HW5

Jon Griffith and Carla Ellefsen

2025-04-14

```
library(leaps)
```

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

```
library(ISLR2)
```

```
## Warning: package 'ISLR2' was built under R version 4.4.3
```

8.

(a)-(b)

```
set.seed(1)

# Generate data
x <- rnorm(100)
e <- rnorm(100)

Y <- 1 + 2*x + 3*x^2 + 4*x^3 + e

# Create dataframe and add names to variables in created data frame
xnames <- c('X1', 'X2', 'X3', 'X4', 'X5', 'X6', 'X7', 'X8', 'X9', 'X10', 'Y')
df <- cbind(x, x^2, x^3, x^4, x^5, x^6, x^7, x^8, x^9, x^10, Y)
colnames(df) <- xnames
df <- as.data.frame(df)
```

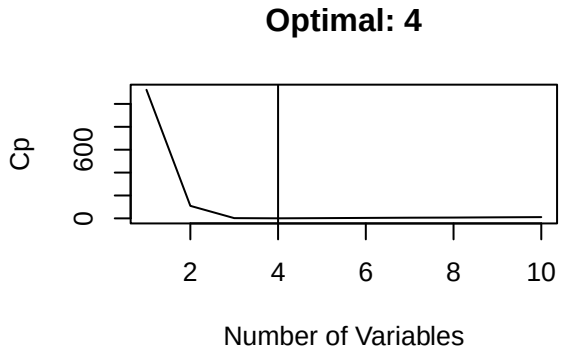
(c)

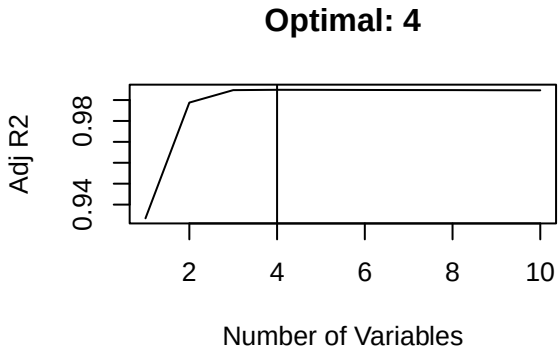
```
regfit.full <- regsubsets(Y ~ ., data=df, nvmax=10)
reg.summary <- summary(regfit.full)
reg.summary
```

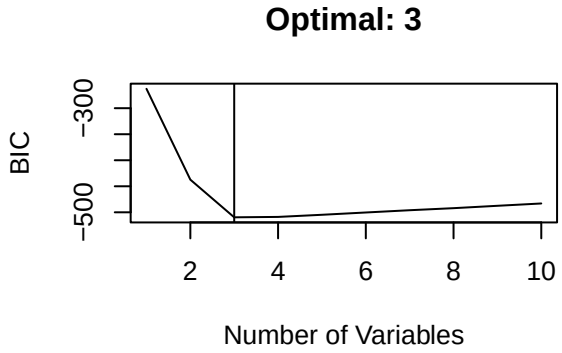
```
## Subset selection object
## Call: regsubsets.formula(Y ~ ., data = df, nvmax = 10)
## 10 Variables  (and intercept)
```

```
##     Forced in Forced out
## X1      FALSE      FALSE
## X2      FALSE      FALSE
## X3      FALSE      FALSE
## X4      FALSE      FALSE
## X5      FALSE      FALSE
## X6      FALSE      FALSE
## X7      FALSE      FALSE
## X8      FALSE      FALSE
## X9      FALSE      FALSE
## X10     FALSE      FALSE
## 1 subsets of each size up to 10
## Selection Algorithm: exhaustive
##           X1  X2  X3  X4  X5  X6  X7  X8  X9  X10
## 1  ( 1 )  " " " " "*" " " " " " " " " " " " " " "
## 2  ( 1 )  " " "*" "*" " " " " " " " " " " " " " "
## 3  ( 1 )  "*" "*" "*" " " " " " " " " " " " " " "
## 4  ( 1 )  "*" "*" "*" " " "*" " " " " " " " " " "
## 5  ( 1 )  "*" "*" "*" " " "*" "*" " " " " " " " "
## 6  ( 1 )  "*" "*" "*" " " " " " " " "*" "*" "*" " "
## 7  ( 1 )  "*" "*" "*" " " "*" "*" " " "*" " " "*"
## 8  ( 1 )  "*" "*" "*" "*" " " "*" " " "*" "*" "*"
## 9  ( 1 )  "*" "*" "*" "*" "*" "*" " " "*" "*" "*"
## 10  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" "*" "*"
```

```r
par(mfrow=c(2,2))
# PLot of cp vs num variables
plot(reg.summary$cp,
     xlab='Number of Variables',
     ylab='Cp',
     type='l',
     main=paste('Optimal:', which.min(reg.summary$cp)))
abline(v=which.min(reg.summary$cp))

# PLot of adjr2 vs num variables
plot(reg.summary$adjr2,
     xlab='Number of Variables',
     ylab='Adj R2',
     type='l',
     main=paste('Optimal:', which.max(reg.summary$adjr2)))
abline(v=which.max(reg.summary$adjr2))

# PLot of bic vs num variables
plot(reg.summary$bic,
     xlab='Number of Variables',
     ylab='BIC',
     type='l',
     main=paste('Optimal:', which.min(reg.summary$bic)))
abline(v=which.min(reg.summary$bic))
```







We see from the above plots that the optimal number of variables based on each respective measure is 4, 4, and 3 for Cp, Adj R2, and BIC, respectively. We know that BIC favors fewer variables so it is no surprise that this is the one that recommends the fewest. We show the coefficients for model with three and 4 variables below.

```
coef(regfit.full, 3)
```

```
## (Intercept)          X1          X2          X3
##    1.061507    1.975280    2.876209    4.017639
```

```
coef(regfit.full, 4)
```

```
## (Intercept)          X1          X2          X3          X5
##  1.07200775  2.38745596  2.84575641  3.55797426  0.08072292
```

Both models chose to include the same predictor variables as the true function. We can see that the three variable model is pretty close to our true function coefficients. The four variable model is also close, but because of the fourth added variable, 'X5', it does not come as close to the true coefficients for the other three.

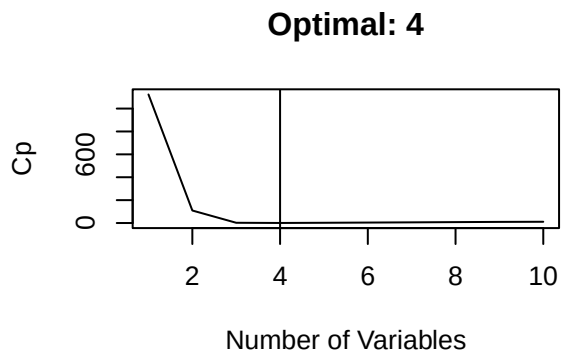
(d)

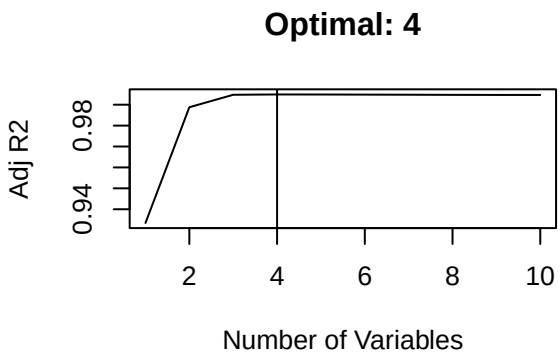
```r
regfit.fwd <- regsubsets(Y ~ ., data=df, nvmax=10, method='forward')
reg.summary <- summary(regfit.fwd)
reg.summary
```

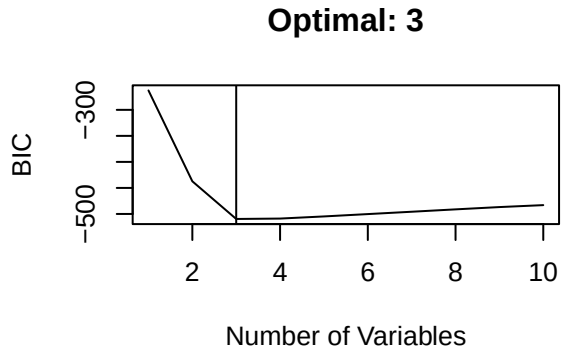
```
## Subset selection object
## Call: regsubsets.formula(Y ~ ., data = df, nvmax = 10, method = "forward")
## 10 Variables  (and intercept)
##     Forced in Forced out
## X1      FALSE      FALSE
## X2      FALSE      FALSE
## X3      FALSE      FALSE
## X4      FALSE      FALSE
## X5      FALSE      FALSE
## X6      FALSE      FALSE
## X7      FALSE      FALSE
## X8      FALSE      FALSE
## X9      FALSE      FALSE
## X10     FALSE      FALSE
## 1 subsets of each size up to 10
## Selection Algorithm: forward
##           X1  X2  X3  X4  X5  X6  X7  X8  X9  X10
## 1  ( 1 )  " " " " "*" " " " " " " " " " " " " " "
## 2  ( 1 )  " " "*" "*" " " " " " " " " " " " " " "
## 3  ( 1 )  "*" "*" "*" " " " " " " " " " " " " " "
## 4  ( 1 )  "*" "*" "*" " " "*" " " " " " " " " " "
## 5  ( 1 )  "*" "*" "*" " " "*" "*" " " " " " " " "
## 6  ( 1 )  "*" "*" "*" " " "*" "*" " " " " "*" " "
## 7  ( 1 )  "*" "*" "*" " " "*" "*" "*" " " "*" " "
## 8  ( 1 )  "*" "*" "*" " " "*" "*" "*" "*" "*" " "
## 9  ( 1 )  "*" "*" "*" " " "*" "*" "*" "*" "*" "*"
## 10  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" "*" "*"
```

```r
par(mfrow=c(2,2))
# PLot of cp vs num variables
plot(reg.summary$cp,
     xlab='Number of Variables',
     ylab='Cp',
     type='l',
     main=paste('Optimal:', which.min(reg.summary$cp)))
abline(v=which.min(reg.summary$cp))

# PLot of adjr2 vs num variables
plot(reg.summary$adjr2,
     xlab='Number of Variables',
     ylab='Adj R2',
     type='l',
     main=paste('Optimal:', which.max(reg.summary$adjr2)))
abline(v=which.max(reg.summary$adjr2))

# PLot of bic vs num variables
plot(reg.summary$bic,
     xlab='Number of Variables',
```

```
     ylab='BIC',
     type='l',
     main=paste('Optimal:', which.min(reg.summary$bic)))
abline(v=which.min(reg.summary$bic))
```





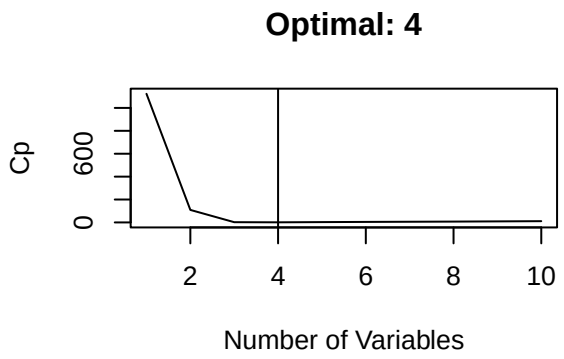


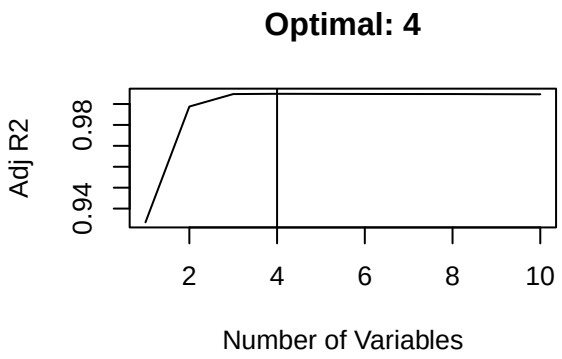

```
regfit.bwd <- regsubsets(Y ~ ., data=df, nvmax=10, method='backward')
reg.summary <- summary(regfit.bwd)
reg.summary
```

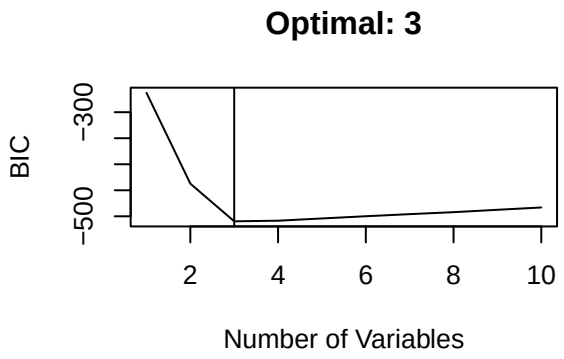
```
## Subset selection object
## Call: regsubsets.formula(Y ~ ., data = df, nvmax = 10, method = "backward")
## 10 Variables  (and intercept)
##     Forced in Forced out
## X1      FALSE      FALSE
## X2      FALSE      FALSE
## X3      FALSE      FALSE
## X4      FALSE      FALSE
## X5      FALSE      FALSE
## X6      FALSE      FALSE
## X7      FALSE      FALSE
## X8      FALSE      FALSE
## X9      FALSE      FALSE
## X10     FALSE      FALSE
## 1 subsets of each size up to 10
## Selection Algorithm: backward
```

```
##           X1  X2  X3  X4  X5  X6  X7  X8  X9  X10
## 1  ( 1 )  " " " " "*" " " " " " " " " " " " " " "
## 2  ( 1 )  " " "*" "*" " " " " " " " " " " " " " "
## 3  ( 1 )  "*" "*" "*" " " " " " " " " " " " " " "
## 4  ( 1 )  "*" "*" "*" " " " " " " " " " " "*" " "
## 5  ( 1 )  "*" "*" "*" " " " " " " " " " "*" "*" " "
## 6  ( 1 )  "*" "*" "*" " " " " " " " " "*" "*" "*"
## 7  ( 1 )  "*" "*" "*" " " " " " " "*" " " "*" "*" "*"
## 8  ( 1 )  "*" "*" "*" "*" " " "*" " " "*" "*" "*"
## 9  ( 1 )  "*" "*" "*" "*" "*" "*" " " "*" "*" "*"
## 10  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" "*" "*"
```

```r
par(mfrow=c(2,2))
# PLot of cp vs num variables
plot(reg.summary$cp,
     xlab='Number of Variables',
     ylab='Cp',
     type='l',
     main=paste('Optimal:', which.min(reg.summary$cp)))
abline(v=which.min(reg.summary$cp))

# PLot of adjr2 vs num variables
plot(reg.summary$adjr2,
     xlab='Number of Variables',
     ylab='Adj R2',
     type='l',
     main=paste('Optimal:', which.max(reg.summary$adjr2)))
abline(v=which.max(reg.summary$adjr2))

# PLot of bic vs num variables
plot(reg.summary$bic,
     xlab='Number of Variables',
     ylab='BIC',
     type='l',
     main=paste('Optimal:', which.min(reg.summary$bic)))
abline(v=which.min(reg.summary$bic))
```








```
paste('Forward Stepwise Regression')
```

```
## [1] "Forward Stepwise Regression"
```

```
coef(regfit.fwd, 3)
```

```
## (Intercept)          X1          X2          X3
##    1.061507    1.975280    2.876209    4.017639
```

```
coef(regfit.fwd, 4)
```

```
## (Intercept)          X1          X2          X3          X5
##  1.07200775  2.38745596  2.84575641  3.55797426  0.08072292
```

```
cat('\n')
```

```
paste('Backward Stepwise Regression')
```

```
## [1] "Backward Stepwise Regression"
```

```
coef(regfit.bwd, 3)
```

```
## (Intercept)          X1          X2          X3
##    1.061507    1.975280    2.876209    4.017639
```

```
coef(regfit.bwd, 4)
```

```
## (Intercept)          X1          X2          X3          X9
## 1.079236362 2.231905828 2.833494180 3.819555807 0.001290827
```

We see that for both forward and backward stepwise regression, we also get the same optimal variable outputs of 4, 4, and 3 for AIC, Adj R2, and BIC, respectively. The three variable models for each all select X1, X2, and X3 which reflect the same variables for the true function and each three variable model obviously have the same coefficient values (since they all use OLS) which approximate the true coefficients pretty well.

Where the models depart from one another is for the four variable models, the forward selection and best subset selection chose the same fourth variable, X5, but the backward selection chose 'X9' as its fourth variable.

10

(a)

```
set.seed(1)
p <- 20
n <- 1000
varnames <- c()
bnames <- c('B0')

X <- c()
betaT <- c(3)

for (i in 1:p){
  varnames <- append(varnames, paste0('X',i))

  X <- cbind(X, rnorm(n, 0, 1/i))

  if (i %% 2 == 0){
    bnames <- append(bnames, paste0('X',i))
    betaT <- cbind(betaT, runif(1,-5,5))
#    Y <- cbind(Y, X[,i])
  }
  else {
    bnames <- append(bnames, paste0('X',i))
    betaT <- cbind(betaT, 0)
  }
}

colnames(X) <- varnames
```

```r
colnames(betaT) <- bnames

X <- cbind(1, X)

Y <- X%*%t(betaT) + rnorm(n, 0, 3)

df <- as.data.frame(cbind(X,Y))
colnames(df)[22] <- 'Y'
df <- df[,2:22]
```

(b)

```r
set.seed(1)

# Train subset
train <- sample(1:n, 100)
```

(c)

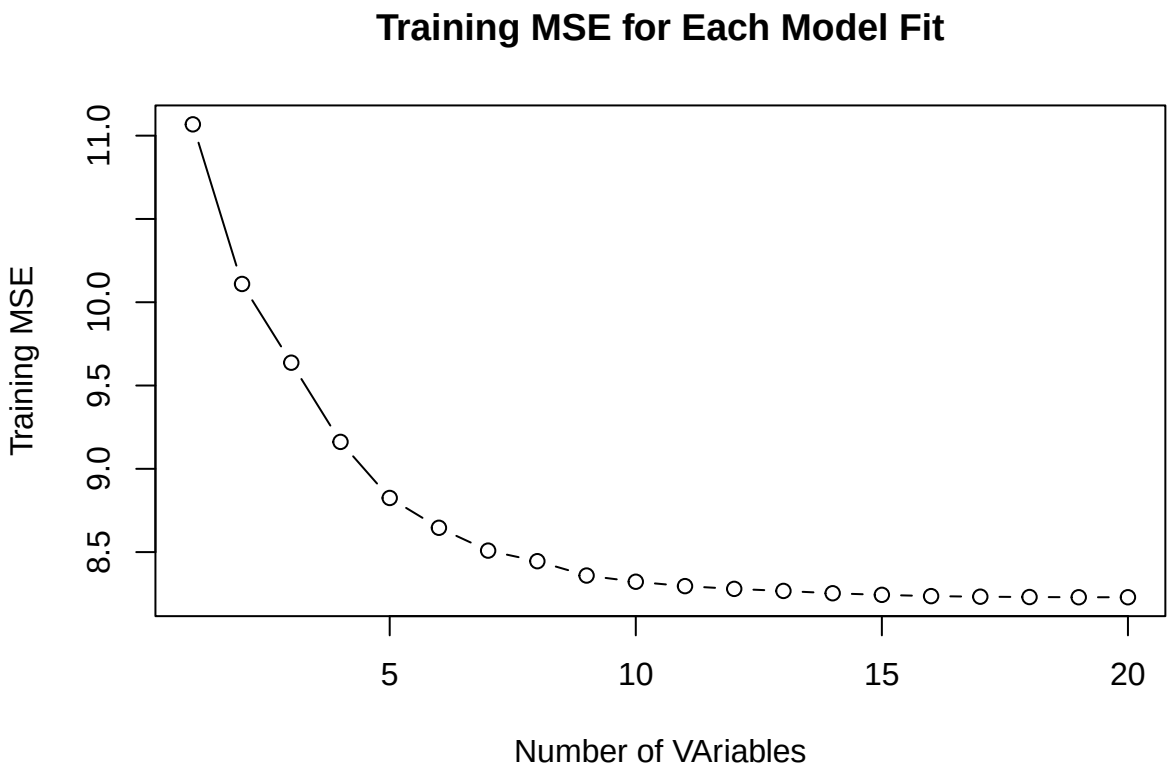
```r
regfit.full <- regsubsets(Y ~ ., data=df[train,], nvmax=20)
summary(regfit.full)
```

```
## Subset selection object
## Call: regsubsets.formula(Y ~ ., data = df[train, ], nvmax = 20)
## 20 Variables  (and intercept)
##     Forced in Forced out
## X1      FALSE      FALSE
## X2      FALSE      FALSE
## X3      FALSE      FALSE
## X4      FALSE      FALSE
## X5      FALSE      FALSE
## X6      FALSE      FALSE
## X7      FALSE      FALSE
## X8      FALSE      FALSE
## X9      FALSE      FALSE
## X10     FALSE      FALSE
## X11     FALSE      FALSE
## X12     FALSE      FALSE
## X13     FALSE      FALSE
## X14     FALSE      FALSE
## X15     FALSE      FALSE
## X16     FALSE      FALSE
## X17     FALSE      FALSE
## X18     FALSE      FALSE
## X19     FALSE      FALSE
## X20     FALSE      FALSE
## 1 subsets of each size up to 20
## Selection Algorithm: exhaustive
##           X1  X2  X3  X4  X5  X6  X7  X8  X9  X10 X11 X12 X13 X14 X15 X16 X17
```

```
## 1  ( 1 )  " " "*" " " " " " " " " " " " " " " " " " " " " " " " " " " " " " "
## 2  ( 1 )  " " "*" " " " " " " "*" " " " " " " " " " " " " " " " " " " " " " "
## 3  ( 1 )  " " "*" " " " " " " "*" " " " " " " "*" " " " " " " " " " " " " " "
## 4  ( 1 )  " " "*" "*" " " " " "*" " " " " " " "*" " " " " " " " " " " " " " "
## 5  ( 1 )  " " "*" "*" "*" " " "*" " " " " " " "*" " " " " " " " " " " " " " "
## 6  ( 1 )  " " "*" "*" "*" " " "*" " " " " " " "*" " " " " " " " " " " " " "*"
## 7  ( 1 )  " " "*" "*" "*" "*" "*" " " " " " " "*" " " " " " " " " " " " " "*"
## 8  ( 1 )  " " "*" "*" "*" "*" "*" " " " " "*" "*" " " " " " " " " " " " " "*"
## 9  ( 1 )  " " "*" "*" "*" "*" "*" " " " " "*" "*" " " " " " " " " "*" " " "*"
## 10  ( 1 ) " " "*" "*" "*" "*" "*" "*" " " "*" "*" " " " " " " " " "*" " " "*"
## 11  ( 1 ) " " "*" "*" "*" "*" "*" "*" " " "*" "*" "*" " " " " " " "*" " " "*"
## 12  ( 1 ) " " "*" "*" "*" "*" "*" "*" " " "*" "*" "*" " " " " " " "*" " " "*"
## 13  ( 1 ) " " "*" "*" "*" "*" "*" "*" " " "*" "*" "*" " " " " " " "*" " " "*"
## 14  ( 1 ) "*" "*" "*" "*" "*" "*" "*" " " "*" "*" "*" " " " " " " "*" " " "*"
## 15  ( 1 ) "*" "*" "*" "*" "*" "*" "*" " " "*" "*" "*" " " " " " " "*" " " "*"
## 16  ( 1 ) "*" "*" "*" "*" "*" "*" "*" " " "*" "*" "*" " " " " " " "*" "*" "*"
## 17  ( 1 ) "*" "*" "*" "*" "*" "*" "*" " " "*" "*" "*" " " "*" " " "*" "*" "*"
## 18  ( 1 ) "*" "*" "*" "*" "*" "*" "*" " " "*" "*" "*" " " "*" "*" "*" "*" "*"
## 19  ( 1 ) "*" "*" "*" "*" "*" "*" "*" " " "*" "*" "*" "*" "*" "*" "*" "*" "*"
## 20  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" "*" "*" "*" "*" "*" "*" "*" "*" "*"
##           X18 X19 X20
## 1  ( 1 )  " " " " " "
## 2  ( 1 )  " " " " " "
## 3  ( 1 )  " " " " " "
## 4  ( 1 )  " " " " " "
## 5  ( 1 )  " " " " " "
## 6  ( 1 )  " " " " " "
## 7  ( 1 )  " " " " " "
## 8  ( 1 )  " " " " " "
## 9  ( 1 )  " " " " " "
## 10  ( 1 ) " " " " " "
## 11  ( 1 ) " " " " " "
## 12  ( 1 ) " " "*" " "
## 13  ( 1 ) " " "*" "*"
## 14  ( 1 ) "*" "*" " "
## 15  ( 1 ) "*" "*" "*"
## 16  ( 1 ) "*" "*" "*"
## 17  ( 1 ) "*" "*" "*"
## 18  ( 1 ) "*" "*" "*"
## 19  ( 1 ) "*" "*" "*"
## 20  ( 1 ) "*" "*" "*"
```

(d)

```
reg.summary <- summary(regfit.full)
plot(reg.summary$rss/100, type='b',
     main='Training MSE for Each Model Fit',
     xlab='Number of VAriables',
     ylab='Training MSE')
```

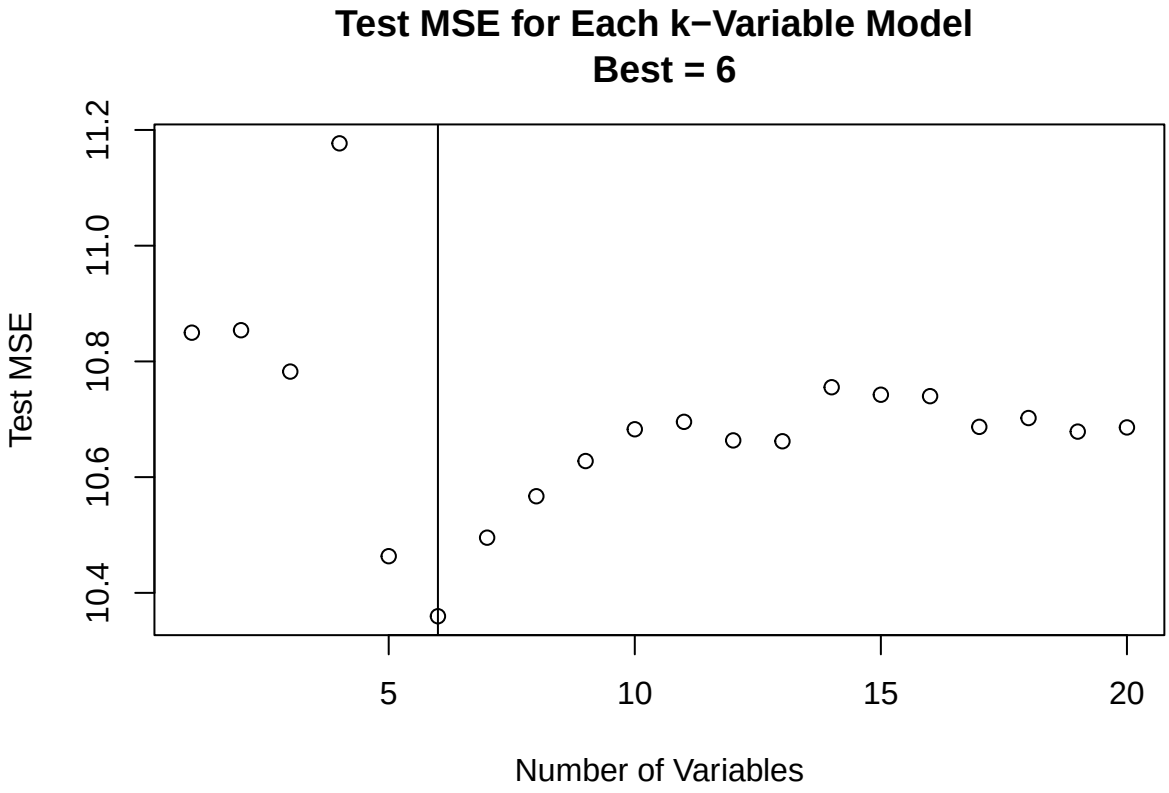


(e)

```
test.mat <- model.matrix(Y ~ ., data=df[-train,])

val.errors <- rep(NA, 20)
for (i in 1:20){
  coefi <- coef(regfit.full, id=i)
  pred <- test.mat[, names(coefi)] %*% coefi
  val.errors[i] <- mean((df$Y[-train] - pred)^2)
}
val.errors
```

```
##  [1] 10.84967 10.85392 10.78243 11.17686 10.46338 10.35960 10.49542 10.56696
##  [9] 10.62782 10.68274 10.69553 10.66348 10.66199 10.75535 10.74241 10.73995
## [17] 10.68687 10.70217 10.67870 10.68593
```

```
plot(val.errors,
     main=paste('Test MSE for Each k-Variable Model \n Best =', which.min(val.errors)),
     xlab='Number of Variables',
     ylab='Test MSE')
abline(v=which.min(val.errors))
```



(e)

The model with six variables takes on the lowest test MSE, which differs from the train MSE which gets higher and higher, as we'd expect from the training phase.

(f)

First we fit the full model on the optimal $k = 6$ variables and then take a look at the coefficients.

```
regfit.fulldata <- regsubsets(Y ~ .,data=df, nvmax=20)

cat('Beta hat values for k=6 variables:\n')
```

```
## Beta hat values for k=6 variables:
```

```
coef(regfit.fulldata, 6)
```

```
## (Intercept)          X2          X4          X6          X8         X10
##    3.200465   -3.319987   -3.376248   -3.376041   -4.640587   -4.120440
##         X13
##    3.629453
```

```
cat('\n')
```

```
cat('True corresponding betas:\n')
```

```
## True corresponding betas:
```

```
sel <- betaT[1, c(1,3,5,7,9,11,14), drop = TRUE]
print(sel)
```

```
##        B0        X2        X4        X6        X8       X10       X13
##  3.000000 -3.122315 -3.723085 -3.663759 -4.793990 -3.497579  0.000000
```

```
cat('\n')
```

```
cat('Differences between beta hats and true betas:\n')
```

```
## Differences between beta hats and true betas:
```

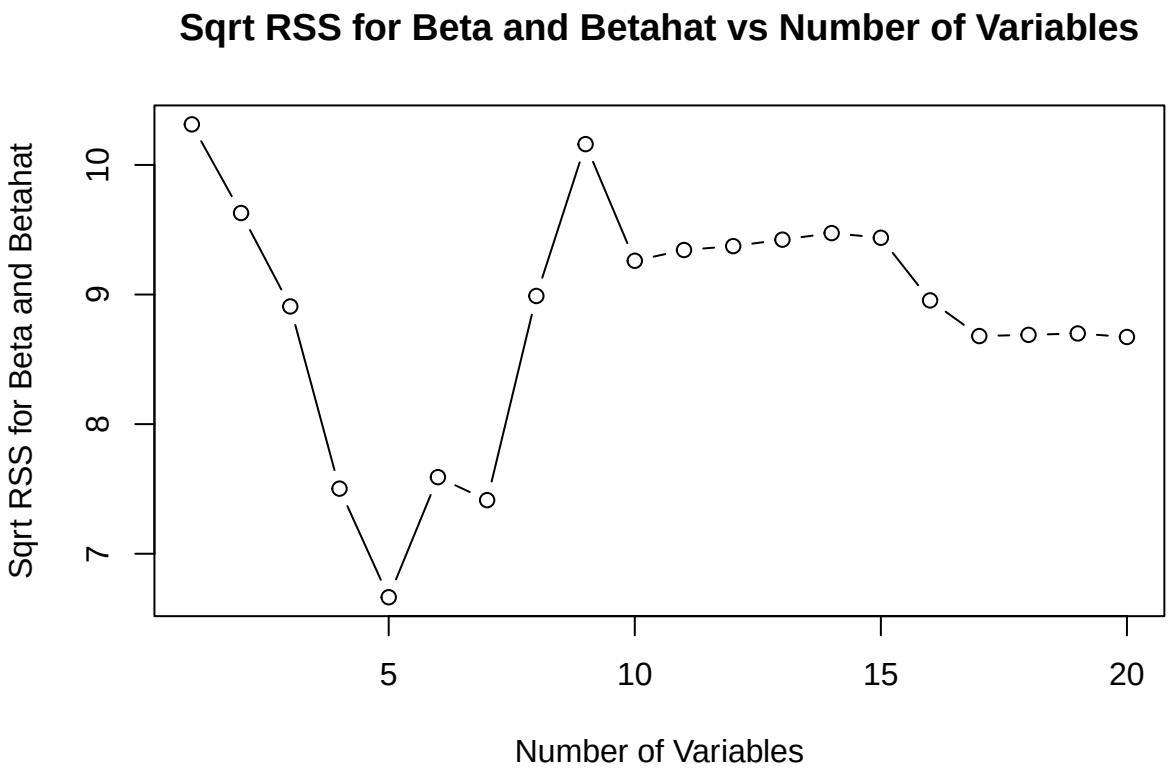
```
coef(regfit.fulldata, 6) - sel
```

```
## (Intercept)          X2          X4          X6          X8         X10
##   0.2004651  -0.1976713   0.3468372   0.2877179   0.1534028  -0.6228611
##         X13
##   3.6294531
```

We see that of the six variables selected, the first four match up pretty closely to the true beta values, the fifth is not quite as close but still close, and the sixth is completely off since the sixth true beta is zero. We can see the differences between the two on the third line above.

(g)

```
beta_true <- as.numeric(betaT[-1])
names(beta_true) <- colnames(betaT)[-1]
beta_diff_norm <- numeric(20)
for(r in 1:20) {
  coef_r <- coef(regfit.fulldata, id = r)
  coef_r <- coef_r[names(coef_r) != "(Intercept)"]

  beta_hat <- setNames(numeric(length(beta_true)), names(beta_true))
  beta_hat[names(coef_r)] <- coef_r

  beta_diff_norm[r] <- sqrt(sum((beta_true - beta_hat)^2))
}

plot(seq_len(p), beta_diff_norm,
     type = "b",
     xlab = "Number of Variables",
     ylab = 'Sqrt RSS for Beta and Betahat',
     main = "Sqrt RSS for Beta and Betahat vs Number of Variables")
```



Using the equation $\sqrt{\sum_{j=1}^{p}(\beta_j - \hat{\beta}_j^r)^2}$, we see that this norm value for each subset of variables has a minimum at the model with five variables, but is still quite low with the model with six variables. Overall, this plot looks pretty similar to our test MSE plot, further validating our choice for the optimal model being the six variable model.