Homework 5

Question 1a

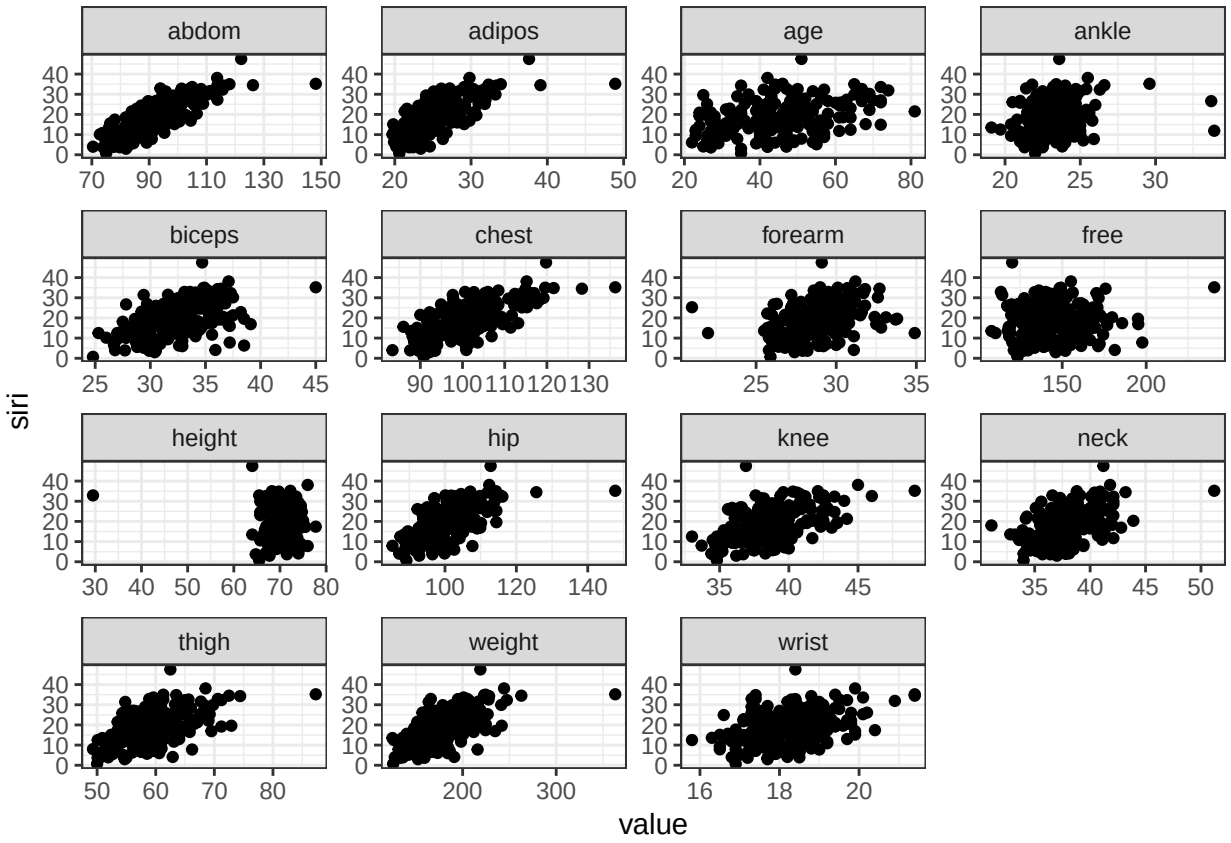
Begin by doing some exploratory analysis. Plot scatterplots to examine the relationships between all the explanatory variables and the response siri. Do you observe any relationship of siri with any of the predictors? Do you visually observe any outlier or high leverage point in any of the plots? Briefly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

```
library(tidyr)
library(ggplot2)
set.seed(123)
data=faraway::fat
data = data[-1]
data = data [-2]
smp_size <- floor(0.8 * nrow(data))
train_ind <- sample(seq_len(nrow(data)), size = smp_size)

train <- data[train_ind, ]
test <- data[-train_ind, ]



train %>%
  gather(-siri, key = "var", value = "value") %>%
  ggplot(aes(x = value, y = siri)) +
    geom_point() +
    facet_wrap(~ var, scales = "free") +
    theme_bw()
```



The above plots show several strong positive relationships with the response variable siri. Abdom, adipos, chest, biceps, hip, thigh, and weight all demonstrate a strong linear relationship with the response. Other variables show a weaker relationship such as age, wrist, forearm and neck. The height variable shows signs of an outlier, but the majority of the data appears consistent.

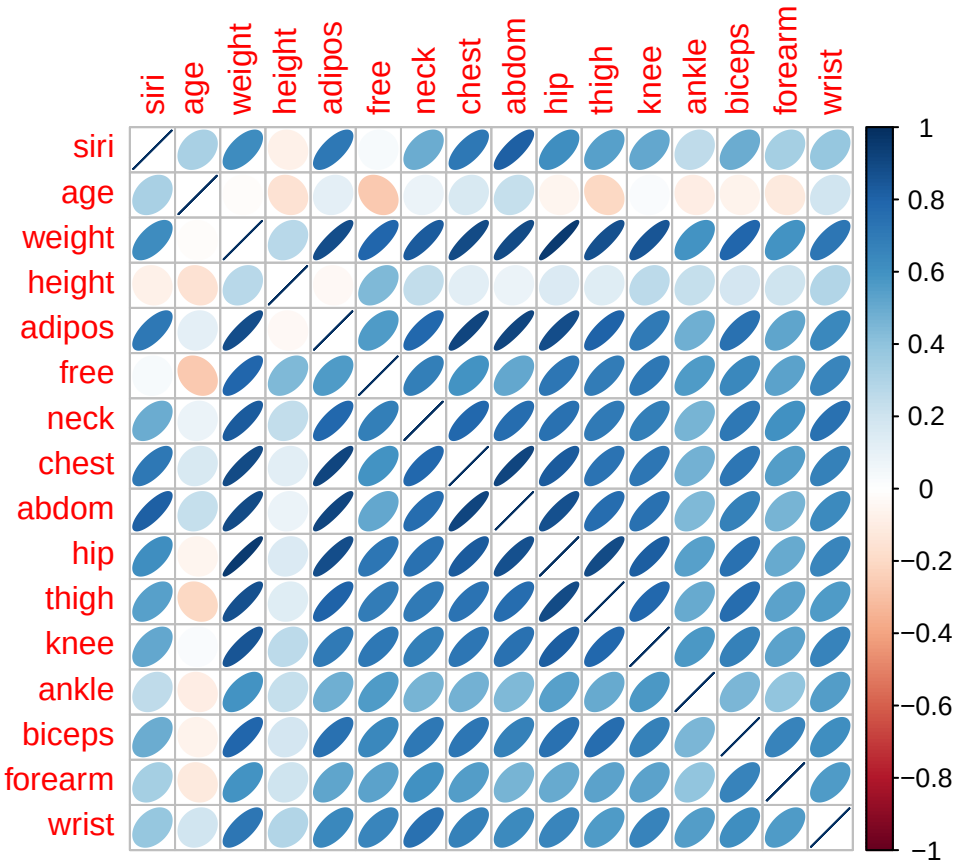
Question 1b

Plot a correlation matrix of all variables vs all other variables:

```
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
M <- cor(train)
corrplot(M, method = "ellipse")
```



The above correlation matrix plot confirms the previous scatterplots. Abdom, adipos, chest, biceps, hip, thigh, and weight all demonstrate a strong linear relationship with the response.

Question 1c

Collinearity leads to imprecise estimates of β. The signs of the coefficients can be the opposite of what intuition about the effect of the predictor might suggest. The standard errors are inflated so that t-tests may fail to reveal significant factors. The fit becomes very sensitive to measurement errors where small changes in y can lead to large changes in β.

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

There appears to be multicollinearity among the predictors. THe plot above shows weight having a higher correlation than the response with many of the predictors including adipos, neck, chest, hip, etc. Adipos also demonstrates a high correlation with many predictors such as chest, abdom, and hip.

Question 1d

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?

```
library(regclass)
```

```
## Loading required package: bestglm
## Loading required package: leaps
## Loading required package: VGAM
## Loading required package: stats4
## Loading required package: splines
##
## Attaching package: 'VGAM'
## The following object is masked from 'package:tidyr':
##
##     fill
## Loading required package: rpart
## Loading required package: randomForest
## randomForest 4.6-14
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
## The following object is masked from 'package:ggplot2':
##
##     margin
## Important regclass change from 1.3:
## All functions that had a . in the name now have an _
## all.correlations -> all_correlations, cor.demo -> cor_demo, etc.
```

```
m = lm(siri ~ ., data = train)

VIF(m)
```

```
##       age    weight    height    adipos      free      neck     chest
##  2.320919 48.027606  2.127750 15.139377  6.563514  4.650246 11.208230
##     abdom       hip     thigh      knee     ankle    biceps   forearm
## 18.908090 19.018327  7.801492  4.853886  1.829493  3.564787  2.204051
##     wrist
##  3.407079
```

Many of the predictors have VIFs >10 including weight, adipos, chest, abdom, and hip.

Question 1e

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?

```
(eigen(t(train)%*%as.matrix(train))$values^-1 * 9.997697e+07 )^.5
```

```
##  [1]    2.228237   42.834394   46.431022   84.803408  183.472721
##  [6]  210.785434  266.002183  364.321924  431.239406  453.108662
## [11]  502.890391  538.911533  636.633000  660.778504  867.694910
## [16] 1421.523145
```

All of the condition numbers are greater than 30, except the largest eigenvalue. This is more evidence of multicollinearity.

Question 2a

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?

```
model1 = lm(siri~ ., data=train)
summary(model1)
```

```
##
## Call:
## lm(formula = siri ~ ., data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.4306 -0.6051  0.2517  0.8736  6.2617
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -12.247702   7.118599  -1.721 0.087010 .
## age           0.009424   0.013297   0.709 0.479381
## weight        0.355839   0.025320  14.053  < 2e-16 ***
## height        0.035693   0.041019   0.870 0.385338
## adipos       -0.491725   0.114378  -4.299 2.77e-05 ***
## free         -0.554260   0.015322 -36.173  < 2e-16 ***
## neck         -0.022372   0.096133  -0.233 0.816236
## chest         0.112640   0.043393   2.596 0.010194 *
## abdom         0.158899   0.043361   3.665 0.000323 ***
## hip          -0.020844   0.064336  -0.324 0.746310
## thigh         0.174544   0.057181   3.052 0.002604 **
## knee          0.133655   0.098783   1.353 0.177703
## ankle         0.130195   0.083551   1.558 0.120876
## biceps        0.128909   0.068733   1.875 0.062301 .
## forearm       0.243944   0.083643   2.916 0.003978 **
## wrist         0.224690   0.216178   1.039 0.299985
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.548 on 185 degrees of freedom
## Multiple R-squared:  0.9686, Adjusted R-squared:  0.9661
## F-statistic: 380.6 on 15 and 185 DF,  p-value: < 2.2e-16
```

The above summary shows that weight, adipos, free, chest, abdom, thigh, and forearm are all significant at the 99% level.

Question 2b

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

```
model2 = lm(siri ~ weight + adipos+ free+chest+abdom+ thigh+  forearm, data = train)
summary(model2)
```

```
##
## Call:
## lm(formula = siri ~ weight + adipos + free + chest + abdom +
##     thigh + forearm, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.6613 -0.6103  0.1547  0.8660  6.8773
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.11330    3.55061  -0.595 0.552412
## weight       0.37924    0.02024  18.733  < 2e-16 ***
## adipos      -0.54247    0.09268  -5.853 2.04e-08 ***
## free        -0.54901    0.01472 -37.300  < 2e-16 ***
## chest        0.11924    0.04177   2.855 0.004778 **
## abdom        0.16116    0.03926   4.105 5.97e-05 ***
## thigh        0.16002    0.04661   3.433 0.000731 ***
## forearm      0.33580    0.07447   4.509 1.13e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.576 on 193 degrees of freedom
## Multiple R-squared:  0.966,  Adjusted R-squared:  0.9648
## F-statistic: 784.2 on 7 and 193 DF,  p-value: < 2.2e-16
```

```
anova(model1, model2)
```

```
## 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 + chest + abdom + thigh + forearm
##   Res.Df    RSS Df Sum of Sq      F  Pr(>F)
## 1    185 443.18
## 2    193 479.63 -8   -36.453 1.9021 0.06197 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The ANOVA test shows that there is a significant difference with the models at the 0.1 level. However, I prefer the second model because it is simpler and still has a comparable R^2 to the first model.

Question 2c

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?

```
library(Metrics)
rmse(train$siri,predict(model1, train[-1]))
```

```
## [1] 1.484876
```

```
rmse(test$siri,predict(model1, test[-1]))
```

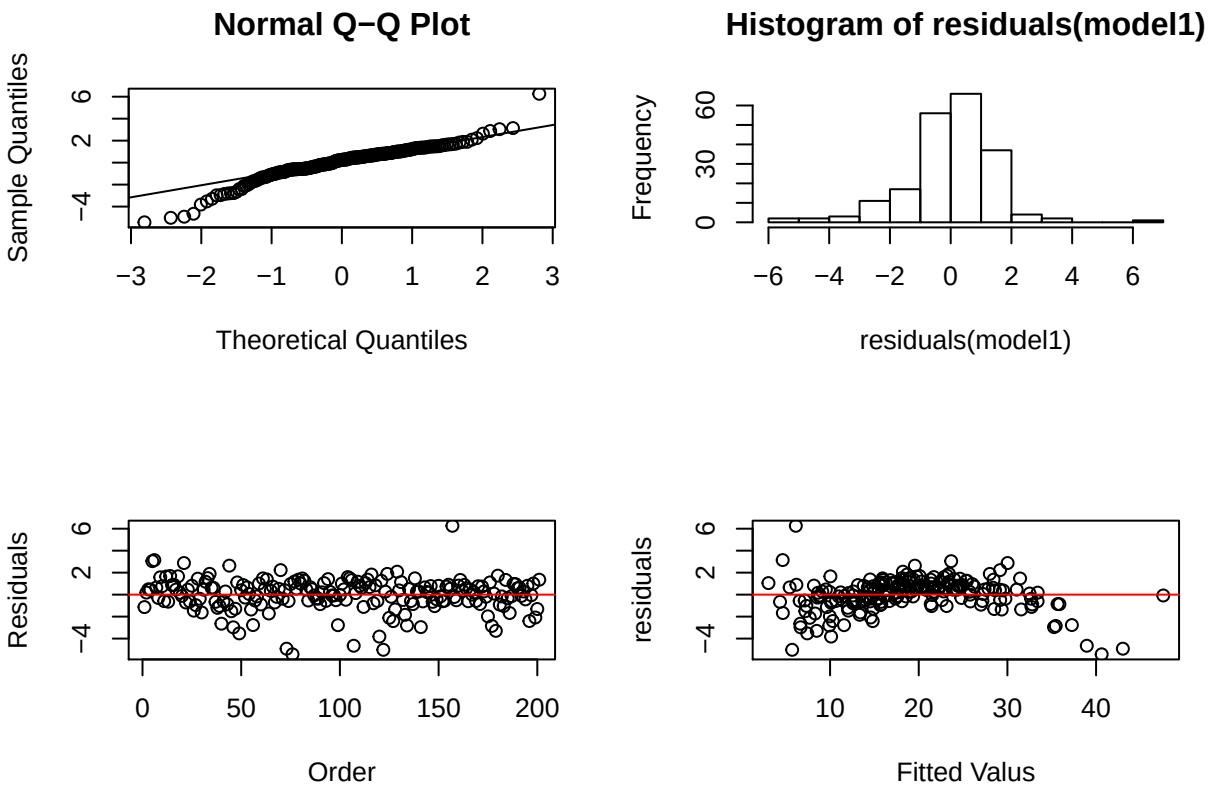
```
## [1] 1.401448
```

The training dataset had a lower RMSE (1.46) than the testing dataset (1.49), whcih makes sense because the model was fit to the training data so it is expected to be slightly overfit to this dataset.

Question 2d

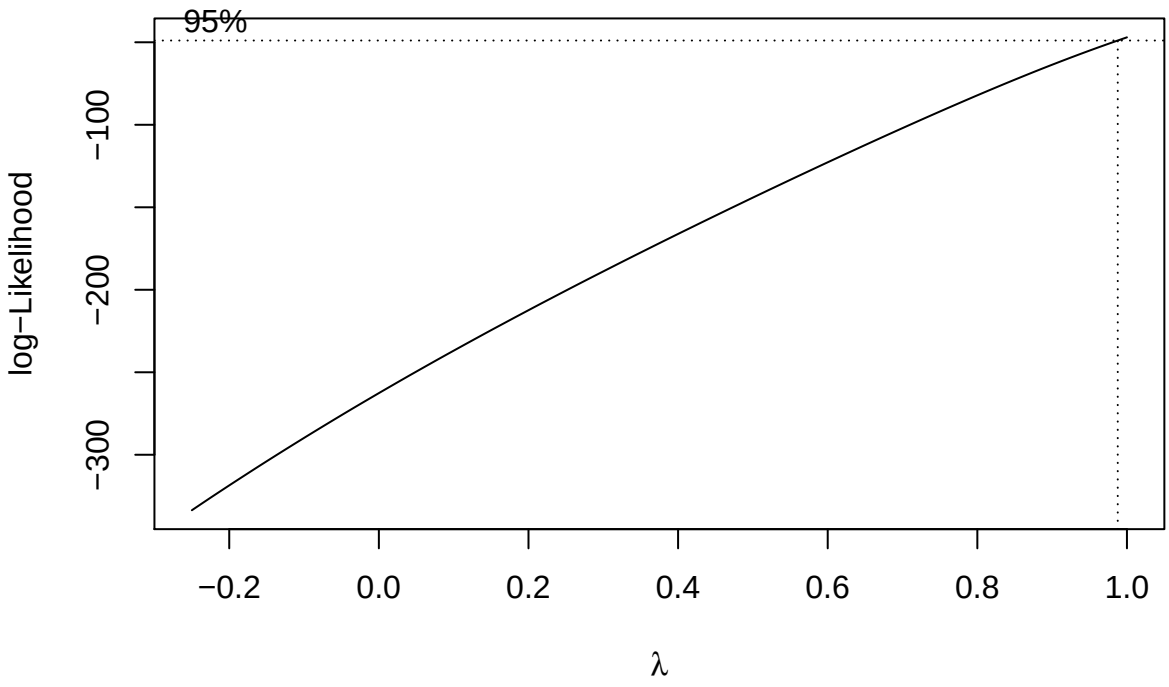
Perform a residual analysis to check for non-constant variance.State your observation. If you find any anomaly, perform a boxcox transformation on the response to remove any heteroscedasticity. What is your optimal choice of lambda? Fit a different model transforming the response by the optimal lambda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

```
par(mfrow=c(2,2))
qqnorm(residuals(model1))
qqline(residuals(model1))
hist(residuals(model1))
plot(residuals(model1), xlab = "Order", ylab = "Residuals")
abline(0,0, lty=1,col="red")
plot(fitted(model1),residuals(model1), xlab="Fitted Valus", ylab = "residuals")
abline(0,0, lty=1,col="red")
```



The third and fourth plot above evaluates the assumptions for constant variance and independence. There is not any pattern in the residuals, which indicates that the assumptions hold reasonably well. Therefore I will not perform a boxcox transform.

```
library(MASS)
b = boxcox(siri+.01~., data = train,lambda = seq(-.25,1, length = 10))
```




```
lambda <- b$x # lambda values

lik <- b$y # log likelihood values for SSE

bc <- cbind(lambda, lik) # combine lambda and lik

sorted_bc <- bc[order(-lik),] # values are sorted to identify the lambda value for the maximum log like

head(sorted_bc, n = 10)
```

```
##          lambda       lik
##  [1,] 1.0000000 -47.00960
##  [2,] 0.9873737 -48.98017
##  [3,] 0.9747475 -50.98905
##  [4,] 0.9621212 -53.03517
##  [5,] 0.9494949 -55.11743
##  [6,] 0.9368687 -57.23477
##  [7,] 0.9242424 -59.38611
##  [8,] 0.9116162 -61.57036
##  [9,] 0.8989899 -63.78646
## [10,] 0.8863636 -66.03331
```

The value with the highest log likelihood is 1, ie no transformation so the original model holds.

Question 3a

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)
x = as.matrix(train[-1])
y = as.matrix(train[1])
out = leaps(x,y, method = "Cp")
#cbind(as.matrix(out$which),out$Cp)
best.model = which(out$Cp==min(out$Cp))
#cbind(as.matrix(out$which), out$Cp)[best.model,]

colnames(train[-1])[cbind(as.matrix(out$which), out$Cp)[best.model,]>0]
```

```
##  [1] "weight"  "adipos"  "free"    "chest"   "abdom"   "thigh"   "knee"
##  [8] "ankle"   "biceps"  "forearm" "wrist"   NA
```

```
te_fr = rmse(train$siri,predict(lm(formula = siri ~ weight+  adipos+ free+chest+  abdom+  thigh+   ankl


tst_fr = rmse(test$siri,predict(lm(formula = siri ~ weight+  adipos+ free+chest+  abdom+  thigh+   ankl
```

The variables of the best model are:"weight" "adipos" "free" "chest" "abdom" "thigh" "ankle" "biceps" "forearm" "wrist" The training error was 1.5 and the test was 1.39.

Question 3b

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)

```
step(model1, direction = "backward")
```

```
## Start:  AIC=190.92
## siri ~ age + weight + height + adipos + free + neck + chest +
##     abdom + hip + thigh + knee + ankle + biceps + forearm + wrist
##
##           Df Sum of Sq    RSS    AIC
## - neck     1      0.13  443.3 188.98
## - hip      1      0.25  443.4 189.04
## - age      1      1.20  444.4 189.47
## - height   1      1.81  445.0 189.74
## - wrist    1      2.59  445.8 190.09
## - knee     1      4.39  447.6 190.90
## <none>                  443.2 190.92
## - ankle    1      5.82  449.0 191.54
## - biceps   1      8.43  451.6 192.71
## - chest    1     16.14  459.3 196.11
## - forearm  1     20.38  463.6 197.96
## - thigh    1     22.32  465.5 198.80
## - abdom    1     32.17  475.3 203.01
## - adipos   1     44.28  487.5 208.06
```

```
## - weight   1    473.12  916.3 334.92
## - free     1   3134.53 3577.7 608.71
##
## Step:  AIC=188.98
## 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  443.5 187.06
## - age      1       1.2  444.5 187.51
## - height   1       1.8  445.1 187.79
## - wrist    1       2.5  445.8 188.09
## <none>                  443.3 188.98
## - knee     1       4.6  447.9 189.04
## - ankle    1       6.1  449.4 189.74
## - biceps   1       8.3  451.7 190.73
## - chest    1      16.6  459.9 194.38
## - forearm  1      20.3  463.6 195.99
## - thigh    1      22.2  465.5 196.80
## - abdom    1      32.2  475.5 201.06
## - adipos   1      46.9  490.2 207.18
## - weight   1     485.6  928.9 335.68
## - free     1    3255.7 3699.0 613.42
##
## Step:  AIC=187.06
## siri ~ age + weight + height + adipos + free + chest + abdom +
##     thigh + knee + ankle + biceps + forearm + wrist
##
##           Df Sum of Sq    RSS    AIC
## - age      1       1.3  444.7 185.63
## - height   1       2.1  445.6 186.02
## - wrist    1       2.4  445.9 186.17
## <none>                  443.5 187.06
## - knee     1       4.4  447.9 187.07
## - ankle    1       6.3  449.7 187.88
## - biceps   1       8.5  452.0 188.88
## - chest    1      19.2  462.6 193.56
## - forearm  1      21.8  465.3 194.71
## - thigh    1      23.1  466.5 195.25
## - abdom    1      32.7  476.2 199.35
## - adipos   1      48.5  492.0 205.94
## - weight   1     574.7 1018.2 352.12
## - free     1    3299.0 3742.4 613.76
##
## Step:  AIC=185.63
## siri ~ weight + height + adipos + free + chest + abdom + thigh +
##     knee + ankle + biceps + forearm + wrist
##
##           Df Sum of Sq    RSS    AIC
## - height   1       1.9  446.6 184.48
## <none>                  444.7 185.63
## - wrist    1       4.6  449.3 185.69
## - ankle    1       5.8  450.5 186.23
## - knee     1       6.2  450.9 186.41
```

```
## - biceps   1       9.1  453.8 187.70
## - chest    1      19.8  464.5 192.37
## - forearm  1      20.7  465.4 192.78
## - thigh    1      22.5  467.2 193.54
## - abdom    1      41.6  486.3 201.60
## - adipos   1      48.9  493.7 204.62
## - weight   1     582.4 1027.2 351.88
## - free     1    3368.2 3812.9 615.51
##
## Step:  AIC=184.48
## siri ~ weight + adipos + free + chest + abdom + thigh + knee +
##     ankle + biceps + forearm + wrist
##
##           Df Sum of Sq    RSS    AIC
## <none>                  446.6 184.48
## - wrist    1       5.2  451.8 184.80
## - knee     1       5.6  452.2 184.97
## - ankle    1       5.9  452.5 185.13
## - biceps   1       9.6  456.2 186.74
## - chest    1      20.0  466.6 191.29
## - forearm  1      21.0  467.6 191.71
## - thigh    1      21.1  467.7 191.75
## - abdom    1      42.7  489.3 200.83
## - adipos   1      81.3  527.9 216.08
## - weight   1     674.7 1121.3 367.51
## - free     1    3379.8 3826.4 614.22
```

```
##
## Call:
## lm(formula = siri ~ weight + adipos + free + chest + abdom +
##     thigh + knee + ankle + biceps + forearm + wrist, data = train)
##
## Coefficients:
## (Intercept)       weight       adipos         free        chest
##    -11.7794       0.3556      -0.5539      -0.5556       0.1194
##       abdom        thigh         knee        ankle       biceps
##      0.1671       0.1474       0.1434       0.1297       0.1364
##     forearm        wrist
##      0.2380       0.2840
```

The final AIC of the last model is 180.29 compared to the original AIC of 185.28. The final model variables are: age + weight + height + adipos + free + chest + abdom + thigh + ankle + biceps + forearm.

```
tr_error_bck = rmse(train$siri,predict(lm(formula = siri ~ age + weight + height + adipos + free + ches
    abdom + thigh + ankle + biceps + forearm, data = train), train[-1]))


tst_error_bck = rmse(test$siri,predict(lm(formula = siri ~ age + weight + height + adipos + free + ches
    abdom + thigh + ankle + biceps + forearm, data = train), test[-1]))
```

The training rmse for the new model is 1.5 and the test is 1.38.

Question 4a

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 λ (lambda.min). Use this λ 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

```
library(glmnet)
```

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

```
##
## Attaching package: 'Matrix'
```

```
## The following object is masked from 'package:tidyr':
##
##     expand
```

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

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

```
##
## Attaching package: 'glmnet'
```

```
## The following object is masked from 'package:Metrics':
##
##     auc
```

```
x = as.matrix(train[-1])
y = as.matrix(train[1])
lambdas = 10^seq(3, -5, by = -.1)

model.cv=cv.glmnet(x,y,alpha=0,nfolds=10)
opt_lambda = model.cv$lambda.min
model3= glmnet(x,y, alpha = 0, lambda = opt_lambda )
coef(model3,s = model.cv$lambda.min)
```

```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                          1
## (Intercept) -50.31144513
## age           0.03859888
## weight        0.07402405
## height        0.11798343
## adipos        0.13411374
## free         -0.34374037
## neck          0.00241986
## chest         0.17519362
## abdom         0.28049916
## hip           0.13131196
## thigh         0.18941063
## knee          0.32377759
## ankle         0.13421424
## biceps        0.14805977
## forearm       0.26438424
## wrist        -0.21745368
```

```
summary(model3)
```

```
##           Length Class     Mode
## a0         1     -none-    numeric
## beta      15     dgCMatrix S4
## df         1     -none-    numeric
## dim        2     -none-    numeric
## lambda     1     -none-    numeric
## dev.ratio  1     -none-    numeric
## nulldev    1     -none-    numeric
## npasses    1     -none-    numeric
## jerr       1     -none-    numeric
## offset     1     -none-    logical
## call       5     -none-    call
## nobs       1     -none-    numeric
```

```
tr_err_ridge = rmse(train$siri,predict(model3, as.matrix(train[-1])))

tst_error_ridge = rmse(test$siri,predict(model3, as.matrix(test[-1])))
```

The training error is 2.31 and the testing error is 2.24 for the ridge regression with an optimal lambda of .73.

Question 4b

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 λ (lambda.min). Use this λ to build the final model. Report the final variables obtained (non 0 coefficients), the model training and testing error

```
model.cv=cv.glmnet(x,y,alpha=1,nfolds=10)
opt_lambda = model.cv$lambda.min
model4= glmnet(x,y, alpha = 1, lambda = opt_lambda )
coef = coef(model4,s = model.cv$lambda.min)
summary(model4)
```

```
##           Length Class     Mode
## a0         1     -none-    numeric
## beta      15     dgCMatrix S4
## df         1     -none-    numeric
## dim        2     -none-    numeric
## lambda     1     -none-    numeric
## dev.ratio  1     -none-    numeric
## nulldev    1     -none-    numeric
## npasses    1     -none-    numeric
## jerr       1     -none-    numeric
## offset     1     -none-    logical
## call       5     -none-    call
## nobs       1     -none-    numeric
```

```
trn_error_lasso = rmse(train$siri,predict(model4, as.matrix(train[-1])))

tst_error_lasso = rmse(test$siri,predict(model4, as.matrix(test[-1])))

coef
```

```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                          1
## (Intercept) -13.37838855
## age           .
## weight        0.27334842
## height        0.06217568
## adipos        .
## free         -0.47806822
## neck          .
## chest         0.04967245
## abdom         0.20962097
## hip           .
## thigh         0.07034997
## knee          0.24434131
## ankle         0.03654206
## biceps        0.08986242
## forearm       0.21426383
## wrist         .
```

The training data had a RMSE of 1.67 and the test data had an rmse of 1.57 with an optimal lambda of 0.13. The coefficients in the model are: weight, height, free, chest, abdom, thigh, knee, ankle, biceps, and forearm.

Question 4c

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?

The forward and backward regression had the lowest test error at 1.39, they also had the lowest number of covariates with 10 for forward and 11 for backward. The lower rmse and lower number of covariates suggest that these models have lower variance. For this data set I would choose forward regression as the best model because it has the lowest rmse and the fewest number of predictors.

Question 5

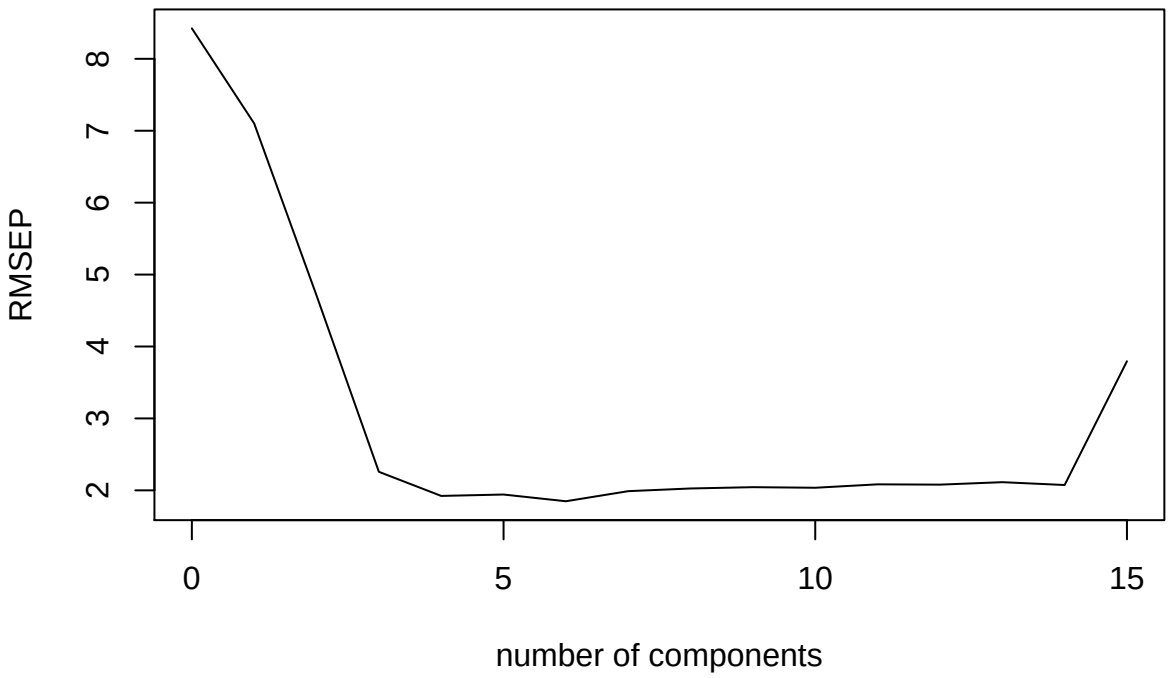
```
library(pls)
```

```
##
## 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, validation="CV", ncomp=15)
pcrCV = RMSEP(pcrmod, estimate="CV")
plot(pcrCV,main="")
```



It looks like 6 PC's has the lowest RMSE.

```
tr_err_p = rmse(train$siri,predict(pcrmod,train,ncomp = 6))
tst_err_p = rmse(test$siri,predict(pcrmod,test,ncomp = 6))
```

The training error was 1.66 and the testing error was 1.4. This has an rmse close to the lowest previous model. I still would choose the previous step regression models because it has the added benefit of being able to use the model as an explanation of the system as a whole. If predictive power was the number one priority, however, I would consider this model also.