498818_SDS496_HW4

2024-09-21

Homework Set 4 SDS 496

Download the prostate data set, which we discussed in lecture 4, under Modules on Canvas. We are interested in predicting lpsa (logarithm of prostate specific antigen) by making use of information on lweight (logarithm of cancer weight).

Problem 1

```
# Load the data
library(data.table)
prostate <- fread('/Users/aidanashrafi/Downloads/prostate.txt')

# Inspect the data
head(prostate)
```

```
##        lcavol lweight   age      lbph   svi      lcp gleason pgg45     lpsa
##         <num>   <num> <int>     <num> <int>    <num>   <int> <int>    <num>
## 1: -0.5798185  2.7695    50 -1.386294     0 -1.38629       6     0 -0.43078
## 2: -0.9942523  3.3196    58 -1.386294     0 -1.38629       6     0 -0.16252
## 3: -0.5108256  2.6912    74 -1.386294     0 -1.38629       7    20 -0.16252
## 4: -1.2039728  3.2828    58 -1.386294     0 -1.38629       6     0 -0.16252
## 5:  0.7514161  3.4324    62 -1.386294     0 -1.38629       6     0  0.37156
## 6: -1.0498221  3.2288    50 -1.386294     0 -1.38629       6     0  0.76547
```

```
str(prostate)
```

```
## Classes 'data.table' and 'data.frame':   97 obs. of  9 variables:
##  $ lcavol : num  -0.58 -0.994 -0.511 -1.204 0.751 ...
##  $ lweight: num  2.77 3.32 2.69 3.28 3.43 ...
##  $ age    : int  50 58 74 58 62 50 64 58 47 63 ...
##  $ lbph   : num  -1.39 -1.39 -1.39 -1.39 -1.39 ...
##  $ svi    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ lcp    : num  -1.39 -1.39 -1.39 -1.39 -1.39 ...
##  $ gleason: int  6 6 7 6 6 6 6 6 6 6 ...
##  $ pgg45  : int  0 0 20 0 0 0 0 0 0 0 ...
##  $ lpsa   : num  -0.431 -0.163 -0.163 -0.163 0.372 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
sum(as.numeric(is.na(prostate)))
```

```
## [1] 0
```

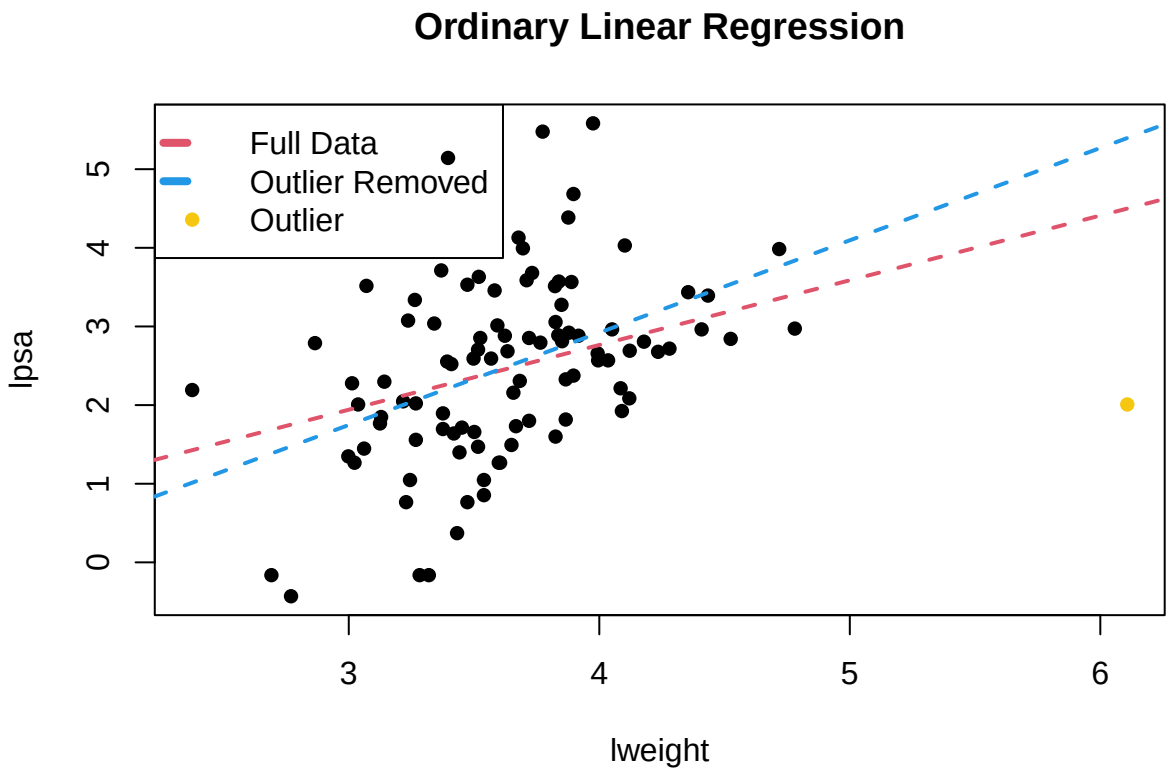
i. Fit a suitable simple linear model. Are there any patients with an exceedingly high influence on the fitted regression line?

ii. Remove the patient with the highest influence on the fitted regression line and fit the simple linear model again. How much has the fitted regression line been impacted?

```
# Fit a suitable simple linear model
slr <- lm(lpsa ~ lweight, data= prostate)
summary(slr)
```

```
##
## Call:
## lm(formula = lpsa ~ lweight, data = prostate)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.4908 -0.6474 -0.1022  0.6290  2.8994
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.5281     0.8220  -0.642 0.522120
## lweight       0.8231     0.2230   3.691 0.000373 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.085 on 95 degrees of freedom
## Multiple R-squared:  0.1254, Adjusted R-squared:  0.1162
## F-statistic: 13.62 on 1 and 95 DF,  p-value: 0.0003729
```

```
outlier = cooks.distance(slr) == max(cooks.distance(slr))
removed <- lm(lpsa~lweight, data=prostate, !outlier)
plot(lpsa~lweight, prostate, col=6*outlier + 1, pch=16, main = "Ordinary Linear Regression")
abline(slr, col=2,lty=2,lwd=2)
abline(removed, col=4,lty=2,lwd=2)
legend('topleft', c('Full Data', 'Outlier Removed', 'Outlier'), col = c(2,4,7), lty = c(2,2,NA), lwd=c(
```




```
print(which.max(outlier))
```

```
## 32
## 32
```

i. Yes, there is a patient with an exceedingly high influence on the fitted regression line. Patient # 32, or the patient in index 32 is the patient that is carrying this high influence.

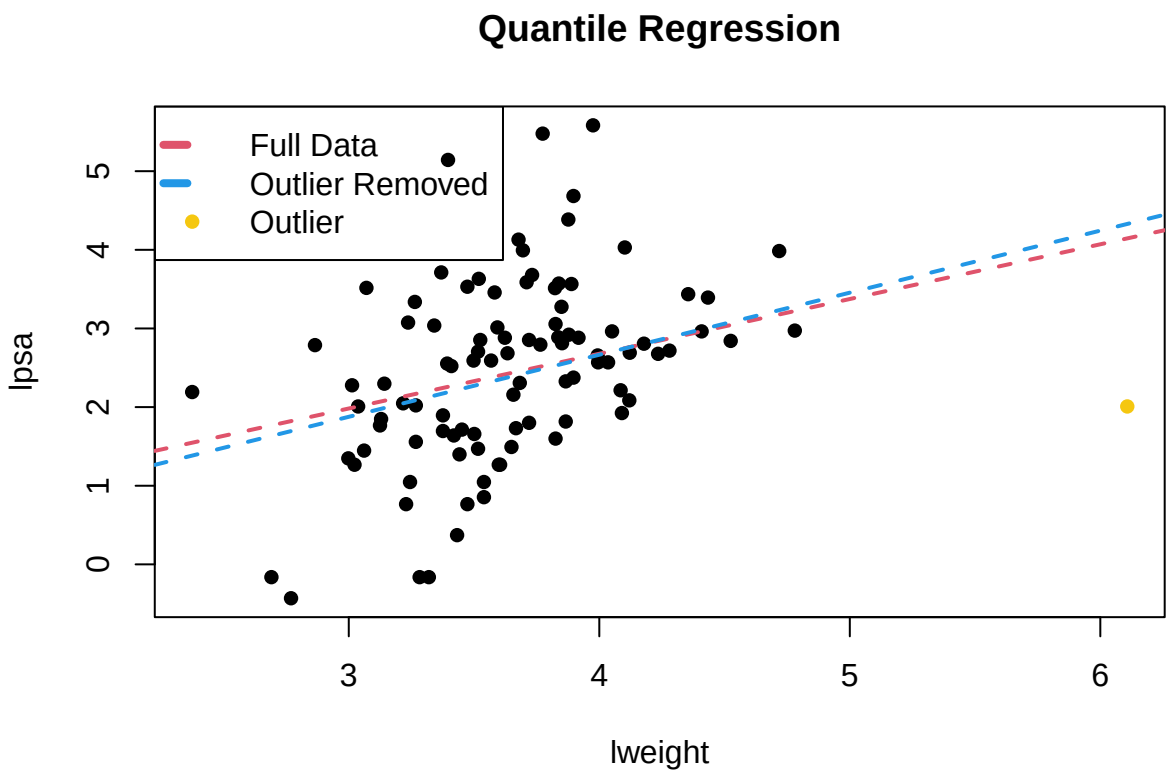
ii. The fitted regression line has been dramatically impacted by the removal of just one data point, verifying the earlier conclusion that there is a patient with an exceedingly high influence on the fitted regression line, the patient in index 32.

iii. Try out 2 different robust regression methods. How much are the fitted regression lines impacted now if you remove the previously identified patient with the highest influence on the ordinary regression line?

```
# Robust Regression Method 1
# Quantile Regression
library(quantreg)
```

```
## Loading required package: SparseM
```

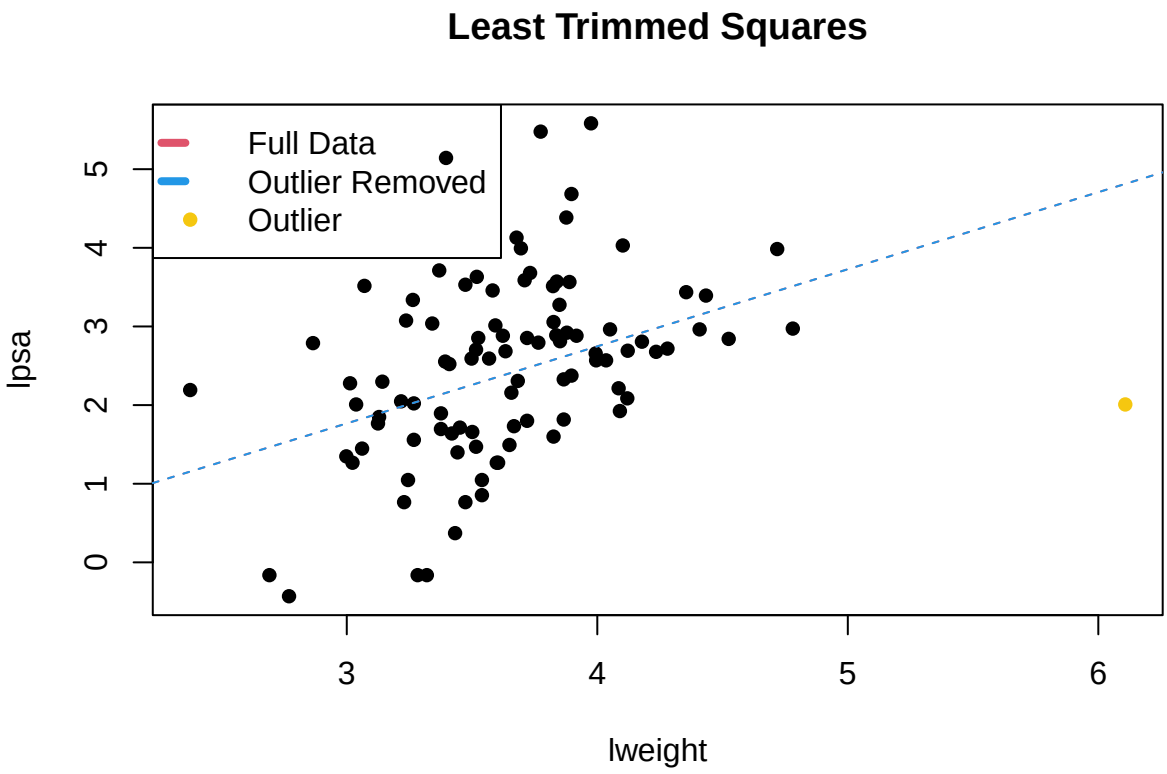
```
qr <- rq(lpsa ~ lweight, prostate, tau=0.5)
removed_qr = rq(lpsa~ lweight, prostate, !outlier, tau=0.5)
plot(lpsa~ lweight, prostate, col = 6*outlier + 1, pch = 16, main = 'Quantile Regression')
abline(qr, col = 2, lwd = 2, lty = 2)
abline(removed_qr, col = 4, lty = 2,lwd = 2)
legend('topleft', c('Full Data', 'Outlier Removed', 'Outlier'), col=c(2,4,7), lty=c(2,2,NA),lwd=c(4,4,N
```



Using a robust regression method, we expect that the influence of outliers on the regression line will be minimal/significantly reduced.

Specifically using Quantile Regression, we can see that conducting Quantile Regression on the full data is almost identical to conducting Quantile Regression on the data with the outlier removed. This agrees with our understanding of robust regression methods.

```
# Robust Regression Method 2
# Least Trimmed Squares
library(MASS)
lts <- lqs(lpsa ~ lweight, prostate)
lts_removed <- lqs(lpsa ~ lweight, prostate, subset =! outlier)
plot(lpsa ~ lweight, prostate, col = 6 * outlier + 1, pch = 16, main = 'Least Trimmed Squares')
abline(lts, col = 2, lty = 2, lwd = 1)
abline(lts_removed, col = 4, lty = 2, lwd=1)
legend('topleft', c('Full Data', 'Outlier Removed', 'Outlier'), col = c(2,4,7), lty = c(2,2,NA), lwd =
```



Specifically using Least Trimmed Squares, we can see that conducting Least Trimmed Squares on the full data is identical to conducting Least Trimmed Squares on the data with the outlier removed. This agrees with our understanding of robust regression methods.

The key insight here is that robust regression methods allow for the use of the full data without the need to remove outliers because of the way these methods handle extremes.

iv. Apply the bootstrap method to obtain an estimate for the standard error of the estimated slope coefficient for each of the robust regression methods you utilized. How do these estimated standard errors compare against the estimated standard error from the ordinary linear regression model?

```
set.seed(123) # For reproducibility
# Apply the bootstrap method
B = 10000
n = dim(prostate)[1]
beta = numeric(B)
beta_qr <- numeric(B)
beta_lts <- numeric(B)
for (k in 1:B){
  boot_sample = sample(n,replace=TRUE)
  boot = lm(lpsa ~ lweight, data = prostate[boot_sample, ])
  beta[k] = boot$coefficients[2]
  # Quantile Regression (QR)
  boot_qr <- rq(lpsa ~ lweight, data = prostate[boot_sample, ], tau = 0.5)
  beta_qr[k] <- boot_qr$coefficients[2]

  # Least Trimmed Squares (LTS)
  boot_lts <- lqs(lpsa ~ lweight, data = prostate[boot_sample, ])
  beta_lts[k] <- boot_lts$coefficients[2]
```

```
}
```

```
## Warning in rq.fit.br(x, y, tau = tau, ...): Solution may be nonunique
## Warning in rq.fit.br(x, y, tau = tau, ...): Solution may be nonunique
```

```
# Non-parametric estimate of the Standard Error for the slope coefficient
sd(beta)
```

```
## [1] 0.3145218
```

```
sd(beta_qr)
```

```
## [1] 0.321103
```

```
sd(beta_lts)
```

```
## [1] 0.5569895
```

Applying the **bootstrap method with SLR** we get an estimated standard error of **0.3145218** for the beta coefficient of the predictor variable. Without the bootstrap method, **SLR** gives us an estimated standard error of **0.2230** for the beta coefficient of the predictor variable.

Applying the **bootstrap method with QR** we get an estimated standard error of **0.321103**. This is very close to the estimated standard error using the bootstrap method with SLR.

It is larger than the estimated standard error for the beta coefficient of the predictor variable using Ordinary SLR. Because QR is robust to outliers and skewness, this can lead to larger standard errors, especially in the presence of outliers and/or skewness. QR focuses on quantiles rather than the mean, which may result in higher standard errors under skewness.

Applying the **bootstrap method with LTS** we get an estimated standard error of **0.5569895**. This is relatively far away from the estimated standard error for the beta coefficient of the predictor variable using bootstrap method with QR, and bootstrap method with SLR.

It is larger than the estimated standard error for the beta coefficient of the predictor variable using Ordinary SLR. This can be because LTS trims the data, and when using bootstrap, we sample from the data alredy in a subsetting manner, so taking a double subset essentially gives us more potential for variation, and thus we get the largest estimated standard error for the beta coefficient of the predictor variable.

The key takeaway here is that QR and LTS tend to have larger standard errors in the presence of outliers, compared to Ordinary SLR. This is because robust methods like QR and LTS are designed to reduce the influence of outliers, but that robustness can increase the variability of the slope estimates (hence the larger standard errors) when outliers are present.

Problem 2

```
# Load the data
seatpos <- fread('/Users/aidanashrafi/Downloads/seatpos.txt')
# Inspect the data
head(seatpos)
```

```
##      Age Weight HtShoes    Ht Seated   Arm Thigh   Leg hipcenter
##    <int>  <int>   <num> <num>  <num> <num> <num> <num>     <num>
## 1:    46    180   187.2 184.9   95.2  36.1  45.3  41.3  -206.300
## 2:    31    175   167.5 165.5   83.8  32.9  36.5  35.9  -178.210
## 3:    23    100   153.6 152.2   82.9  26.0  36.6  31.0   -71.673
## 4:    19    185   190.3 187.4   97.3  37.4  44.1  41.0  -257.720
## 5:    23    159   178.0 174.1   93.9  29.5  40.1  36.9  -173.230
```

```
## 6:    47    170   178.7 177.0   92.4  36.0  43.2  37.4  -185.150
```

```
sum(is.na(seatpos))
```

```
## [1] 0
```

```
str(seatpos)
```

```
## Classes 'data.table' and 'data.frame':   38 obs. of  9 variables:
##  $ Age      : int  46 31 23 19 23 47 30 28 23 29 ...
##  $ Weight   : int  180 175 100 185 159 170 137 192 150 120 ...
##  $ HtShoes  : num  187 168 154 190 178 ...
##  $ Ht       : num  185 166 152 187 174 ...
##  $ Seated   : num  95.2 83.8 82.9 97.3 93.9 92.4 87.7 96.9 91.4 85.2 ...
##  $ Arm      : num  36.1 32.9 26 37.4 29.5 36 32.5 35.8 29.4 26.6 ...
##  $ Thigh    : num  45.3 36.5 36.6 44.1 40.1 43.2 35.6 39.9 35.5 31 ...
##  $ Leg      : num  41.3 35.9 31 41 36.9 37.4 36.2 43.1 33.4 32.8 ...
##  $ hipcenter: num  -206.3 -178.2 -71.7 -257.7 -173.2 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

i. Split the data set into a training set and a test set. Fit a multiple linear model with all available predictors on the test set and provide a summary of its results.

```
# Training and Test Split
# Set seed for reproducibility
set.seed(123)
n = nrow(seatpos)
train = sample(n, 0.6*n)
test = setdiff(1:n,train)
mlr_train <- lm(hipcenter ~ . , data = seatpos, subset = train)
mlr_test <- lm(hipcenter ~ . , data = seatpos, subset = test)
summary(mlr_test)
```

```
##
## Call:
## lm(formula = hipcenter ~ ., data = seatpos, subset = test)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -39.59 -22.64   0.10  11.97  59.65
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 299.6964   258.6916   1.159    0.285
## Age           0.5662     1.0306   0.549    0.600
## Weight       -0.2650     0.4355  -0.608    0.562
## HtShoes      -2.9596    21.0135  -0.141    0.892
## Ht            5.0455    22.7185   0.222    0.831
## Seated       -2.2681     8.5958  -0.264    0.799
## Arm          -3.2935     8.2100  -0.401    0.700
## Thigh        -4.5404     4.7507  -0.956    0.371
## Leg          -8.4005    10.4462  -0.804    0.448
##
## Residual standard error: 39.49 on 7 degrees of freedom
## Multiple R-squared:  0.7937, Adjusted R-squared:  0.5578
## F-statistic: 3.365 on 8 and 7 DF,  p-value: 0.06368
```

None of the predictors in the model are statistically significant, and the Adjusted R squared value is **0.5578**, indicating that while the model explains some of the variance in the response variable, it does not provide strong explanatory power.

The standard errors for the predictors HtShoes and Ht are extremely large, which suggests the presence of multicollinearity. Multicollinearity occurs when two or more predictors are highly correlated, making it difficult for the model to distinguish their individual effects on the response variable. In this case, HtShoes and Ht are likely correlated, and this leads to inflated standard errors and wider confidence intervals for these variables, reducing their statistical significance.

Multicollinearity not only impacts individual predictor estimates but also harms the model's overall stability. It can result in unreliable coefficient estimates, and even though the Adjusted R-squared is moderately high, the lack of significant predictors suggests that the model's interpretability and generalizability are compromised.

After the Change in Instructions this is no longer true, however for personal understanding and reference, I left the information below.

Please do not grade this, but I would appreciate it if you could provide feedback on it.

When fitting a multiple linear regression model with all available predictors on the test set, several issues arose due to the small sample size and the improper use of the test set for model fitting:

No Residual Degrees of Freedom: Since the test set had the same number of observations (8) as the number of estimated coefficients (8), the model perfectly fit the test set, leaving no residuals to evaluate. This resulted in NaN (Not a Number) values for standard errors, t-values, and p-values, indicating that the model could not meaningfully estimate uncertainty or test hypotheses about the predictors.

Overfitting: The model achieved a perfect fit, but this is misleading because it is overfitted to the test data, meaning it will not generalize well to new, unseen data.

Singularity Issues: One predictor was automatically dropped due to multicollinearity, further complicating the model fit and highlighting the problem of fitting a model with highly correlated predictors on a small dataset.

Continue Grading

ii. Apply the backward elimination procedure on the training set with the Bayesian information criterion (BIC) as both the predictor selection criterion and the stopping criterion. Fit the final linear model on the test set and provide a summary of its results.

```
set.seed(123) # Reproducibility
# Make sure that the penalty used is the correct number based on the training set
w = nrow(seatpos[train, ])
mlr_bic_optimal <- step(mlr_train,  formula(mlr_train), direction = 'backward', k = log(w))
```

```
## Start:  AIC=181.49
## hipcenter ~ Age + Weight + HtShoes + Ht + Seated + Arm + Thigh +
##     Leg
##
##           Df Sum of Sq   RSS    AIC
## - Arm      1      0.76 23773 178.40
## - HtShoes  1    175.52 23948 178.56
## - Weight   1    186.80 23959 178.57
## - Seated   1    395.53 24168 178.77
## - Thigh    1    633.64 24406 178.98
## - Ht       1    752.03 24524 179.09
## - Age      1   1279.34 25051 179.56
```

```
## - Leg      1   2548.94 26321 180.64
## <none>                 23772 181.49
##
## Step:  AIC=178.4
## hipcenter ~ Age + Weight + HtShoes + Ht + Seated + Thigh + Leg
##
##           Df Sum of Sq   RSS    AIC
## - HtShoes  1     176.3 23949 175.47
## - Weight   1     195.3 23968 175.49
## - Seated   1     464.8 24238 175.74
## - Thigh    1     668.7 24442 175.92
## - Ht       1     762.2 24535 176.01
## - Age      1    1605.7 25378 176.75
## <none>                 23773 178.40
## - Leg      1    4014.9 27788 178.75
##
## Step:  AIC=175.48
## hipcenter ~ Age + Weight + Ht + Seated + Thigh + Leg
##
##          Df Sum of Sq   RSS    AIC
## - Weight  1     118.0 24067 172.49
## - Thigh   1     516.2 24465 172.85
## - Seated  1     649.5 24598 172.97
## - Age     1    2051.9 26001 174.19
## - Ht      1    3365.4 27314 175.28
## <none>                 23949 175.47
## - Leg     1    3872.2 27821 175.68
##
## Step:  AIC=172.49
## hipcenter ~ Age + Ht + Seated + Thigh + Leg
##
##          Df Sum of Sq   RSS    AIC
## - Thigh   1     438.3 24505 169.80
## - Seated  1    1120.6 25188 170.40
## - Ht      1    3253.3 27320 172.19
## <none>                 24067 172.49
## - Leg     1    3777.3 27844 172.61
## - Age     1    4545.3 28612 173.21
##
## Step:  AIC=169.8
## hipcenter ~ Age + Ht + Seated + Leg
##
##          Df Sum of Sq   RSS    AIC
## - Seated  1     878.6 25384 167.48
## - Ht      1    2928.1 27434 169.19
## <none>                 24505 169.80
## - Leg     1    4210.8 28716 170.20
## - Age     1    5036.1 29542 170.82
##
## Step:  AIC=167.48
## hipcenter ~ Age + Ht + Leg
##
##        Df Sum of Sq   RSS    AIC
## - Ht    1    2444.1 27828 166.41
```

```
## - Leg   1    3807.8 29192 167.47
## <none>              25384 167.48
## - Age   1    4158.6 29543 167.73
##
## Step:  AIC=166.41
## hipcenter ~ Age + Leg
##
##        Df Sum of Sq   RSS    AIC
## <none>              27828 166.41
## - Age   1      6007 33835 167.62
## - Leg   1     44234 72062 184.25
```

```
# Use the best fitted model from BIC on the test set
bic_optimal <- lm(formula(mlr_bic_optimal), data = seatpos, subset = test)
summary(bic_optimal)
```

```
##
## Call:
## lm(formula = formula(mlr_bic_optimal), data = seatpos, subset = test)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -42.875 -28.136   2.305  13.749  53.319
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 314.81005   88.08949   3.574   0.0034 **
## Age          -0.07485    0.58162  -0.129   0.8996
## Leg         -12.97953    2.20298  -5.892 5.31e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 33.09 on 13 degrees of freedom
## Multiple R-squared:  0.731,  Adjusted R-squared:  0.6896
## F-statistic: 17.66 on 2 and 13 DF,  p-value: 0.0001967
```

The adjusted R squared for this model is significantly better than the full model, increasing to **0.6896**. The predictor leg is now statistically significant, whereas in the full model, none of the predictors were statistically significant. This suggests that backward elimination improved the model by removing irrelevant predictors, leading to a more interpretable and reliable result.

iii. Apply the forward selection procedure on the training set with the t test of statistical significance as a predictor selection criterion and the adjusted coefficient of determination as a model selection criterion. Fit the best linear model on the test set and provide a summary of its results.

```
library(car)
```

Manual Iteration:

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

```
# Iterate through starting from an empty model to apply forward selection
manual_model <- lm(hipcenter ~ 1, data = seatpos, subset = train)
add1(manual_model, ~ Age + Weight + Ht + HtShoes + Seated + Arm + Leg + Thigh, test = 'F')
```

```
## Single term additions
##
## Model:
## hipcenter ~ 1
##         Df Sum of Sq   RSS    AIC F value    Pr(>F)
## <none>               78507 181.96
## Age      1      6444 72062 182.07  1.7885 0.1961252
## Weight   1     22216 56291 176.64  7.8932 0.0108276 *
## Ht       1     46413 32093 164.28 28.9237 2.900e-05 ***
## HtShoes  1     46076 32431 164.51 28.4150 3.230e-05 ***
## Seated   1     40334 38173 168.09 21.1320 0.0001746 ***
## Arm      1     18295 60212 178.12  6.0769 0.0228704 *
## Leg      1     44672 33835 165.44 26.4056 5.001e-05 ***
## Thigh    1     17570 60936 178.38  5.7669 0.0261701 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
manual_model <- lm(hipcenter ~ HtShoes, data=seatpos, subset=train)
add1(manual_model, ~ Age + Weight + Ht + Seated + Arm + Leg + HtShoes + Thigh, test='F')
```

```
## Single term additions
##
## Model:
## hipcenter ~ HtShoes
##        Df Sum of Sq   RSS    AIC F value Pr(>F)
## <none>              32431 164.51
## Age     1   2983.99 29447 164.38  1.9254 0.1813
## Weight  1   1047.76 31383 165.78  0.6343 0.4356
## Ht      1    341.69 32089 166.28  0.2023 0.6579
## Seated  1      2.50 32428 166.51  0.0015 0.9699
## Arm     1    164.63 32266 166.40  0.0969 0.7589
## Leg     1   2601.56 29829 164.67  1.6571 0.2135
## Thigh   1    732.55 31698 166.00  0.4391 0.5155
```

Based on the predictor selection criterion, we will only add HtShoes. This does not seem great as prior to adding HtShoes, there were numerous statistically significant predictor variables, so let's see if extreme multicollinearity could be causing this issue.

Mathematical Theory and Reasoning:

The reason we could manually iterate through from an empty model to apply forward selection on the training set with the t test of statistical significance as a predictor selection criterion is because the partial F test is identical to the 'marginal'/individual t test for each predictor variable when the null hypothesis is testing two-sided significance (existence or not), i.e., beta hat for the predictor variable is 0 or not. As a result of this, when iterating through and using add1() we can actually consider the partial F test statistic equivalent to the t test for each predictor because the F test is determining if the variable should be added to the model or not, and this directly translates to the beta hat coefficient being nonzero or not. As a result of this understanding, I greatly simplified the forward selection procedure.

```
full_model <- lm(hipcenter ~ ., data = seatpos, subset=train)
summary(full_model)
```

```
##
## Call:
## lm(formula = hipcenter ~ ., data = seatpos, subset = train)
##
```

```
## Residuals:
##     Min      1Q  Median      3Q     Max
## -65.630 -20.747   1.446  19.868  78.178
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 511.3694   356.9121   1.433    0.176
## Age           0.8611     1.0295   0.836    0.418
## Weight        0.2864     0.8962   0.320    0.754
## HtShoes       4.3323    13.9834   0.310    0.762
## Ht           -9.2985    14.4995  -0.641    0.532
## Seated        3.7079     7.9726   0.465    0.650
## Arm          -0.1460     7.1642  -0.020    0.984
## Thigh         2.4709     4.1975   0.589    0.566
## Leg          -9.5595     8.0968  -1.181    0.259
##
## Residual standard error: 42.76 on 13 degrees of freedom
## Multiple R-squared:  0.6972, Adjusted R-squared:  0.5109
## F-statistic: 3.742 on 8 and 13 DF,  p-value: 0.0173
```

```
vif(full_model)
```

```
##        Age     Weight    HtShoes         Ht     Seated        Arm      Thigh
##   2.963459   5.705923 215.339639 228.739174  13.964673   6.950958   3.005497
##        Leg
##   6.610805
```

We can see that the VIF values for some of the predictor variables in this model are extremely high, > 200, which could be the source of confusion earlier seeing several predictor variables deemed statistically significant individually with respect to the response variable, but once the most significant predictor was added, others were no longer significant.

Because the predictor selection criterion has led us to choose only 1 predictor variable, HtShoes, there is no further testing necessary to determine which model is the best based on adjusted R squared because we can only look to optimize our model selection from various choices based on adjusted R squared criterion if there is more than 1 predictor variable (more than 1 possible model).

There are no other choices to make essentially, no other models to compare with, so the predictor selection criterion has also acted as the model selection criterion in this specific case.

iv. Further split the training set into a smaller training set and a validation set.

```
set.seed(123)  # Reproducibility

n <- nrow(seatpos)  # Get total number of rows
n
```

```
## [1] 38
```

```
# First, split into 60% training and 40% test
train <- sample(1:n, 0.6 * n)  # 60% of total data
test <- setdiff(1:n, train)  # The remaining 40% for the test set

# Further split the 60% training set: 2/3 of the 60% for final training (40% of total), 1/3 for validat
final_train <- sample(train, 2/3 * length(train))  # Correctly sample 2/3 of the 60% train set
validation <- setdiff(train, final_train)  # The remaining 1/3 of the 60% train set for validation
```

```r
# Summary of data splitting
length(final_train)  # 40% of the original data for final training
```

```
## [1] 14
```

```r
length(validation)   # 20% of the original data for validation
```

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

```r
length(test)         # 40% of the original data for testing
```

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

iv. Apply the backward elimination procedure on the training set with the t test of statistical significance as a predictor selection criterion.

Manual Approach

```r
final_train_model <- lm(hipcenter ~ . , data = seatpos, subset = final_train)
summary(final_train_model)
```

```
##
## Call:
## lm(formula = hipcenter ~ ., data = seatpos, subset = final_train)
##
## Residuals:
##        7       38       26       33       35       19       28       29
##   4.7531 -24.7059  -1.1512  12.6497 -25.6724  43.0110   0.4546  -6.3420
##        5        9       34       10       36        4
## -20.0993 -20.1995   4.0731   3.1500   6.6038  23.4750
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 149.0620   377.8483   0.395   0.7095
## Age           3.3747     1.0437   3.233   0.0231 *
## Weight        1.0283     1.2784   0.804   0.4577
## HtShoes     -22.4445    19.7714  -1.135   0.3078
## Ht           19.4056    18.4541   1.052   0.3411
## Seated        5.6662     7.3576   0.770   0.4760
## Arm         -19.4695     9.5715  -2.034   0.0976 .
## Thigh         3.5717     4.6245   0.772   0.4748
## Leg          -0.7491    11.0037  -0.068   0.9484
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 30.91 on 5 degrees of freedom
## Multiple R-squared:  0.8706, Adjusted R-squared:  0.6637
## F-statistic: 4.207 on 8 and 5 DF,  p-value: 0.06514
```

```r
# Remove Leg
final_train_model <- update(final_train_model, . ~ . - Leg)
summary(final_train_model)
```

```
##
## Call:
## lm(formula = hipcenter ~ Age + Weight + HtShoes + Ht + Seated +
##     Arm + Thigh, data = seatpos, subset = final_train)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -26.339 -16.339   2.046   5.951  43.206
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 161.4255   302.6081   0.533  0.61290
## Age           3.4080     0.8419   4.048  0.00674 **
## Weight        1.0833     0.9046   1.197  0.27628
## HtShoes     -23.1856    15.0732  -1.538  0.17492
## Ht           19.9166    15.3970   1.294  0.24339
## Seated        5.6798     6.7172   0.846  0.43022
## Arm         -19.9489     5.9192  -3.370  0.01504 *
## Thigh         3.7423     3.5500   1.054  0.33240
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 28.23 on 6 degrees of freedom
## Multiple R-squared:  0.8705, Adjusted R-squared:  0.7195
## F-statistic: 5.763 on 7 and 6 DF,  p-value: 0.02431
```

```
# Remove Seated
final_train_model <- update(final_train_model, . ~ . - Seated)
summary(final_train_model)
```

```
##
## Call:
## lm(formula = hipcenter ~ Age + Weight + HtShoes + Ht + Arm +
##     Thigh, data = seatpos, subset = final_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -28.981  -8.881  -3.806   7.944  53.294
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 236.8838   283.2022   0.836  0.43053
## Age           3.2729     0.8096   4.042  0.00492 **
## Weight        1.0333     0.8841   1.169  0.28078
## HtShoes     -19.3146    14.0657  -1.373  0.21207
## Ht           18.9754    15.0408   1.262  0.24751
## Arm         -21.0663     5.6511  -3.728  0.00738 **
## Thigh         3.1462     3.4077   0.923  0.38660
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 27.65 on 7 degrees of freedom
## Multiple R-squared:  0.8551, Adjusted R-squared:  0.7309
## F-statistic: 6.885 on 6 and 7 DF,  p-value: 0.01129
```

```
# Remove Thigh
final_train_model <- update(final_train_model, . ~ . - Thigh)
summary(final_train_model)
```

```
##
```

```
## Call:
## lm(formula = hipcenter ~ Age + Weight + HtShoes + Ht + Arm, data = seatpos,
##     subset = final_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -27.674 -15.021  -4.021   8.158  53.819
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 109.0501   244.7569   0.446  0.66774
## Age           3.3391     0.7990   4.179  0.00308 **
## Weight        0.6225     0.7569   0.822  0.43467
## HtShoes     -18.3205    13.8945  -1.319  0.22382
## Ht           19.5985    14.8864   1.317  0.22446
## Arm         -20.0447     5.4904  -3.651  0.00649 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 27.39 on 8 degrees of freedom
## Multiple R-squared:  0.8375, Adjusted R-squared:  0.7359
## F-statistic: 8.243 on 5 and 8 DF,  p-value: 0.005113
```

```r
# Remove Weight
final_train_model <- update(final_train_model, . ~ . - Weight)
summary(final_train_model)
```

```
##
## Call:
## lm(formula = hipcenter ~ Age + HtShoes + Ht + Arm, data = seatpos,
##     subset = final_train)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -27.23 -16.78  -1.55  13.55  55.54
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -26.8747   177.2496  -0.152  0.88283
## Age           3.4909     0.7633   4.573  0.00134 **
## HtShoes     -16.9360    13.5419  -1.251  0.24260
## Ht           19.5068    14.6159   1.335  0.21478
## Arm         -19.9519     5.3896  -3.702  0.00491 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 26.9 on 9 degrees of freedom
## Multiple R-squared:  0.8237, Adjusted R-squared:  0.7454
## F-statistic: 10.51 on 4 and 9 DF,  p-value: 0.001909
```

```r
# Remove HtShoes
final_train_model <- update(final_train_model, . ~ . - HtShoes)
summary(final_train_model)
```

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

```
## lm(formula = hipcenter ~ Age + Ht + Arm, data = seatpos, subset = final_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -35.379 -20.146   1.973  15.175  51.960
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  34.4488   175.0708   0.197  0.84795
## Age           3.0792     0.7078   4.350  0.00144 **
## Ht            1.3336     1.6156   0.825  0.42836
## Arm         -16.2492     4.6289  -3.510  0.00563 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 27.64 on 10 degrees of freedom
## Multiple R-squared:  0.7931, Adjusted R-squared:  0.731
## F-statistic: 12.78 on 3 and 10 DF,  p-value: 0.0009342
```

```
# Remove Ht
final_train_model <- update(final_train_model, . ~ . - Ht)
summary(final_train_model)
```

```
##
## Call:
## lm(formula = hipcenter ~ Age + Arm, data = seatpos, subset = final_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -44.735 -17.518   3.613  13.422  51.364
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 166.2644    70.7136   2.351 0.038403 *
## Age           2.7079     0.5385   5.028 0.000385 ***
## Arm         -12.9794     2.3600  -5.500 0.000186 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 27.24 on 11 degrees of freedom
## Multiple R-squared:  0.779,  Adjusted R-squared:  0.7388
## F-statistic: 19.38 on 2 and 11 DF,  p-value: 0.000248
```

```
# Final model has Arm + Age
```

Use the mean squared prediction error on the validation set as a model selection criterion

```
# Make sure we have the right updated model
final_train_model
```

```
##
## Call:
## lm(formula = hipcenter ~ Age + Arm, data = seatpos, subset = final_train)
##
## Coefficients:
```

```
## (Intercept)          Age          Arm
##     166.264        2.708      -12.979
```

```r
# Predict using validation set
predictions <- predict(final_train_model, newdata = seatpos[validation, ])
mpse <- mean((seatpos$hipcenter[validation] - predictions)^2)

# Fit a model with just 'Arm' on the smaller training set
model_arm <- lm(hipcenter ~ Arm, data = seatpos, subset = final_train)
# Predict using validation set
predictions_arm <- predict(model_arm, newdata = seatpos[validation, ])
mpse_arm <- mean((seatpos$hipcenter[validation] - predictions_arm)^2)

# Fit a model with just 'Age' on the smaller training set
model_age <- lm(hipcenter ~ Age, data = seatpos, subset = final_train)
# Predict using validation set
predictions_age <- predict(model_age, newdata = seatpos[validation, ])
mpse_age <- mean((seatpos$hipcenter[validation] - predictions_age)^2)

# Output MPSE for each model
mpse
```

```
## [1] 3527.807
```

```r
mpse_arm
```

```
## [1] 4213.43
```

```r
mpse_age
```

```
## [1] 5252.792
```

Based on the model selection criterion, we will choose the model with the lowest MPSE, this is the model with both Age and Arm as predictors for hipcenter.

Fit the best linear model on the test set and provide a summary of its results.

```r
best_linear_model <- lm(hipcenter ~ Age + Arm, data = seatpos, subset = test)
summary(best_linear_model)
```

```
##
## Call:
## lm(formula = hipcenter ~ Age + Arm, data = seatpos, subset = test)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -69.56 -12.62   3.53  16.24  85.94
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 243.1215    99.2862   2.449 0.029285 *
## Age           1.1339     0.7052   1.608 0.131828
## Arm         -13.9873     3.1080  -4.500 0.000597 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 39.63 on 13 degrees of freedom
```

```
## Multiple R-squared:  0.614,  Adjusted R-squared:  0.5546
## F-statistic: 10.34 on 2 and 13 DF,  p-value: 0.002055
```

The results show an Adjusted R squared of **0.5546**, which is not great. The training set for this problem has become smaller, potentially leading to the model failing to capture the overall information as well as previous models with a higher training set spli because of less information and variability in the data.

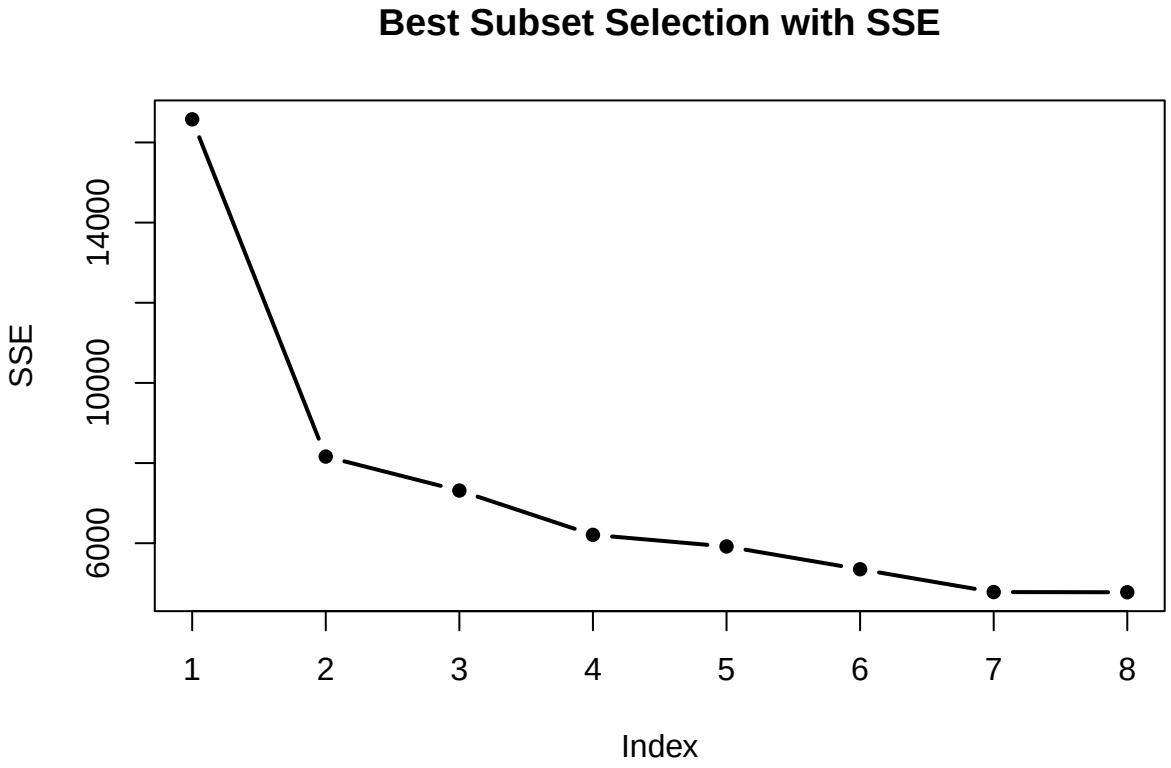
v. Apply the best subset selection method on the training set with the sum of squared errors as a predictor selection criterion.

```
library(leaps)
seatpos_df <-as.data.frame(seatpos)

full <- lm(hipcenter ~ . , seatpos, subset = final_train)
predictors = seatpos_df[, c('Age', 'Weight', 'HtShoes', 'Ht', 'Seated', 'Arm', 'Thigh', 'Leg')]
P = dim(predictors)[2]
SSE = numeric(P)
MSPE = numeric(P)
sub = regsubsets(formula(full), seatpos, nvmax = P, subset = final_train)
summary(sub)$which
```

```
##   (Intercept)   Age Weight HtShoes    Ht Seated   Arm Thigh   Leg
## 1        TRUE FALSE  FALSE   FALSE FALSE  FALSE FALSE FALSE  TRUE
## 2        TRUE  TRUE  FALSE   FALSE FALSE  FALSE  TRUE FALSE FALSE
## 3        TRUE  TRUE   TRUE   FALSE FALSE  FALSE  TRUE FALSE FALSE
## 4        TRUE  TRUE  FALSE   FALSE FALSE   TRUE  TRUE FALSE  TRUE
## 5        TRUE  TRUE  FALSE    TRUE  TRUE  FALSE  TRUE FALSE  TRUE
## 6        TRUE  TRUE   TRUE    TRUE  TRUE  FALSE  TRUE  TRUE FALSE
## 7        TRUE  TRUE   TRUE    TRUE  TRUE   TRUE  TRUE  TRUE FALSE
## 8        TRUE  TRUE   TRUE    TRUE  TRUE   TRUE  TRUE  TRUE  TRUE
```

We are asked to calculate using SSE, so we do not take the mean as the code in lecture does for this part.

```
for (p in 1:P){
  fit = lm(seatpos_df$hipcenter ~ ., predictors[,summary(sub)$which[p,-1], drop=FALSE],final_train)
  predictions = predict(fit,predictors[final_train,summary(sub)$which[p,-1],drop=FALSE])
  SSE[p] = sum((seatpos$hipcenter[final_train] - predictions)^2)
}

plot(SSE,type = 'b', pch = 16, lwd = 2, ylab = 'SSE', main = 'Best Subset Selection with SSE')
```




```
p = which.min(SSE)


fit = lm(seatpos_df$hipcenter ~ ., predictors[,summary(sub)$which[p,-1], drop=FALSE],test)
summary(fit)
```

```
##
## Call:
## lm(formula = seatpos_df$hipcenter ~ ., data = predictors[, summary(sub)$which[p,
##     -1], drop = FALSE], subset = test)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -39.59 -22.64   0.10  11.97  59.65
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 299.6964   258.6916   1.159    0.285
## Age           0.5662     1.0306   0.549    0.600
## Weight       -0.2650     0.4355  -0.608    0.562
## HtShoes      -2.9596    21.0135  -0.141    0.892
## Ht            5.0455    22.7185   0.222    0.831
## Seated       -2.2681     8.5958  -0.264    0.799
## Arm          -3.2935     8.2100  -0.401    0.700
## Thigh        -4.5404     4.7507  -0.956    0.371
## Leg          -8.4005    10.4462  -0.804    0.448
##
## Residual standard error: 39.49 on 7 degrees of freedom
## Multiple R-squared:  0.7937, Adjusted R-squared:  0.5578
## F-statistic: 3.365 on 8 and 7 DF,  p-value: 0.06368
```

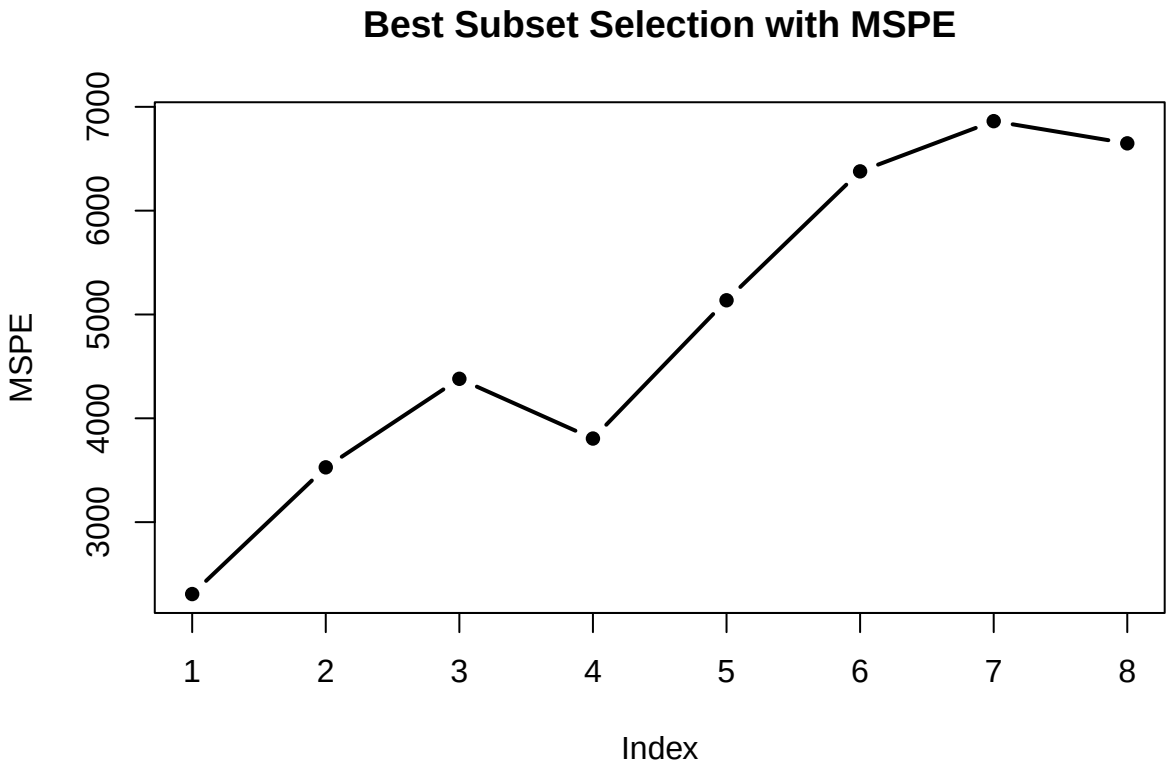
As theory suggests, the SSE of a linear model will always decrease (or stay the same) as we add more predictors, therefore this result agrees with our intuition. We will now conduct Best Subset Selection with MPSE to derive more conclusive results.

Use the mean squared prediction error on the validation set as a model selection criterion.

```
for (p in 1:P){
  fit = lm(seatpos_df$hipcenter ~ ., predictors[,summary(sub)$which[p,-1], drop=FALSE],final_train)
  predictions = predict(fit,predictors[validation,summary(sub)$which[p,-1],drop=FALSE])
  MSPE[p] = mean((seatpos$hipcenter[validation] - predictions)^2)
}
# Which variable
p = which.min(MSPE)
summary(sub)$which[p,-1]
```

```
##      Age  Weight HtShoes      Ht  Seated     Arm   Thigh     Leg
##    FALSE   FALSE   FALSE   FALSE   FALSE   FALSE   FALSE    TRUE
```

```
# Plot
plot(MSPE,type = 'b', pch = 16, lwd = 2, ylab = 'MSPE', main = 'Best Subset Selection with MSPE')
```



Fit the best linear model on the test set and provide a summary of its results.

```
fit = lm(seatpos_df$hipcenter ~ ., predictors[,summary(sub)$which[p,-1], drop=FALSE],test)
summary(fit)
```

```
##
## Call:
## lm(formula = seatpos_df$hipcenter ~ ., data = predictors[, summary(sub)$which[p,
##     -1], drop = FALSE], subset = test)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -41.918 -27.321   1.834  14.050  54.646
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  310.320     77.991   3.979  0.00137 **
## Leg          -12.936      2.099  -6.162 2.47e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 31.91 on 14 degrees of freedom
## Multiple R-squared:  0.7306, Adjusted R-squared:  0.7114
## F-statistic: 37.97 on 1 and 14 DF,  p-value: 2.467e-05
```

Based on Best Subset Selection with MSPE, we choose the best model with 1 predictor as the MPSE with one predictor is the lowest. The specific model is the model with Leg as the predictor, and hipcenter as the response.

vi. Apply 5-fold cross-validation on the larger training set to obtain the optimal lambda value in ridge regression. Fit the optimal ridge regression model on the test set.

vii. Repeat the previous part for the lasso regression method.

```
library(glmnet)
```

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

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

```
## The following object is masked from 'package:SparseM':
##
##     det
```

```
## Loaded glmnet 4.1-8
```

```
# Setting seed for reproducibility
set.seed(123)

# Prepare the data
x_train <- as.matrix(seatpos[train, c('Age', 'Weight', 'Ht', 'HtShoes', 'Arm', 'Seated', 'Thigh', 'Leg'
y_train <- seatpos$hipcenter[train]

# Perform 5-fold cross-validation for Ridge Regression on the training set
cv.ridge <- cv.glmnet(x_train, y_train, alpha = 0, nfolds = 5)

# Perform 5-fold cross-validation for Lasso Regression on the training set
cv.lasso <- cv.glmnet(x_train, y_train, alpha = 1, nfolds=5)

# Determine the optimal lambda Ridge
optimal_lambda_ridge <- cv.ridge$lambda.min

# Determine the optimal lambda Lasso

optimal_lambda_lasso <- cv.lasso$lambda.min
```

```r
# Prepare test data
x_test <- as.matrix(seatpos[test, c('Age', 'Weight', 'Ht', 'HtShoes', 'Arm', 'Seated', 'Thigh', 'Leg')]
y_test <- seatpos$hipcenter[test]

# Fit the optimal ridge regression model on the test set
model_ridge_test <- glmnet(x_test, y_test, alpha = 0, lambda = optimal_lambda_ridge)

model_lasso_test <- glmnet(x_test,y_test, alpha = 1, lambda = optimal_lambda_lasso)
# Print the coefficients of the optimal ridge model
print(coef(model_ridge_test))
```

```
## 9 x 1 sparse Matrix of class "dgCMatrix"
##                       s0
## (Intercept) 309.7106404
## Age           0.1782699
## Weight       -0.1438969
## Ht           -0.5102578
## HtShoes      -0.5084568
## Arm          -1.5908786
## Seated       -0.9197823
## Thigh        -1.9758242
## Leg          -2.0732230
```

```r
# Print the coefficients of the optimal Lasso model
print(coef(model_lasso_test))
```

```
## 9 x 1 sparse Matrix of class "dgCMatrix"
##                       s0
## (Intercept) 272.2313957
## Age           .
## Weight       -0.1706706
## Ht           -0.5482486
## HtShoes       .
## Arm           .
## Seated        .
## Thigh        -2.6844389
## Leg          -5.8102834
```

What can you observe about the estimated ridge regression coefficients? Are your observations consistent with the collinearity results on the seatpos data set we obtained in the previous problem set?

The estimated ridge regression coefficients are extremely different from the regression coefficients when fitting a full model without any regularization techniques. The observations are consistent with the collinearity results we saw earlier because the coefficients for Ht and HtShoes were positive and negative, and with ridge regression both coefficients became negative.

HtShoes was -2.9596, Ht was 5.0455 with OLS.

HtShoes is -0.5084568, Ht is -0.5102578.

The estimated lasso regression coefficients show why lasso is widely used in real world analysis. Lasso acts as a predictor selection criterion as well, filtering out variables that exhibit collinearity and are not neeeded for the model based on its regularization techniques.

Apply the bootstrap method on the test set to obtain estimates for the standard errors of the estimated ridge regression coefficients.

vii. Repeat the previous part for the lasso regression method.

```
set.seed(123)  # For reproducibility

# Set the number of bootstrap iterations
B <- 10000

# Use test set for bootstrapping
n <- dim(seatpos[test, ])[1]

# Initialize matrices to store coefficients for Ridge and Lasso
coef_ridge_bootstrap <- matrix(0, nrow = B, ncol = length(coef(model_ridge_test)))  # Store ridge coeff
coef_lasso_bootstrap <- matrix(0, nrow = B, ncol = length(coef(model_lasso_test)))  # Store lasso coeff

# Bootstrap loop
for (k in 1:B) {
  # Generate a bootstrap sample from the test set
  boot_sample <- sample(1:n, replace = TRUE)

  # Bootstrap data for the test set
  x_boot <- x_test[boot_sample, ]
  y_boot <- y_test[boot_sample]

  # Fit a ridge regression model on the bootstrapped sample using the optimal lambda
  boot_ridge <- glmnet(x_boot, y_boot, alpha = 0, lambda = optimal_lambda_ridge)
  coef_ridge_bootstrap[k, ] <- as.vector(coef(boot_ridge))  # Store coefficients

  # Fit a lasso regression model on the bootstrapped sample using the optimal lambda
  boot_lasso <- glmnet(x_boot, y_boot, alpha = 1, lambda = optimal_lambda_lasso)
  coef_lasso_bootstrap[k, ] <- as.vector(coef(boot_lasso))  # Store coefficients
}

# Calculate standard errors of the bootstrapped coefficients
ridge_se <- apply(coef_ridge_bootstrap, 2, sd)
lasso_se <- apply(coef_lasso_bootstrap, 2, sd)
```

```
variables <- c("(Intercept)", "Age", "Weight", "Ht", "HtShoes", "Arm", "Seated", "Thigh", "Leg")

# Print the standard errors for Ridge Regression with variable names
print("Standard Errors for Ridge Regression:")
```

```
## [1] "Standard Errors for Ridge Regression:"
```

```
ridge_se_named <- setNames(ridge_se, variables)
print(ridge_se_named)
```

```
##  (Intercept)          Age       Weight           Ht      HtShoes          Arm
## 108.23982703   0.19723294   0.04723338   0.12599765   0.12990565   0.61600196
##       Seated        Thigh          Leg
##   0.47876722   0.73266868   0.61727902
```

```
# Print the standard errors for Lasso Regression with variable names
print("Standard Errors for Lasso Regression:")
```

```
## [1] "Standard Errors for Lasso Regression:"
```

```
lasso_se_named <- setNames(lasso_se, variables)
print(lasso_se_named)
```

```
## (Intercept)         Age      Weight          Ht     HtShoes         Arm
## 153.8653135   0.1634259   0.1886768   0.5839520   0.4735273   1.6126023
##      Seated       Thigh         Leg
##   1.8987122   2.1737448   4.0990034
```

```
full_ols_se <- summary(full_model)$coefficients[, "Std. Error"]

# Print the standard errors for Lasso Regression with variable names
print("Standard Errors for OLS:")
```

```
## [1] "Standard Errors for OLS:"
```

```
full_named <- setNames(full_ols_se, variables)
print(full_named)
```

```
## (Intercept)         Age      Weight          Ht     HtShoes         Arm
## 356.9121234   1.0294639   0.8962364  13.9834265  14.4995408   7.9725972
##      Seated       Thigh         Leg
##   7.1641507   4.1974667   8.0968464
```

How do they compare against the estimated standard errors of the ordinary linear regression coefficients from part i?

The estimated standard errors from Ridge and Lasso regression are much smaller than the estimated standard errors of the OLS model. This is due to the regularization techniques used by Ridge and Lasso, which help to reduce the effects of multicollinearity.

Ridge performs well by shrinking coefficients and maintaining all variables, leading to lower variability in the coefficients.

Lasso, on the other hand, performs variable selection by setting some coefficients to zero. The remaining coefficients may have slightly higher variability, but overall, regularization reduces the standard errors compared to OLS.

Ridge is likely the better method in this case because it retains all variables and reduces the effect of multicollinearity, whereas Lasso removes some variables entirely, potentially leading to higher variability for the remaining coefficients.