Homework 3

STAT 471

Jacob Kahn
Devesh Dayal
Meghana Jayam

30 October, 2016

Preflight Tasks

We have a lot of existing variables, so we'll clean them out.

```
rm(list=ls())
```

Check our working directory. This changed for various group members, so we each set it locally:

```
# setwd(dir)
# getwd() # check that working directory
```

Problem 1

Part a

Generate a predictor X of length n=100, as well as a noise vector epsilon of length n=100.

```
set.seed(10)
X <- rnorm(100)
epsilon <- rnorm(100)
```

Part b

Generate a response vector Y of length n=100 with B0, B1, B2, B3

```
B0 <- -2
B1 <- 0.1
B2 <- 1
B3 <- 5
Y <- B0+B1*X+B2*X^2+B3*X^3+epsilon
```

Part c

Perform best subset selection (find Cp, BIC, adjr2)

```r
library(leaps) # for regsubsets
data <- data.frame(y=Y, x=X)
regsub <- regsubsets(y~poly(x,degree=10,raw=TRUE), data=data, nvmax=10)
reg.sum <- summary(regsub)
names(reg.sum)
```

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

```r
# find optimal size by getting min Cp, BIC, adjr2
which.min(reg.sum$cp)       # 2
```

```
## [1] 2
```

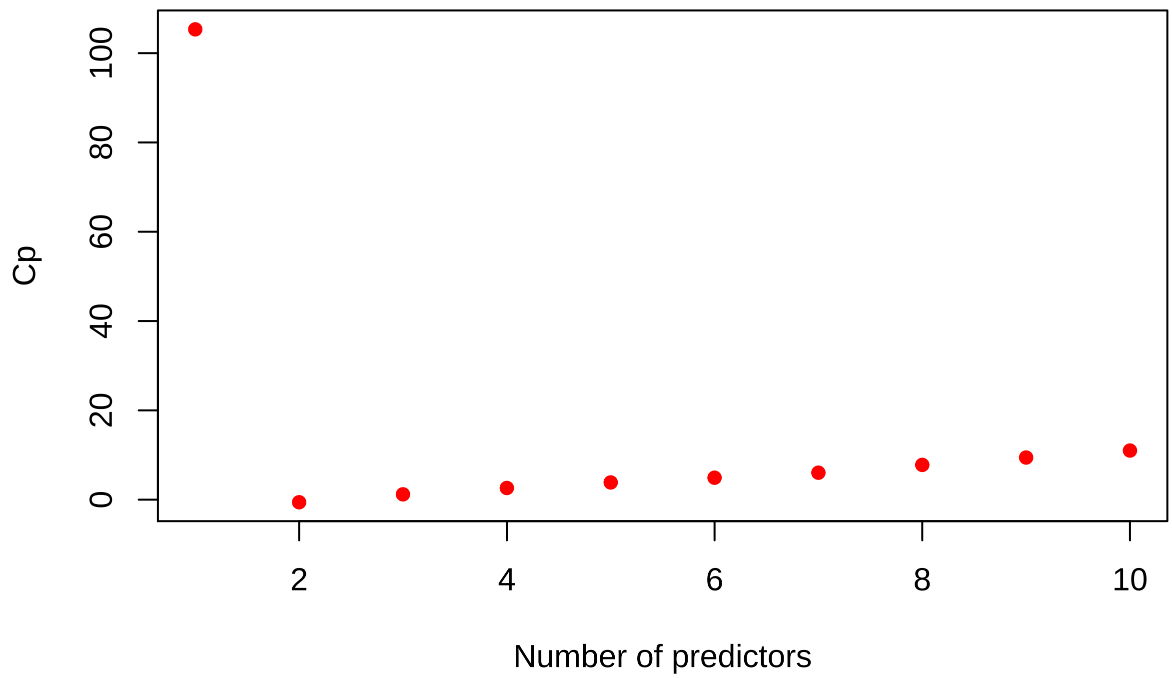
```r
which.min(reg.sum$bic)      # 2
```

```
## [1] 2
```

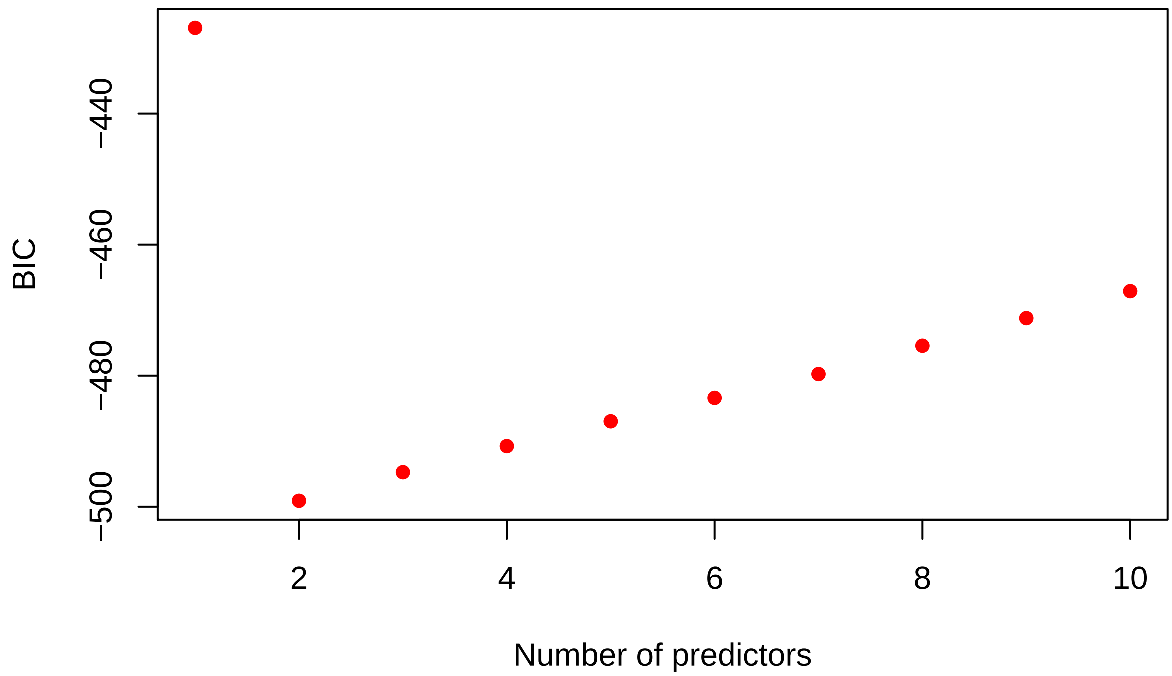
```r
which.max(reg.sum$adjr2)    # 2
```

```
## [1] 2
```

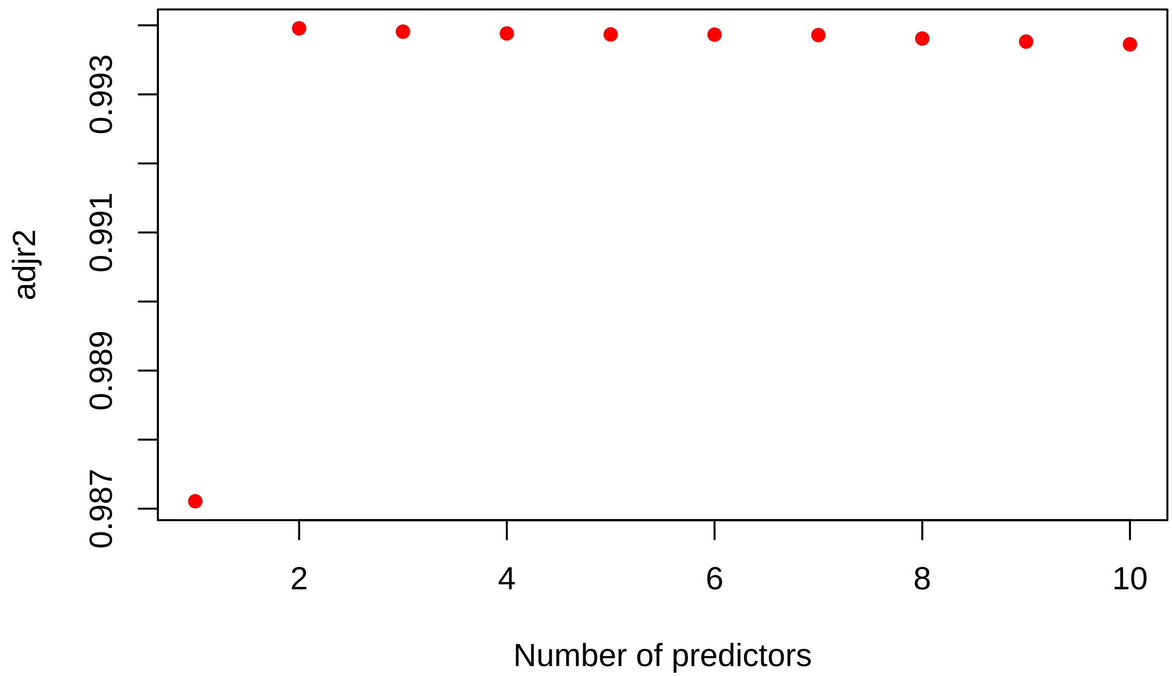
```r
# plot Cp, BIC, adjr2
plot(reg.sum$cp, xlab="Number of predictors", ylab="Cp", col="red", type="p", pch=16) # exhibit 1
```



```r
plot(reg.sum$bic, xlab="Number of predictors", ylab="BIC", col="red", type="p", pch=16) # exhibit 2
```




```r
plot(reg.sum$adjr2, xlab="Number of predictors", ylab="adjr2", col="red", type="p", pch=16) # exhibit 3
```



As seen by the plots, we would use a 2-variable model with Cp, a 2-variable model with BIC, and a 2-variable model with Adjusted R^2.

```r
coef(regsub, id=2)
```

```
##                       (Intercept) poly(x, degree = 10, raw = TRUE)2
##                        -2.0710776                         0.9642645
## poly(x, degree = 10, raw = TRUE)3
##                         5.0163042
```

```
## (Intercept) poly(x, degree = 10, raw = TRUE)2 poly(x, degree = 10, raw = TRUE)3
## -2.0710776                        0.9642645                         5.0163042
```

The model will use x^2 and x^3.

Part d

Use forward selection and the backward selection to compare esults with part c. First, we will do forward selection:

```r
regsub.f <- regsubsets(y~poly(x,degree=10,raw=TRUE), data=data, nvmax=10, method="forward")
reg.sum.f <- summary(regsub.f)

which.min(reg.sum.f$cp)       # 2
```

```
## [1] 2
```

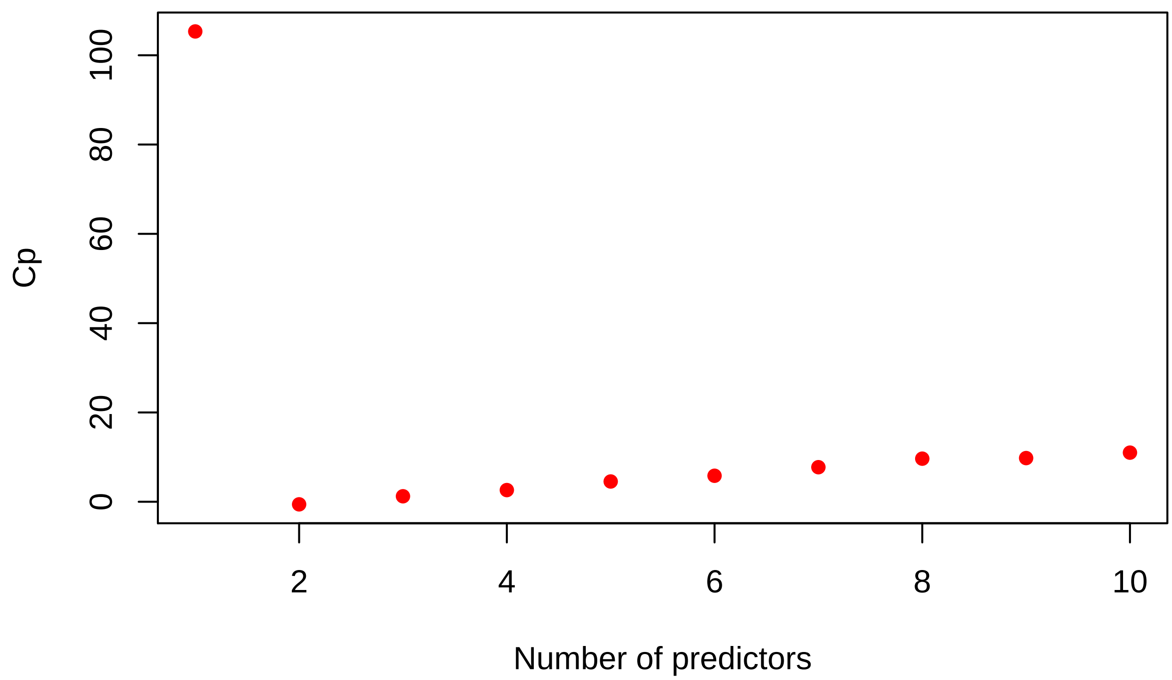
```r
which.min(reg.sum.f$bic)       # 2
```

```
## [1] 2
```

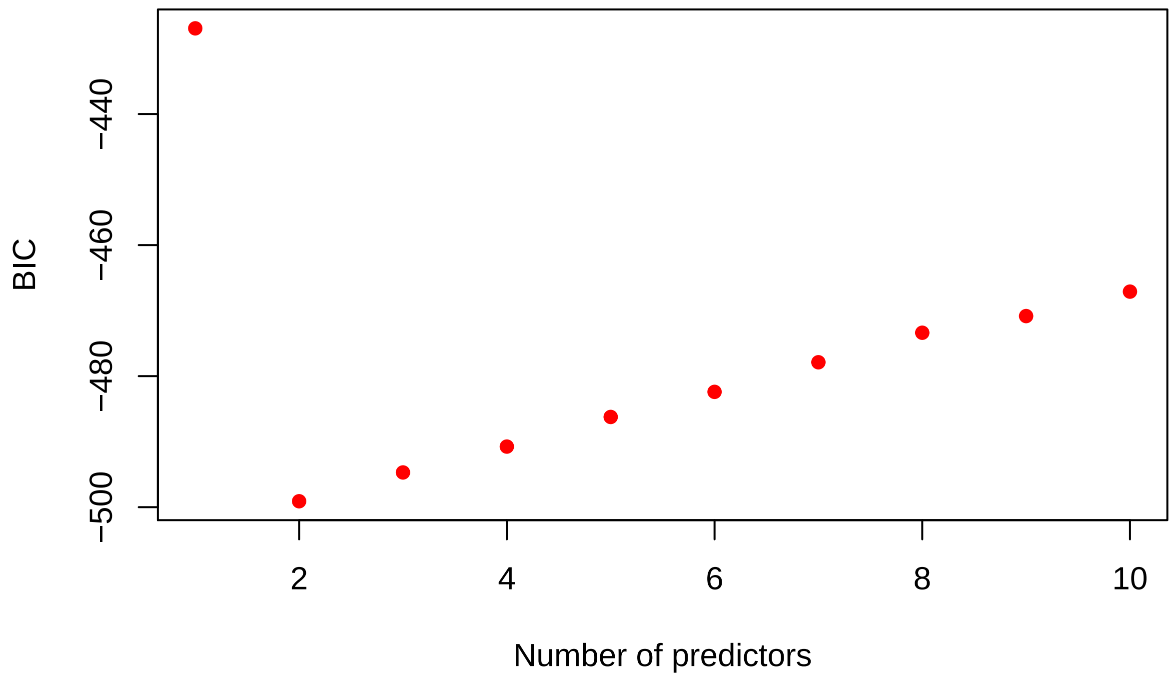
```r
which.max(reg.sum.f$adjr2)     # 2
```

```
## [1] 2
```

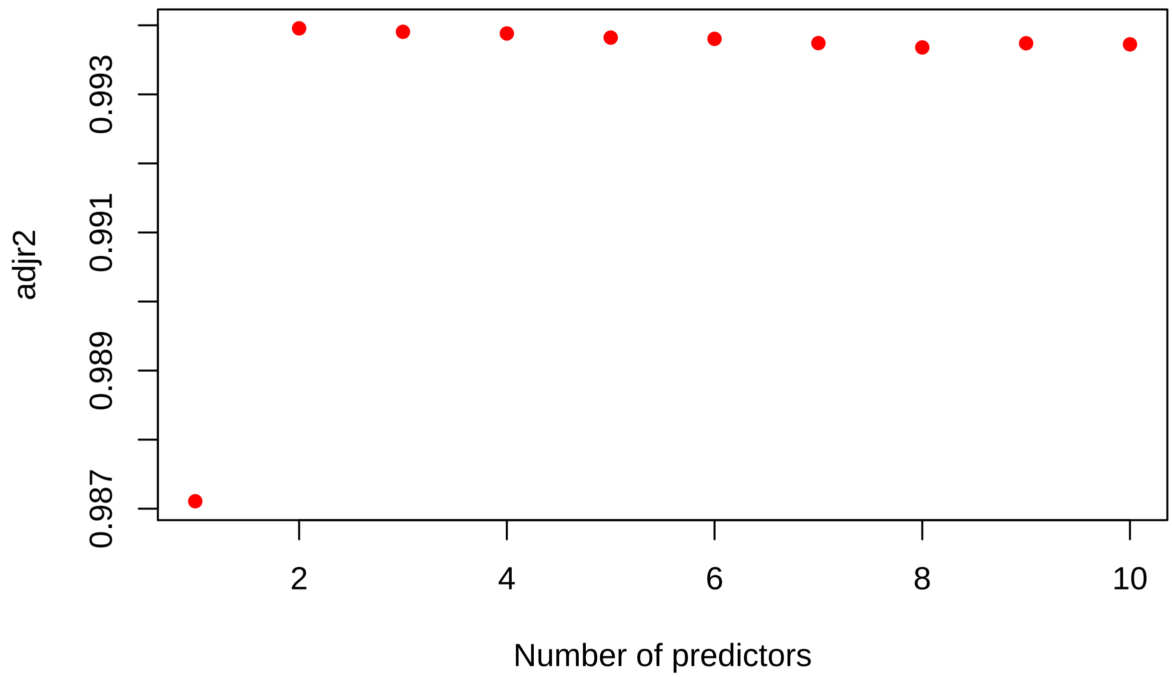
```r
plot(reg.sum.f$cp, xlab="Number of predictors", ylab="Cp", col="red", type="p", pch=16) # exhibit 4
```




```r
plot(reg.sum.f$bic, xlab="Number of predictors", ylab="BIC", col="red", type="p", pch=16) # exhibit 5
```




```
plot(reg.sum.f$adjr2, xlab="Number of predictors", ylab="adjr2", col="red", type="p", pch=16) # exhibit
```




```r
coef(regsub.f, id=2)
```

```
##                       (Intercept) poly(x, degree = 10, raw = TRUE)2
##                        -2.0710776                         0.9642645
## poly(x, degree = 10, raw = TRUE)3
##                         5.0163042
```

```
 ## (Intercept) poly(x, degree = 10, raw = TRUE)2 poly(x, degree = 10, raw = TRUE)3
 ## -2.0710776                         0.9642645                         5.0163042
```

For forward selection, we see the same reults as in part c. Now, we will do backward selection:

```r
regsub.b <- regsubsets(y~poly(x,degree=10,raw=TRUE), data=data, nvmax=10, method="backward")
reg.sum.b <- summary(regsub.b)

which.min(reg.sum.b$cp)       # 2
```

```
## [1] 2
```

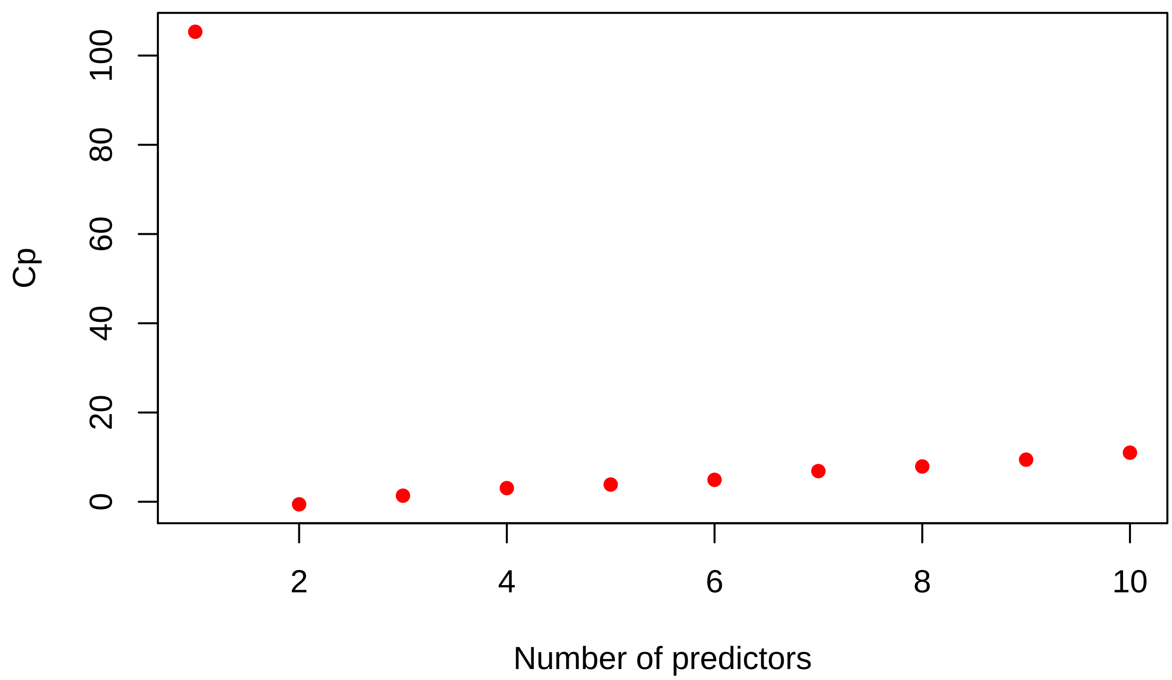
```r
which.min(reg.sum.b$bic)      # 2
```

```
## [1] 2
```

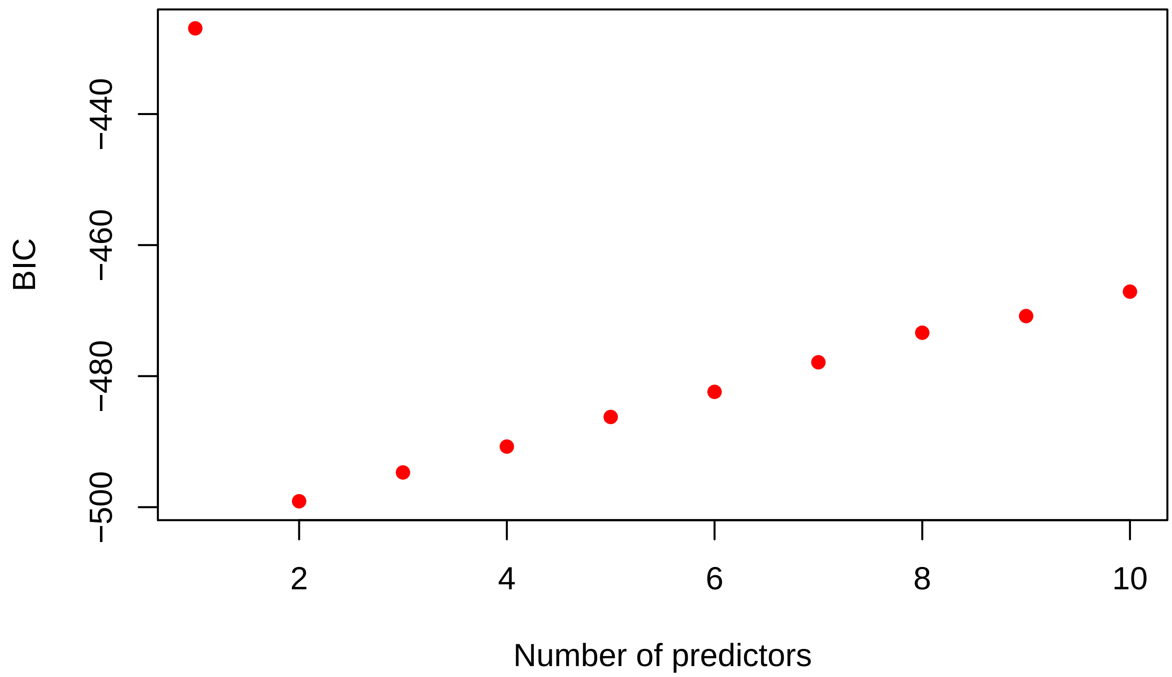
```r
which.max(reg.sum.b$adjr2)     # 2
```

```
## [1] 2
```

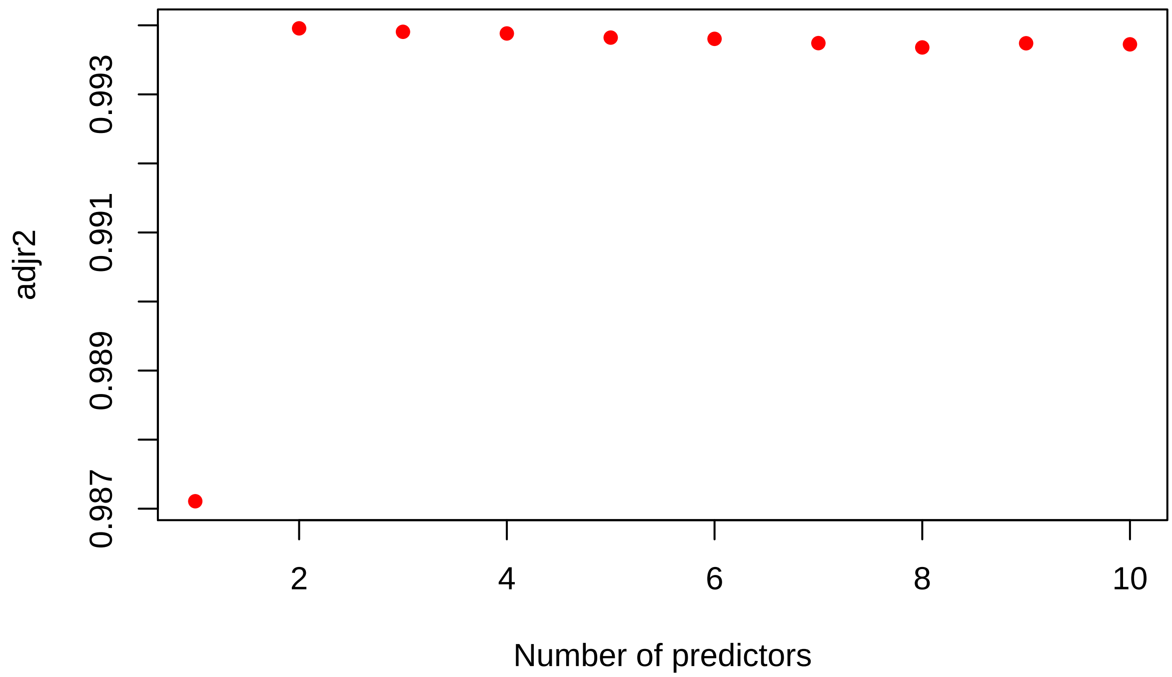
```r
plot(reg.sum.b$cp, xlab="Number of predictors", ylab="Cp", col="red", type="p", pch=16) # exhibit 7
```




```r
plot(reg.sum.f$bic, xlab="Number of predictors", ylab="BIC", col="red", type="p", pch=16) # exhibit 8
```




```
plot(reg.sum.f$adjr2, xlab="Number of predictors", ylab="adjr2", col="red", type="p", pch=16) # exhibit
```




```r
coef(regsub.b, id=2)
```

```
##                      (Intercept) poly(x, degree = 10, raw = TRUE)2
##                       -2.0710776                         0.9642645
## poly(x, degree = 10, raw = TRUE)3
##                        5.0163042
```

```
## (Intercept) poly(x, degree = 10, raw = TRUE)2 poly(x, degree = 10, raw = TRUE)3
## -2.0710776                         0.9642645                         5.0163042
```

Again, we see the same model and values.

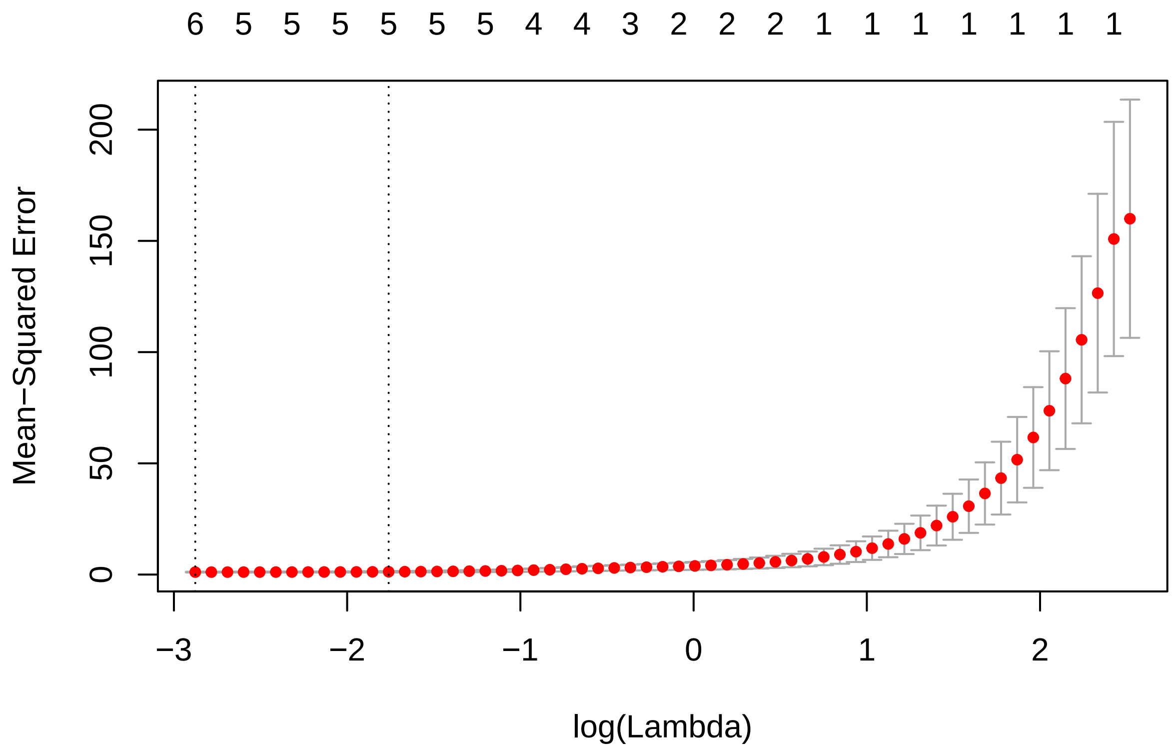
Part e

Fit a LASSO model and use cross-validation to select the optimal value for lambda.

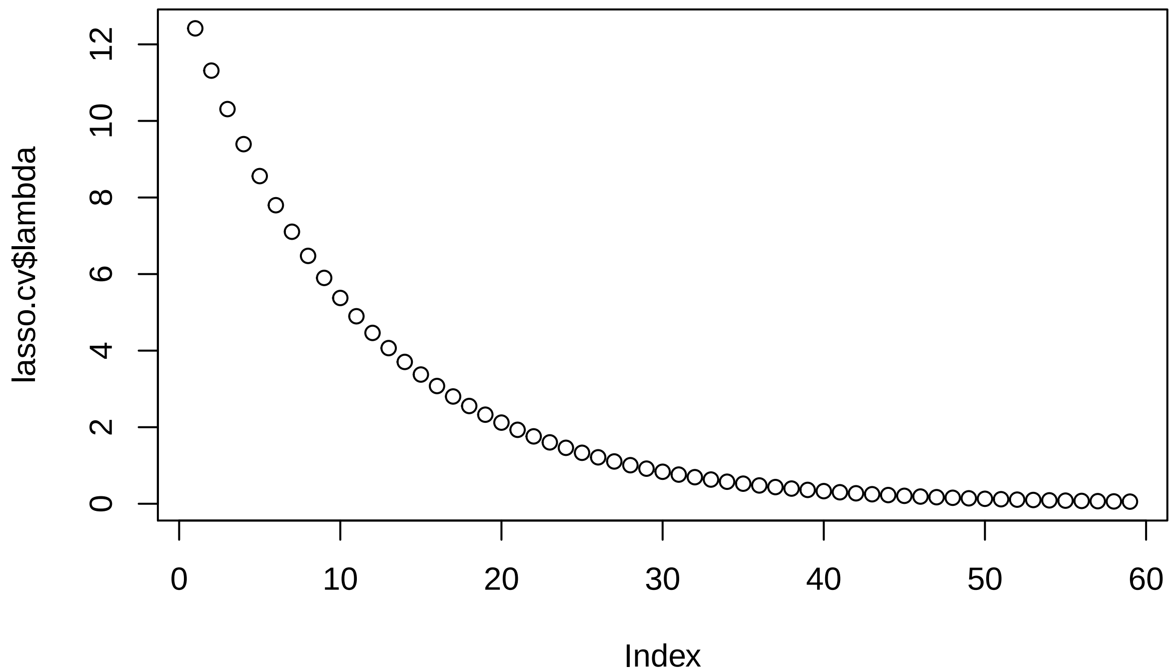
```r
library(glmnet) # for LASSO
```

```
## Loading required package: Matrix

## Loading required package: foreach

## Loaded glmnet 2.0-5
```

```r
X.lasso <- model.matrix(y~poly(x,degree=10,raw=TRUE), data=data)[,-1]
Y.lasso <- Y
lasso.cv <- cv.glmnet(X.lasso, Y.lasso, alpha=1, nfolds=10)
plot(lasso.cv) # exhibit 10
```




```r
plot(lasso.cv$lambda) # exhibit 11
```




```
lambda.final <- lasso.cv$lambda.min # 0.05631109

beta <- coef(lasso.cv, s=lambda.final)
beta <- as.matrix(beta)
beta
```

```
##                                             1
## (Intercept)                     -1.9625901145
## poly(x, degree = 10, raw = TRUE)1   0.0056439028
## poly(x, degree = 10, raw = TRUE)2   0.7397802716
## poly(x, degree = 10, raw = TRUE)3   4.9391488225
## poly(x, degree = 10, raw = TRUE)4   0.0361210858
## poly(x, degree = 10, raw = TRUE)5   0.0041259611
## poly(x, degree = 10, raw = TRUE)6   0.0000000000
## poly(x, degree = 10, raw = TRUE)7   0.0000000000
## poly(x, degree = 10, raw = TRUE)8   0.0000000000
## poly(x, degree = 10, raw = TRUE)9   0.0003331596
## poly(x, degree = 10, raw = TRUE)10  0.0000000000
```

```
 #                                             1
 #  (Intercept)                     -1.9625901145
 #  poly(x, degree = 10, raw = TRUE)1   0.0056439028
 #  poly(x, degree = 10, raw = TRUE)2   0.7397802716
 #  poly(x, degree = 10, raw = TRUE)3   4.9391488225
 #  poly(x, degree = 10, raw = TRUE)4   0.0361210858
```

```
# poly(x, degree = 10, raw = TRUE)5   0.0041259611
# poly(x, degree = 10, raw = TRUE)6   0.0000000000
# poly(x, degree = 10, raw = TRUE)7   0.0000000000
# poly(x, degree = 10, raw = TRUE)8   0.0000000000
# poly(x, degree = 10, raw = TRUE)9   0.0003331596
# poly(x, degree = 10, raw = TRUE)10  0.0000000000
```

Using the LASSO method, the model picks x, x^2, x^3, x^4, x^5, x^9. The coefficients fo x^9, x, and x^5 are more negligible than the others.

Part f

```
B7 <- 7
Y.f <- B0 + B7*X^7 + epsilon
data.f <- data.frame(y=Y.f, x=X)
```

Model selection using regsubsets:

```
regsub.partf <- regsubsets(y~poly(x,degree=10,raw=TRUE), data=data.f, nvmax=10)
reg.sum.partf <- summary(regsub.partf)

which.min(reg.sum.partf$cp)       # 1
```

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

```
which.min(reg.sum.partf$bic)      # 1
```

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

```
which.max(reg.sum.partf$adjr2)    # 1
```

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

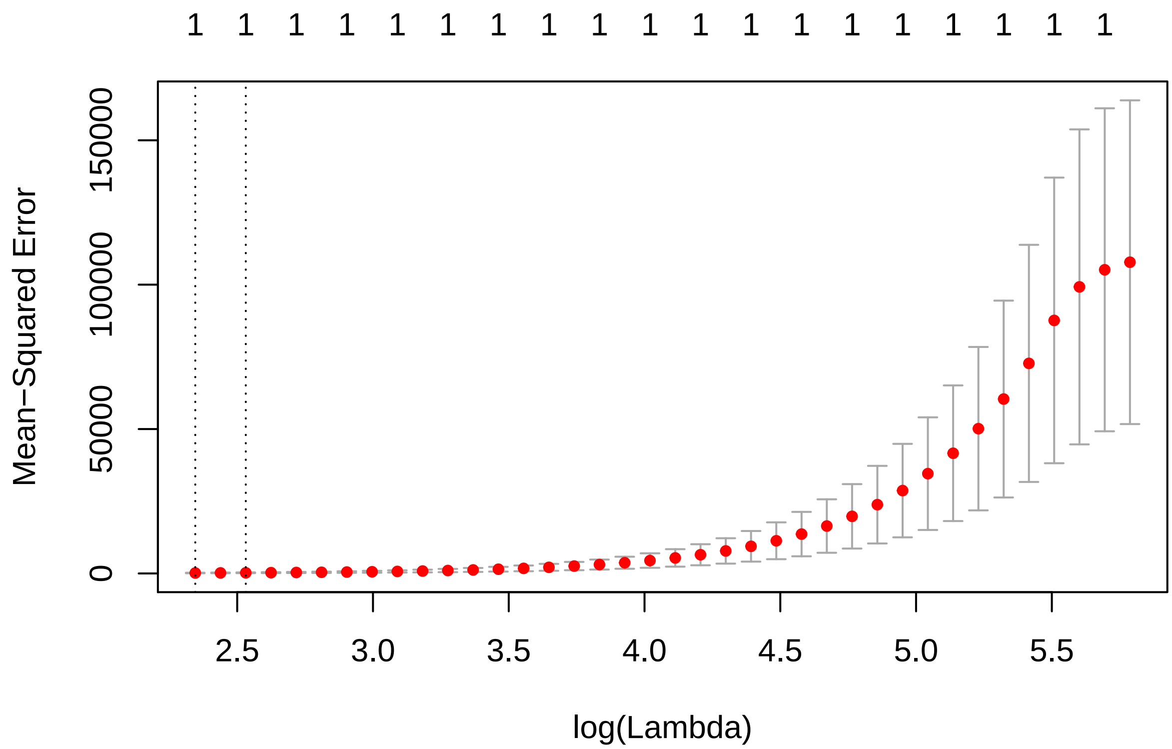
```
coef(regsub.partf, id=1)
```

```
##                        (Intercept) poly(x, degree = 10, raw = TRUE)7
##                          -2.095239                          6.999867
```

```
# (Intercept) poly(x, degree = 10, raw = TRUE)7
# -2.095239                         6.999867
```

All three methods chose one-variable models. Now, using LASSO:

```
X.lasso.f <- model.matrix(y~poly(x,degree=10,raw=TRUE), data=data.f)[,-1]
Y.lasso.f <- Y.f
lasso.cv.f <- cv.glmnet(X.lasso.f, Y.lasso.f, alpha=1, nfolds=10)
plot(lasso.cv.f) # exhibit 12
```




```r
lambda.final.f <- lasso.cv.f$lambda.min # 10.43878

beta <- coef(lasso.cv.f, s=lambda.final.f)
beta <- as.matrix(beta)
beta
```

```
##                                         1
## (Intercept)                     -2.561234
## poly(x, degree = 10, raw = TRUE)1   0.000000
## poly(x, degree = 10, raw = TRUE)2   0.000000
## poly(x, degree = 10, raw = TRUE)3   0.000000
## poly(x, degree = 10, raw = TRUE)4   0.000000
## poly(x, degree = 10, raw = TRUE)5   0.000000
## poly(x, degree = 10, raw = TRUE)6   0.000000
## poly(x, degree = 10, raw = TRUE)7   6.775923
## poly(x, degree = 10, raw = TRUE)8   0.000000
## poly(x, degree = 10, raw = TRUE)9   0.000000
## poly(x, degree = 10, raw = TRUE)10  0.000000
```

```r
 #                                         1
 # (Intercept)                     -2.561234
 # poly(x, degree = 10, raw = TRUE)1   0.000000
 # poly(x, degree = 10, raw = TRUE)2   0.000000
 # poly(x, degree = 10, raw = TRUE)3   0.000000
 # poly(x, degree = 10, raw = TRUE)4   0.000000
```

```
# poly(x, degree = 10, raw = TRUE)5   0.000000
# poly(x, degree = 10, raw = TRUE)6   0.000000
# poly(x, degree = 10, raw = TRUE)7   6.775923
# poly(x, degree = 10, raw = TRUE)8   0.000000
# poly(x, degree = 10, raw = TRUE)9   0.000000
# poly(x, degree = 10, raw = TRUE)10  0.000000
```

Using LASSO, the model selected is also a one-variable model.

Problem 2