974411 0.02388011 0.02542917 0.02638681
##        207        216        175         39
```

```
## 0.02679949 0.02970340 0.03929327 0.27651294
```

Comment:

When I saw the residual plot v.s. fitted values and stanardized residual v.s. fitted values, there is slightly heteroscedascity (as fitted value goes up, variance of residual/erros seem decrease) but I cannot really see any significant outliers. As we learned in the lectures, usually, in convention, we say that when absolute magnitude of standardized residual is bigger than 3, it is large, but I cannot see any. And, for residuals v.s. standardized residual plot, I can see a dot far off on the left hand side, and slightly off near x axis being 1. As I found (please refer to the code above), **39th and 175th** observations might have unusual leverage. (not on the blue line) Also, on residual v.s. predicted residual plot, **39th and 175th** observatinos have shown quite enough evidence of having high leverages. The residual v.s. leverage graph makes it more clear. No observation was above cook's distance = 0.5, which is a good signl however, 39th (the highest leverage - although it was not really high: little bit above 0.30 but less than 0.35) and 175th are showing higher leverage than usual observations. Also, this diagram shows the possibility of outliers: **negative residuals: 224th, 225th, and 204th & positive residuals: 81th and 82th** Also, **36th, 159th, and 206th** might be potential high leverage observations. Lastly, when I looked at the cook's distance v.s. observations number, I can obivously find **39th and 175th**. Also, unexpectedly, 216th, 221th, and 207th observations show quite high cook's distance.

I will say 39th and 175th are outliers, and 36th, 159th, and 206th might be potential high leverage points, and 224th, 225th, 204th, 81th, and 82th might be potential outliers.

Part c - v)

```
x <- lists[[xnumber[which.min(mse)]]]
combined <- cbind(data$bodyfat, x); colnames(combined)[1] <- "y"
combined_deleted <- combined[-c(39, 175), ]

refit <- lm(y ~., data = combined_deleted)

print("Original fit:")
```

```
## [1] "Original fit:"
```

```
summary(fit)
```

```
##
## Call:
```

```
## lm(formula = y ~ ., data = combined)
##
## 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
```

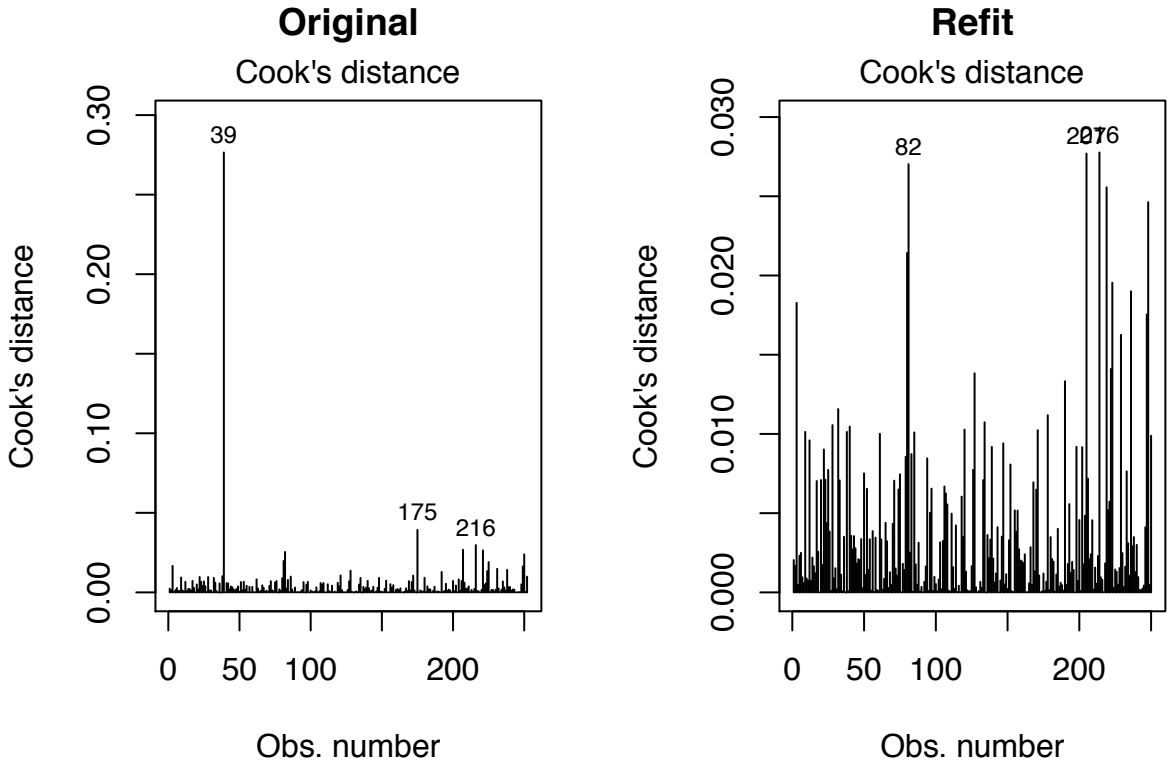
```
print("Refit:")
```

```
## [1] "Refit:"
```

```
summary(refit)
```

```
##
## Call:
## lm(formula = y ~ ., data = combined_deleted)
##
## 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
```

```r
par(mfrow = c(1,2))
plot(fit, which = 4, main = "Original")
plot(refit, which = 4, main = "Refit")
```



Comment:

I am only going to take out two significantly infulential & outlying observations: **39th and 175th**.

After refitting, I can see the cook's distances are significantly go down (ever observation is now below 0.030)!!!