Homework 5 Peer Assignment

```
library(faraway)
```

```
## Warning: package 'faraway' was built under R version 3.5.3
```

```
data = faraway::fat
attach(data)
new_data <- data[, -c(1,3)]
dim(new_data)
```

```
## [1] 252  16
```

```
head(new_data)
```

```
##   siri age weight height adipos  free neck chest abdom   hip thigh knee
## 1 12.3  23 154.25  67.75   23.7 134.9 36.2  93.1  85.2  94.5  59.0 37.3
## 2  6.1  22 173.25  72.25   23.4 161.3 38.5  93.6  83.0  98.7  58.7 37.3
## 3 25.3  22 154.00  66.25   24.7 116.0 34.0  95.8  87.9  99.2  59.6 38.9
## 4 10.4  26 184.75  72.25   24.9 164.7 37.4 101.8  86.4 101.2  60.1 37.3
## 5 28.7  24 184.25  71.25   25.6 133.1 34.4  97.3 100.0 101.9  63.2 42.2
## 6 20.9  24 210.25  74.75   26.5 167.0 39.0 104.5  94.4 107.8  66.0 42.0
##   ankle biceps forearm wrist
## 1  21.9   32.0    27.4  17.1
## 2  23.4   30.5    28.9  18.2
## 3  24.0   28.8    25.2  16.6
## 4  22.8   32.4    29.4  18.2
## 5  24.0   32.2    27.7  17.7
## 6  25.6   35.7    30.6  18.8
```

```
set.seed(123)
train <- sample(1:nrow(new_data), size = round(nrow(new_data)* 0.8), replace = FALSE)
train_data <- new_data[train,]
test_data <- new_data[-train,]
dim(train_data)
```

```
## [1] 202  16
```

```
dim(test_data)
```

```
## [1] 50 16
```

Question 1: Exploratory Data Analysis

a). Begin by doing some exploratory analysis. Plot scatterplots to examine the relationships between all the explanatory variables and the response siri. Do you observe any relationship of siri with any of the predictors? Do you visually observe any outlier or high leverage point in any of the plots? Briefly note down your observations.

```
library(car)
```

```
## Warning: package 'car' was built under R version 3.5.3
```

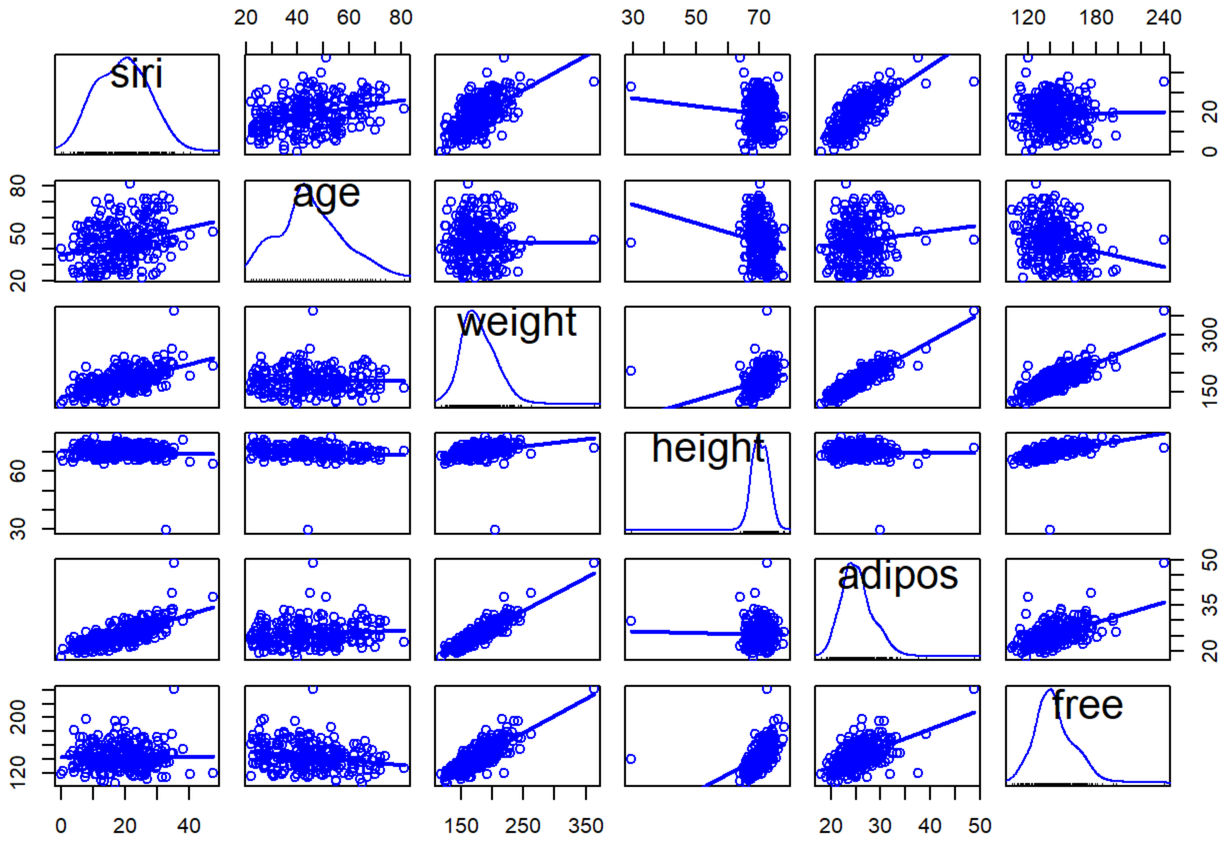
```
## Loading required package: carData
```

```
## Warning: package 'carData' was built under R version 3.5.2
```

```
##
## Attaching package: 'car'
```

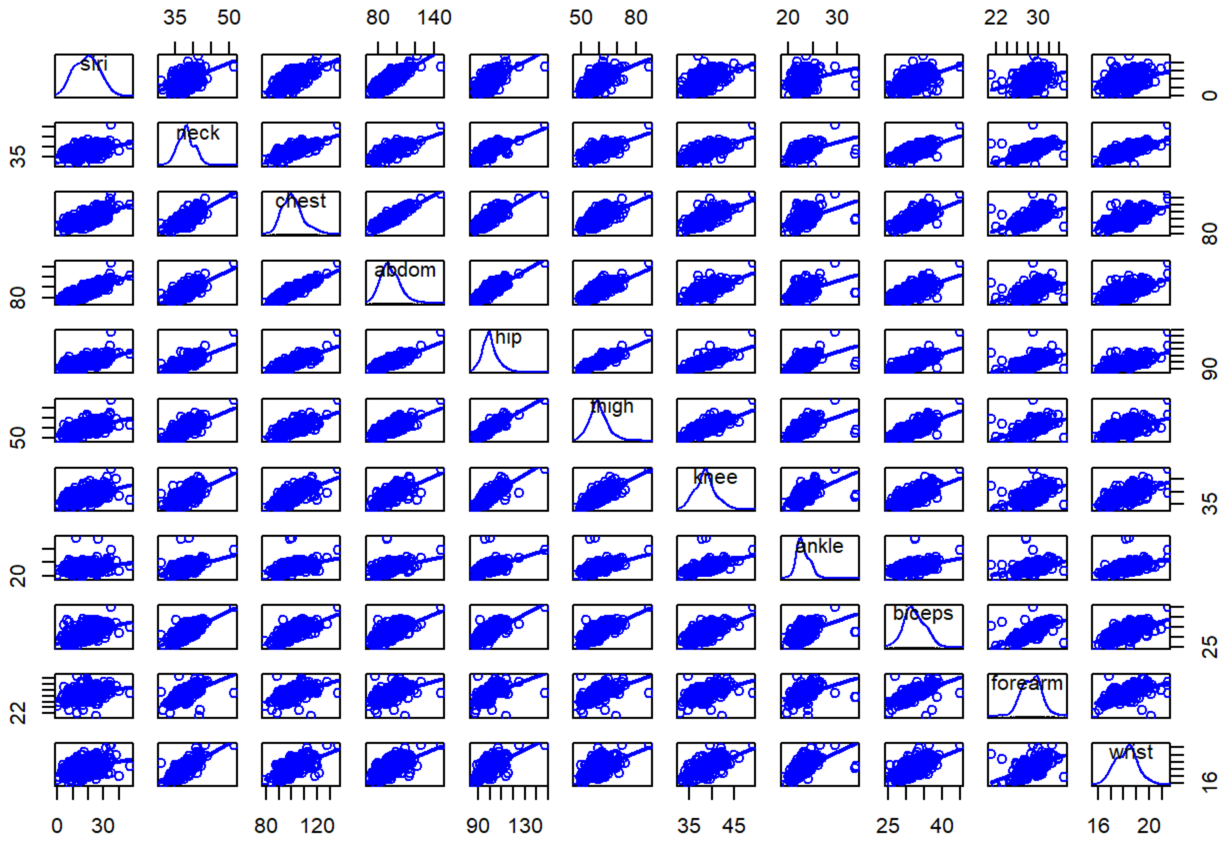
```
## The following objects are masked from 'package:faraway':
##
##     logit, vif
```

```
scatterplotMatrix(~ siri + age + weight + height + adipos + free, data = new_data, sm
ooth = FALSE)
```



#### **Answer:** For clear intepretation, we divide into two groups for scatter plots. This scatter plot is about the relationships between siri and several explatory variables respectively. We found that the relatiship is very strong and positive between siri/weight, siri/adipos. However, the relationship is weak between siri/age, siri/height, siri/free. In addition, there are several obvious outliers, which can be almostly found in every scatter plot such as plots of siri/weight, siri/height, siri/adipos, siri/free and others.

```
scatterplotMatrix(~ siri + neck + chest + abdom + hip + thigh + knee + ankle + biceps
+ forearm + wrist, data = new_data, smooth = FALSE)
```



#### **Answer:** From the scatter plots of siri and circumference measurements, we found some of them have some strong relationship, especially siri/abdom, siri/chest, siri/hip, siri/thigh, siri/knee and siri/biceps. Among them, the siri/abdom seems to be the strongest positive relationship. There are also some very weak relationship, such as siri/neck, siri/ankle, siri/forearm and siri/wrist. In addition, there are also several outliers which are almostly found in every scatterplot.

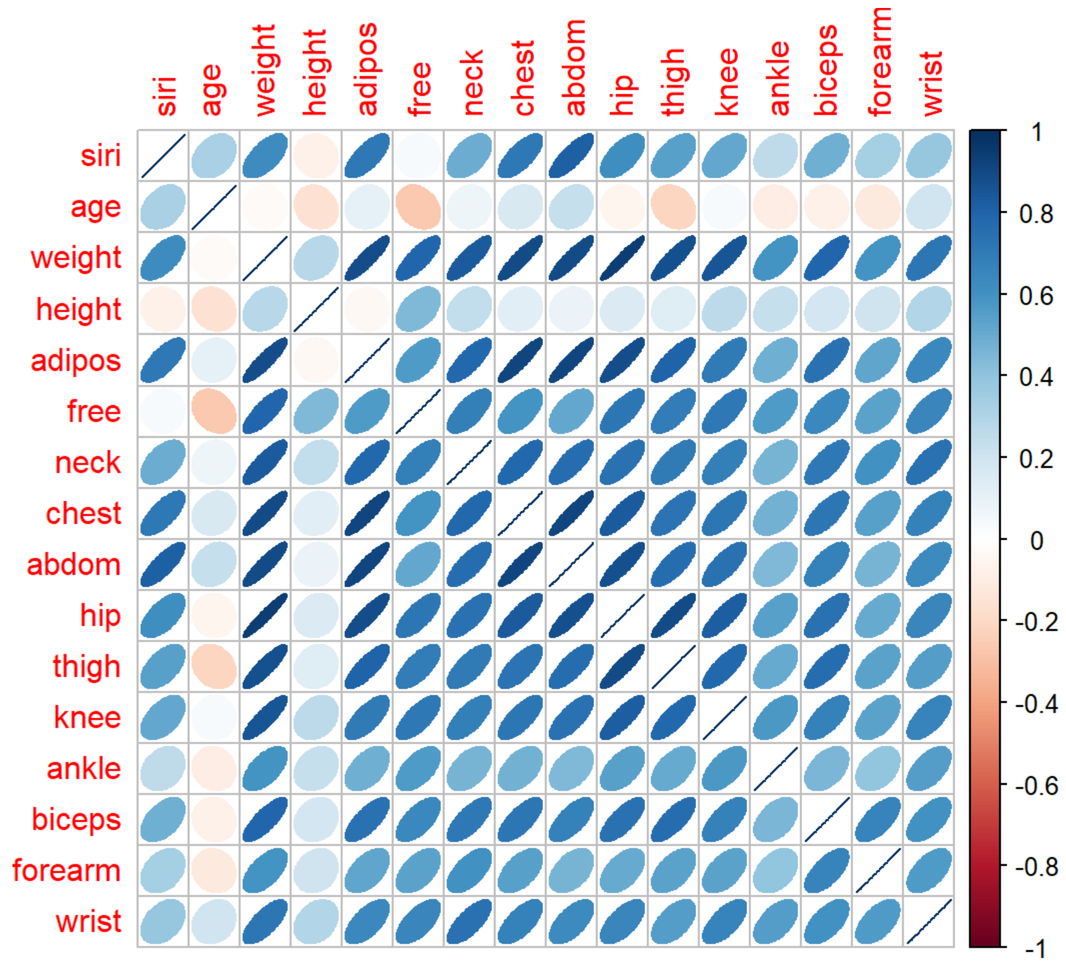
b). Plot a correlation matrix of all variables vs all other variables. Which are the variables that are most strongly correlated with siri? Did you expect these results based on your insights from the previous scatterplots?

```
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 3.5.1
```

```
## corrplot 0.84 loaded
```

```
corrplot(cor(train_data), method = 'ellipse')
```



Answer: From the plot of correlation matrix, we found that the most strongly correlated relationship is between siriand abdom. In addition, siri/adipos, siri/chest, siri/weight also have strong relationship. The very week relationship includes siri/height, siri/free and others. All the results are consistent with scatter plot in figure 1a.

[Note] The above code plots an inter-correlation matrix of all predictors. The darker and thinner ellipses mean stronger correlation while lighter and thicker ellipses mean weak correlations. Blue stands for a positive and red for a negative relationship.

c). From the plot in 1b, do you find any multicollinearity among the predictors? Which set of predictors are correlated with each other?

Answer: Among exploatory variables, weight seems to have multicollinearity because it has strong relationships with other exploatory variables (such as adipos, neck, chest, abdom, hip, thigh, knee). In addition, adipos seems to have strong relationships with weight, chest, abdom and hip; abdom seems to have strong relationships with other variables such as weight, adipos, chest and hip; hip seems to have strong relationship with variables such as weight, adipos, abdom and thigh. Also, chest seems to have strong relationships with weight, adipos and abdom.

d). Another way to detect multicollinearity in a model is by using Variance Inflation Factor (VIF). Fit a linear regression model on your training data with siri as response vs all other variables as predictors. Use the vif() function in R to calculate VIFs of the predictors. Which variables have VIFs more than 10? Do you detect multicollinearity among the predictors?

```
model_1 = lm(siri ~., data = train_data)
vif(model_1)
```

```
##       age    weight    height    adipos      free      neck     chest
##  2.352316 47.192294  2.128310 15.118745  6.510406  4.644633 11.202413
##     abdom       hip     thigh      knee     ankle    biceps   forearm
## 18.664060 18.939566  7.810564  4.847792  1.830010  3.568352  2.197986
##     wrist
##  3.401136
```

Answer: According to the result of VIF above, the vif values of weight, adipos, chest, abdom and hip are above 10, which means multicollinearity, these results are consistent with 1a and 1b.

e). A third way to detect multicollinearity is to calculate the condition numbers of the predictor covariance matrix. Use the eigen() function in R to calculate the eigenvalues of the covariance matrix. Then calculate the condition numbers associated with all eigenvalues relative to the largest eigenvalue. How many condition numbers are greater than 30? Do you detect multicollinearity?

```
# XTX which is the un-scaled and un-centered, we need scale for co-variance.
scaled <- scale(train_data, center = TRUE, scale = TRUE)
res_cov <- cov(scaled)
# Eigensystem analysis
val = eigen(res_cov)$values
val
```

```
##  [1] 9.783451849 1.908899530 1.056147847 0.710299663 0.657566412
##  [6] 0.523073189 0.334616959 0.267148603 0.227270860 0.189526125
## [11] 0.132531917 0.077413654 0.058018457 0.038337038 0.029599304
## [16] 0.006098593
```

```
# Condition number:
kj = sqrt(max(val)/min(val))
kj
```

```
## [1] 40.05265
```

```
## or method:  sqrt(kappa(cor(scaled), exact = TRUE))
```

Answer: The condition numbers kj = 40.05, and kj > 30, which means that there do exist multicollinearity among exploatroy variables.

Question 2: Fitting Linear Regression Model.

a). Fit a linear regression model on your training data with siri as response vs all other variables as predictors (same as you did before). Which predictors are significant at the 99% confidence level?

```
model_2 <- lm(siri ~., data = train_data)
summary(model_2)
```

```
##
## Call:
## lm(formula = siri ~ ., data = train_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.4741 -0.6090  0.2642  0.8741  6.3193
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -11.835476   7.065828  -1.675 0.095609 .
## age           0.009308   0.013270   0.701 0.483927
## weight        0.357696   0.025048  14.281  < 2e-16 ***
## height        0.035673   0.040942   0.871 0.384706
## adipos       -0.489356   0.114082  -4.289 2.87e-05 ***
## free         -0.555035   0.015229 -36.446  < 2e-16 ***
## neck         -0.020285   0.095878  -0.212 0.832678
## chest         0.111983   0.043295   2.586 0.010460 *
## abdom         0.156034   0.042967   3.632 0.000364 ***
## hip          -0.023124   0.064082  -0.361 0.718622
## thigh         0.175200   0.057061   3.070 0.002458 **
## knee          0.131478   0.098519   1.335 0.183655
## ankle         0.130614   0.083391   1.566 0.118980
## biceps        0.130303   0.068558   1.901 0.058898 .
## forearm       0.241527   0.083371   2.897 0.004220 **
## wrist         0.216391   0.215247   1.005 0.316052
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.545 on 186 degrees of freedom
## Multiple R-squared:  0.9686, Adjusted R-squared:  0.966
## F-statistic: 382.1 on 15 and 186 DF,  p-value: < 2.2e-16
```

Answer: Those predictors are significant at the 99% confidence level (0.01) includes varibles of weight, adipos, free, abdom, thigh and forearm.

b). Build a new model on the training data with only the predictors that are statistically significant at the 99% confidence level. Perform an ANOVA test to compare this new model with the full model. Which one would you prefer? Explain

```
model_3 <- lm(siri ~ weight + adipos + free + abdom + thigh + forearm, data = train_d
ata)
summary(model_3)
```

```
##
## Call:
## lm(formula = siri ~ weight + adipos + free + abdom + thigh +
##     forearm, data = train_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.9369 -0.5786  0.1471  0.8991  7.0016
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.30283    3.05924   1.080   0.2816
## weight       0.39255    0.01992  19.704  < 2e-16 ***
## adipos      -0.44286    0.08736  -5.069 9.25e-07 ***
## free        -0.54799    0.01485 -36.911  < 2e-16 ***
## abdom        0.19368    0.03727   5.197 5.09e-07 ***
## thigh        0.11691    0.04476   2.612   0.0097 **
## forearm      0.37368    0.07429   5.030 1.11e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.602 on 195 degrees of freedom
## Multiple R-squared:  0.9646, Adjusted R-squared:  0.9635
## F-statistic: 884.9 on 6 and 195 DF,  p-value: < 2.2e-16
```

```
anova(model_2, model_3)
```

```
## Analysis of Variance Table
##
## Model 1: siri ~ age + weight + height + adipos + free + neck + chest +
##     abdom + hip + thigh + knee + ankle + biceps + forearm + wrist
## Model 2: siri ~ weight + adipos + free + abdom + thigh + forearm
##   Res.Df    RSS Df Sum of Sq      F   Pr(>F)
## 1    186 443.90
## 2    195 500.25 -9   -56.347 2.6233 0.007059 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Answer: p-value of ANOVA is approximately 0, thus reject the null hypothesis. We conclude that at least one other predictor among the predictors (age, height, neck, chest, hip, knee, ankle, biceps and wrist) will be significantly associated to the siri.Therefore, therefore, we prefer model_2, which contains enough informations, and we can not simply choose model_3.

c). Use the full model in 2a and predict() function in R to predict the response on the testing data. Calculate and report the RMSE (root mean squared error) of the response obtained on both the training and testing data.Why do you think there is a difference in the errors between the 2 datasets?

```
model_2 <- lm(siri ~., data = train_data)
pred1_test <- predict(model_2, test_data)
pred1_train <- predict(model_2, train_data)

RMSE1_test <- sqrt(mean((pred1_test - test_data$siri)^2))
RMSE1_test
```

```
## [1] 1.407183
```

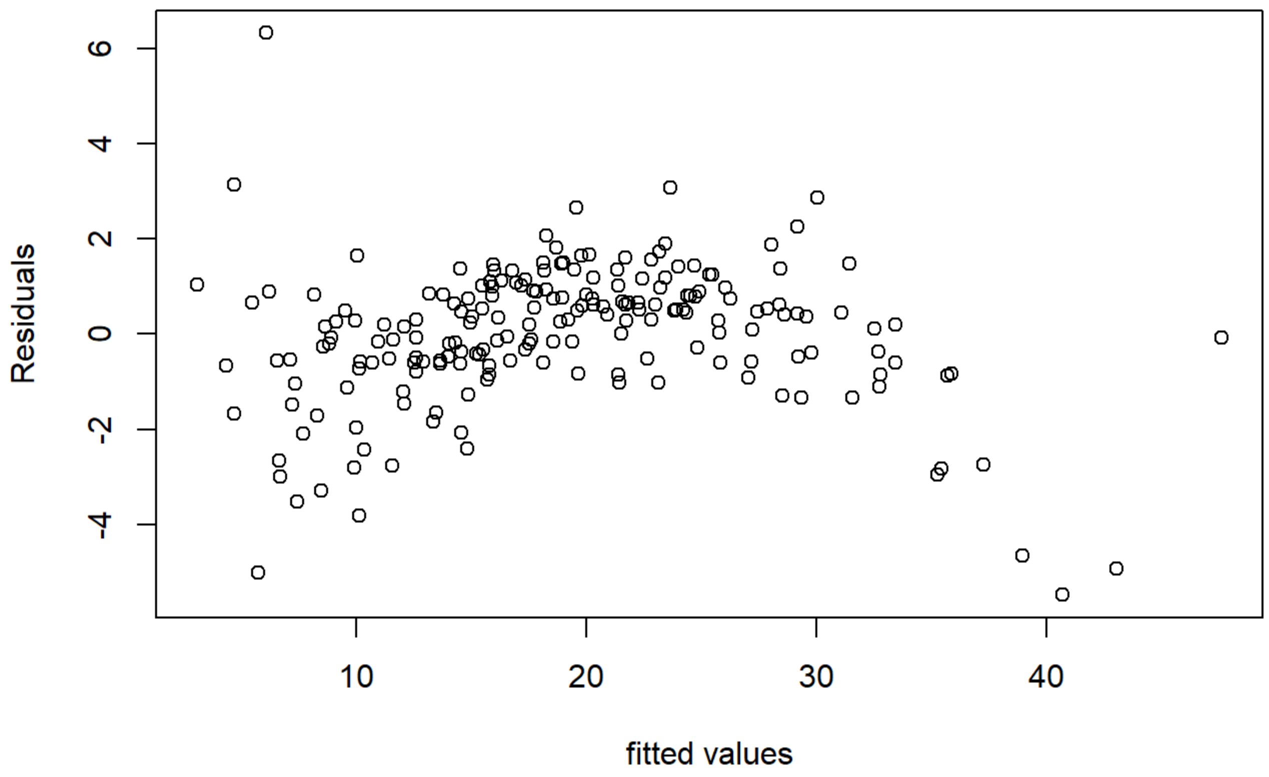
```
RMSE1_train <- sqrt(mean((pred1_train - train_data$siri)^2))
RMSE1_train
```

```
## [1] 1.482409
```

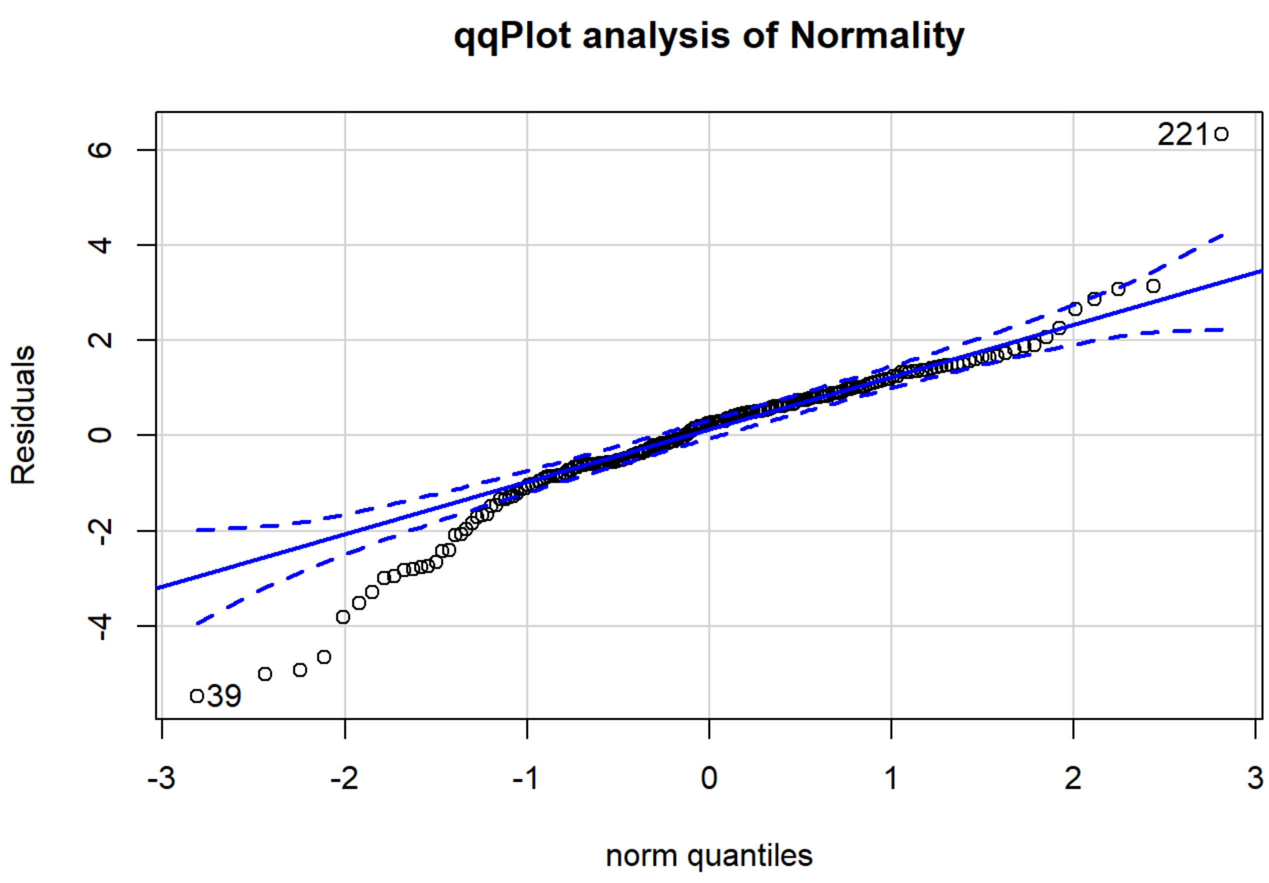
Answer: We found a little difference of errors between train and test datasets, the train error is a little bit higher than test error. The difference may be caused by that the data have been used twice (for modeling and prediction) in train dataset. The train error is a little higher than test error, which probably means that the model is not the best one, thus we need to improve the model firstly.

d). Perform a residual analysis to check for non-constant variance.State your observation. If you find any anomaly, perform a boxcox transformation on the response to remove any heteroscedasticity. What is your optimal choice of lambda? Fit a different model transforming the response by the optimal lambda and check for non-constant variance again. What do you observe?

```
#### Check Constant Variance & Uncorrelated Errors
model_2 <- lm(siri ~., data = train_data)
full.fitted = fitted(model_2)
full.resid = residuals(model_2)
par(mfrow=c(1,1))
plot(full.fitted, full.resid, xlab="fitted values", ylab="Residuals")
```



```
qqPlot(full.resid, ylab="Residuals", main = "qqPlot analysis of Normality")
```



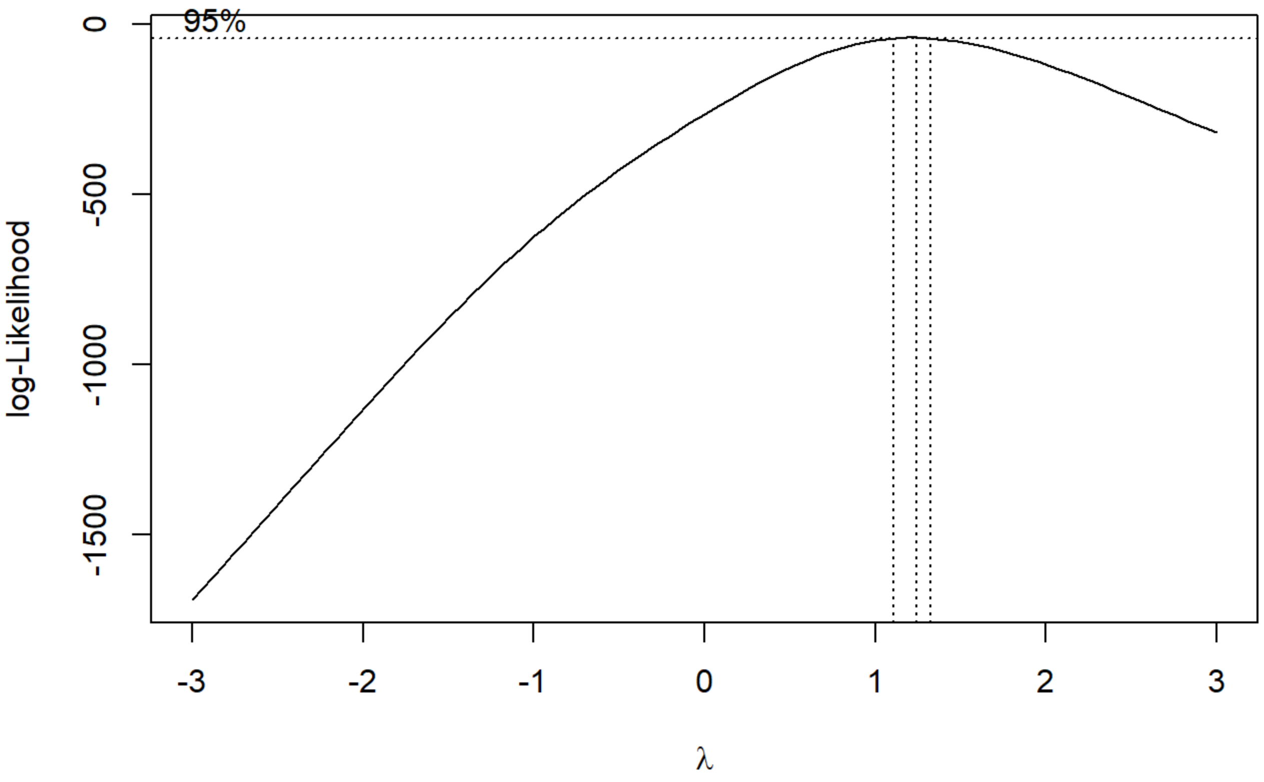
```
## 221  39
## 157  76
```

```
####

library(MASS)
```

```
## Warning: package 'MASS' was built under R version 3.5.1
```

```
bc = boxcox(model_2, lambda = seq(-3,3))
```



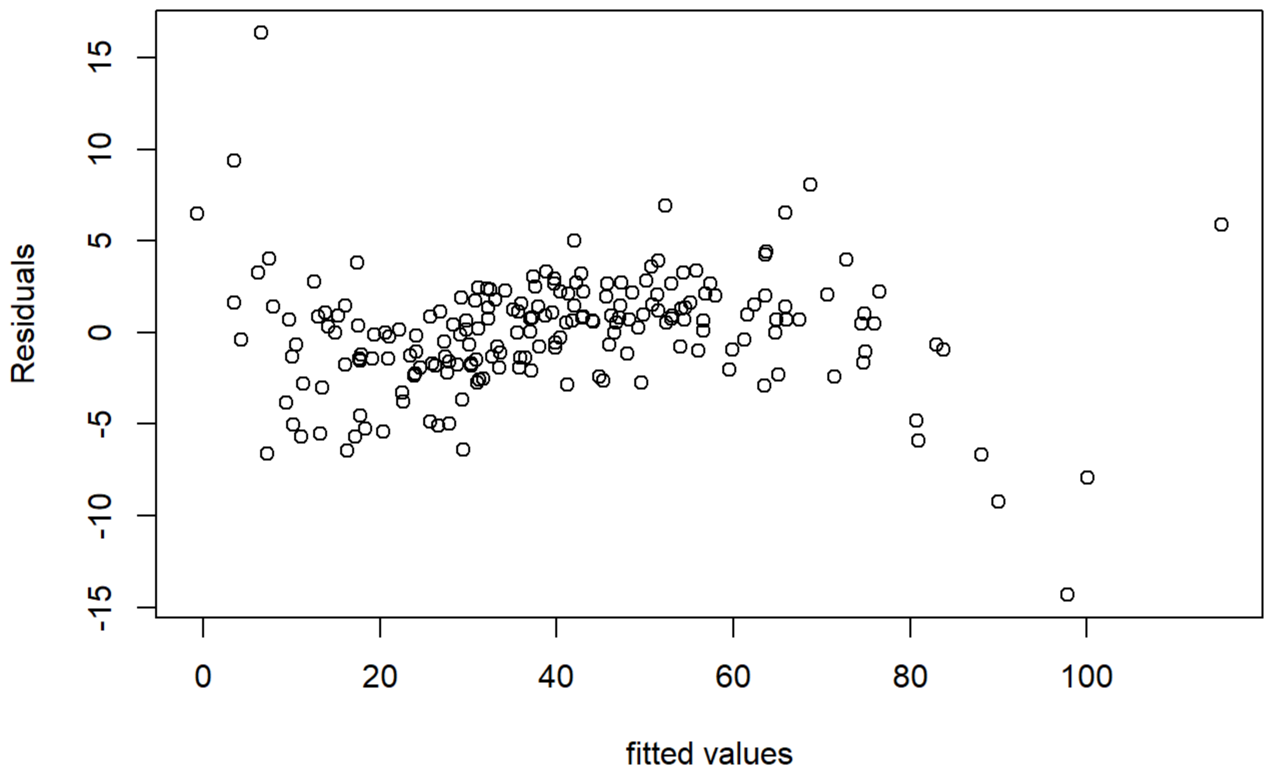
```
#### extract best lambda
best = bc$x[which(bc$y == max(bc$y))]
best
```

```
## [1] 1.242424
```

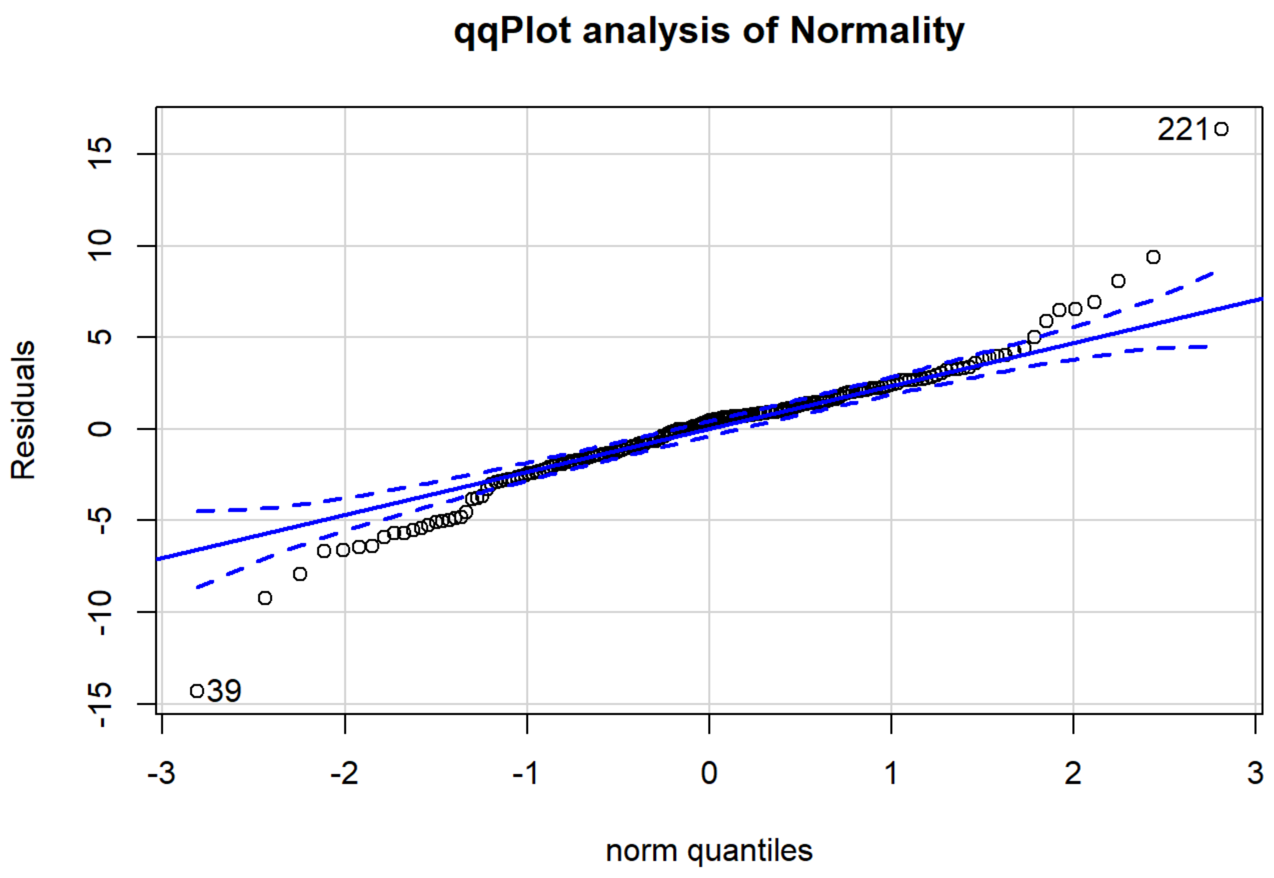
```
bc_model <- lm(siri**best ~., data = train_data)
summary(bc_model)
```

```
##
## Call:
## lm(formula = siri^best ~ ., data = train_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.3203  -1.5566   0.4546   1.6045  16.3348
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -28.200199  15.315405  -1.841 0.067171 .
## age           0.005518   0.028763   0.192 0.848075
## weight        0.923474   0.054291  17.010  < 2e-16 ***
## height        0.050480   0.088743   0.569 0.570154
## adipos       -0.976118   0.247277  -3.947 0.000112 ***
## free         -1.410004   0.033009 -42.715  < 2e-16 ***
## neck          0.009102   0.207820   0.044 0.965111
## chest         0.259596   0.093843   2.766 0.006242 **
## abdom         0.348775   0.093132   3.745 0.000240 ***
## hip           0.012666   0.138901   0.091 0.927443
## thigh         0.268572   0.123682   2.171 0.031162 *
## knee          0.241671   0.213543   1.132 0.259208
## ankle         0.355664   0.180752   1.968 0.050590 .
## biceps        0.272503   0.148602   1.834 0.068284 .
## forearm       0.573997   0.180710   3.176 0.001746 **
## wrist         0.441994   0.466554   0.947 0.344685
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.349 on 186 degrees of freedom
## Multiple R-squared:  0.9769, Adjusted R-squared:  0.975
## F-statistic: 524.4 on 15 and 186 DF,  p-value: < 2.2e-16
```

```
bc.resid = residuals(bc_model)
bc.fitted = fitted(bc_model)
par(mfrow=c(1,1))
plot(bc.fitted, bc.resid, xlab="fitted values", ylab="Residuals")
```



```
qqPlot(bc.resid, ylab="Residuals", main = "qqPlot analysis of Normality")
```



```
## 221  39
## 157  76
```

Answer: After boxcox transformation, there seems no improvements or very little changes, probably we need to seek for other better methods for best model.

e). Use the transformed model in 2d. Calculate and report the RMSE of the response obtained on both the training and testing data. Did the difference in training and testing errors increase or decrease as compared to the results of question 2c? Explain the change. Which results do you prefer? Note: Please remember to back transform your predicted value to original scale before calculating. Also, if any predicted value is negative, convert them to 0 and then proceed.

```
bc_model <- lm(siri^best ~., data = train_data)
pred_bc_test <- predict(bc_model, test_data)
pred_bc_train <- predict(bc_model, train_data)

#### Back transform the predicted value to original scale before calculating.
#InvBoxCox <- function(x, lambda){
#  if (lambda == 0) exp(x) else (lambda*x + 1)^(1/lambda)
#}                 This is an alternative function method

origin_pred_test = exp(log(best * pred_bc_test + 1)/best)
origin_pred_train = exp(log(best * pred_bc_train + 1)/best)

RMSE_bc_test <- sqrt(mean((origin_pred_test - test_data$siri)^2))
RMSE_bc_test
```

```
## [1] 4.264987
```

```
RMSE_bc_train <- sqrt(mean((origin_pred_train - train_data$siri)^2))
RMSE_bc_train
```

```
## [1] 4.517949
```

Answer: To remove heteroscedasticity, we used Box-Cox transformation for model improvement. the RMSE results have increased a little compared to previous results (Question 2c). This method seems it doesn't work well , so I still prefer the previous result, and also we should continue to find the optimal model by using other methods.

Question 3: Variable selection using Stepwise Regression

a). Use the leaps function in the leaps package and perform an all subset regression on the training data by "minimizing" Mallow's Cp statistics (method="Cp"). Report the variables of the best model, its training and testing error

```
library(leaps)
```

```
## Warning: package 'leaps' was built under R version 3.5.3
```

```
out = leaps(train_data[, -1], train_data$siri, method = 'Cp')
#cbind(as.matrix(out$which), out$Cp)
best.model = which(out$Cp==min(out$Cp))
best.model
```

```
## [1] 101
```

```
cbind(as.matrix(out$which), out$Cp)[best.model, ]
```

```
##        1        2        3        4        5        6        7        8 
## 0.000000 1.000000 0.000000 1.000000 1.000000 0.000000 1.000000 1.000000 
##        9        A        B        C        D        E        F 
## 0.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 9.467832
```

```
model_4 <- lm(siri ~  weight + adipos + free + chest + abdom + thigh + knee + ankle +
biceps + forearm + wrist, data = train_data)
summary(model_4)
```

```
## 
## Call:
## lm(formula = siri ~ weight + adipos + free + chest + abdom + 
##     thigh + knee + ankle + biceps + forearm + wrist, data = train_data)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -5.3678 -0.6557  0.1959  0.8618  6.3800 
## 
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)    
## (Intercept) -11.44559    4.51985  -2.532  0.01214 *  
## weight        0.35729    0.02081  17.172  < 2e-16 ***
## adipos       -0.55219    0.09425  -5.859 2.02e-08 ***
## free         -0.55639    0.01460 -38.109  < 2e-16 ***
## chest         0.11919    0.04096   2.910  0.00405 ** 
## abdom         0.16372    0.03881   4.219 3.80e-05 ***
## thigh         0.14740    0.04925   2.993  0.00313 ** 
## knee          0.14035    0.09313   1.507  0.13348    
## ankle         0.13013    0.08168   1.593  0.11278    
## biceps        0.13811    0.06760   2.043  0.04243 *  
## forearm       0.23650    0.07965   2.969  0.00337 ** 
## wrist         0.27630    0.19069   1.449  0.14901    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 1.535 on 190 degrees of freedom
## Multiple R-squared:  0.9683, Adjusted R-squared:  0.9665 
## F-statistic: 527.9 on 11 and 190 DF,  p-value: < 2.2e-16
```

```
pred4_test <- predict(model_4, test_data)
pred4_train <- predict(model_4, train_data)

RMSE4_test <- sqrt(mean((pred4_test - test_data$siri)^2))
RMSE4_test
```

```
## [1] 1.405536
```

```
RMSE4_train <- sqrt(mean((pred4_train - train_data$siri)^2))
RMSE4_train
```

```
## [1] 1.488247
```

Answer: The best model with respect to Mallow’s Cp criterion: weight, adipos, free, chest, abdom, thigh, knee, ankle, biceps, forearm and wrist. This training error and testing errors seems no changes.

b). Use the step() function on the original model in 2a to perform a backward stepwise regression by minimizing AIC. What was the change in AIC from the original model? Report the variables of the final model, its training and testing error. (Keep trace=FALSE)

```
model_2 <- lm(siri ~., data = train_data)
model_back <- step(model_2, direction = 'backward', trace = 1)
```

```
## Start:  AIC=191.04
## siri ~ age + weight + height + adipos + free + neck + chest +
##     abdom + hip + thigh + knee + ankle + biceps + forearm + wrist
##
##           Df Sum of Sq    RSS    AIC
## - neck     1       0.1  444.0 189.09
## - hip      1       0.3  444.2 189.18
## - age      1       1.2  445.1 189.58
## - height   1       1.8  445.7 189.86
## - wrist    1       2.4  446.3 190.14
## - knee     1       4.3  448.2 190.97
## <none>                  443.9 191.04
## - ankle    1       5.9  449.8 191.69
## - biceps   1       8.6  452.5 192.93
## - chest    1      16.0  459.9 196.18
## - forearm  1      20.0  463.9 197.96
## - thigh    1      22.5  466.4 199.03
## - abdom    1      31.5  475.4 202.88
## - adipos   1      43.9  487.8 208.10
## - weight   1     486.7  930.6 338.57
## - free     1    3170.1 3614.0 612.63
##
## Step:  AIC=189.09
## siri ~ age + weight + height + adipos + free + chest + abdom +
##     hip + thigh + knee + ankle + biceps + forearm + wrist
##
##           Df Sum of Sq    RSS    AIC
## - hip      1       0.2  444.2 187.20
## - age      1       1.1  445.1 187.61
## - height   1       1.8  445.8 187.90
## - wrist    1       2.3  446.3 188.14
## <none>                  444.0 189.09
## - knee     1       4.4  448.4 189.09
## - ankle    1       6.2  450.2 189.87
## - biceps   1       8.6  452.6 190.94
## - chest    1      16.4  460.4 194.43
## - forearm  1      20.0  464.0 195.99
## - thigh    1      22.4  466.4 197.03
## - abdom    1      31.5  475.5 200.93
## - adipos   1      46.4  490.4 207.18
## - weight   1     501.1  945.1 339.70
## - free     1    3288.3 3732.4 617.14
##
## Step:  AIC=187.2
## siri ~ age + weight + height + adipos + free + chest + abdom +
##     thigh + knee + ankle + biceps + forearm + wrist
##
##           Df Sum of Sq    RSS    AIC
## - age      1       1.2  445.5 185.76
## - height   1       2.2  446.4 186.18
## - wrist    1       2.3  446.5 186.23
## - knee     1       4.3  448.5 187.13
## <none>                  444.2 187.20
## - ankle    1       6.3  450.5 188.04
## - biceps   1       8.7  453.0 189.13
## - chest    1      19.1  463.3 193.69
## - forearm  1      21.5  465.8 194.77
## - thigh    1      23.1  467.3 195.42
## - abdom    1      31.9  476.2 199.22
## - adipos   1      48.2  492.4 205.99
## - weight   1     588.6 1032.9 355.63
## - free     1    3338.8 3783.1 617.86
##
## Step:  AIC=185.76
## siri ~ weight + height + adipos + free + chest + abdom + thigh +
##     knee + ankle + biceps + forearm + wrist
##
##           Df Sum of Sq    RSS    AIC
## - height   1       1.9  447.4 184.63
## - wrist    1       4.3  449.8 185.72
## <none>                  445.5 185.76
## - ankle    1       5.8  451.3 186.39
## - knee     1       6.0  451.5 186.45
## - biceps   1       9.3  454.8 187.95
## - chest    1      19.7  465.2 192.50
## - forearm  1      20.5  466.0 192.84
## - thigh    1      22.5  468.0 193.72
## - abdom    1      40.9  486.3 201.49
## - adipos   1      48.6  494.1 204.67
## - weight   1     597.0 1042.5 355.50
## - free     1    3408.7 3854.2 619.63
##
## Step:  AIC=184.63
## siri ~ weight + adipos + free + chest + abdom + thigh + knee +
##     ankle + biceps + forearm + wrist
##
##           Df Sum of Sq    RSS    AIC
## <none>                  447.4 184.63
## - wrist    1       4.9  452.3 184.85
## - knee     1       5.3  452.8 185.03
## - ankle    1       6.0  453.4 185.31
## - biceps   1       9.8  457.2 187.02
## - chest    1      19.9  467.3 191.44
## - forearm  1      20.8  468.2 191.79
## - thigh    1      21.1  468.5 191.94
## - abdom    1      41.9  489.3 200.72
## - adipos   1      80.8  528.2 216.18
## - weight   1     694.4 1141.8 371.88
## - free     1    3419.8 3867.2 618.31
```

```
model_back
```

```
##
## Call:
## lm(formula = siri ~ weight + adipos + free + chest + abdom +
##     thigh + knee + ankle + biceps + forearm + wrist, data = train_data)
##
## Coefficients:
## (Intercept)       weight       adipos         free        chest
##    -11.4456       0.3573      -0.5522      -0.5564       0.1192
##       abdom        thigh         knee        ankle       biceps
##      0.1637       0.1474       0.1403       0.1301       0.1381
##     forearm        wrist
##      0.2365       0.2763
```

```
AIC(model_back)
```

```
## [1] 759.881
```

```
AIC(model_2)
```

```
## [1] 766.2932
```

```
summary(model_back)
```

```
##
## Call:
## lm(formula = siri ~ weight + adipos + free + chest + abdom +
##     thigh + knee + ankle + biceps + forearm + wrist, data = train_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.3678 -0.6557  0.1959  0.8618  6.3800
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -11.44559    4.51985  -2.532  0.01214 *
## weight        0.35729    0.02081  17.172  < 2e-16 ***
## adipos       -0.55219    0.09425  -5.859 2.02e-08 ***
## free         -0.55639    0.01460 -38.109  < 2e-16 ***
## chest         0.11919    0.04096   2.910  0.00405 **
## abdom         0.16372    0.03881   4.219 3.80e-05 ***
## thigh         0.14740    0.04925   2.993  0.00313 **
## knee          0.14035    0.09313   1.507  0.13348
## ankle         0.13013    0.08168   1.593  0.11278
## biceps        0.13811    0.06760   2.043  0.04243 *
## forearm       0.23650    0.07965   2.969  0.00337 **
## wrist         0.27630    0.19069   1.449  0.14901
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.535 on 190 degrees of freedom
## Multiple R-squared:  0.9683, Adjusted R-squared:  0.9665
## F-statistic: 527.9 on 11 and 190 DF,  p-value: < 2.2e-16
```

```
pred5_test <- predict(model_back, test_data)
pred5_train <- predict(model_back, train_data)

RMSE5_test <- sqrt(mean((pred5_test - test_data$siri)^2))
RMSE5_test
```

```
## [1] 1.405536
```

```
RMSE5_train <- sqrt(mean((pred5_train - train_data$siri)^2))
RMSE5_train
```

```
## [1] 1.488247
```

Answer: The changes of AIC between step() and model_2 are very little, also the RMSE of test and train seems to have no changes.

Question 4: Variable selection using Ridge, Lasso and

Elasticnet regression

a). Use the glmnet() function in the library glmnet to build a Ridge Regression model by using the full model matrix of 2a as the training dataset.Perform a 10 fold cross validation with the training data and report the optimal r (lambda.min). Use this r to build the final model and report its training and testing error Note: Remove the intercept column from the model matrix of the full model

```
set.seed(123)
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 3.5.1
```

```
## Loading required package: Matrix
```

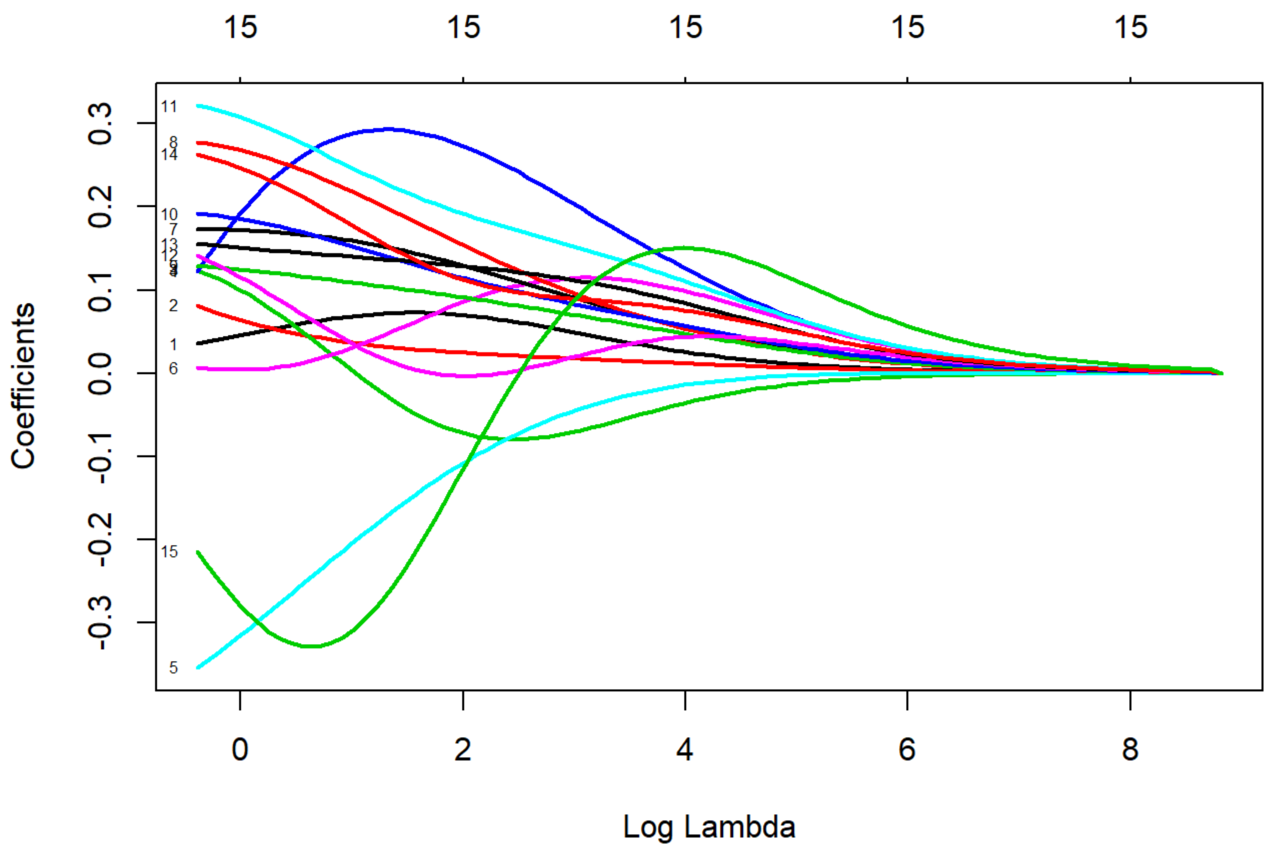
```
## Loading required package: foreach
```

```
## Warning: package 'foreach' was built under R version 3.5.1
```

```
## Loaded glmnet 2.0-16
```

```
Xpred = as.matrix(train_data[, -1])
y <- as.matrix(train_data[, 1])

# Find the optimal lambda using 10-fold CV
r_model.cv = cv.glmnet(Xpred, y, alpha = 0, nfolds = 10)
## Fit lasso model with 100 values for lambda
r_model = glmnet(Xpred, y, alpha = 0, nlambda = 100)
plot(r_model, xvar = 'lambda', label = TRUE, lwd = 2)
```



```
## Extract coefficients at optimal lambda
coef(r_model, s = r_model.cv$lambda.min)
```

```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                         1
## (Intercept) -49.631417447
## age           0.037548047
## weight        0.076619611
## height        0.118873364
## adipos        0.140463163
## free         -0.345027006
## neck          0.005330011
## chest         0.172825058
## abdom         0.276104648
## hip           0.127900922
## thigh         0.190466974
## knee          0.319078385
## ankle         0.135723465
## biceps        0.154252245
## forearm       0.259721911
## wrist        -0.231565224
```

```
s = r_model.cv$lambda.min
s
```

```
## [1] 0.7436835
```

```
pred6_test <- predict(r_model.cv, s = 'lambda.min', newx = as.matrix(test_data[, -
1]))
pred6_train <- predict(r_model.cv, s = 'lambda.min', newx = as.matrix(train_data[, -
1]))

RMSE6_test <- sqrt(mean((pred6_test - test_data$siri)^2))
RMSE6_test
```

```
## [1] 2.215763
```

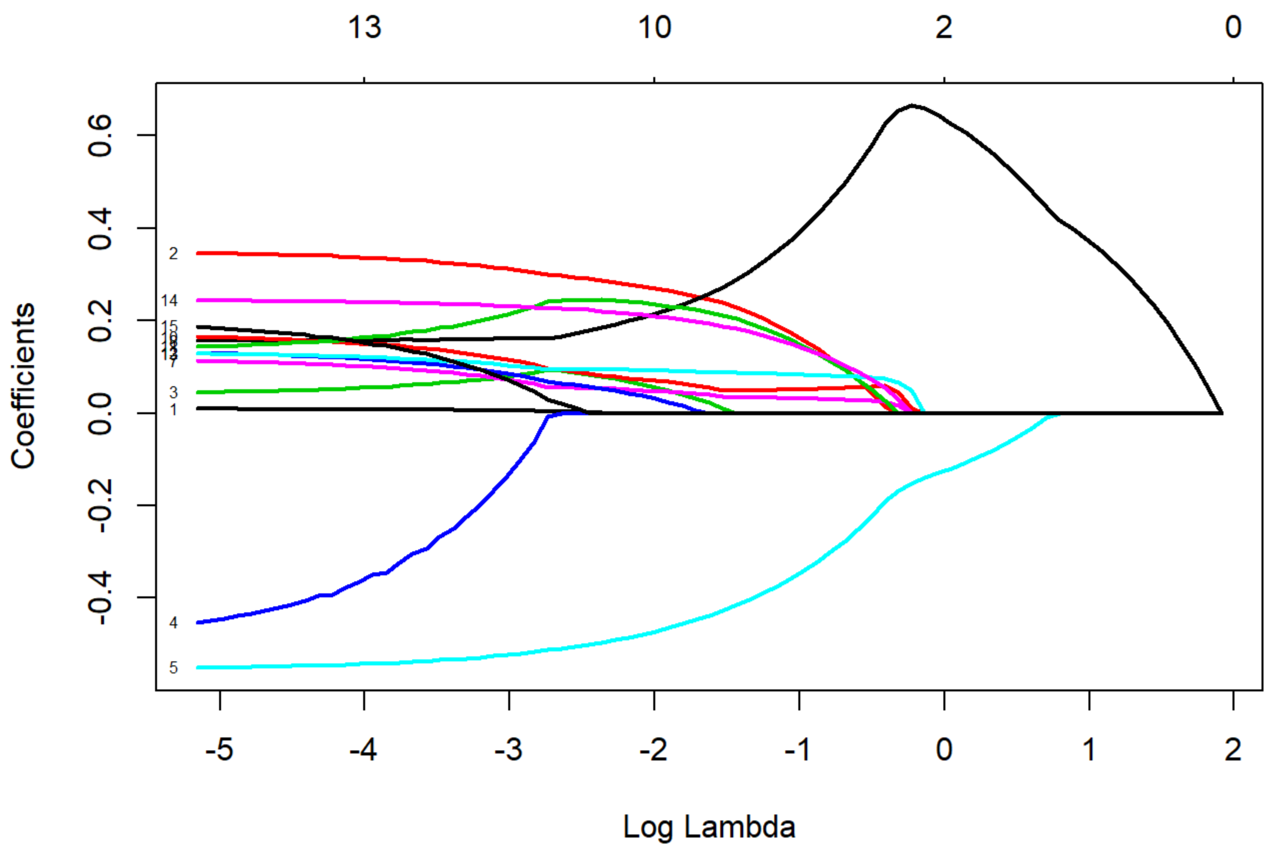
```
RMSE6_train <- sqrt(mean((pred6_train - train_data$siri)^2))
RMSE6_train
```

```
## [1] 2.306854
```

b). Use the glmnet() function to build a lasso Regression model by using the full model matrix of 2a as the training dataset.Perform a 10 fold cross validation with the training data and report the optimal (lambda.min). Use this to build the final model. Report the final variables obtained (non 0 coefficients), the model training and testing error

```
set.seed(123)
library(glmnet)
Xpred = as.matrix(train_data[, -1])
y <- as.matrix(train_data[, 1])

# Find the optimal lambda using 10-fold CV
La_model.cv = cv.glmnet(Xpred, y, alpha = 1, nfolds = 10)
## Fit lasso model with 100 values for lambda
La_model = glmnet(Xpred, y, alpha = 1, nlambda = 100)
plot(La_model, xvar = 'lambda', label = TRUE, lwd = 2)
```



```
## Extract coefficients at optimal lambda
coef(La_model, s = La_model.cv$lambda.min)
```

```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept) -13.18678306
## age            .
## weight       0.27449821
## height       0.06243061
## adipos         .
## free        -0.47872274
## neck           .
## chest        0.04913153
## abdom        0.20705645
## hip            .
## thigh        0.07303150
## knee         0.23822665
## ankle        0.03699487
## biceps       0.09257090
## forearm      0.21227912
## wrist          .
```

```
s2 = La_model.cv$lambda.min
s2
```

```
## [1] 0.1240539
```

```
pred7_test <- predict(La_model.cv, s = 'lambda.min', newx = as.matrix(test_data[, -
1]))
pred7_train <- predict(La_model.cv, s = 'lambda.min', newx = as.matrix(train_data[, -
1]))

RMSE7_test <- sqrt(mean((pred7_test - test_data$siri)^2))
RMSE7_test
```

```
## [1] 1.57399
```

```
RMSE7_train <- sqrt(mean((pred7_train - train_data$siri)^2))
RMSE7_train
```

```
## [1] 1.668157
```

c). Among all the variable selection models you built, which model has the lowest testing error? Which one is a low variance model? Which variable selection model would you prefer for predictive purposes?

Answer: Among all the models above, we found that these models such as Subset Regression using Mallow’s Cp, Backward Stepwide Regression and LASSO have lowest testing error (values are about 1.4), the LASSO Regression has the low variance model with the least explatory variables(10 exploratory variables for LASSO, 11 exploratory variables for Subset Regression using Mallow’s Cp and Backward Stepwide Regression), but have very good performance as full model. Therefore, we prefer LASSO Regression model.

Question 5: Principal Component Regression

In this question you will directly fit a PCR model on the training data and perform cross validation by minimizing RMSE to obtain the desired number of PCs. Use the following code to plot the Cross Validation curve:

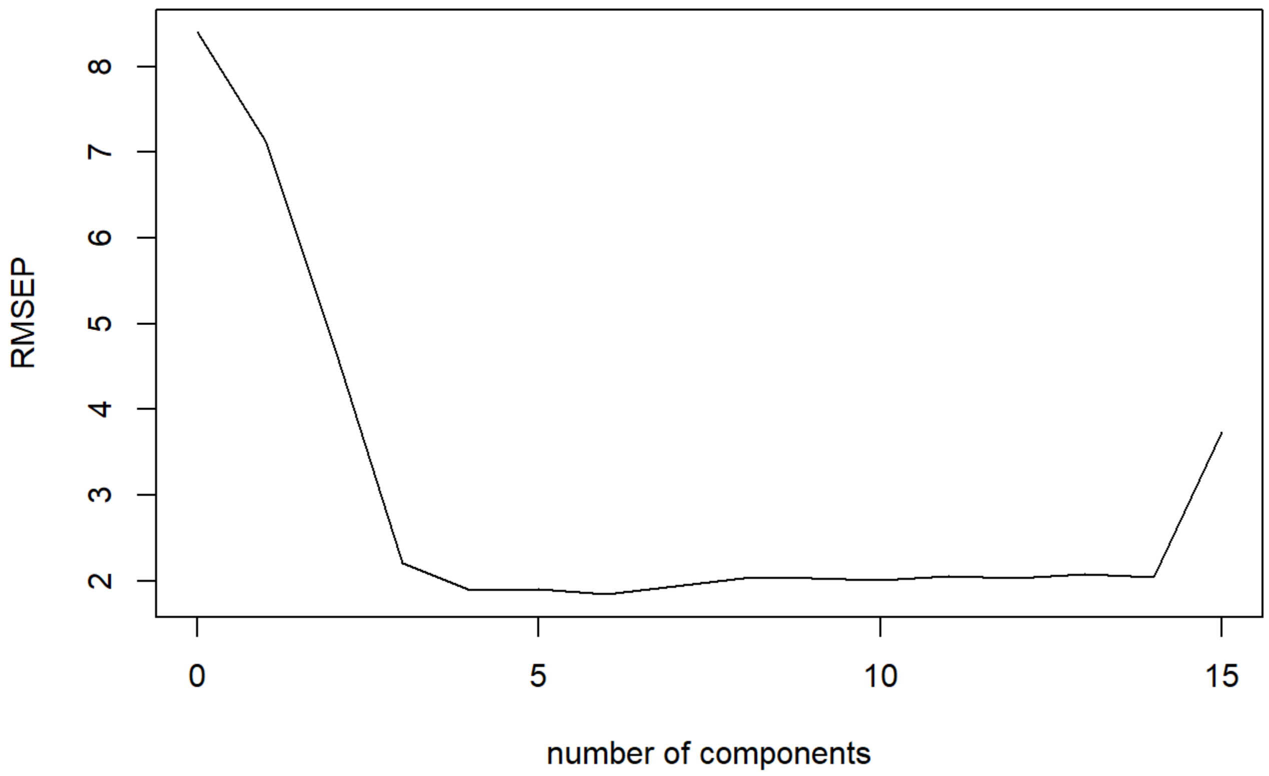
```
library(pls)
```

```
## Warning: package 'pls' was built under R version 3.5.3
```

```
##
## Attaching package: 'pls'
## The following object is masked from 'package:corrplot':
##
##     corrplot
```

```
## The following object is masked from 'package:stats':
##
##     loadings
```

```
set.seed(123)
pcrmod <- pcr(siri ~ ., data=train_data, validation='CV', ncomp=15)
#summary(pcrmod)
pcrCV = RMSEP(pcrmod, estimate='CV')
plot(pcrCV,main='')
```



```
which.min(pcrCV$val)
```

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

```
#predict(pcrmod,train,ncomp = best # of PCs)
pred8_test <- predict(pcrmod, test_data, ncomp = 7)
pred8_train <- predict(pcrmod, train_data, ncomp = 7)

RMSE8_test <- sqrt(mean((pred8_test - test_data$siri)^2))
RMSE8_test
```

```
## [1] 1.361429
```

```
RMSE8_train <- sqrt(mean((pred8_train - train_data$siri)^2))
RMSE8_train
```

```
## [1] 1.619852
```

Answer: This PCA is one of the best methods for variables selection, because the RMSE of test error is also very low as other models such as LASSO, but the dimensions are very low of PCA, which can perform excellent as other models or full model. Therefore, PCA is one of the best method we should consider for model selection.