rm4064

Ronan McNally

2024-10-05

R Markdown

Problem 1

HW #2, Prob 1,

Ridge regression...

$$\text{minimize} \quad RSS + \lambda \sum_{j=1}^{p} \hat{\beta}_j^2$$

$$\text{minimize:} \sum_{i=1}^{n} \left(y_i - \hat{\beta}_0 - \sum_{j=1}^{p} \hat{\beta}_j x_{ij}\right)^2 + \lambda \sum_{j=1}^{p} \hat{\beta}_j^2$$

$n=2$
$p=2$

$$\left[y_1 - \hat{\beta}_0 - (\hat{\beta}_1 x_{11} + \hat{\beta}_2 x_{12})\right]^2 + \left[y_2 - \hat{\beta}_0 - (\hat{\beta}_1 x_{21} + \hat{\beta}_2 x_{22})\right]^2 + \lambda\left(\hat{\beta}_1^2 + \hat{\beta}_2^2\right)$$

$x_{11} = x_{12} = 'x_1'$
$x_{21} = x_{22} = 'x_2'$
$y_1 + y_2 = 0$
$\left.\begin{matrix} x_{11} + x_{21} = 0 \\ x_{12} + x_{22} = 0 \end{matrix}\right\} \to x_1 + x$
$\hat{\beta}_0 = 0$

$$\left[y_1 - \cancel{\hat{\beta}_0}^{\,0} - x_1(\hat{\beta}_1 + \hat{\beta}_2)\right]^2 + \left[y_2 - \cancel{\hat{\beta}_0}^{\,0} - x_2(\hat{\beta}_1 + \hat{\beta}_2)\right]^2 + \lambda(\hat{\beta}_1^2 + \hat{\beta}_2^2)$$

$$\boxed{(a) \min_{\beta}\left[\left[y_1 - x_1(\hat{\beta}_1 + \hat{\beta}_2)\right]^2 + \left[y_2 - x_2(\hat{\beta}_1 + \hat{\beta}_2)\right]^2 + \lambda(\hat{\beta}_1^2 + \hat{\beta}_2^2)\right]}$$

(b)

$$\frac{\partial}{\partial \hat{\beta}_1}(\text{ridge expression}) = \underbrace{2\left[y_1 - x_1(\hat{\beta}_1 + \hat{\beta}_2)\right][-x_1] + 2\left[y_2 - x_2(\hat{\beta}_1 + \hat{\beta}_2)\right][-x_2]}_{\text{refer to as 'Q'}} + 2\lambda\hat{\beta}_1 = 0$$

$$\frac{\partial}{\partial \hat{\beta}_2}(\text{ridge expression}) = \overbrace{2\left[y_1 - x_1(\hat{\beta}_1 + \hat{\beta}_2)\right][-x_1] + 2\left[y_2 - x_2(\hat{\beta}_1 + \hat{\beta}_2)\right][-x_2]} + 2\lambda\hat{\beta}_2 = 0$$

$$\left.\begin{matrix} Q + 2\lambda\hat{\beta}_1 = 0 \longrightarrow \hat{\beta}_1 = \dfrac{-Q}{2\lambda} \\ Q + 2\lambda\hat{\beta}_2 = 0 \longrightarrow \hat{\beta}_2 = \dfrac{-Q}{2\lambda} \end{matrix}\right\} \text{equivalent}, \ \therefore \hat{\beta}_1 = \hat{\beta}_2$$

(c) Lasso optimization

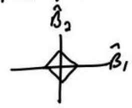
$$\text{minimize} \quad RSS + \lambda \sum_{j=1}^{p} |\beta_j|$$

$$\text{minimize:} \sum_{i=1}^{n} \left(y_i - \hat{\beta}_0 - \sum_{j=1}^{p} \hat{\beta}_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} |\hat{\beta}_j|$$

$$\text{minimize:} [y_1 - x_1(\hat{\beta}_1 + \hat{\beta}_2)]^2 + [y_2 - x_2(\hat{\beta}_1 + \hat{\beta}_2)]^2 + \lambda [|\hat{\beta}_1| + |\hat{\beta}_2|]$$

(d)

minimizing penalty is equivalent to $|\hat{\beta}_1| + |\hat{\beta}_2| \leq s$ as seen in class.



minimizing RSS term...

$$\min: [y_1 - x_1(\hat{\beta}_1 + \hat{\beta}_2)]^2 + [y_2 - x_2(\hat{\beta}_1 + \hat{\beta}_2)]^2$$

$y_1 + y_2 = 0$
$y_1 = -y_2$
$x_1 + x_2 = 0$
$x_1 = -x_2$

$$\equiv [y_1 - x_1(\hat{\beta}_1 + \hat{\beta}_2)]^2 + [-y_1 + x_1(\hat{\beta}_1 + \hat{\beta}_2)]^2$$

$$\equiv [y_1 - x_1(\hat{\beta}_1 + \hat{\beta}_2)]^2 + [(-1)(y_1 - x_1(\hat{\beta}_1 + \hat{\beta}_2))]^2$$

$$\equiv [y_1 - x_1(\hat{\beta}_1 + \hat{\beta}_2)]^2 + [y_1 - x_1(\hat{\beta}_1 + \hat{\beta}_2)]^2$$

$$\equiv 2[y_1 - x_1(\hat{\beta}_1 + \hat{\beta}_2)]^2$$

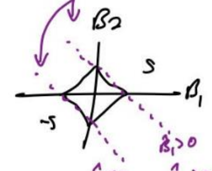
minimized when $x_1(\hat{\beta}_1 + \hat{\beta}_2) = y_1$

$$\therefore \hat{\beta}_1 + \hat{\beta}_2 = \frac{y_1}{x_1}$$

$\hat{\beta}_2 = -\hat{\beta}_1 + \frac{y_1}{x_1}$

negative slope,

Solution lines



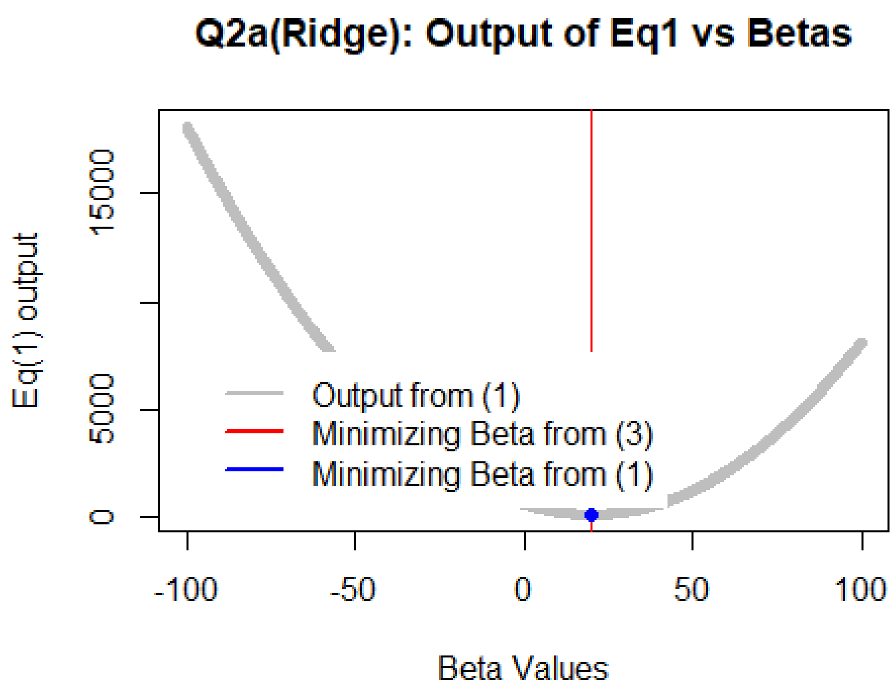
Comment:

For this minimization, there is a set of $\{\hat{\beta}_1, \hat{\beta}_2\}$ for which their sum is $\hat{\beta}_1 + \hat{\beta}_2 \leq s$ & $\hat{\beta}_1 + \hat{\beta}_2 = \frac{y_1}{x_1}$. This is effectively being on the edge of the lasso diamond for which there exists a line of points that satisfy the conditions found here. $\therefore \{\beta_1, \beta_2\}$ solution is NOT unique

Problem 2

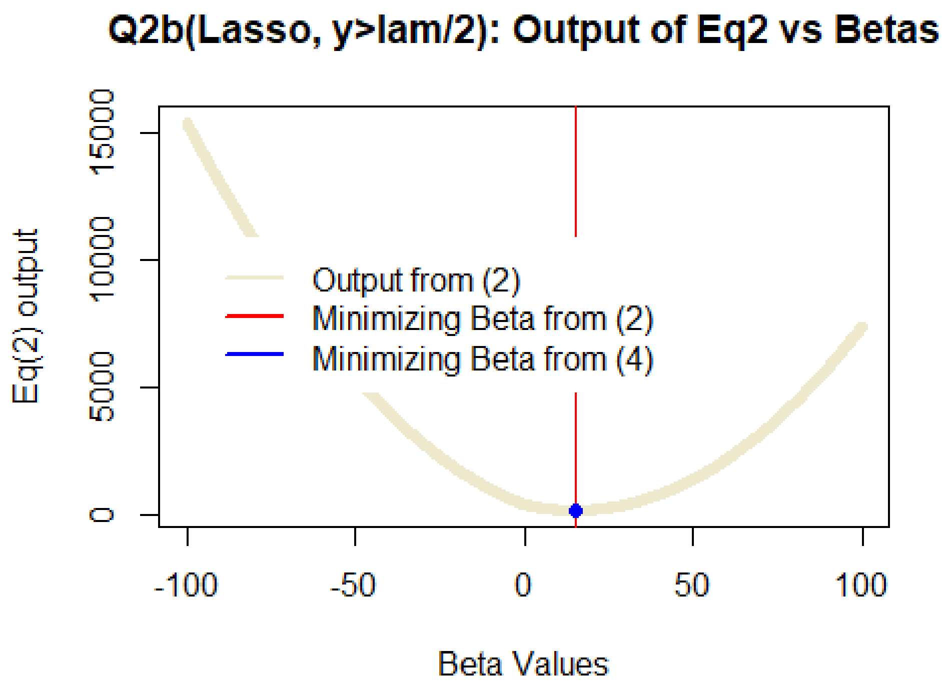
```r
# P2 PART A

lambda.p2 <- 0.25
y1.p2 <- 25
Beta.p2 <- seq(-100, 100, by=.01)
plot(Beta.p2,((1+lambda.p2)*Beta.p2^2)-(2*y1.p2*Beta.p2)+(y1.p2^2), pch=20,
col="grey", xlab = "Beta Values", ylab = "Eq(1) output", main = "Q2a(Ridge):
Output of Eq1 vs Betas")

print("Question 2 PART A:")

## [1] "Question 2 PART A:"

print("Corresponding Beta value for minimum of ridge regression function
(1)")

## [1] "Corresponding Beta value for minimum of ridge regression function
(1)"

Beta.p2[which.min(((1+lambda.p2)*Beta.p2^2)-(2*y1.p2*Beta.p2)+(y1.p2^2))]

## [1] 20

Bmin.exp1 <- Beta.p2[which.min(((1+lambda.p2)*Beta.p2^2)-
(2*y1.p2*Beta.p2)+(y1.p2^2))]

print("Beta value calculated through (3)")

## [1] "Beta value calculated through (3)"

y1.p2/(1+lambda.p2)

## [1] 20

Bmin.exp3 <- y1.p2/(1+lambda.p2)

print("Are the two equal?")

## [1] "Are the two equal?"

setequal(Bmin.exp1, Bmin.exp3)

## [1] TRUE

abline(v=y1.p2/(1+lambda.p2), col="red")
points(Bmin.exp1, ((1+lambda.p2)*Bmin.exp1^2)-(2*y1.p2*Bmin.exp1)+(y1.p2^2),
col="blue", pch=19)

legend("bottomleft", inset=.05,c("Output from (1)","Minimizing Beta from
```

```
(3)","Minimizing Beta from (1)"), lwd=2, lty=c(1, 1, 1),
col=c("grey","red","blue"), box.lty=0)
```



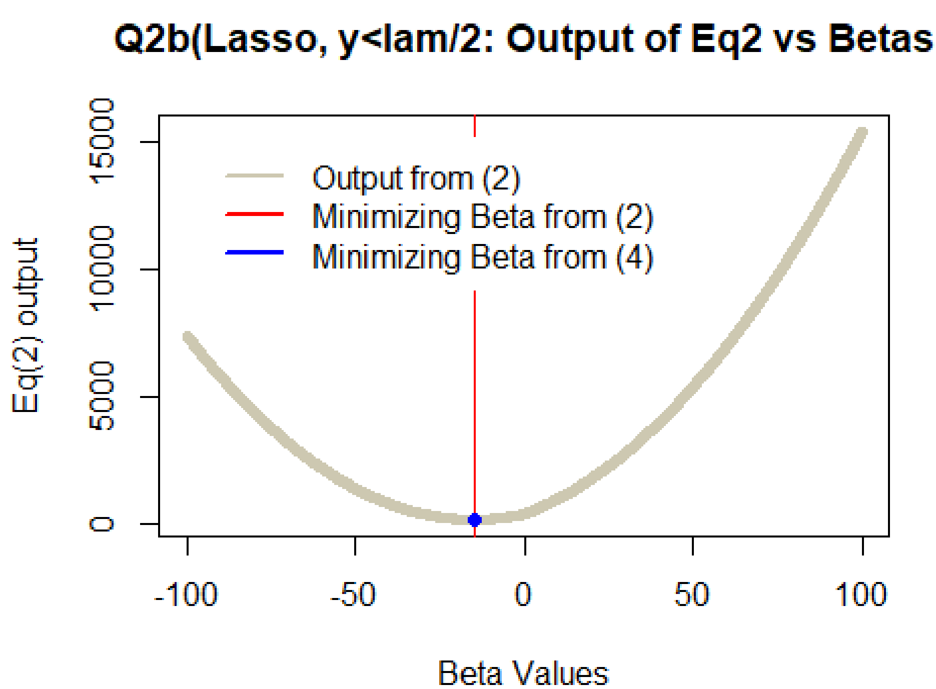

```
# P2 PART B
print("Question 2 PART A:")

## [1] "Question 2 PART A:"

lambda.p2b <- 10

y1.p2b.op1 <- 20
y1.p2b.op2 <- -20
y1.p2b.op3 <- 1


# OPTION 1: y>lambda/2
Beta.p2 <- seq(-100, 100, by=.01)

plot(Beta.p2,Beta.p2^2 - 2*y1.p2b.op1*Beta.p2 + y1.p2b.op1^2 +
lambda.p2b*abs(Beta.p2), pch=20, col="cornsilk2", xlab = "Beta Values", ylab
= "Eq(2) output", main = "Q2b(Lasso, y>lam/2): Output of Eq2 vs Betas")


print("Corresponding Beta value for minimum of lasso regression function (2)
when y>lam/2")
```

```
## [1] "Corresponding Beta value for minimum of lasso regression function (2)
when y>lam/2"

Beta.p2[which.min(Beta.p2^2 - 2*y1.p2b.op1*Beta.p2 + y1.p2b.op1^2 +
lambda.p2b*abs(Beta.p2))]

## [1] 15

Bmin.exp2.p2.op1 <- Beta.p2[which.min(Beta.p2^2 - 2*y1.p2b.op1*Beta.p2 +
y1.p2b.op1^2 + lambda.p2b*abs(Beta.p2))]

print("Beta value calculated through (4), y>lam/2")

## [1] "Beta value calculated through (4), y>lam/2"

y1.p2b.op1 - (lambda.p2b/2)

## [1] 15

Bmin.exp4.p2.op1 <- y1.p2b.op1-(lambda.p2b/2)

print("Are the two equal?")

## [1] "Are the two equal?"

setequal(Bmin.exp2.p2.op1, Bmin.exp4.p2.op1)

## [1] TRUE

abline(v=Bmin.exp4.p2.op1, col="red")
points(Bmin.exp2.p2.op1, Bmin.exp2.p2.op1^2 - 2*y1.p2b.op1*Bmin.exp2.p2.op1 +
y1.p2b.op1^2 + lambda.p2b*abs(Bmin.exp2.p2.op1), col="blue", pch=19)

legend("left", inset=.05,c("Output from (2)","Minimizing Beta from
(2)","Minimizing Beta from (4)"), lwd=2, lty=c(1, 1, 1),
col=c("cornsilk2","red","blue"), box.lty=0)
```




```
# OPTION 2: y<lambda/2

plot(Beta.p2,Beta.p2^2 - 2*y1.p2b.op2*Beta.p2 + y1.p2b.op2^2 +
lambda.p2b*abs(Beta.p2), pch=20, col="cornsilk3", xlab = "Beta Values", ylab
= "Eq(2) output", main = "Q2b(Lasso, y<lam/2: Output of Eq2 vs Betas")


print("Corresponding Beta value for minimum of lasso regression function (2)
when y<lam/2")

## [1] "Corresponding Beta value for minimum of lasso regression function (2)
when y<lam/2"

Beta.p2[which.min(Beta.p2^2 - 2*y1.p2b.op2*Beta.p2 + y1.p2b.op2^2 +
lambda.p2b*abs(Beta.p2))]

## [1] -15

Bmin.exp2.p2.op2 <- Beta.p2[which.min(Beta.p2^2 - 2*y1.p2b.op2*Beta.p2 +
y1.p2b.op2^2 + lambda.p2b*abs(Beta.p2))]

print("Beta value calculated through (4), y<lam/2")

## [1] "Beta value calculated through (4), y<lam/2"

y1.p2b.op2 + (lambda.p2b/2)

## [1] -15
```

```r
Bmin.exp4.p2.op2 <- y1.p2b.op2+(lambda.p2b/2)

print("Are the two equal?")

## [1] "Are the two equal?"

setequal(Bmin.exp2.p2.op2, Bmin.exp4.p2.op2)

## [1] TRUE

abline(v=Bmin.exp4.p2.op2, col="red")
points(Bmin.exp2.p2.op2, Bmin.exp2.p2.op2^2 - 2*y1.p2b.op2*Bmin.exp2.p2.op2 +
y1.p2b.op2^2 + lambda.p2b*abs(Bmin.exp2.p2.op2), col="blue", pch=19)

legend("topleft", inset=.05,c("Output from (2)","Minimizing Beta from
(2)","Minimizing Beta from (4)"), lwd=2, lty=c(1, 1, 1),
col=c("cornsilk3","red","blue"), box.lty=0)
```




```r
# OPTION 3: |y|(<or=)lambda/2

plot(Beta.p2,Beta.p2^2 - 2*y1.p2b.op3*Beta.p2 + y1.p2b.op3^2 +
lambda.p2b*abs(Beta.p2), pch=20, col="cornsilk4", xlab = "Beta Values", ylab
= "Eq(2) output", main = "Q2b(Lasso, |y|(<or=)lam/2: Output of Eq2 vs Betas")


print("Corresponding Beta value for minimum of lasso regression function (2)
when |y|(<or=)lam/2")
```

```
## [1] "Corresponding Beta value for minimum of lasso regression function (2)
when |y|(<or=)lam/2"
```

```
Beta.p2[which.min(Beta.p2^2 - 2*y1.p2b.op3*Beta.p2 + y1.p2b.op3^2 +
lambda.p2b*abs(Beta.p2))]
```

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

```
Bmin.exp2.p2.op3 <- Beta.p2[which.min(Beta.p2^2 - 2*y1.p2b.op3*Beta.p2 +
y1.p2b.op3^2 + lambda.p2b*abs(Beta.p2))]

print("Beta value calculated through (4), |y|(<or=)lam/2")
```

```
## [1] "Beta value calculated through (4), |y|(<or=)lam/2"
```

```
0
```

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

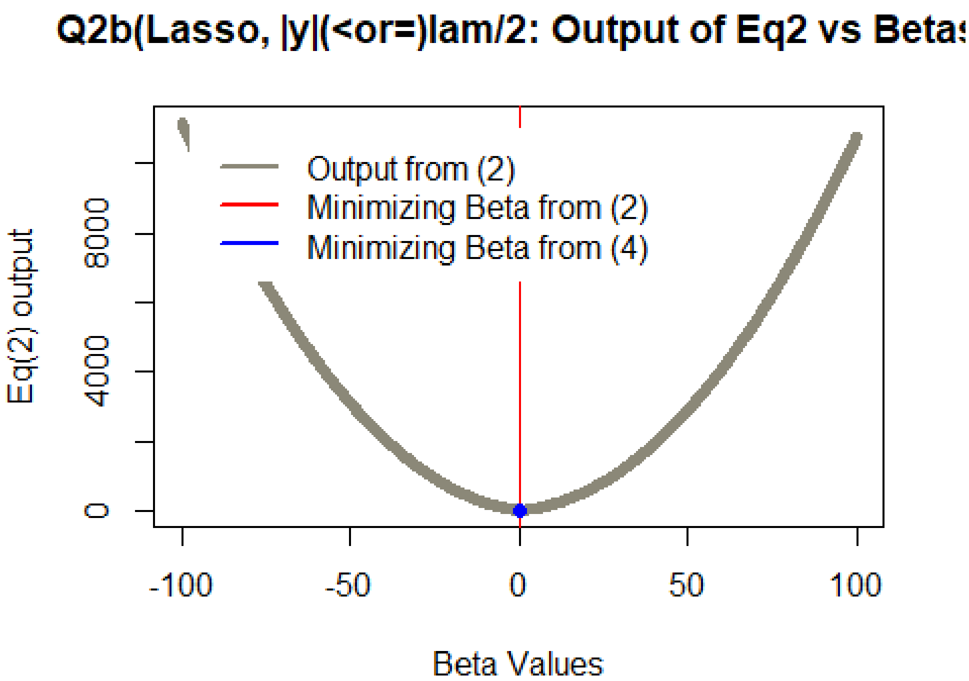
```
Bmin.exp4.p2.op3 <- 0

print("Are the two equal?")
```

```
## [1] "Are the two equal?"
```

```
setequal(Bmin.exp2.p2.op3, Bmin.exp4.p2.op3)
```

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

```
abline(v=Bmin.exp4.p2.op3, col="red")
points(Bmin.exp2.p2.op3, Bmin.exp2.p2.op3^2 - 2*y1.p2b.op3*Bmin.exp2.p2.op3 +
y1.p2b.op3^2 + lambda.p2b*abs(Bmin.exp2.p2.op3), col="blue", pch=19)

legend("topleft", inset=.05,c("Output from (2)","Minimizing Beta from
(2)","Minimizing Beta from (4)"), lwd=2, lty=c(1, 1, 1),
col=c("cornsilk4","red","blue"), box.lty=0)
```


Q2b(Lasso, |y|(<or=)lam/2: Output of Eq2 vs Betas
Output from (2)
Minimizing Beta from (2)
Minimizing Beta from (4)
Eq(2) output
0
4000
8000
-100
-50
0
50
100
Beta Values

Problem 3

Ronan McNally

HW #2

P3

(a) Write out the likelihood of $y_i = \beta_0 + \sum_{j=1}^{p} x_{ij}\beta_j + \epsilon_i$ (ϵ_i is i.i.d. & normally distributed)

$\epsilon_i = y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j$

$$L(Y|X,\beta) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} e^{\left(-\frac{(y_i - \beta_0 \sum_{j=1}^{p} x_{ij}\beta_j)^2}{2\sigma^2}\right)} = \left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n e^{-\frac{1}{2\sigma^2}\left(\sum_{i=1}^{n}(y_i - \beta_0 + \sum_{j=1}^{p} x_{ij}\beta_j)\right)} = \left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n e^{-\frac{1}{2\sigma^2}\left(\sum_{i=1}^{n}(\epsilon_i)^2\right)}$$

$e^a e^b e^c = e^{a+b+c}$

$$\rightarrow \boxed{L(Y|X,\beta) = \left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n e^{-\frac{1}{2\sigma^2}\left(\sum_{i=1}^{n}(\epsilon_i)^2\right)}}$$

(b)

posterior $\propto$ (likelihood)(prior)

$$p(\beta|X,Y) \propto L(Y|X,\beta)\, p(\beta|X) = L(Y|X,\beta)\, p(\beta)$$

$$p(\beta) = \frac{1}{2b} e^{\left(\frac{-|\beta|}{b}\right)}$$

$$\boxed{p(\beta|X,Y) \propto L(Y|X,\beta)\, p(\beta) = \left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n e^{\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\epsilon_i^2\right)} \left[\frac{1}{2b} e^{\left(-\frac{|\beta|}{b}\right)}\right]}$$

(c) The mode of any set is the value that shows up most commonly.

In a Bayesian context...

posterior $\propto$ likelihood $\times$ prior

posterior: corresponding conclusions we can draw

likelihood: data we have

prior: beliefs about that data

For linear regression $y_i = \beta_0 + \sum_{j=1}^{p} x_{ij}\beta_j + \epsilon_i$, this is the...

probability of a set of Betas $\propto$ probability of ys for βs & xs we have $\times$ belief about distribution of βs

We've already found likelihood & assumed a double exponential prior for β.

Maximizing the posterior distribution will give the most likely values for β.

Problem 3 part C (continued)

If we can show maximizing the posterior distribution results in the expression for LASSO regression, then we have shown the LASSO estimate is the mode for β under the posterior distribution . . .

$$\max_{\beta}\left(P(\beta|X,Y)\right) \equiv \max_{\beta}\left(L(Y|X,\beta)\,p(\beta)\right)$$

$$\equiv \max_{\beta}\left(\left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n e^{\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\epsilon_i^2\right)}\left[\frac{1}{2b}e^{\left(-\frac{|\beta|}{b}\right)}\right]\right) \equiv \max_{\beta}\left(\left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n\left(\frac{1}{2b}\right)e^{\left[-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\epsilon_i^2-\frac{|\beta|}{b}\right]}\right)$$

take $\ln() \rightarrow (\max(A) \propto \max(\ln(A)) \rightarrow$

$$\equiv \max_{\beta}\left[\underbrace{\ln\left[\left(\frac{1}{\sigma\sqrt{2\pi}}\right)\left(\frac{1}{2b}\right)\right]}_{\text{some constant value } C} - \underbrace{\frac{1}{2\sigma^2}\sum_{i=1}^{n}\epsilon_i^2 - \frac{|\beta|}{b}}_{\text{minimizing this maximizes overall expression}}\right] \equiv \min_{\beta}\left[\frac{1}{2\sigma^2}\sum_{i=1}^{n}\epsilon_i^2-\frac{|\beta|}{b}\right]$$

$$\min_{\beta}\left[\frac{1}{2\sigma^2}\sum_{i=1}^{n}\epsilon_i^2 - \frac{|\beta|}{b}\right] \equiv \min_{\beta}\frac{1}{2\sigma^2}\left[\underbrace{\sum_{i=1}^{n}\epsilon_i^2}_{\text{equivalent to RSS}} - \overset{\text{rewrite as penalty term } \lambda}{\frac{2\sigma^2}{b}}\sum_{j=1}^{p}|\beta_j|\right] \equiv \min_{\beta}\left[RSS - \lambda\sum_{j=1}^{p}|\beta_j|\right]$$

$$\lambda = \frac{2\sigma^2}{b}$$

- we know $|\beta| = \sum_{j=1}^{p}|\beta_j|$

and THIS is our LASSO estimate reflecting mode for β.

(d) Assuming β distribution is normal...

$$p(\beta \mid Y, X) \propto \underbrace{p(Y \mid X, \beta)}_{\text{already have}} \underbrace{p(\beta)}_{\text{for a multivariate gaussian distribution}}$$

for a <u>multivariate</u> gaussian distribution

$$p(\beta) = \prod_{i=1}^{n} \frac{1}{\sigma\sqrt{2\pi}} e^{[(\beta_i - \mu_i)^2 / 2\sigma^2]}$$

$$= \left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n e^{\sum_{i=1}^{n} \frac{(\beta_i - \mu_i)^2}{2\sigma^2}}$$

$\mu_j = 0$

$$= \left[\left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n e^{\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n} \epsilon_i^2\right)}\right]\left[\left(\frac{1}{\sigma\sqrt{2\pi}}\right)^p e^{\sum_{j=1}^{p}\left(-\frac{(\beta_j - \mu_j)^2}{2\sigma^2}\right)}\right]$$

c c c c

$$\boxed{= \left[\left(\frac{1}{c\sqrt{2\pi}}\right)^n e^{\left(-\frac{1}{2c^2}\sum_{i=1}^{n} \epsilon_i^2\right)}\right]\left[\left(\frac{1}{c\sqrt{2\pi}}\right)^p e^{\left(\sum_{j=1}^{p}\left(-\frac{\beta_j^2}{2c^2}\right)\right)}\right]}$$

posterior

(e) for confirming the mode for ridge, similar process as before...

$$\max_\beta \left[p(\beta \mid Y, X) \right] \equiv \max_\beta \left[p(Y \mid X, \beta)\, p(\beta) \right] \equiv \max_\beta \left[\left[\left(\frac{1}{c\sqrt{2\pi}} \right)^n e^{\left(-\frac{1}{2c^2} \sum_{i=1}^n \epsilon_i^2 \right)} \right] \left[\left(\frac{1}{c\sqrt{2\pi}} \right)^p e^{\sum_{j=1}^p \left(-\frac{\beta_j^2}{2c^2} \right)} \right] \right]$$

combine exp()
& apply ln()

$$\equiv \max_\beta \left[\ln\left[\left(\frac{1}{c\sqrt{2\pi}} \right)^{n+p} \right] - \underbrace{\left[\frac{1}{2c^2} \sum_{i=1}^n \epsilon_i^2 + \sum_{j=1}^p \left(\frac{\beta_j^2}{2c^2} \right) \right]}_{\text{equiv. to minimizing this}} \right] \equiv \min_\beta \frac{1}{2c^2} \left[\overbrace{\sum_{i=1}^n \epsilon_i^2}^{RSS} + \sum_{j=1}^p \frac{\beta_j^2}{2c^2} \right] \equiv \min_\beta \left[RSS + \underbrace{\frac{1}{2c}}_{\text{make } \lambda = \frac{1}{2c}} \sum_{j=1}^p \beta_j^2 \right]$$

$$\equiv \min_\beta \underbrace{\left[RSS + \lambda \sum_{j=1}^p \beta_j^2 \right]}_{\text{eq. for ridge regression}}$$

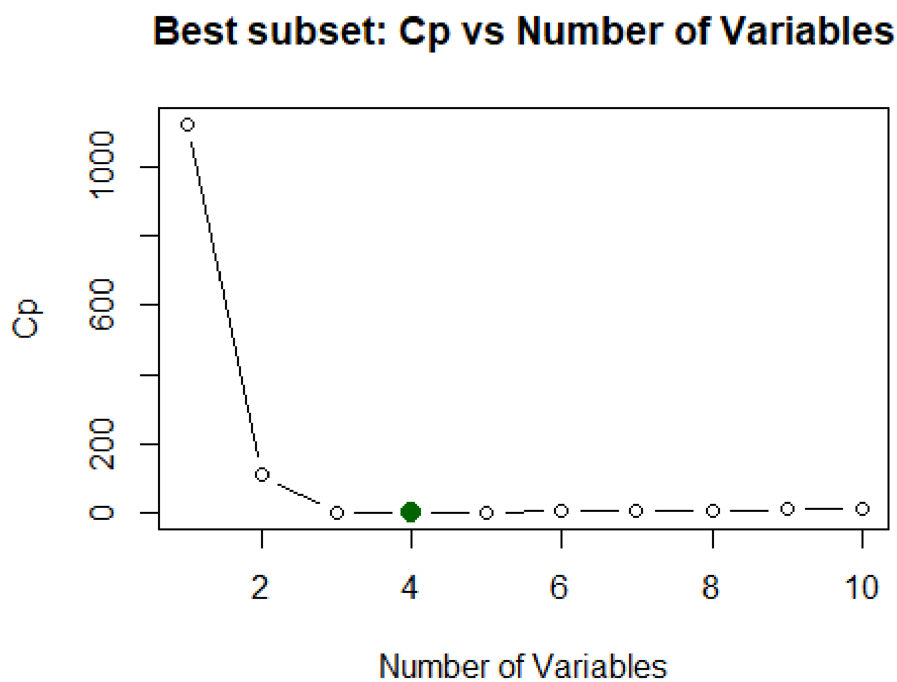
eq. for ridge regression $\longrightarrow$ good estimate for β mode and given a normal distribution, the mean & mode are equivalent.

Problem 4

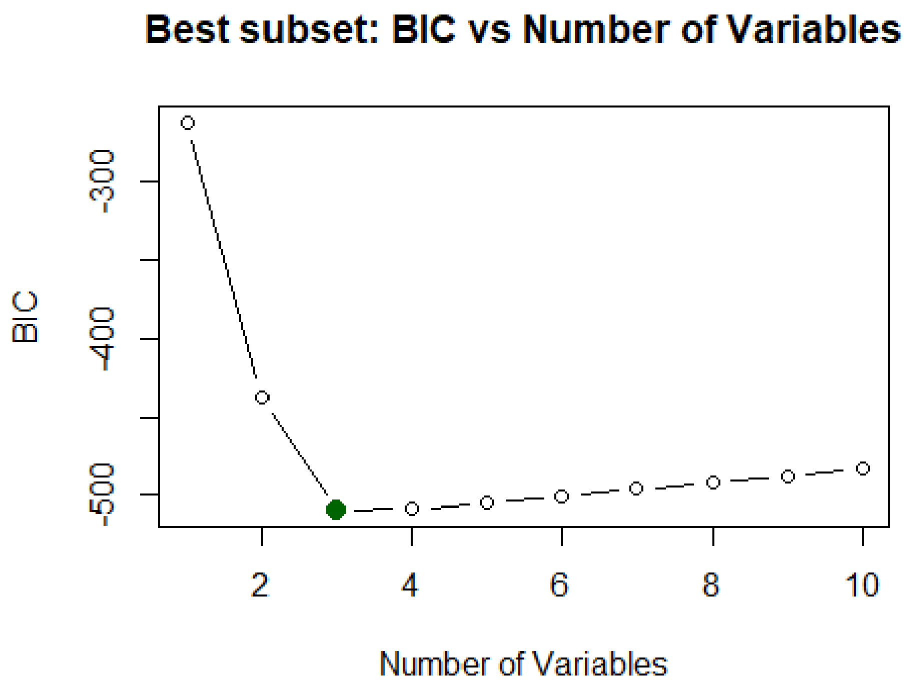
```
# P4 PART A

set.seed(1)

# length of vectors
n <- 100

# Generate predictor X of length n=100 as well as a noise vector eps of
length n=100

X <- rnorm(n, mean = 0, sd = 1)

eps <- rnorm(n, mean = 0, sd = 1)


# P4 PART B: Generate a response vector Y of length n=100 according to the
model shown in HW2 doc, where B0, B1, B2, and B3 are constant of your choice.

B0 <- 1
B1 <- 2
B2 <- 3
B3 <- 4

Y <- B0 + B1*X + B2*(X^2) + B3*(X^3) + eps


# P4 PART C: Use
#install.package("leaps")
library(leaps)
library(ggplot2)

X1 <- X^1
X2 <- X^2
X3 <- X^3
X4 <- X^4
X5 <- X^5
X6 <- X^6
X7 <- X^7
X8 <- X^8
X9 <- X^9
X10 <- X^10

writeLines("-----------------------------------------------------------------
------------")
```

```r
## ---------------------------------------------------------------------------
---

writeLines("\n BEST SUBSET SELECTION \n")

##
##  BEST SUBSET SELECTION

data <- data.frame(X, Y)
#data <- data.frame(Y, X1, X2, X3, X4, X5, X6, X7, X8, X9, X10)

subset_fits <- regsubsets(Y ~ X + X2 + X3 + X4 + X5 + X6 + X7 + X8 + X9 +
X10, data = data, nvmax=10)
#subset_fits <- regsubsets(Y ~ poly(X, 10, raw = TRUE), data = data,
nvmax=10)


regfits_summary <- summary(subset_fits)

writeLines("Best Selection Coefficients (Cp, BIC, R2 respectively):")

## Best Selection Coefficients (Cp, BIC, R2 respectively):

coef(subset_fits, which.min(regfits_summary$cp))

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

writeLines("\n")

coef(subset_fits, which.min(regfits_summary$bic))

## (Intercept)           X          X2          X3
##    1.061507    1.975280    2.876209    4.017639

writeLines("\n")

coef(subset_fits, which.max(regfits_summary$adjr2))

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

writeLines("\n")

regfits_summary$cp

##  [1] 1123.2892318  109.3256041    2.1859433    0.6067483    2.1782005
##  [6]    3.9955812    5.7869063    7.1694092    9.1535580   11.0000000

writeLines(paste("Lowest Cp model number: ", which.min(regfits_summary$cp)))

## Lowest Cp model number:  4
```

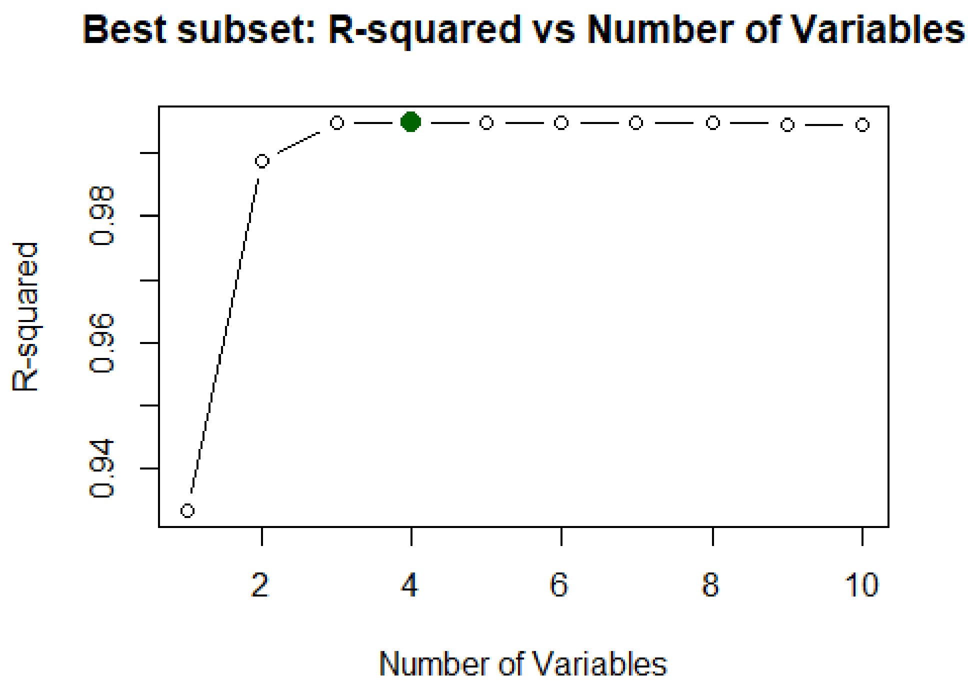
```r
plot(regfits_summary$cp, xlab = "Number of Variables", ylab = "Cp", main =
"Best subset: Cp vs Number of Variables", type = "b")
points(which.min(regfits_summary$cp),
regfits_summary$cp[which.min(regfits_summary$cp)], col="darkgreen", cex = 2,
pch = 20)
```




```r
regfits_summary$bic

##  [1] -262.7744 -437.2907 -509.6393 -508.9084 -504.7773 -500.3748 -496.0018
##  [8] -492.0868 -487.4994 -483.0666

writeLines(paste("Lowest BIC model: ", which.min(regfits_summary$bic)))

## Lowest BIC model:  3

plot(regfits_summary$bic, xlab = "Number of Variables", ylab = "BIC", main =
"Best subset: BIC vs Number of Variables", type = "b")
points(which.min(regfits_summary$bic),
regfits_summary$bic[which.min(regfits_summary$bic)], col="darkgreen", cex =
2, pch = 20)
```

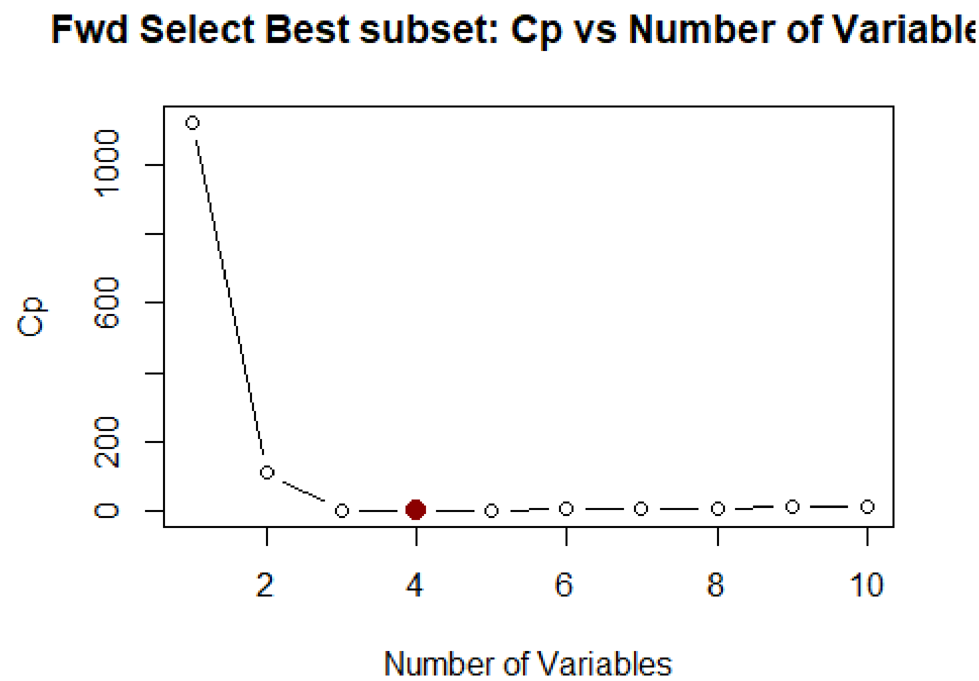



```r
regfits_summary$adjr2

##  [1] 0.9334429 0.9887867 0.9947516 0.9948979 0.9948680 0.9948233 0.9947792
##  [8] 0.9947581 0.9947008 0.9946505

writeLines(paste("Greatest R-squared model number: ",
which.max(regfits_summary$adjr2)))

## Greatest R-squared model number:  4

plot(regfits_summary$adjr2, xlab = "Number of Variables", ylab = "R-squared",
main = "Best subset: R-squared vs Number of Variables", type = "b")
points(which.max(regfits_summary$adjr2),
regfits_summary$adjr2[which.max(regfits_summary$adjr2)], col="darkgreen", cex
= 2, pch = 20)
```

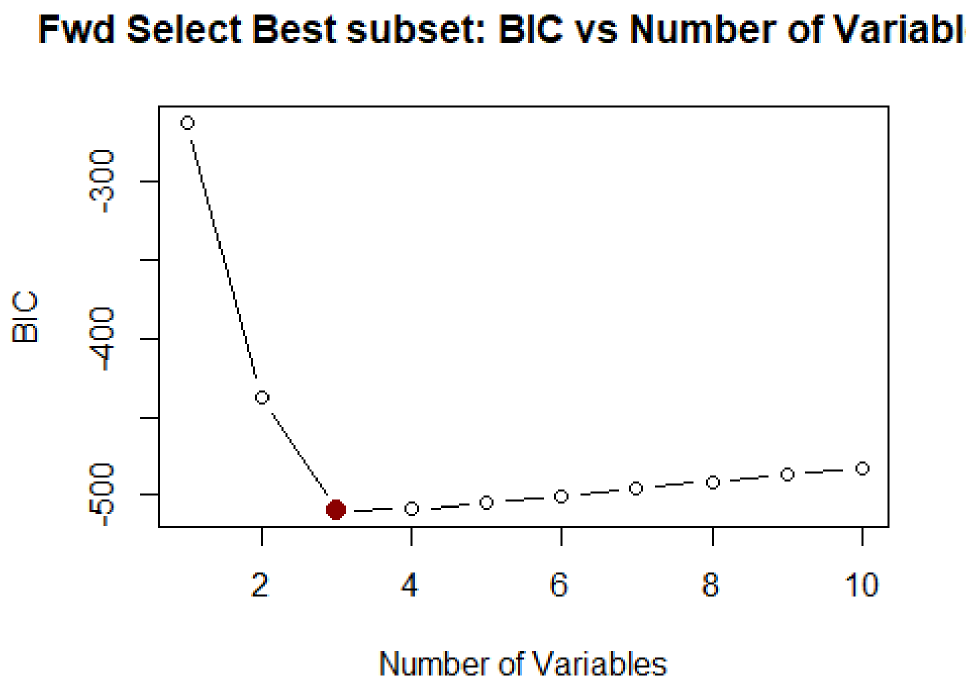



```
# P4 PART D:


# forward selection
writeLines("------------------------------------------------------------------------------")

## ---------------------------------------------------------------------------

writeLines("\n FORWARD SELECTION \n")

##
##  FORWARD SELECTION

#regfit.fwd <- regsubsets(Y ~ poly(X, 10, raw = TRUE), data = data, nvmax =
10, method = "forward")
regfit.fwd <- regsubsets(Y ~ X + X2 + X3 + X4 + X5 + X6 + X7 + X8 + X9 + X10,
data = data, nvmax = 10, method = "forward")
regfits_summary.fwd <- summary(regfit.fwd)

writeLines("Fwd Selection Coefficients (Cp, BIC, R2 respectively):")

## Fwd Selection Coefficients (Cp, BIC, R2 respectively):

coef(regfit.fwd, which.min(regfits_summary.fwd$cp))
```

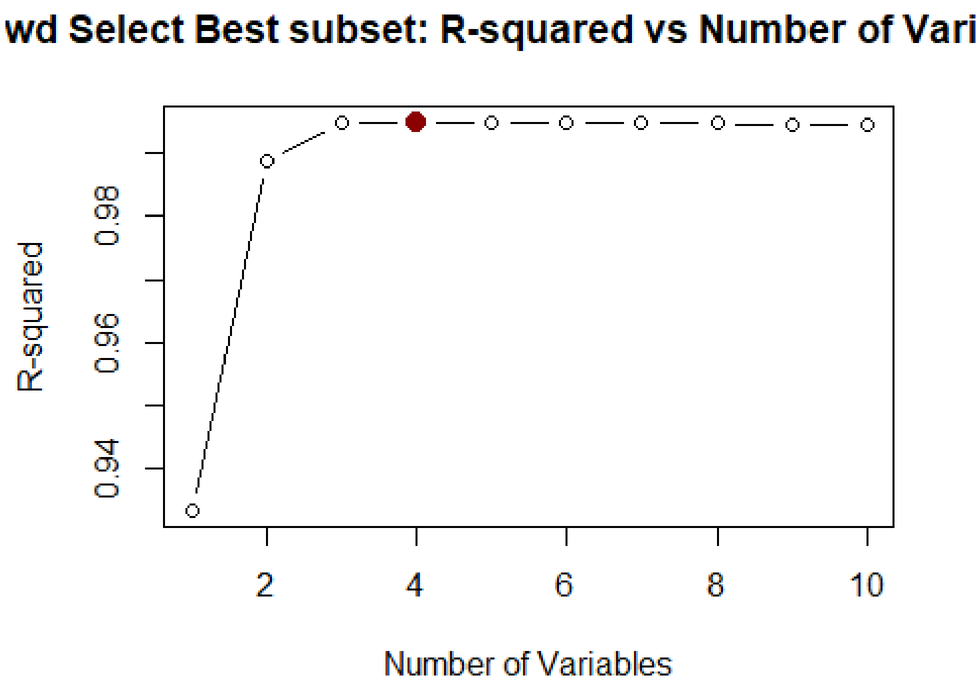
```
## (Intercept)           X          X2          X3          X5
##  1.07200775  2.38745596  2.84575641  3.55797426  0.08072292

writeLines("\n")

coef(regfit.fwd, which.min(regfits_summary.fwd$bic))

## (Intercept)           X          X2          X3
##    1.061507    1.975280    2.876209    4.017639

writeLines("\n")

coef(regfit.fwd, which.max(regfits_summary.fwd$adjr2))

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

writeLines("\n")

regfits_summary.fwd$cp

##  [1] 1123.2892318  109.3256041    2.1859433    0.6067483    2.1782005
##  [6]    4.0621068    6.0165107    7.9582570    9.7842795   11.0000000

writeLines(paste("Fwd Select Lowest Cp model number: ",
which.min(regfits_summary.fwd$cp)))

## Fwd Select Lowest Cp model number:  4

plot(regfits_summary.fwd$cp, xlab = "Number of Variables", ylab = "Cp", main
= "Fwd Select Best subset: Cp vs Number of Variables", type = "b")
points(which.min(regfits_summary.fwd$cp),
regfits_summary.fwd$cp[which.min(regfits_summary.fwd$cp)], col="darkred", cex
= 2, pch = 20)
```




```
regfits_summary.fwd$bic

##  [1] -262.7744 -437.2907 -509.6393 -508.9084 -504.7773 -500.3010 -495.7464
##  [8] -491.2060 -486.7944 -483.0666

writeLines(paste("Fwd Select Lowest BIC model: ",
which.min(regfits_summary.fwd$bic)))

## Fwd Select Lowest BIC model:  3

plot(regfits_summary.fwd$bic, xlab = "Number of Variables", ylab = "BIC",
main = "Fwd Select Best subset: BIC vs Number of Variables", type = "b")
points(which.min(regfits_summary.fwd$bic),
regfits_summary.fwd$bic[which.min(regfits_summary.fwd$bic)], col="darkred",
cex = 2, pch = 20)
```

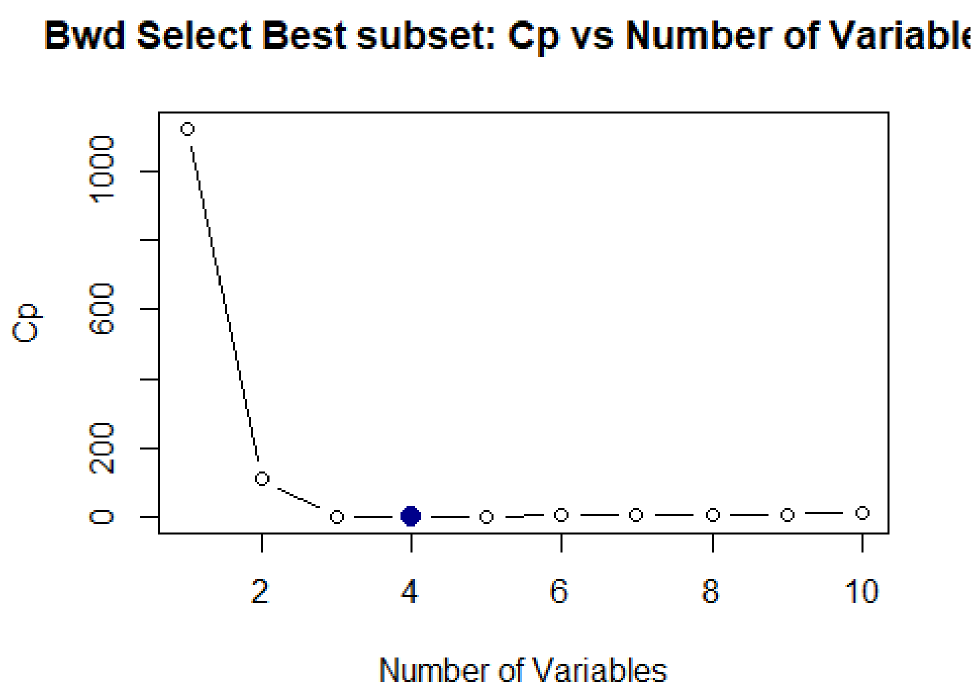



```
regfits_summary.fwd$adjr2

##  [1] 0.9334429 0.9887867 0.9947516 0.9948979 0.9948680 0.9948195 0.9947658
##  [8] 0.9947117 0.9946633 0.9946505

writeLines(paste("Fwd Select Greatest R-squared model number: ",
which.max(regfits_summary.fwd$adjr2)))

## Fwd Select Greatest R-squared model number:  4

plot(regfits_summary.fwd$adjr2, xlab = "Number of Variables", ylab = "R-
squared", main = "Fwd Select Best subset: R-squared vs Number of Variables",
type = "b")
points(which.max(regfits_summary.fwd$adjr2),
regfits_summary.fwd$adjr2[which.max(regfits_summary.fwd$adjr2)],
col="darkred", cex = 2, pch = 20)
```

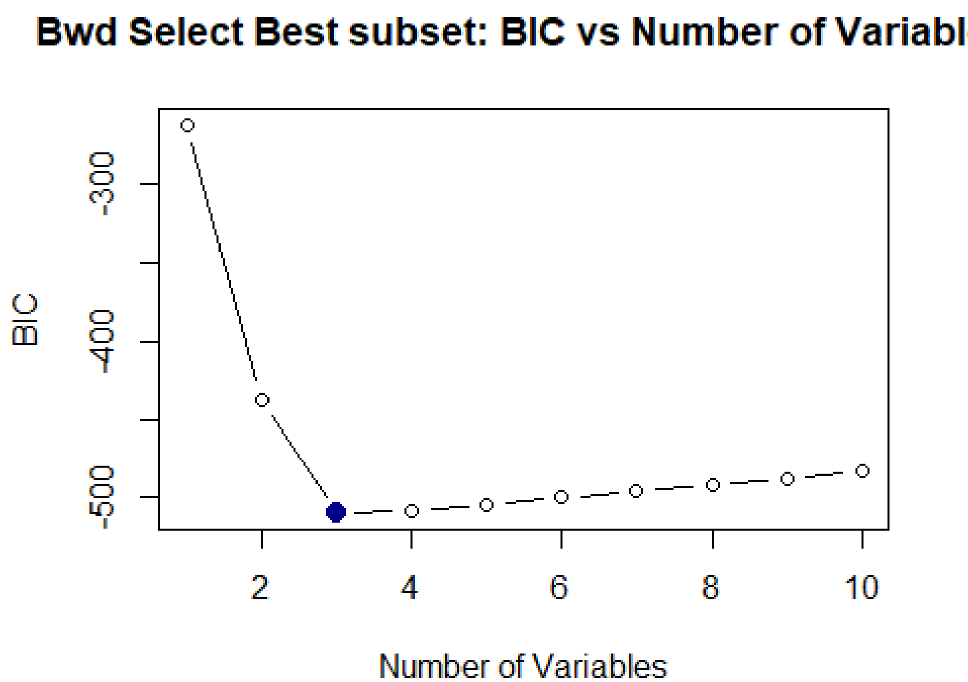



```
# backward selection
writeLines("------------------------------------------------------------------------------")

## ------------------------------------------------------------------------------

writeLines("\n BACKWARD SELECTION \n")

##
##  BACKWARD SELECTION

#regfit.bwd <- regsubsets(Y ~ poly(X, 10, raw = TRUE), data = data, nvmax = 10, method = "backward")
regfit.bwd <- regsubsets(Y ~ X + X2 + X3 + X4 + X5 + X6 + X7 + X8 + X9 + X10, data = data, nvmax = 10, method = "backward")
regfits_summary.bwd <- summary(regfit.bwd)

writeLines("Bwd Selection Coefficients (Cp, BIC, R2 respectively):")

## Bwd Selection Coefficients (Cp, BIC, R2 respectively):

coef(regfit.bwd, which.min(regfits_summary.bwd$cp))

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

writeLines("\n")
```

```r
coef(regfit.bwd, which.min(regfits_summary.bwd$bic))

## (Intercept)           X          X2          X3
##    1.061507    1.975280    2.876209    4.017639

writeLines("\n")

coef(regfit.bwd, which.max(regfits_summary.bwd$adjr2))

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

writeLines("\n")

regfits_summary.bwd$cp

##  [1] 1123.2892318  109.3256041    2.1859433    0.9808795    2.7310150
##  [6]    4.4181003    5.9250326    7.1694092    9.1535580   11.0000000

writeLines(paste("Bwd Select Lowest Cp model number: ",
which.min(regfits_summary.bwd$cp)))

## Bwd Select Lowest Cp model number:  4

plot(regfits_summary.bwd$cp, xlab = "Number of Variables", ylab = "Cp", main
= "Bwd Select Best subset: Cp vs Number of Variables", type = "b")
points(which.min(regfits_summary.bwd$cp),
regfits_summary.bwd$cp[which.min(regfits_summary.bwd$cp)], col="darkblue",
cex = 2, pch = 20)
```

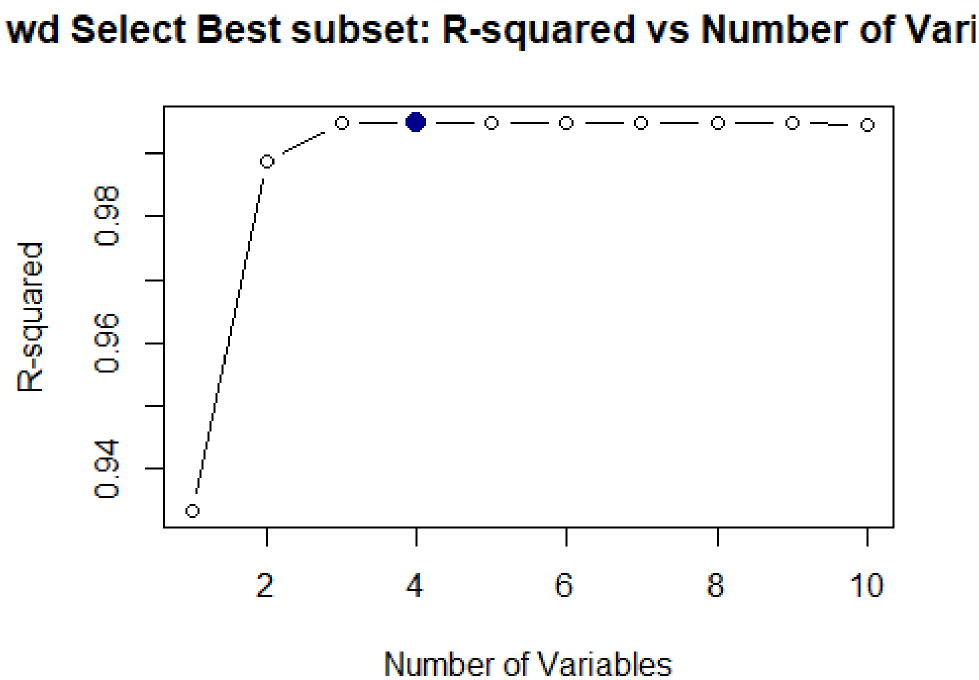


```
regfits_summary.bwd$bic

##  [1] -262.7744 -437.2907 -509.6393 -508.4963 -504.1661 -499.9065 -495.8481
##  [8] -492.0868 -487.4994 -483.0666

writeLines(paste("Bwd Select Lowest BIC model: ",
which.min(regfits_summary.bwd$bic)))

## Bwd Select Lowest BIC model:  3

plot(regfits_summary.bwd$bic, xlab = "Number of Variables", ylab = "BIC",
main = "Bwd Select Best subset: BIC vs Number of Variables", type = "b")
points(which.min(regfits_summary.bwd$bic),
regfits_summary.bwd$bic[which.min(regfits_summary.bwd$bic)], col="darkblue",
cex = 2, pch = 20)
```




```
regfits_summary.bwd$adjr2

##  [1] 0.9334429 0.9887867 0.9947516 0.9948768 0.9948365 0.9947990 0.9947711
##  [8] 0.9947581 0.9947008 0.9946505

writeLines(paste("Bwd Select Greatest R-squared model number: ",
which.max(regfits_summary.bwd$adjr2)))

## Bwd Select Greatest R-squared model number:  4

plot(regfits_summary.bwd$adjr2, xlab = "Number of Variables", ylab = "R-
squared", main = "Bwd Select Best subset: R-squared vs Number of Variables",
type = "b")
```

```
points(which.max(regfits_summary.bwd$adjr2),
regfits_summary.bwd$adjr2[which.max(regfits_summary.bwd$adjr2)],
col="darkblue", cex = 2, pch = 20)
```

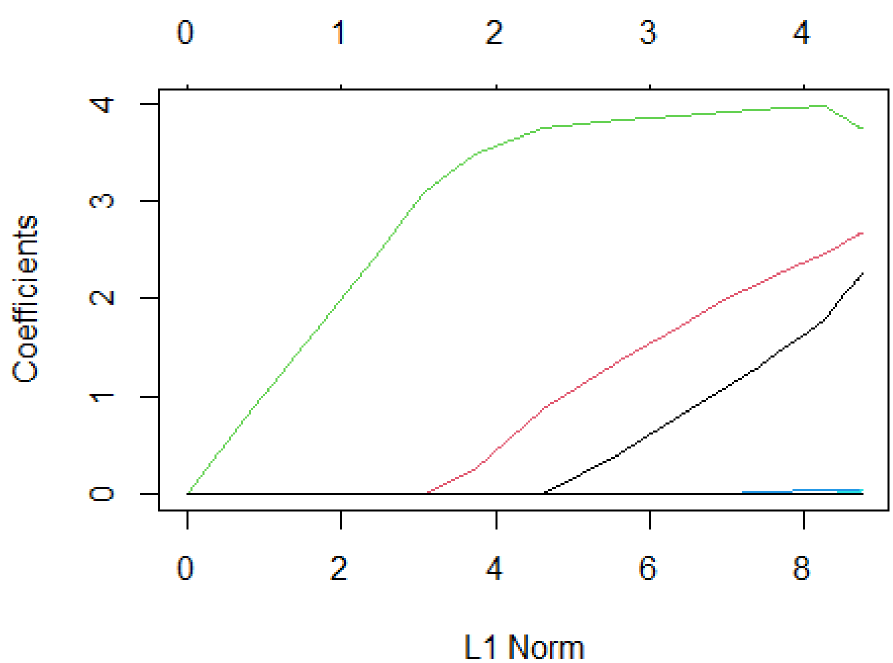


4D COMMENT:

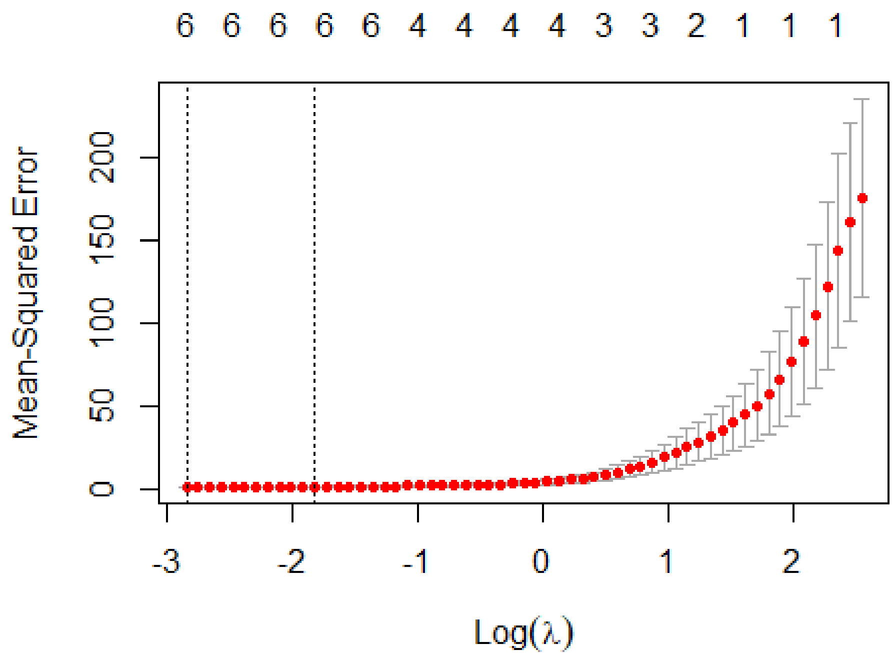
Between the forward and backward stepwise selection of part (d) and the best subset selection of part (C) (note: I realize the titles of my plots may be a bit confusing given I include the phrase “best subset” in the forward (fwd) and backward (bwd) stepwise selections), they all appear to select the best model size according to Cp, BIC, and R^2 in the order of 4 parameters, 3 parameters, and 4 parameters respectively. All contain parameters for X, X^2, and X^3. In the case where a 4^{th} parameter is chosen, the forward stepwise selection and best subset selection agree on X^5 while the backward stepwise selection disagrees and chooses a parameter of X^9.

```
# P4 PART E: LASSO and Cross Validation

# first must create x-matrix and y-vector
library(glmnet)

## Loading required package: Matrix

## Loaded glmnet 4.1-8
```

```
set.seed(1)

X_mat = model.matrix(Y ~ X + X2 + X3 + X4 + X5 + X6 + X7 + X8 + X9 + X10,
data, nvmax = 10) [,-1]
Y_vec = data$Y


sample <- sample(c(TRUE, FALSE), nrow(X_mat), replace=TRUE, prob=c(0.7,0.3))
train <- X_mat[sample,]
test <- X_mat[!sample,]
grid <- 10^seq(10,-2, length = 100)
lasso.regfit <-glmnet(X_mat, Y_vec, alpha = 1, lambda = grid)
plot(lasso.regfit)

## Warning in regularize.values(x, y, ties, missing(ties), na.rm = na.rm):
## collapsing to unique 'x' values
```




```
cv.out <-cv.glmnet(X_mat, Y_vec, alpha = 1)
plot(cv.out)
```




```
bestlam <- cv.out$lambda.min
lasso.pred <-predict(lasso.regfit, s = bestlam, newx = train)
dev.off
```

```
## function (which = dev.cur())
## {
##     if (which == 1)
##         stop("cannot shut down device 1 (the null device)")
##     .External(C_devoff, as.integer(which))
##     dev.cur()
## }
## <bytecode: 0x0000022ae1c99cd8>
## <environment: namespace:grDevices>
```

```
bestmodel <- glmnet(X_mat, Y_vec, alpha = 1)
predict(bestmodel, s = bestlam, type = "coefficients")
```

```
## 11 x 1 sparse Matrix of class "dgCMatrix"
##                      s1
## (Intercept) 1.168794337
## X           2.164793590
## X2          2.639485133
## X3          3.800683773
## X4          0.041512567
## X5          0.014068421
## X6          .
## X7          0.004039751
```

```
## X8           .
## X9           .
## X10          .
```

PART 4E COMMENT:

Using the lasso model, we obtain 6 coefficients (for X, X^2, X^3, and (weakly) X^4, X^5, and X^7). All parameters for which a true B was programmed have been selected.

```
# P4 PART F
B7 <- 4.5

Y_4f <- B0 + B7*X7 + eps

data_4f <- data.frame(Y_4f, X)
model_4f <- regsubsets(Y_4f ~ X + X2 + X3 + X4 + X5 + X6 + X7 + X8 + X9 +
X10, data = data_4f, nvmax = 10)
model_4f_sum <- summary(model_4f)

print("Results from Best Subset Selection")

## [1] "Results from Best Subset Selection"

print("Minimum Cp, 4f")

## [1] "Minimum Cp, 4f"

which.min(model_4f_sum$cp)

## [1] 2

print("Minimum BIC, 4f")

## [1] "Minimum BIC, 4f"

which.min(model_4f_sum$bic)

## [1] 1

print("Maximum R^2, 4f")

## [1] "Maximum R^2, 4f"

which.max(model_4f_sum$adjr2)
```

```
## [1] 4

print("Coefs of Model with 2 variables")

## [1] "Coefs of Model with 2 variables"

coefficients(model_4f, id=2)

## (Intercept)          X2          X7
##   1.0704904  -0.1417084   4.5015552

print("Coefs of Model with 1 variables")

## [1] "Coefs of Model with 1 variables"

coefficients(model_4f, id=1)

## (Intercept)          X7
##   0.9589402   4.5007705

print("Coefs of Model with 4 variables")

## [1] "Coefs of Model with 4 variables"

coefficients(model_4f, id=4)

## (Intercept)           X          X2          X3          X7
##   1.0762524   0.2914016  -0.1617671  -0.2526527   4.5091338

print("Results from Lasso Regression")

## [1] "Results from Lasso Regression"

cv.out.4f <-cv.glmnet(X_mat, Y_vec, alpha = 1)
bestlam.4f <- cv.out$lambda.min
bestmodel.4f <- glmnet(X_mat, Y_4f, alpha = 1)
predict(bestmodel.4f, s = bestlam.4f, type = "coefficients")

## 11 x 1 sparse Matrix of class "dgCMatrix"
##                   s1
## (Intercept) 1.512650
## X           .
## X2          .
## X3          .
## X4          .
## X5          .
## X6          .
## X7          4.369571
## X8          .
## X9          .
## X10         .
```

4f COMMENT:

The best subset selection of this model which now only has 1 parameter chooses model seizes 2, 1, and 4 for Cp, BIC, and R^2 respectively. The lasso only selects one parameter (parameter 7, which has a coefficient approximately what was set for B7=4.50. This makes sense given the lasso penalty would reduce overfitting and would consequently result in a smaller model size.

QUESTION 5 (uses ‘college’ dataset)

```
# P5 PART A: Split the data into a training and a test set

library(ISLR2)
data(College)


sample5 <- sample(c(TRUE, FALSE), nrow(College), replace=TRUE,
prob=c(0.7,0.3))
train5 <- College[sample5, ]
test5 <- College[!sample5, ]




# P5 PART B: fit a linear model on the training set using the least squares
method (using lm() command) and report the test error obtained

lm.q5 <- lm(train5$Apps ~ ., train5) # trains linear regression model on
training data (Enroll ~ all other variables)

predictions.q5b <- predict(lm.q5, test5[,-2]) # makes predictions on test
data for 'yhat', test5[,-4] removes response variable from test data

residuals.q5b <- test5$Apps - predictions.q5b # take residuals (actual -
predicted)

MSE.lm.q5b <- mean(residuals.q5b^2) # find residual mean square error (large?
possibly due to qualitative ")

print("MSE from least squares method:")

## [1] "MSE from least squares method:"

MSE.lm.q5b

## [1] 826642.8
```

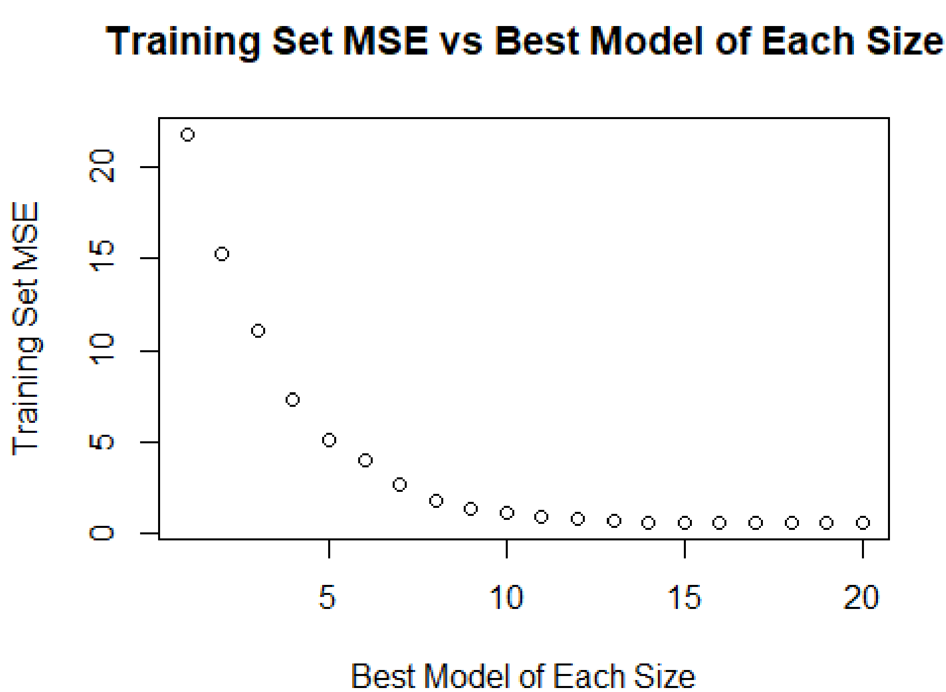
```r
# P5 PART C

train_xmat_q5 <- model.matrix(train5$Apps ~ ., train5) [,-1]
train_yvec_q5 <- train5$Apps

test_xmat_q5 <- model.matrix(test5$Apps ~ ., test5) [,-1]
test_yvec_q5 <- test5$Apps

cv.out.q5c <- cv.glmnet(train_xmat_q5, train_yvec_q5, alpha = 0)
lambda_best_q5c <- cv.out.q5c$lambda.min

ridge.mod.q5c <- glmnet(train_xmat_q5, train_yvec_q5, alpha = 0, lambda =
lambda_best_q5c)

predictions.q5c <- predict(ridge.mod.q5c, s=lambda_best_q5c, newx =
test_xmat_q5)

residuals.q5c <- test_yvec_q5 - predictions.q5c

MSE.glm.q5c <- mean(residuals.q5c^2)

print("MSE from ridge method:")

## [1] "MSE from ridge method:"

MSE.glm.q5c

## [1] 885108.3

# P5 PART D

cv.out.q5d <- cv.glmnet(train_xmat_q5, train_yvec_q5, alpha = 1)
lambda_best_q5d <- cv.out.q5d$lambda.min

lasso.mod.q5d <- glmnet(train_xmat_q5, train_yvec_q5, alpha = 1, lambda =
lambda_best_q5d)

predictions.q5d <- predict(lasso.mod.q5d, s=lambda_best_q5d, newx =
test_xmat_q5)

residuals.q5d <- test_yvec_q5 - predictions.q5d

MSE.glm.q5d <- mean(residuals.q5d^2)
print("MSE from lasso method:")

## [1] "MSE from lasso method:"

MSE.glm.q5d

## [1] 807822.6
```

```r
# number of non-zero rows for lasso regression (note: always finding the
coefficients using the training on the full dataset)

full_xmat_q5 <- model.matrix(College$Apps ~ ., College) [,-1]
full_yvec_q5 <- College$Apps

cv.out.q5dcoeffs <- cv.glmnet(full_xmat_q5, full_yvec_q5, alpha = 1)
lambda_best_q5dcoeffs <- cv.out.q5dcoeffs$lambda.min

lasso.mod.q5dcoeffs <- glmnet(full_xmat_q5, full_yvec_q5, alpha = 1, lambda =
lambda_best_q5dcoeffs)

print("Number of non-zero coefficients:")

## [1] "Number of non-zero coefficients:"

colSums(lasso.mod.q5dcoeffs$beta !=0)

## s0
## 15
```

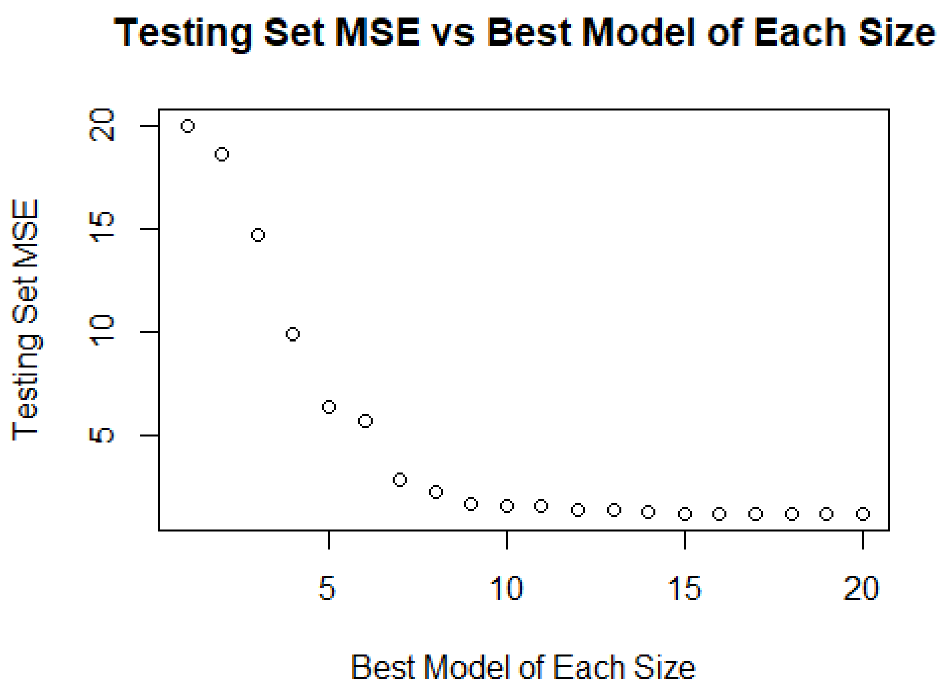
Problem 6

```r
# P6 PART A

set.seed(1)
library(leaps)

n <- 1000
p <- 20

X <- matrix(rnorm(n*p),nrow=n,ncol=p)
X[,1] = rnorm(n, mean = 0, sd = .25)

B <- rnorm(p)

B[6] <- 0
B[7] <- 0
B[15] <- 0

eps <- rnorm(n)

Y <- X%*%B + eps


# P6 PART B


sample <- sample(1:n, size = 100, replace = FALSE)
```

```r
X_train_p6b <- X[sample, ]
X_test_p6b <- X[-sample, ]

Y_train_p6b <- Y[sample]
Y_test_p6b <- Y[-sample]



# P6 PART C

df.p6c.train <- data.frame(X_train_p6b,Y_train_p6b)

regfit <- regsubsets(Y_train_p6b ~ ., data = df.p6c.train, nvmax = p)

train.mat <- model.matrix(Y_train_p6b ~ ., data = df.p6c.train)

val.errors.train <- rep(NA, p)

for (i in 1:p) {
  coefi <- coef(regfit, id = i)
  pred <- train.mat[, names(coefi)] %*% coefi
  val.errors.train[i] <- mean((Y_train_p6b - pred)^2)
}
plot(1:p, val.errors.train, xlab="Best Model of Each Size", ylab="Training
Set MSE", main="Training Set MSE vs Best Model of Each Size")
```



```r
# P6 PART D
df.p6d.test <- data.frame(X_test_p6b,Y_test_p6b)
test.mat <- model.matrix(Y_test_p6b ~ ., data = df.p6d.test)
val.errors.test <- rep(NA, p)

for (i in 1:p) {
  coefi <- coef(regfit, id = i)
  pred <- test.mat[, names(coefi)] %*% coefi
  val.errors.test[i] <- mean((Y_test_p6b - pred)^2)
}

plot(1:p, val.errors.test, xlab="Best Model of Each Size", ylab="Testing Set
MSE", main="Testing Set MSE vs Best Model of Each Size")
```




```r
# P6 PART E
print("Model size of minimum train error:")

## [1] "Model size of minimum train error:"

which.min(val.errors.train)

## [1] 20

print("Model size of minimum test error:")

## [1] "Model size of minimum test error:"

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

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

```r
print("Model size of 15 has minimum test error")
```

```
## [1] "Model size of 15 has minimum test error"
```

6e COMMENT:

The model size for which the test set MSE is minimized is 15.

It makes sense that this would be a smaller model size for minimizing error compared to the training dataset model size of 20 for minimizing MSE as 20 overfits the data and would result in an increased MSE.

```r
# P6 PART F
print("")
```

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

```r
print("coefficients of true model")
```

```
## [1] "coefficients of true model"
```

```r
B
```

```
##  [1]  0.60109155 -2.76711578  0.18152306  2.26188715  0.71197130
0.00000000
##  [7]  0.00000000 -1.09914106  0.37242349 -1.78435004  0.49926364 -
0.37971579
## [13]  0.27895349  0.02597137  0.00000000 -1.68496253 -1.89403426 -
1.38330270
## [19] -0.83727797  0.00256629
```

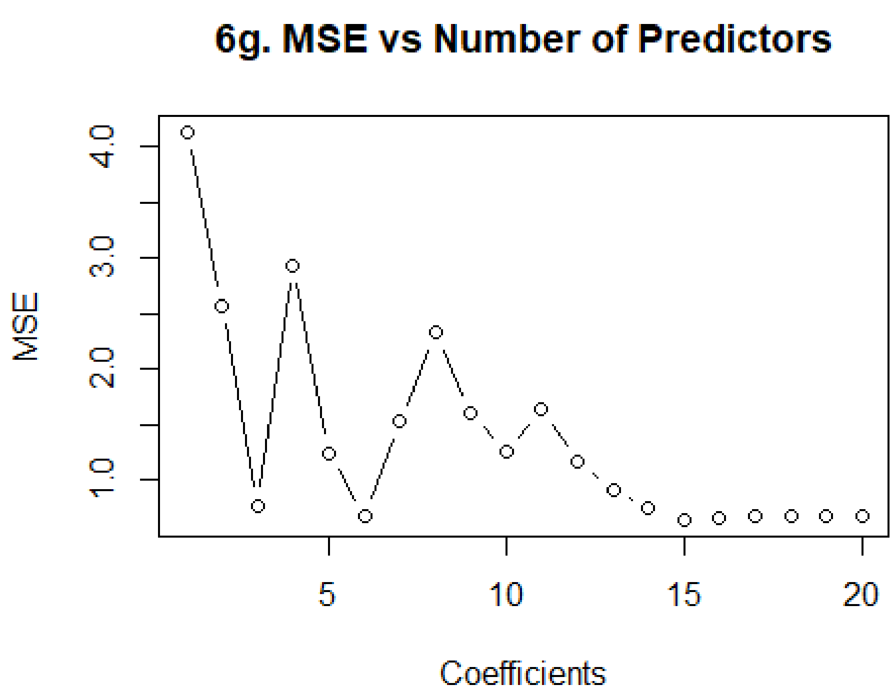
```r
print("coefficients of 15-parameter model")
```

```
## [1] "coefficients of 15-parameter model"
```

```r
coefficients(regfit, id=15)
```

```
## (Intercept)          X1          X2          X3          X4          X5
## -0.08833152  1.09665639 -2.58290864  0.15507539  2.26306776  0.69340258
##          X8          X9         X10         X11         X12         X13
## -1.18918205  0.60818664 -1.65894768  0.58148580 -0.47335698  0.28556215
##         X16         X17         X18         X19
## -1.76086912 -1.76894424 -1.31965338 -0.74813909
```

```
6f COMMENT:

The model parameter we get for the test set have no zero values as their
coefficients compared to the true Beta values which does contain zero values.



# P6 PART G

df.p6g.full <- data.frame(Y,X)
val.errors.6g <- rep(NA,p)
xcolumns <- colnames(df.p6g.full)[-1]

for (i in 1:p) {
  coefi <- coef(regfit, id = i)
  val.errors.6g[i] <- sqrt(sum((B[xcolumns %in% names(coefi)] -
coefi[names(coefi) %in% xcolumns])^2) + sum(B[!(xcolumns %in%
names(coefi))])^2)
}

plot(val.errors.6g, xlab = "Coefficients", ylab = "MSE", main = "6g. MSE vs
Number of Predictors", type = "b")
```




```
print("6g. Minimum error model size:")
```

```
## [1] "6g. Minimum error model size:"

which.min(val.errors.6g)

## [1] 15

# Minimum error in both cases is 15 parameters
val.errors.6g

##  [1] 4.1309490 2.5730520 0.7702071 2.9330592 1.2445126 0.6667725 1.5210634
##  [8] 2.3211045 1.5984242 1.2475036 1.6336103 1.1664059 0.9181042 0.7422308
## [15] 0.6401354 0.6606039 0.6823455 0.6764207 0.6753607 0.6765429
```

```

6g COMMENT: While there is a precipitous drop and then a “damped oscillating decline” towards 15, the model size which minimizes MSE for both cases is 15.
```