Homework 5

Jared Andreatta

2025-04-14

Problem 1

a.

The best subset selection method will have the lowest RSS. Since this method is essentially minimizing RSS over a set of combinations for all k predictors, then it will "find" the model that achieves this. The stepwise selection methods are not guaranteed to select the globally optimal model since they only optimize over a subset of predictors. However, this does not mean that it is impossible for these methods to select the best model.

b.

This is not possible to say without more information. It is likely that the best subset selection method will minimize test RSS, however, this method can be prone to overfitting, especially for a large amount of predictors. So it is largely possible that the subset selection methods could minimize test RSS relative to all other models.

c.

(i) True
(ii) True
(iii) False
(iv) False
(v) False

Problem 8

Chosen params:

$$\beta_0 = 3$$
$$\beta_1 = 2$$
$$\beta_2 = \pi$$
$$\beta_3 = e$$

```r
library(leaps)
```

```
## Warning: package 'leaps' was built under R version 4.4.3
```

```r
# a.
set.seed(1234)

X <- rnorm(100)
eps <- rnorm(100)

# b.
Y <- 3 + 2*X + pi * X^2 + exp(1) * X^3 + eps

# c.
### Adj R^2 chooses the 4 variable models, while the BIC and Cp method choose the 3 variable model. How
data <- data.frame(X,Y)
regfit <- regsubsets(Y~poly(X,10), data=data)

regfit.summary <- summary(regfit)
names(regfit.summary)
```

```
## [1] "which"  "rsq"    "rss"    "adjr2"  "cp"     "bic"    "outmat" "obj"
```

```r
which.max(regfit.summary$adjr2) # Adj R^2
```

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

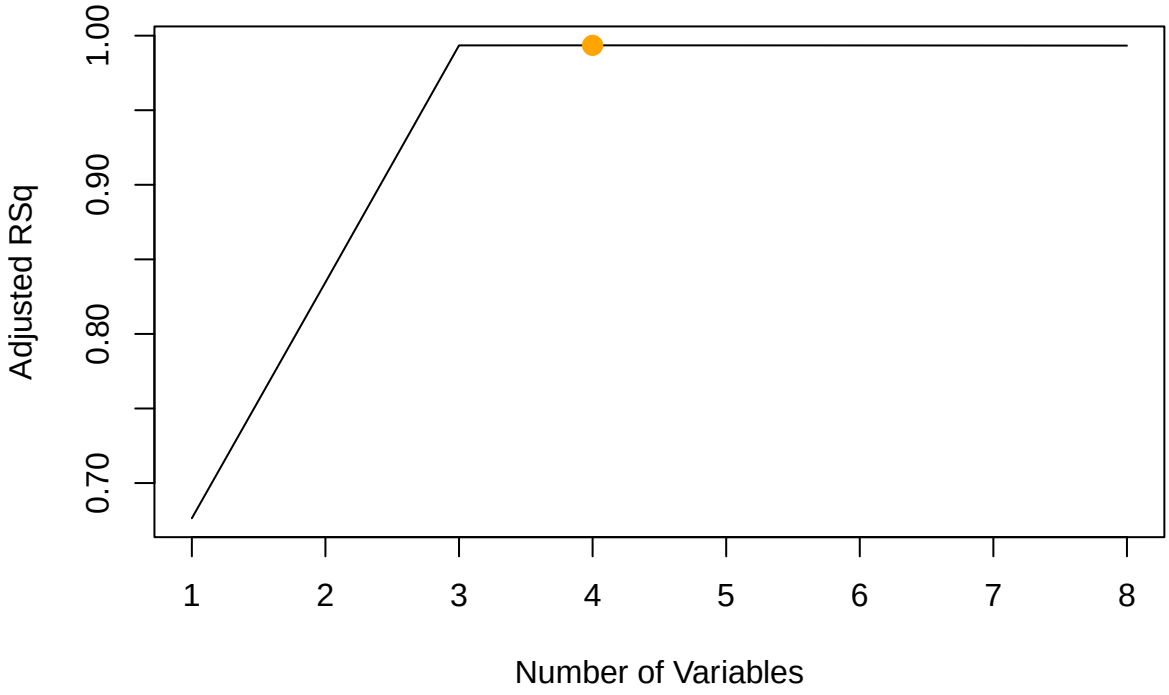
```r
which.min(regfit.summary$cp) # C_p
```

```
## [1] 3
```

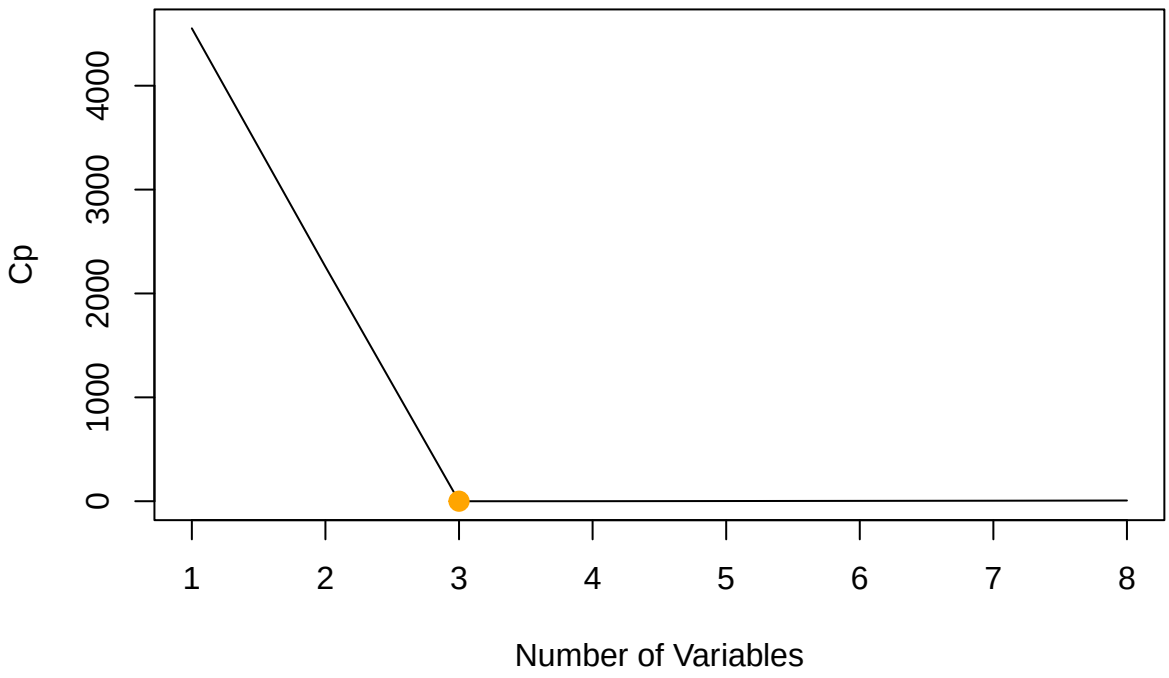
```r
which.min(regfit.summary$bic) # BIC
```

```
## [1] 3
```

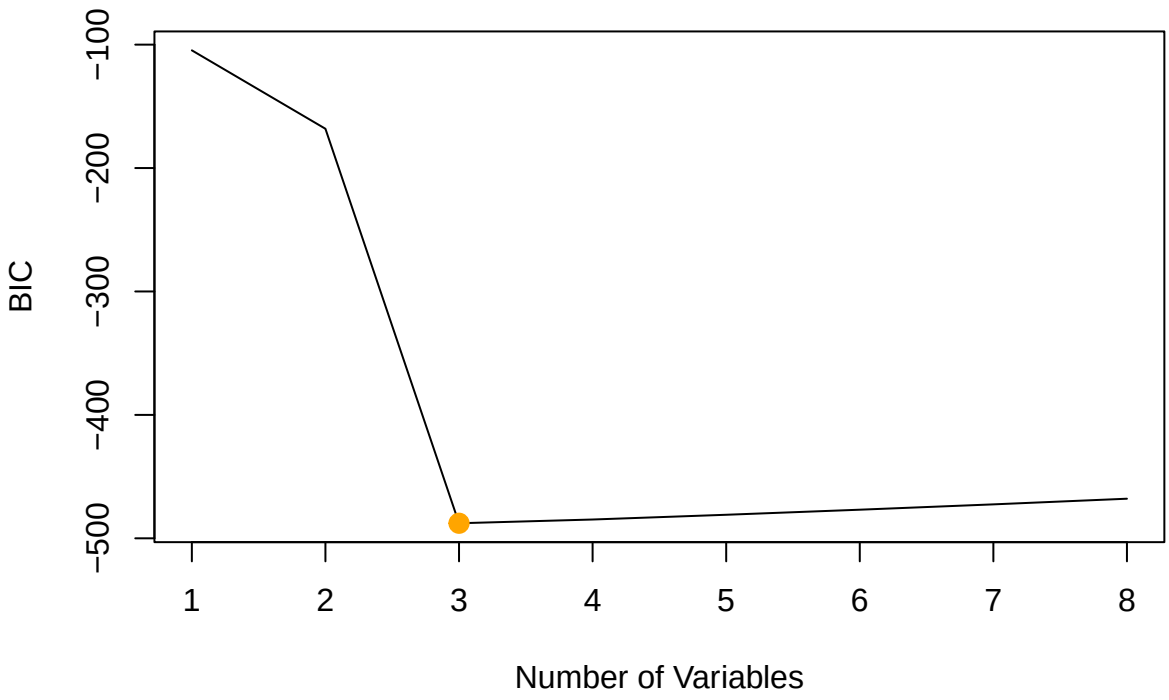
```r
plot(regfit.summary$adjr2, xlab = "Number of Variables", ylab = "Adjusted RSq", type = "l") # Plotting
points(4, regfit.summary$adjr2[4], col = "orange", cex = 2, pch = 20) # Plotting point on line
```




```
plot(regfit.summary$cp, xlab = "Number of Variables", ylab = "Cp", type = "l")
points(3, regfit.summary$cp[4], col = "orange", cex = 2, pch = 20)
```




```
plot(regfit.summary$bic, xlab = "Number of Variables", ylab = "BIC", type = "l")
points(3, regfit.summary$bic[3], col = "orange", cex = 2, pch = 20)
```



d.

The answers are mostly identical with the results obtained in part (c). Adj R^2 chooses 4 variables again, while the other two methods choose 3 variables again.

```
### BACKWARDS ###
regfit.back <- regsubsets(Y~poly(X,10), data=data, method="backward")

back.summary <- summary(regfit.back)
names(back.summary)
```

```
## [1] "which"  "rsq"    "rss"    "adjr2"  "cp"     "bic"    "outmat" "obj"
```

```
which.max(back.summary$adjr2) # Adj R^2
```

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

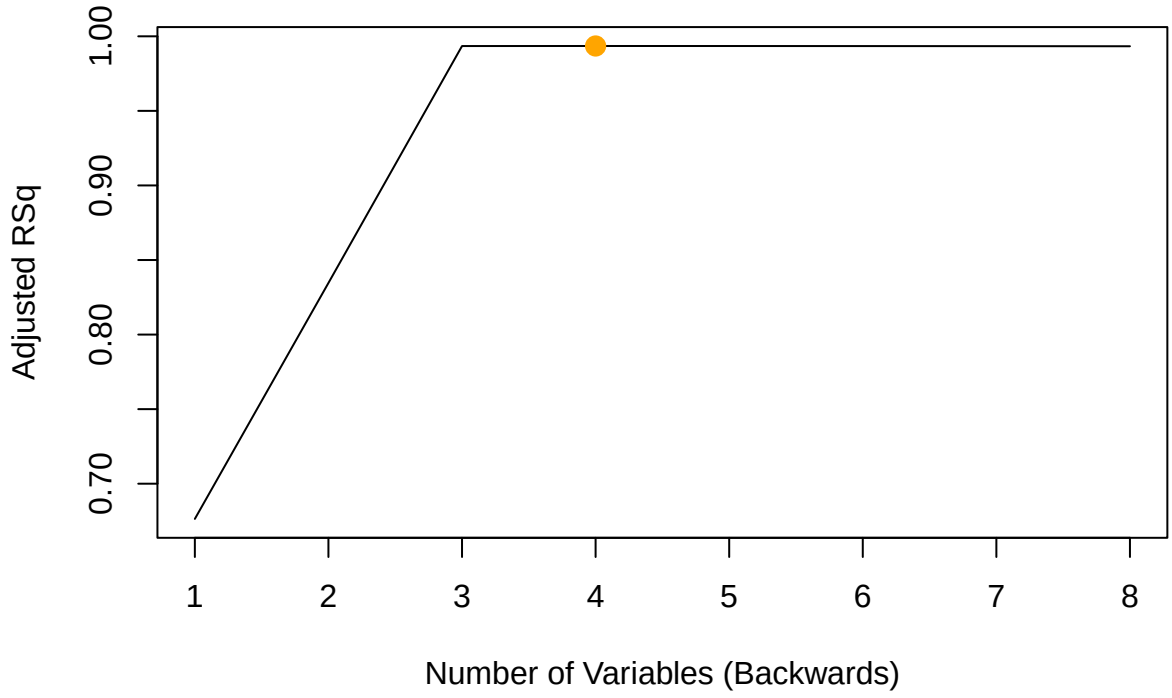
```
which.min(back.summary$cp) # C_p
```

```
## [1] 3
```

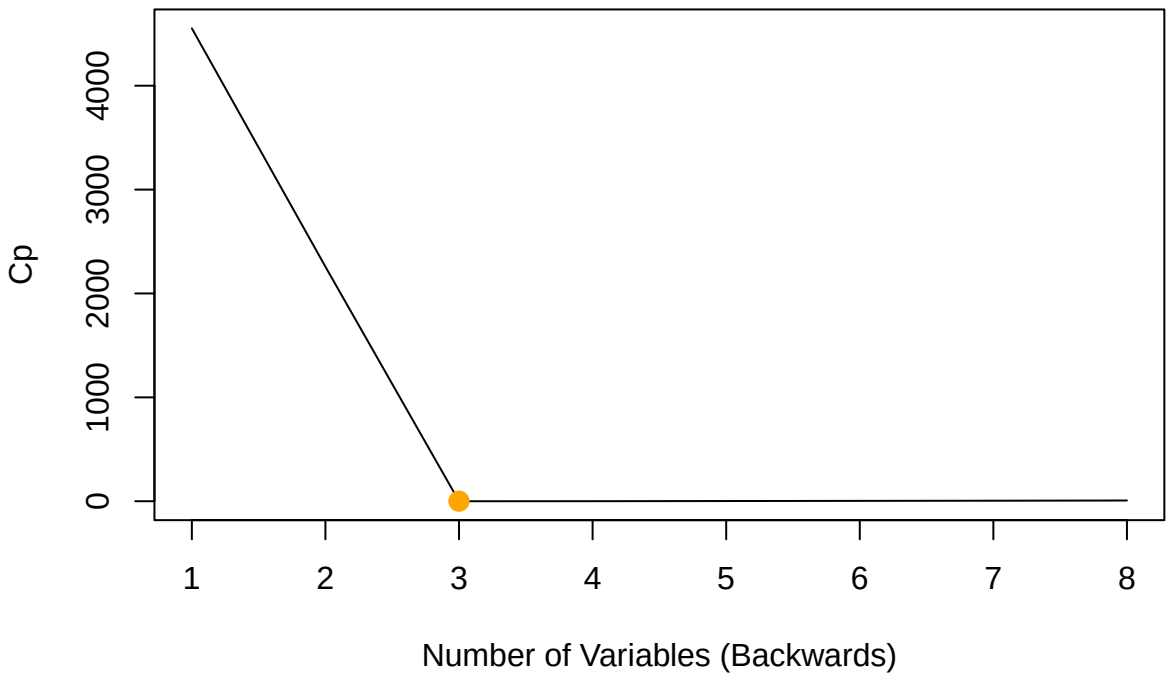
```
which.min(back.summary$bic) # BIC
```

```
## [1] 3
```

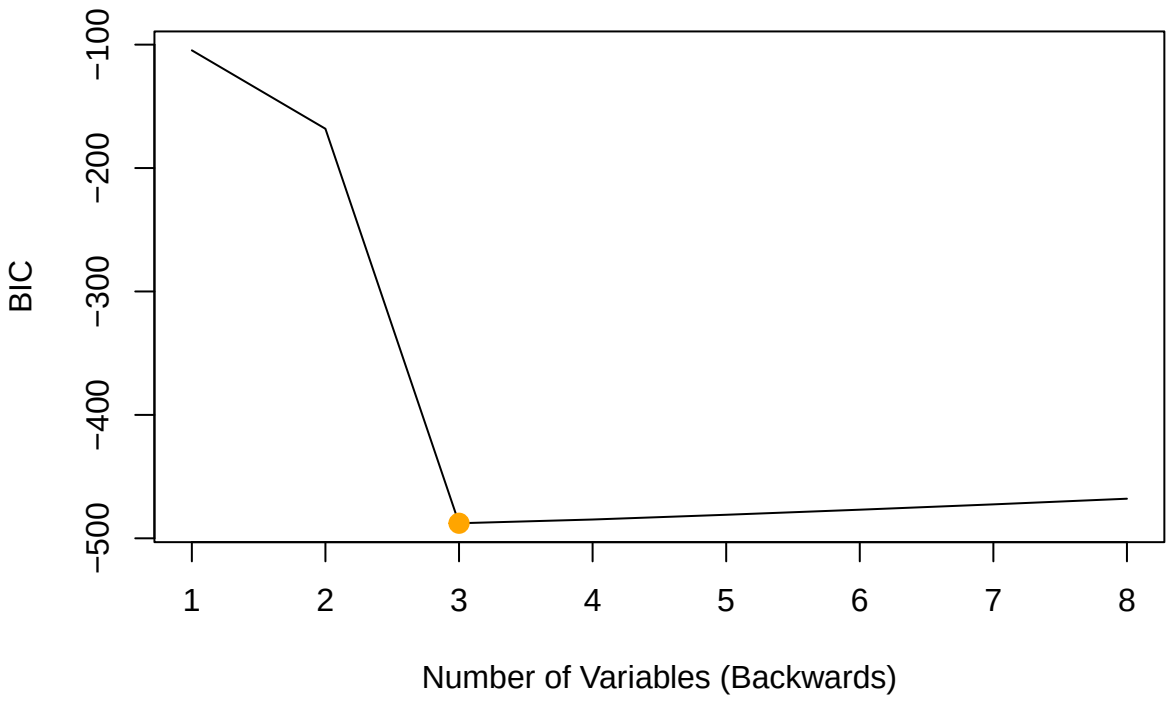
```r
plot(back.summary$adjr2, xlab = "Number of Variables (Backwards)", ylab = "Adjusted RSq", type = "l") #
points(4, back.summary$adjr2[4], col = "orange", cex = 2, pch = 20) # Plotting point on line
```




```r
plot(back.summary$cp, xlab = "Number of Variables (Backwards)", ylab = "Cp", type = "l")
points(3, back.summary$cp[4], col = "orange", cex = 2, pch = 20)
```




```
plot(back.summary$bic, xlab = "Number of Variables (Backwards)", ylab = "BIC", type = "l")
points(3, back.summary$bic[3], col = "orange", cex = 2, pch = 20)
```




```
### FORWARDS ###
regfit.forward <- regsubsets(Y~poly(X,10), data=data, method="forward")

forward.summary <- summary(regfit.forward)
names(forward.summary)
```

```
## [1] "which"  "rsq"    "rss"    "adjr2"  "cp"     "bic"    "outmat" "obj"
```

```
which.max(forward.summary$adjr2) # Adj R^2
```

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

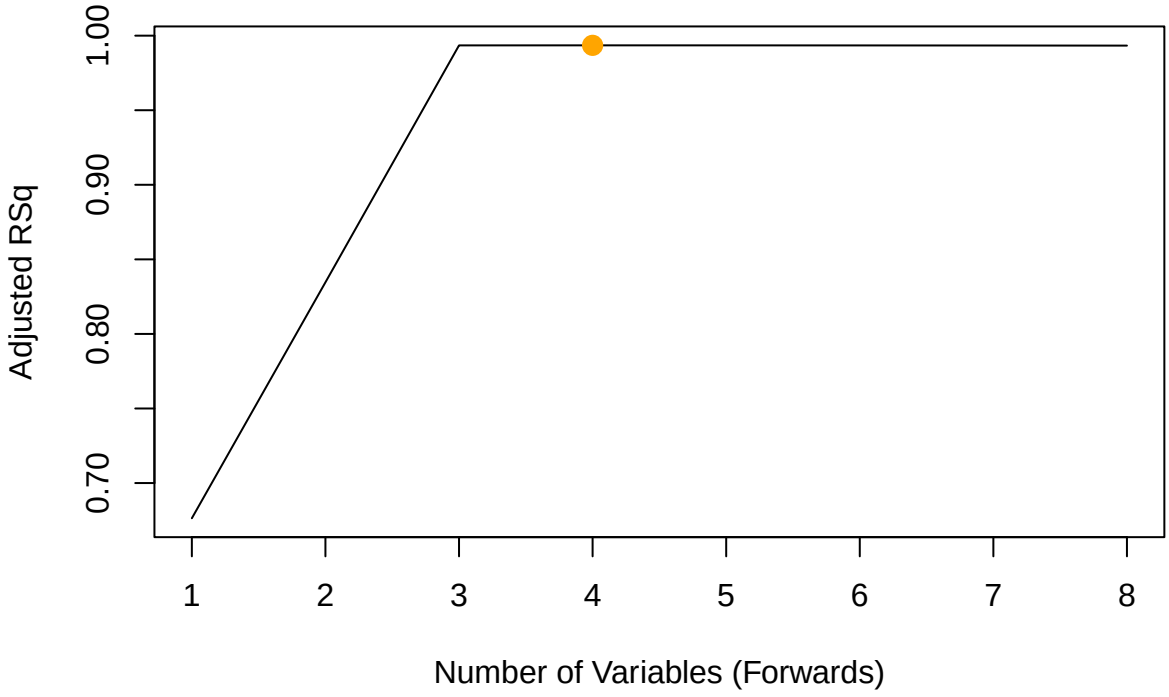
```
which.min(forward.summary$cp) # C_p
```

```
## [1] 3
```

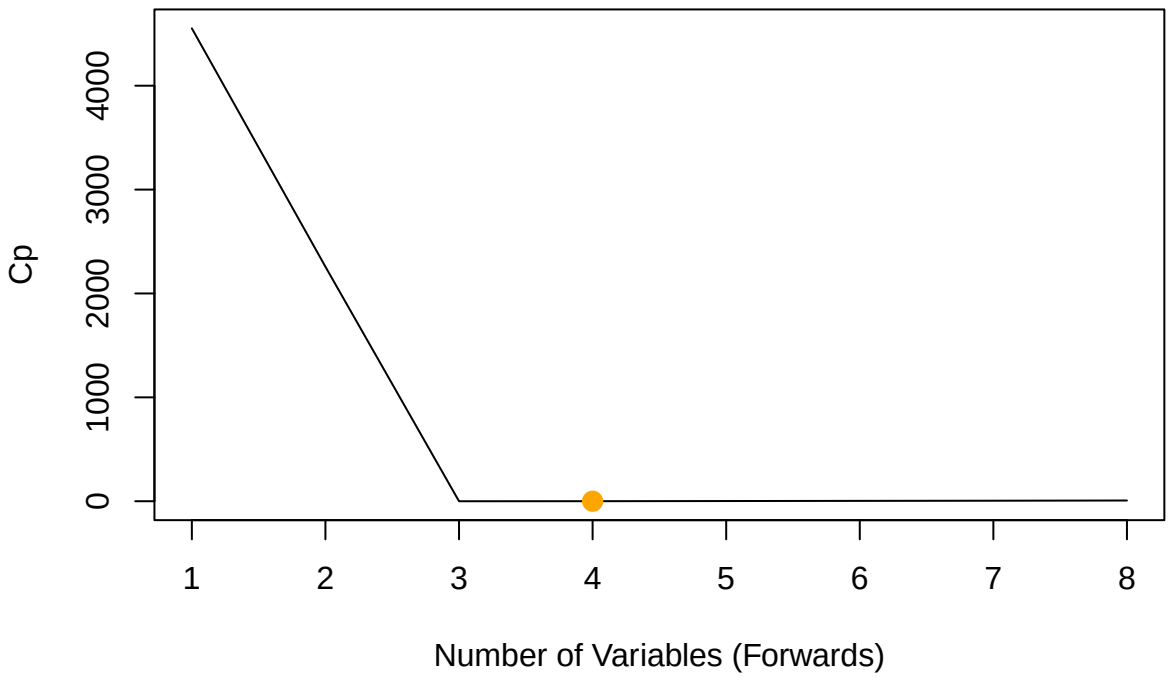
```
which.min(forward.summary$bic) # BIC
```

```
## [1] 3
```

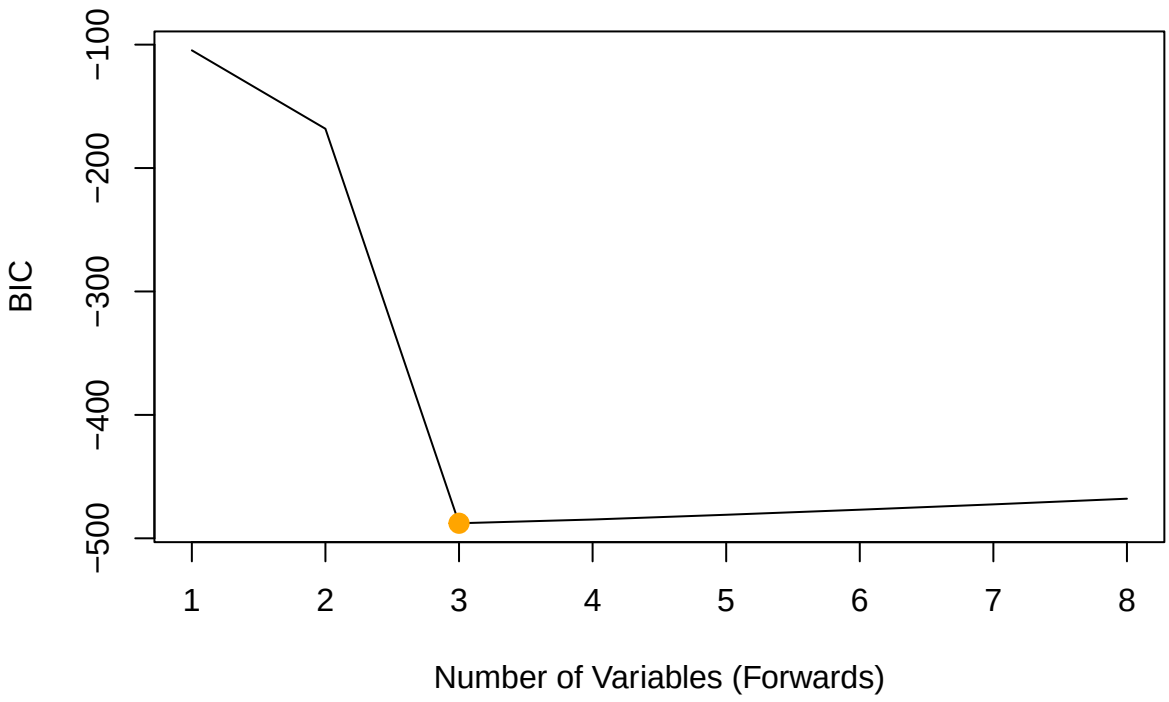
```
plot(forward.summary$adjr2, xlab = "Number of Variables (Forwards)", ylab = "Adjusted RSq", type = "l")
points(4, forward.summary$adjr2[4], col = "orange", cex = 2, pch = 20) # Plotting point on line
```




```
plot(forward.summary$cp, xlab = "Number of Variables (Forwards)", ylab = "Cp", type = "l")
points(4, forward.summary$cp[4], col = "orange", cex = 2, pch = 20)
```




```
plot(forward.summary$bic, xlab = "Number of Variables (Forwards)", ylab = "BIC", type = "l")
points(3, forward.summary$bic[3], col = "orange", cex = 2, pch = 20)
```



Problem 10

```r
# a.
set.seed(32)

p <- 20
n <- 1000

X <- matrix(data = rnorm(n * p, mean = 0, sd = 10), nrow = n, ncol = p)
beta <- runif(p, min=-5, max=5)
beta[c(3,12,19)] <- 0 # Fixing 3 of the beta elements to be 0
eps <- rnorm(1000, mean=0, sd=5)

Y <- X %*% beta + eps


# b.
train <- sample(1:n, size = 100, replace = FALSE)
train_X <- X[train,]
train_eps <- eps[train]
train_Y <- Y[train]
train_data <- data.frame(Y = train_Y, train_X)
```

```
test <- setdiff(1:n, train)
test_X <- X[test, ]
test_eps <- eps[test]
test_Y <- Y[test]
test_data <- data.frame(Y = test_Y, test_X)
```

c.

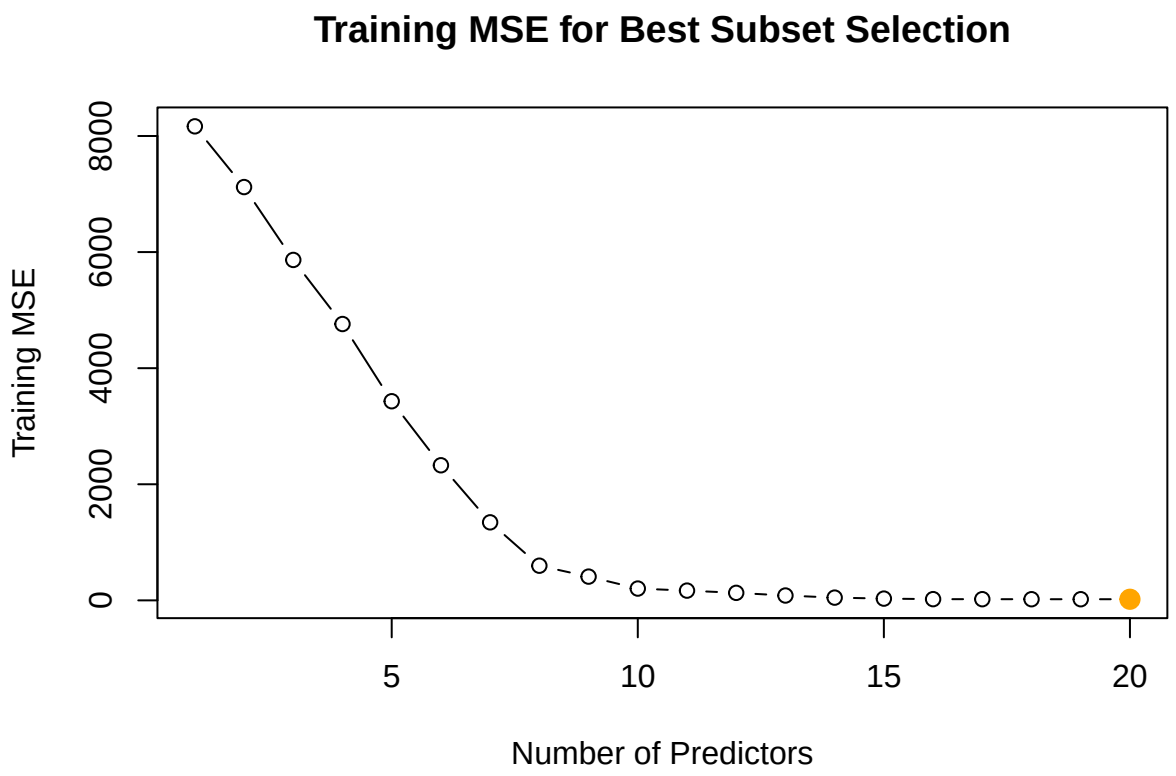
The train MSE is minimized when we use the model with 20 predictors, which is a consistent result.

```
# c.
regfit.best <- regsubsets(Y~., data=train_data, nvmax=20)
regfit.summary <- summary(regfit.best)

# From lab 5
predict.regsubsets <- function(object, newdata, id, ...) {
  form <- as.formula(object$call[[2]])
  mat <- model.matrix(form, newdata)
  coef <- coef(object, id = id)
  mat[, names(coef)] %*% coef
}

# Calculating train mse vector
train_mse <- rep(NA, p)
for (i in 1:p) {
  pred <- predict.regsubsets(regfit.best, newdata = train_data, id = i)
  train_mse[i] <- mean((train_data$Y - pred)^2)
}

# Plotting
plot(1:p, train_mse, type = "b",
     xlab = "Number of Predictors",
     ylab = "Training MSE",
     main = "Training MSE for Best Subset Selection")
points(which.min(train_mse), train_mse[which.min(train_mse)], col = "orange", cex = 2, pch = 20)
```

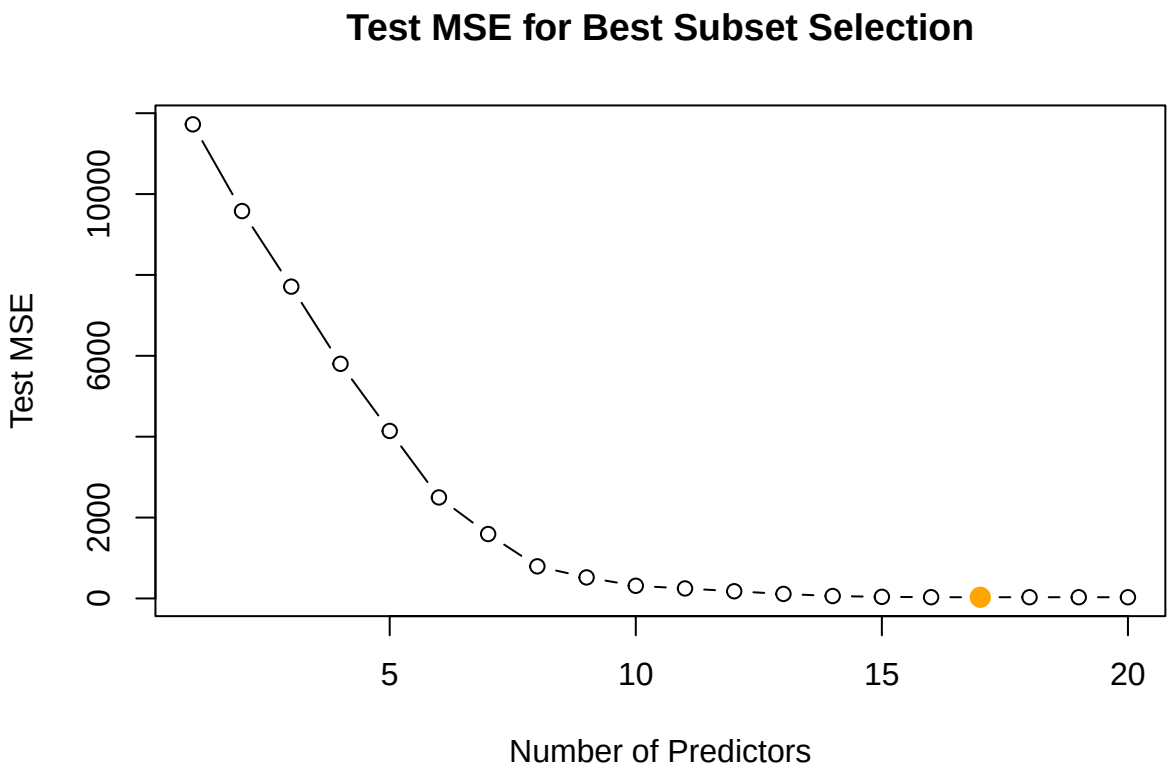


d. and e.

The test set MSE is minimized for the model with 17 predictors. This is consistent with my code above; I hard-coded 3 of the coefficients to be 0. The best subset selection model was able to effectively remove these redundant regressors and reduce the overfitting that came from fitting on all 18+ predictors, as indicated by the MSE plots.

```r
# Calculating test mse vector
test_mse <- rep(NA, p)
for (i in 1:p) {
  pred <- predict.regsubsets(regfit.best, newdata = test_data, id = i)
  test_mse[i] <- mean((test_data$Y - pred)^2)
}

# Plotting
plot(1:p, test_mse, type = "b",
     xlab = "Number of Predictors",
     ylab = "Test MSE",
     main = "Test MSE for Best Subset Selection")
points(which.min(test_mse), test_mse[which.min(test_mse)], col = "orange", cex = 2, pch = 20)
```



f.

The coefficient estimates all approximate the true beta values well. Additionally, the subset selection was able to filter out the three 0 coefficients, namely β_3, β_{12} and β_{19}.

```
coef(regfit.best, 17)
```

```
## (Intercept)          X1          X2          X4          X5          X6
##   0.3846988   4.3041194  -4.1475984  -0.1077145   2.7457397  -0.9113538
##          X7          X8          X9         X10         X11         X13
##   3.9055127   4.1186867  -0.3518370   0.8416197   0.4799948  -0.5974331
##         X14         X15         X16         X17         X18         X20
##  -1.6790210  -3.1093631   1.6260025  -0.7188440  -4.4329110   2.6975236
```

```
beta
```

```
##  [1]  4.3815629 -4.1784648  0.0000000 -0.1180469  2.7834849 -0.8447497
##  [7]  3.9662903  4.1787763 -0.2721908  0.8510338  0.4454490  0.0000000
## [13] -0.6212886 -1.6984895 -3.1751743  1.6017552 -0.7364440 -4.4674840
## [19]  0.0000000  2.8126935
```

g.

```r
# Create a vector to store the sum of squared differences for each model size r
coef_error <- numeric(p)

names(beta) <- paste0("X", 1:p)

for (r in 1:p) {

  best_coef <- coef(regfit.best, id = r)

  est_beta <- rep(0, p)
  names(est_beta) <- paste0("X", 1:p)

  predictors <- names(best_coef)[-1]
  est_beta[predictors] <- best_coef[predictors]
  coef_error[r] <- sum((beta - est_beta)^2)
}

plot(1:p, coef_error, type = "b",
     xlab = "Number of Predictors (r)",
     ylab = "Parameter Differences",
     main = "Coefficient Error vs. Model Size")
```