hw5

Norman Hong

March 24, 2019

8

In this exercise, we will generate simulated data, and will then use this data to perform best subset selection.

(a)

Use the rnorm() function to generate a predictor X of length n=100, as well as a noise vector e of length n=100.

```
set.seed(100)
x <- rnorm(100, 0, 1)
e <- rnorm(100, 0, 1)
```

(b)

Generate a response vector Y of length n=100 according to the model: $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3$ where the coefficients are constants of your choice.

```
response <- function(x, e){
  y <- 1 + 2*x + 3*(x*x) + 2*(x*x*x) + e
  return(y)
}
Y <- response(x,e)
```

(c)

Use the regsubsets() function to perform best subset selection in order to choose the best model containing the predictors $X, X^2, ..., X^{10}$. What is the best model obtained according to C_p, BIC, and adjusted R^2? Show some plots to provide evidence for your answer, and report the coefficients of the best model obtained. Note you will need to use the data.frame() function to create a single data set containing both X and Y.

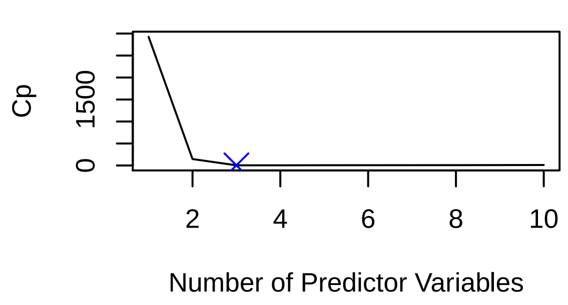
The best model obtained according to Cp, adjusted r-squared, and BIC is the model with 3 predictor variables: x, x^2, x^3. The coefficients for intercept, x, x^2, x^3 associated with this model is $\beta_0 = .98, \beta_1 = 1.871, \beta_2 = 2.96, \beta_3 = 2.01$, respectively.

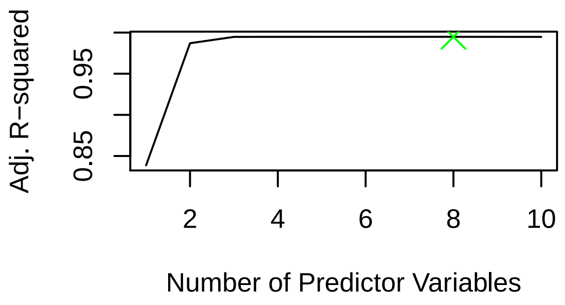
```
data <- data.frame(Y,x)
reg.fit <- regsubsets(Y ~ x + I(x^2) + I(x^3) + I(x^4) + I(x^5) + I(x^6) + I(x^7) + I(x^8) + I(x^9) + I
reg.sum <- summary(reg.fit)
summary(reg.fit)
```

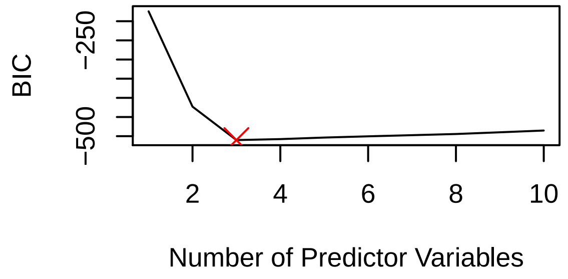
```
## Subset selection object
## Call: regsubsets.formula(Y ~ x + I(x^2) + I(x^3) + I(x^4) + I(x^5) +
##     I(x^6) + I(x^7) + I(x^8) + I(x^9) + I(x^10), data = data,
##     nvmax = 10)
## 10 Variables  (and intercept)
##         Forced in Forced out
## x           FALSE      FALSE
## I(x^2)      FALSE      FALSE
## I(x^3)      FALSE      FALSE
## I(x^4)      FALSE      FALSE
## I(x^5)      FALSE      FALSE
## I(x^6)      FALSE      FALSE
## I(x^7)      FALSE      FALSE
## I(x^8)      FALSE      FALSE
## I(x^9)      FALSE      FALSE
## I(x^10)     FALSE      FALSE
## 1 subsets of each size up to 10
## Selection Algorithm: exhaustive
##           x   I(x^2) I(x^3) I(x^4) I(x^5) I(x^6) I(x^7) I(x^8) I(x^9)
## 1  ( 1 )  " " " "    "*"    " "    " "    " "    " "    " "    " "
## 2  ( 1 )  " " "*"    "*"    " "    " "    " "    " "    " "    " "
## 3  ( 1 )  "*" "*"    "*"    " "    " "    " "    " "    " "    " "
## 4  ( 1 )  "*" "*"    "*"    "*"    " "    " "    " "    " "    " "
## 5  ( 1 )  "*" "*"    "*"    "*"    "*"    " "    " "    " "    " "
## 6  ( 1 )  "*" "*"    "*"    " "    "*"    " "    " "    "*"    " "
## 7  ( 1 )  "*" "*"    "*"    "*"    " "    "*"    " "    "*"    " "
## 8  ( 1 )  "*" "*"    "*"    "*"    "*"    "*"    " "    "*"    " "
## 9  ( 1 )  "*" "*"    "*"    "*"    " "    "*"    "*"    "*"    "*"
## 10  ( 1 ) "*" "*"    "*"    "*"    "*"    "*"    "*"    "*"    "*"
##           I(x^10)
## 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))
adjr2.max <- which.max(reg.sum$adjr2)
cp.min <- which.min(reg.sum$cp)
bic.min <- which.min(reg.sum$bic)

# Created plots to determine the best number of predictors when evaluating with different metrics.
plot(reg.sum$cp, xlab="Number of Predictor Variables", ylab="Cp", type='l')
points(cp.min, reg.sum$cp[cp.min], col='blue', cex=2, pch=4)
plot(reg.sum$adjr2, xlab='Number of Predictor Variables', ylab="Adj. R-squared", type='l')
points(adjr2.max, reg.sum$adjr2[adjr2.max], col='green', cex=2, pch=4)
plot(reg.sum$bic, xlab='Number of Predictor Variables', ylab='BIC', type='l')
points(bic.min, reg.sum$bic[bic.min], col='red', cex=2, pch=4)
```







```
coef(reg.fit, 3) # Determine the coefficients associated with the best model that has 3 predictors.
```

```
## (Intercept)           x      I(x^2)      I(x^3)
##   0.9395742   1.8067474   3.0655041   2.0249081
```

(d)

Repeat (c), using forward stepwise selection and also using backwards stepwise selection. How does your answer compare to the results in (c)?

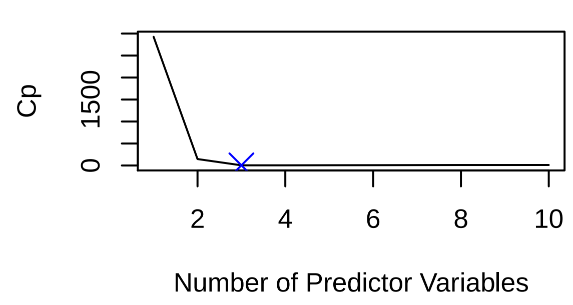
For forward selection, all 3 metrics pointed to 3 predictor variables as being the best model. In addition, the 3 predictors to select are the same: x, x^2, x^3. The same with backward stepwise selection. However, the chosen best model for a fixed number of parameters differ between each method.

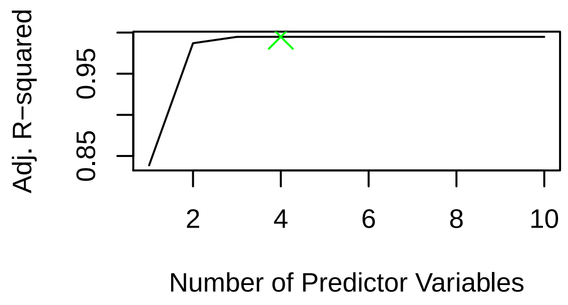
```
# forward stepwise selection
reg.fit <- regsubsets(Y ~ x + I(x^2) + I(x^3) + I(x^4) + I(x^5) + I(x^6) + I(x^7) + I(x^8) + I(x^9) + I
reg.sum <- summary(reg.fit)
summary(reg.fit)
```

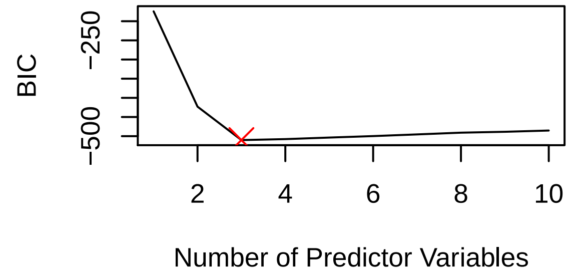
```
## Subset selection object
## Call: regsubsets.formula(Y ~ x + I(x^2) + I(x^3) + I(x^4) + I(x^5) +
##     I(x^6) + I(x^7) + I(x^8) + I(x^9) + I(x^10), data = data,
##     nvmax = 10, method = "forward")
## 10 Variables  (and intercept)
##        Forced in Forced out
## x          FALSE      FALSE
## I(x^2)     FALSE      FALSE
## I(x^3)     FALSE      FALSE
## I(x^4)      FALSE      FALSE
## I(x^5)      FALSE      FALSE
## I(x^6)      FALSE      FALSE
## I(x^7)      FALSE      FALSE
## I(x^8)      FALSE      FALSE
## I(x^9)      FALSE      FALSE
## I(x^10)     FALSE      FALSE
## 1 subsets of each size up to 10
## Selection Algorithm: forward
##           x   I(x^2) I(x^3) I(x^4) I(x^5) I(x^6) I(x^7) I(x^8) I(x^9)
## 1  ( 1 )  " " " "    "*"    " "    " "    " "    " "    " "    " "   
## 2  ( 1 )  " " "*"    "*"    " "    " "    " "    " "    " "    " "   
## 3  ( 1 )  "*" "*"    "*"    " "    " "    " "    " "    " "    " "   
## 4  ( 1 )  "*" "*"    "*"    "*"    " "    " "    " "    " "    " "   
## 5  ( 1 )  "*" "*"    "*"    "*"    "*"    " "    " "    " "    " "   
## 6  ( 1 )  "*" "*"    "*"    "*"    "*"    " "    " "    " "    "*"   
## 7  ( 1 )  "*" "*"    "*"    "*"    "*"    " "    "*"    " "    "*"   
## 8  ( 1 )  "*" "*"    "*"    "*"    "*"    "*"    "*"    " "    "*"   
## 9  ( 1 )  "*" "*"    "*"    "*"    "*"    "*"    "*"    "*"    "*"   
## 10  ( 1 ) "*" "*"    "*"    "*"    "*"    "*"    "*"    "*"    "*"   
##           I(x^10)
## 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))
adjr2.max <- which.max(reg.sum$adjr2)
cp.min <- which.min(reg.sum$cp)
bic.min <- which.min(reg.sum$bic)

# Created plots to determine the best number of predictors when evaluating with different metrics.
plot(reg.sum$cp, xlab="Number of Predictor Variables", ylab="Cp", type='l')
points(cp.min, reg.sum$cp[cp.min], col='blue', cex=2, pch=4)
plot(reg.sum$adjr2, xlab='Number of Predictor Variables', ylab="Adj. R-squared", type='l')
points(adjr2.max, reg.sum$adjr2[adjr2.max], col='green', cex=2, pch=4)
plot(reg.sum$bic, xlab='Number of Predictor Variables', ylab='BIC', type='l')
points(bic.min, reg.sum$bic[bic.min], col='red', cex=2, pch=4)
```





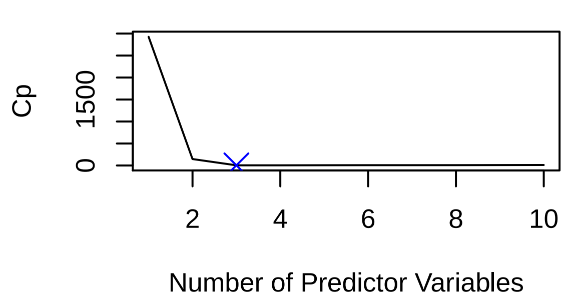


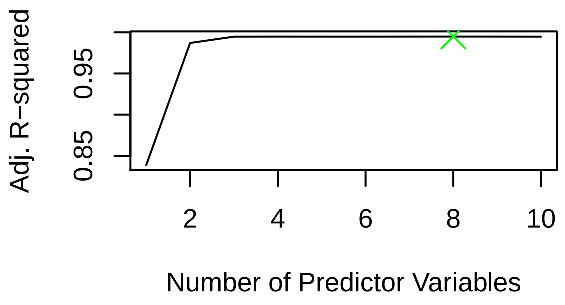
```r
# backward stepwise selection
reg.fit <- regsubsets(Y ~ x + I(x^2) + I(x^3) + I(x^4) + I(x^5) + I(x^6) + I(x^7) + I(x^8) + I(x^9) + I
reg.sum <- summary(reg.fit)
summary(reg.fit)
```

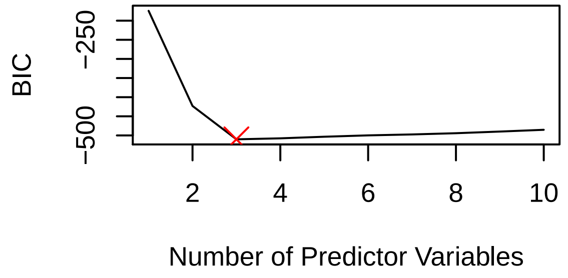
```
## Subset selection object
## Call: regsubsets.formula(Y ~ x + I(x^2) + I(x^3) + I(x^4) + I(x^5) +
##     I(x^6) + I(x^7) + I(x^8) + I(x^9) + I(x^10), data = data,
##     nvmax = 10, method = "backward")
## 10 Variables  (and intercept)
##         Forced in Forced out
## x           FALSE      FALSE
## I(x^2)      FALSE      FALSE
## I(x^3)      FALSE      FALSE
## I(x^4)      FALSE      FALSE
## I(x^5)      FALSE      FALSE
## I(x^6)      FALSE      FALSE
## I(x^7)      FALSE      FALSE
## I(x^8)      FALSE      FALSE
## I(x^9)      FALSE      FALSE
## I(x^10)     FALSE      FALSE
## 1 subsets of each size up to 10
## Selection Algorithm: backward
##           x   I(x^2) I(x^3) I(x^4) I(x^5) I(x^6) I(x^7) I(x^8) I(x^9)
## 1  ( 1 )  " " " "    "*"    " "    " "    " "    " "    " "    " "
## 2  ( 1 )  " " "*"    "*"    " "    " "    " "    " "    " "    " "
## 3  ( 1 )  "*" "*"    "*"    " "    " "    " "    " "    " "    " "
## 4  ( 1 )  "*" "*"    "*"    " "    " "    "*"    " "    " "    " "
## 5  ( 1 )  "*" "*"    "*"    " "    " "    "*"    " "    "*"    " "
## 6  ( 1 )  "*" "*"    "*"    "*"    " "    "*"    " "    "*"    " "
## 7  ( 1 )  "*" "*"    "*"    "*"    " "    "*"    " "    "*"    " "
## 8  ( 1 )  "*" "*"    "*"    "*"    " "    "*"    "*"    "*"    " "
## 9  ( 1 )  "*" "*"    "*"    "*"    " "    "*"    "*"    "*"    "*"
## 10  ( 1 ) "*" "*"    "*"    "*"    "*"    "*"    "*"    "*"    "*"
##           I(x^10)
## 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))
adjr2.max <- which.max(reg.sum$adjr2)
cp.min <- which.min(reg.sum$cp)
bic.min <- which.min(reg.sum$bic)

# Created plots to determine the best number of predictors when evaluating with different metrics.
plot(reg.sum$cp, xlab="Number of Predictor Variables", ylab="Cp", type='l')
points(cp.min, reg.sum$cp[cp.min], col='blue', cex=2, pch=4)
plot(reg.sum$adjr2, xlab='Number of Predictor Variables', ylab="Adj. R-squared", type='l')
points(adjr2.max, reg.sum$adjr2[adjr2.max], col='green', cex=2, pch=4)
plot(reg.sum$bic, xlab='Number of Predictor Variables', ylab='BIC', type='l')
points(bic.min, reg.sum$bic[bic.min], col='red', cex=2, pch=4)
```







Extra 43

Suppose you are given a single data set with p=3 predictors X_1, X_2, X_3. In order to add flexibility to a linear model, you have decided to include also all powers X_i^2 and all interaction terms X_iX_j as variables.

(a)

How many variables are there now? How many different models are there with exactly 3 variables? You don't have to list them.

There are $6 + 3 = 9$ variables total in the model. There are 9 choose 3 different models with exactly 3 variables. This means that are 84 different models with 3 variables.

(b)

List all models with exactly 3 variables that satisfy the hierarchy principle. $Y = X__1 + X__2 + X__3 $, $Y = X_1 + X_2 + X_1^2$, $Y = X_1 + X_2 + X_2^2$, $Y = X_1 + X_2 + X_2 \cdot X_1$, $Y = X_1 + X_3 + X_1^2$, $Y = X_1 + X_3 + X_3^2$, $Y = X_3 + X_1 + X_3 \cdot X_3$, $Y = X_3 + X_2 + X_2^2$, $Y = X_3 + X_2 + X_3^2$, $Y = X_3 + X_2 + X_2 \cdot X_3$

(c)

Propose a modification of the forward selection method such that all selected models satisfy the hierarchy principle. Explain it in words. You don't have to write down a formal algorithm.

Start at model with 0 predictor variables. Select a variable and put into model and determine the best 1-variable model based on the lowest RSS. If the best variable is X_1 or X_2 or X_3, then continue to select a 2-variable model. Determine the best 2-variable model based on the lowest RSS. If the best variable selected is not an interaction variable or quadratic, then continue to select a 3-variable model. Next add an interaction term or term with degree 2 and determine the best 3-variable model by selecting the model with lowest RSS.

If the best variable selected for a 1-variable model is a quadratic, then must select the corresponding term with degree 1 for the 2-variable model. Then continue to select the best variable for the 3-variable model.

If the best variable selected for a 1-variable model is an interaction term, then must select the the corresponding single variables with degree 1. Therefore, the 3-variable model is selected.

If the best variable selected for a 2-variable model is an interaction term, then must select the corresponding left out degree 1 term when selecting the 3-variable model.

If the best variable selected for a 2-variable model is a quadratic term, then must select the corresponding degree 1 term as the variable for 3-variable model.

Extra 44

Consider the concrete strength data from problem 37. There are eight predictors and 1 response. Use forward selection and backward selection to identify good models with $k=1, 2, . . . , 8 variables. Plot the BIC values for both sequences of models in the same graph. Do the sequence of models agree?

The BIC values closely match, but not perfectly inline. In addition, there are differences in the selected predictor variables for models between 2 predictors and 5 predictors. THe best model is the same for both.

```r
reg.fit.concf <- regsubsets(y~., data=conc, nvmax=8, method='forward')
reg.sum.concf <- summary(reg.fit.concf)
summary(reg.fit.concf)
```

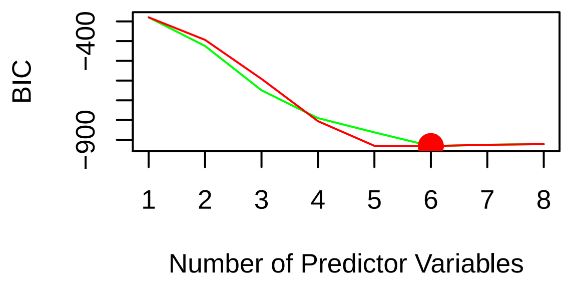
```
## Subset selection object
## Call: regsubsets.formula(y ~ ., data = conc, nvmax = 8, method = "forward")
## 8 Variables  (and intercept)
##    Forced in Forced out
## x1     FALSE      FALSE
## x2     FALSE      FALSE
## x3     FALSE      FALSE
## x4     FALSE      FALSE
## x5     FALSE      FALSE
## x6     FALSE      FALSE
## x7     FALSE      FALSE
## x8     FALSE      FALSE
## 1 subsets of each size up to 8
## Selection Algorithm: forward
##          x1  x2  x3  x4  x5  x6  x7  x8 
## 1  ( 1 ) "*" " " " " " " " " " " " " " "
## 2  ( 1 ) "*" " " " " " " "*" " " " " " "
## 3  ( 1 ) "*" " " " " " " "*" " " " " "*"
## 4  ( 1 ) "*" "*" " " " " "*" " " " " "*"
## 5  ( 1 ) "*" "*" " " "*" "*" " " " " "*"
## 6  ( 1 ) "*" "*" "*" "*" "*" " " " " "*"
## 7  ( 1 ) "*" "*" "*" "*" "*" "*" " " "*"
## 8  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*"
```

```r
reg.fit.concb <- regsubsets(y~., data=conc, nvmax=8, method='backward')
reg.sum.concb <- summary(reg.fit.concb)
summary(reg.fit.concb)
```

```
## Subset selection object
## Call: regsubsets.formula(y ~ ., data = conc, nvmax = 8, method = "backward")
## 8 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
## 1 subsets of each size up to 8
## Selection Algorithm: backward
##          x1  x2  x3  x4  x5  x6  x7  x8 
## 1  ( 1 ) "*" " " " " " " " " " " " " " "
## 2  ( 1 ) "*" " " " " " " " " " " " " "*"
## 3  ( 1 ) "*" " " " " "*" " " " " " " "*"
## 4  ( 1 ) "*" "*" " " "*" " " " " " " "*"
## 5  ( 1 ) "*" "*" "*" "*" " " " " " " "*"
## 6  ( 1 ) "*" "*" "*" "*" "*" " " " " "*"
## 7  ( 1 ) "*" "*" "*" "*" "*" "*" " " "*"
## 8  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*"
```

```r
par(mfrow=c(2,2))
bic.minf <- which.min(reg.sum.concf$bic)
bic.minb <- which.min(reg.sum.concb$bic)

# Created plots to determine the best number of predictors when evaluating with different metrics.
plot(reg.sum.concf$bic, xlab='Number of Predictor Variables', ylab="BIC", type='l', col='green')
points(bic.minf, reg.sum.concf$bic[bic.minf], col='green', cex=3, pch=20)
lines(reg.sum.concb$bic, col='red')
points(bic.minb, reg.sum.concb$bic[bic.minb], col='red', cex=3, pch=20)
```



1

We perform best subset, forward stepwise, and backward stepwise selection on a single data set. For each approach, we obtain p+1 models, containing 0, 1, 2, . . . , p predictors. Explain your answers.

(a)

Which of the three models with k predictors has the smallest training RSS?

Best subset selection method will always produce the most optimal model for every fixed number of predictors when compared to forward stepwise and backward stepwise method. The reason being is that forward subset and backward subset is based on a greedy approach that selects the locally best model at each iteration. These 2 methods do not search all possible models.

(b)

Which of the three models with k predictors has the smallest test RSS?

Because models are selected based on training RSS, it is not possible that a model with best training RSS will have a high test RSS.

(c)

1. The predictors in the k-variable model identified by forward stepwise are a subset of the predictors in the (k+1)-variable model identified by forward stepwise selection. True.

2. The predictors in the k-variable model identified by backward stepwise are a subset of the predictors in the k+1 variable model idenfitied by backward stepwise selection. True.
3. The predictors in the k-variable model identified by backward stepwise are a subset of the predictors in the k+1 variable model identified by forward stepwise selection. False. Backward stepwise starts from a model with all predictors and forward stepwise selection starts from a model with 0 predictors. It is possible that the two methods might identify different best models for each intermediate step.
4. The predictors in the k-variable model identified by forward stepwise are a subset of the predictors in the k+1 variable model identified by backward stepwise selection. False. Same reason as number 3.
5. The predictors in the k-variable model identified by best subset are a subset of the predictors in the k+1 variable model identified by best subset selection. False. The reason is that at each iteration, the model from the previous iteration is not used in determining the best model for the current iteration.

10

We have seen that as the number of features used in a model increases, the training error will necessarily decrease, but the test error may not. We will now explore this in a simulated data set.

(a)

Generate a data set with p=20 features, n=1000 observations, and an associated quantitative response vector generated according to the model $y = X \cdot \beta + \epsilon$, where β has some elements that are exactly equal to 0.

```
set.seed(10)
x <- matrix(rnorm(1000*20, 0, 1), 1000, 20)
b <- rnorm(20, 4, 1)
b[10] = b[2] = b[17] = b[5] = b[11] = 0
e <- rnorm(1000, 0,5)
y <- x %*% b + e
data <- data.frame(x,y)
```

(b)

Split your data set into a training set containing 100 observations and a test set containing 900 observations.

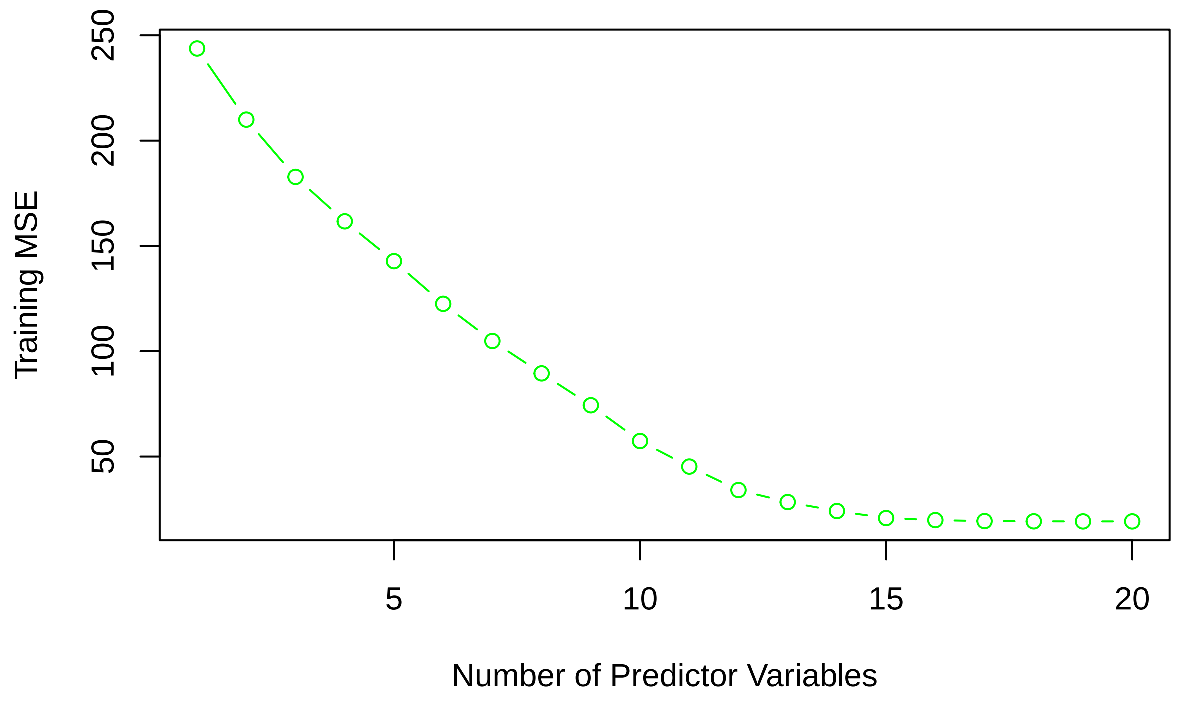
```
set.seed(100)
train <- sample(1:1000, 100, replace=FALSE)
test <- -train

data.train <- data[train,]
data.test <- data[-train,]
```

(c)

Perform best subset selection on the training set, and plot the training set MSE associated with the best model of each size.

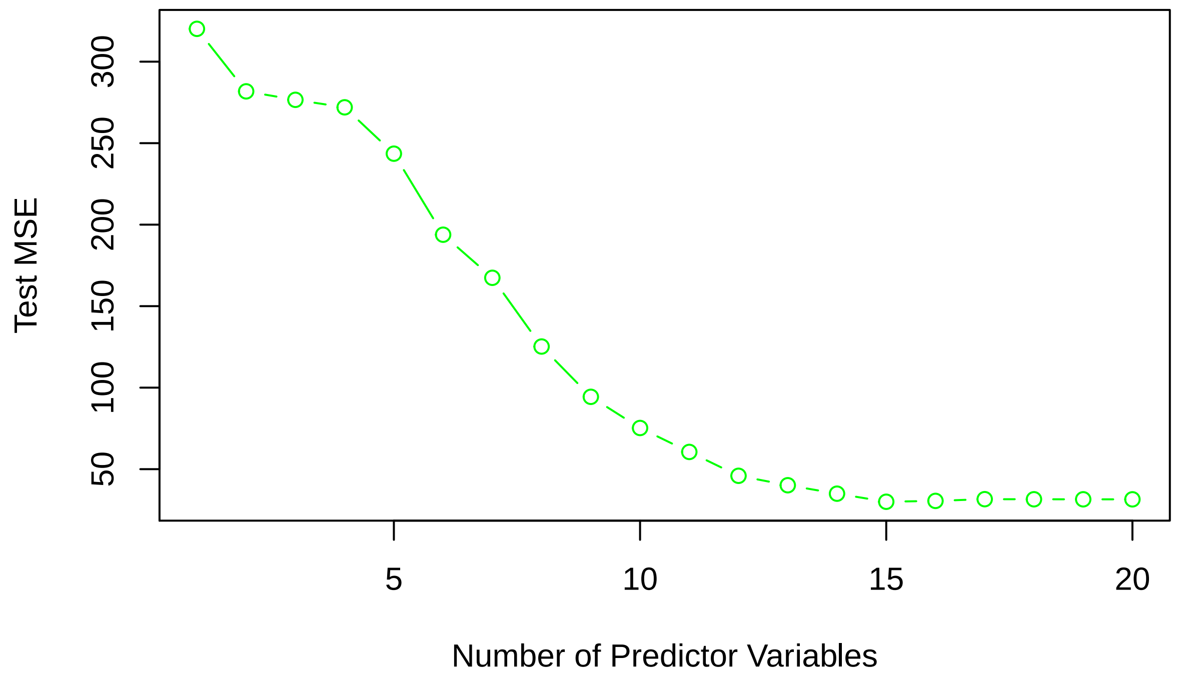
```
reg.best.train <- regsubsets(y~., data=data.train, nvmax=20)
train.mat = model.matrix(y~., data=data.train, nvmax=20)
val.errors = rep(NA, 20)
for (i in 1:20) {
    coefi = coef(reg.best.train, id = i)
    pred = train.mat[, names(coefi)] %*% coefi
    val.errors[i] = mean((pred - data.train[,'y'])^2)
}
plot(val.errors, xlab='Number of Predictor Variables', ylab="Training MSE", type='b', col='green')
```



(d)

Plot the test set MSE associated with the best model of each size.

```
test.mat <- model.matrix(y~., data=data.test, nvmax=20)
val.errors = rep(NA, 20)
for (i in 1:20) {
    coefi = coef(reg.best.train, id = i)
    pred = test.mat[, names(coefi)] %*% coefi
    val.errors[i] = mean((pred - data.test[,'y'])^2)
}
plot(val.errors, xlab='Number of Predictor Variables', ylab="Test MSE", type='b', col='green')
```



(e)

For which model size does the test set MSE take on its minimum value? Comment on your results. If it takes on its minimum value for a model containing only an intercept or a model containing all of the features, then play around with the way that you are generating the data in (a) until you come up with a scenario in which the test set MSE is minimized for an intermediate model size.

Lowest test MSE is when there are 15 predictor variables in the model.

```
which.min(val.errors)
```

```
## [1] 15
```

(f)

How does the model at which the test set MSE is minimized compare to the true model used to generate the data? Comment on the coefficient values?

The best subset selection approach was able to predict the correct variables and the estimated coefficients were very close to the true coefficients. The coefficients that had a true value of 0 were not included into the model.

```
coef(reg.best.train, 15)
```

```
## (Intercept)          X1          X3          X4          X6          X7
##   0.4469421   4.9080392   6.1284470   5.3480666   2.1951345   6.4698639
##          X8          X9         X12         X13         X14         X15
##   2.7758909   2.3092343   4.6085787   6.5909270   5.7549532   4.5148671
##         X16         X18         X19         X20
##   3.9248754   3.8428133   4.1333086   4.8716054
```

```
b
```

```
##  [1] 4.974740 0.000000 5.567393 4.474008 0.000000 2.456266 6.654931
##  [8] 1.851727 2.231923 0.000000 0.000000 5.035979 6.236543 5.263022
## [15] 5.138449 3.571157 0.000000 4.090042 3.981642 4.238047
```

(g)

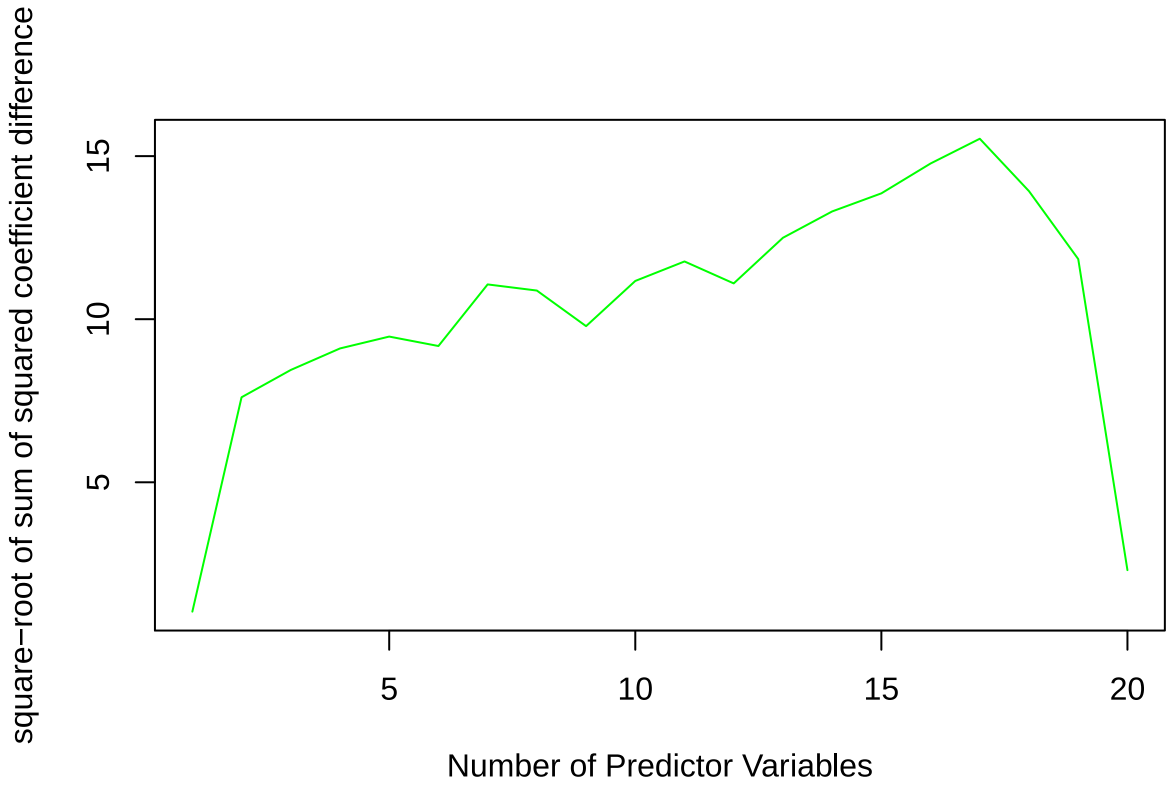
Create a plot displaying the $\sqrt{\sum_{j=1}^{p}(\beta_j - \beta_j^r)^2}$ for a range of values of r, where β_j^r is the jth coefficient estimate for the best model containing r coefficients. Comment on what you observe. How does this compare to the test MSE plot from (d)?

The coefficients seem to increase as the number of predictors go up, but then goes back down. This is different from the test MSE because the test MSE was monotone decreasing.

```
sssc <- rep(NA, 20)

for (i in 1:20){ # Iterating through the number of predictors from 1 to 20.
  coef.test <- coef(reg.best.train, i)
  dif <- b[1:(length(coef.test)-1)] - coef.test[2:length(coef.test)]
  sssc[i] <- sqrt(sum(dif*dif))
}

plot(sssc, xlab='Number of Predictor Variables', ylab="square-root of sum of squared coefficient differe
```



Extra 46

We'll use the MNIST image classification data again, available as mnist_all.RData that were used in class during the last 2 weeks. We want to distinguish between 1 and 3. Extract the relevant training data and place them in a data frame. Remove all variables (pixels) that have 0 variance, i.e. pixels that have the same value for both digits. Repeat this for the test data.

In this problem, you will do 2 forward selection steps for finding good logistic models.

```r
# Creating training set.  digit 1 will be class 0.  digit 3 will be class 1.
train.1 <- train$x[train$y == 1,]
train.3 <- train$x[train$y == 3,]
train.1 <- as.data.frame(x = train.1)
train.3 <- as.data.frame(x = train.3)
train.1$y <- 0
train.3$y <- 1
train.13 <- rbind(train.1, train.3)

# Creating test set. digit 1 will be class 0.  digit 3 will be class 1
test.1 <- test$x[test$y == 1,]
test.3 <- test$x[test$y == 3,]
test.1 <- as.data.frame(x = test.1)
test.3 <- as.data.frame(x = test.3)
test.1$y <- 0
test.3$y <- 1
digit.t = rbind(test.1, test.3)
test.13 <- rbind(test.1, test.3)

# Remove column with 0 variance.
```

```r
train.13.var0 <- lapply(train.13, function(x) 0 == var(x)) # True if the column has var 0.  false if it
train.names <- names(train.13.var0[train.13.var0 == 0]) # False = 0.  True = 1.  Select those that are
test.13.var0 <- lapply(test.13, function(x) 0 == var(x))
test.names <- names(test.13.var0[test.13.var0 == 0])
uniq.names <- unique(test.names, train.names)
train.13.var0 <- subset(train.13, select = uniq.names)
test.13.var0 <- subset(test.13, select = uniq.names)
```

(a)

Find the pixel that gives the best logistic model for the training data, using the area under the ROC curve as a criterion. Do this with a complete search. Do not show the output of all logistic models!

The variable that gives the best logistic model for the training data is V490, and the area under the ROC curve for this model is 0.94.

```r
i = 1
auc.max = 0
auc.maxname = ""

for (c in colnames(train.13.var0)[1:ncol(train.13.var0)-1]) {
  log.fit = suppressWarnings(glm(y ~ get(c), data = train.13.var0, family = binomial))
  pred = suppressWarnings(predict(log.fit, train.13.var0, type = "response"))
  auc = auc(roc(train.13.var0$y ~ pred))[1]
  if(auc > auc.max){
    auc.max = auc
    auc.maxname = c
  }
}
auc.max
```

```
## [1] 0.9438168
```

```
auc.maxname
```

```
## [1] "V490"
```

(b)

Now find one more pixel such that the resulting logistic model using the pixel from a) together with the new one has the best area under the ROC curve. Do this with a complete search. Minimize the output.

The 2-variable model that gives the best logistic model for the training data is V495, and the area under the ROC curve for this model is .974.

```
i = 1
auc.max = 0
auc.maxname = ""

for (c in colnames(train.13.var0)[1:ncol(train.13.var0)-1]) {
  log.fit = suppressWarnings(glm(y ~ V490 + get(c), data = train.13.var0, family = binomial))
  pred = suppressWarnings(predict(log.fit, train.13.var0, type = "response"))
  auc = auc(roc(train.13.var0$y ~ pred))[1]
  if(auc > auc.max){
    auc.max = auc
    auc.maxname = c
  }
}
auc.max
```

```
## [1] 0.974748
```

```
auc.maxname
```

```
## [1] "V495"
```

(c)

Use the test data to decide whether the second model is really better than the first one.

The auc is higher for the 2-variable model.

```
# model with only V490
log.fit = glm(y ~ V490, data = test.13.var0, family = binomial)
pred = predict(log.fit, test.13.var0, type = "response")
auc(roc(test.13.var0$y ~ pred))[1]
```

```
## [1] 0.9575496
```

```
# model with both predictors V490 and V495
log.fit = glm(y ~ V490 + V495, data = test.13.var0, family = binomial)
pred = predict(log.fit, test.13.var0, type = "response")
auc(roc(test.13.var0$y ~ pred))[1]
```

```
## [1] 0.9824739
```

(d)

How many logistic models altogether have you examined? How many will you have to examine if you want to continue this process and make the best logistic model with 10 pixels?

The sum of the number of logistic models examined in part a and b is $544 + 543 = 1087$. If I continued this process, the total number of models examined is $544+543+542+541+540+539+538+537+536+535 = 5395$

```
length(colnames(train.13.var0)[1:ncol(train.13.var0)-1]) # number of models for part a.
```

```
## [1] 544
```