tw2906_HW2

Tong Wu

2022-10-01

Problem 1

Part a

The ridge regression can be written as the following:

$$min\ (RSS + \lambda \sum_{j=1}^{p} \beta_j^2)$$

Where RSS could written as the following:

$$RSS = \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij})^2$$

Where according to the problem, $n = 2, \ \ p = 2, \ \ \hat{\beta}_0 = 0$, so the final form of ridge regression can be written as below:

$$min\ (\sum_{i=1}^{2} (y_i - \sum_{j=1}^{2} \beta_j x_{ij})^2 + \lambda \sum_{j=1}^{2} \beta_j^2$$

$$min\ (y_1 - \hat{\beta}_1 x_{11} - \hat{\beta}_2 x_{12})^2 + (y_2 - \hat{\beta}_1 x_{21} - \hat{\beta}_2 x_{22})^2 + \lambda(\hat{\beta}_1^2 + \hat{\beta}_2^2)$$

Part b

According to the problem, $x_{11} = x_{12}, \ x_{21} = x_{22}, \ y_1 + y_2 = 0, \ x_{11} + x_{21} = 0, \ x_{12} + x_{22} = 0$

$$\beta_1 = \frac{x_1 y_1 + x_2 y_2 - \beta_2 (x_1^2 + x_2^2)}{\lambda + x_1^2 + x_2^2}$$

$$\beta_2 = \frac{x_1 y_1 + x_2 y_2 - \beta_2 (x_1^2 + x_2^2)}{\lambda + x_1^2 + x_2^2}$$

$$\beta_1 = \beta_2$$

Part c

Lasso optimisation:

$$min\ (RSS + \lambda \sum_{j=1}^{p} |\beta_j|)$$

$$min\ (\sum_{i=1}^{2} (y_i - \sum_{j=1}^{2} \beta_j x_{ij})^2 + \lambda(|\hat{\beta}_1| + |\hat{\lambda}_2|))$$

$$min\ (y_1 - \hat{\beta}_1 x_{11} - \hat{\beta}_2 x_{12})^2 + (y_2 - \hat{\beta}_1 x_{21} - \hat{\beta}_2 x_{22})^2 + \lambda(|\hat{\beta}_1^2| + |\hat{\beta}_2^2|)$$

Part d

The best subset for lasso is:

$$|\beta_1| + |\beta_2| \leq s$$

According to the problem, $x_{11} = x_{12},\ x_{21} = x_{22},\ y_1 + y_2 = 0,\ x_{11} + x_{21} = 0,\ x_{12} + x_{22} = 0$ So the answer in part c can be minimized as following:

$$min\ 2(y_1 - (\hat{\beta}_1 + \hat{\beta}_2)x_{11})^2$$

Where the minimum is that $y_1 - (\hat{\beta}_1 + \hat{\beta}_2)x_{11} = 0$, where $\frac{y_1}{x_{11}} = \hat{\beta}_1 + \hat{\beta}_2$ Since the minimum are at the same gradient as the best subset, $\beta_1 + \beta_2 = s$.So two solutions can be determined in here:

$$\hat{\beta}_1 + \hat{\beta}_2 = s$$

$$\hat{\beta}_1 + \hat{\beta}_2 = -s$$

Problem 2

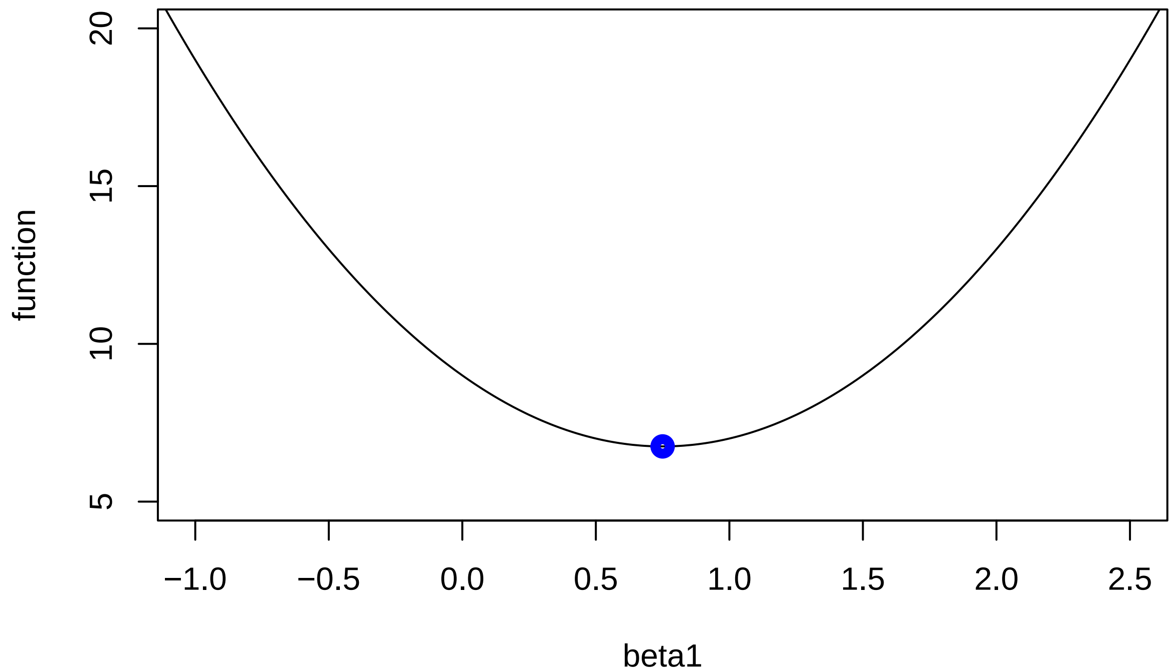
Part a

For p=1, the equation 1 should be:

$$(y - \beta_1)^2 + \lambda\beta_1^2 = (1 + \lambda)\beta_1^2 - 2y\beta_1 + y^2$$

Where $\hat{\beta}_1 = y_1/(1 + \lambda)$ So $(1 + \lambda) - 2y\beta_1 + y^2)' = (1 + \lambda)\beta_1 - 2y_1 = 0$ Set y and lambda are both 3.

```
y <- 3
lambda <- 3
beta1<- seq(-20, 20, 0.01)
fun <- (y-beta1)^2+lambda*beta1^2
fun2 <- (y) / (1+lambda)
fun3 <- (y-fun2)^2+lambda*fun2^2
plot(beta1, fun, type="l", xlab = "beta1", ylab = "function", xlim=c(-1,2.5),
     ylim=c(5,20))
points(fun2, fun3, col="blue", lwd=5)
```



Where the blue dot is the minimum value for the function by β_1, which should same as the value of $\hat{\beta}_1 = y_1/(1+\lambda) = 3/(1+3) = 0.75$

Part b

For p=1, equation 2 can be written as:

$$(y - \beta_1)^2 + \lambda|\beta_1|$$

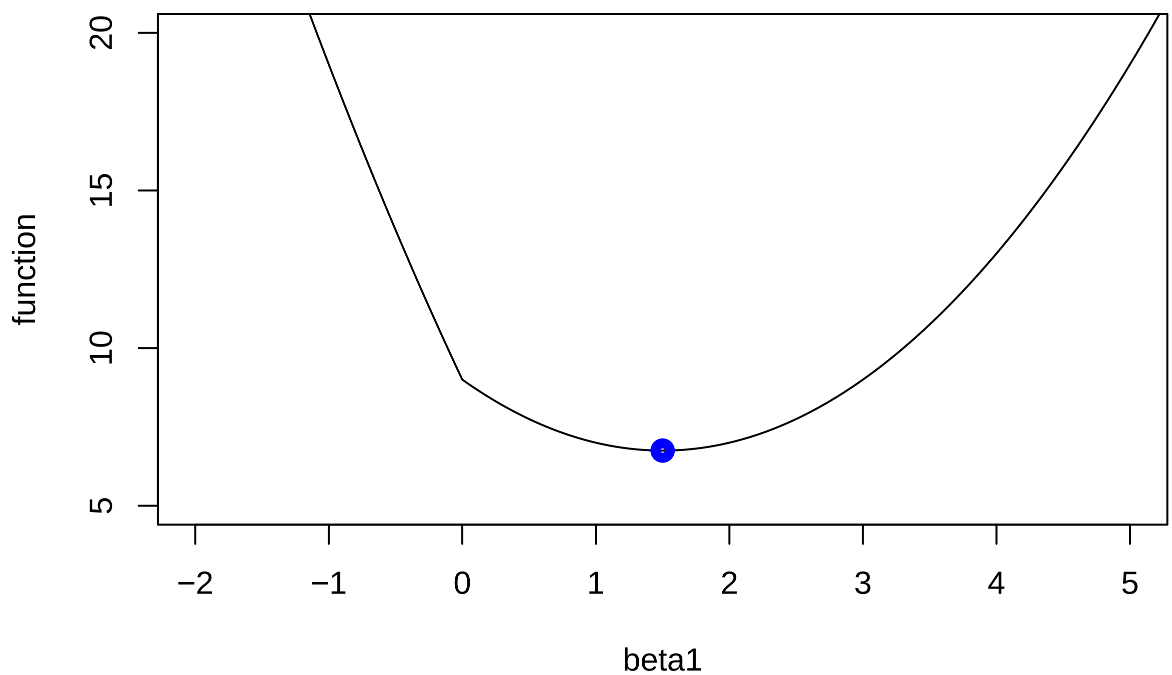
Which can be re-written as:

$$(y - \beta_1)^2 + \lambda\beta_1$$

$$(y - \beta_1)^2 - \lambda\beta_1$$

Assume $y = 3, \quad \lambda = 3,$

```
y <- 3
lambda <- 3
beta1 <- seq(-20, 20, 0.01)
fun <- (y-beta1)^2+lambda*abs(beta1)
fun2 <- y-lambda/2
fun3 <- (y-fun2)^2+lambda*abs(fun2)
plot(beta1, fun, type="l", xlab = "beta1", ylab = "function", xlim=c(-2,5),
     ylim=c(5,20))
points(fun2, fun3, col="blue", lwd=5)
```



Where the blue dot shows the minimum value of the function by β_1, which should be same as the result of the function:

$$\beta_1 = y - \frac{\lambda}{2} = 3 - \frac{3}{2} = 1.5$$

Question 3

Part a

Where likelihood of the data which distributed from $\mathbf{N}(0, \sigma^2)$ distribution for y_i can be written as:

$$\prod_{i=1}^{n} \frac{1}{\sigma\sqrt{2\pi} exp(\frac{-(y_i - (\beta_0 + \sum_{j=1}^{n} \beta_j x_{ij}))^2}{2\sigma^2})}$$

$$\frac{1}{\sigma\sqrt{2\pi}} exp(\frac{-1}{2\sigma^2} \sum_{i=1}^{n} (y_i - (\beta_0 + \sum_{j=1}^{n} \beta_j x_{ij}))^2)$$

Part b

For $p(\beta) = \frac{1}{2b} exp(-|\beta|/b)$ and $|\beta| = \sum_{j=1}^{p} |\beta_j|$, the likelihodd function can be written from the answer in part a:

$$\frac{1}{\sigma\sqrt{2\pi}} exp(\frac{-1}{2\sigma^2} \sum_{i=1}^{n} (y_i - (\beta_0 + \sum_{j=1}^{n} \beta_j x_{ij}))^2) \frac{1}{2b} exp(-|\beta|/b)$$

Part c

```r
knitr::include_graphics("./src/1.jpg")
```

Part d

```r
knitr::include_graphics("./src/2.jpg")
```

Part e

```r
knitr::include_graphics("./src/3.jpg")
```

Question 3

part c.

according to part b, posterior of β is

$$= \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-1}{2\sigma^2} \sum_{i=1}^{n} \left(y_i - (\beta_0 + \sum_{j=1}^{n} \beta_j x_{ij}) \right)^2 \right.$$

$$\frac{1}{2b} \exp(-|\beta|/b)$$

$$= \ln\left(\frac{1}{\sigma\sqrt{2\pi}} \frac{1}{2b} \right) - \frac{1}{2\sigma^2} \sum \left(y_i - (\beta_0 + \sum_{j=1}^{n} \beta_j x_{ij}) \right)^2$$

$$\arg\max \left(\ln\left(\frac{1}{\sigma\sqrt{2\pi}} \frac{1}{2b} \right) \right) - \frac{1}{2\sigma^2} \cdots$$

$$= \arg\min \left(\frac{1}{2\sigma^2} \sum \left(y_i - (\beta_0 + \sum \beta_j x_{ij})^2 \right) \right)$$

$$= \arg\min \left(\sum (y_i - \beta_0 + \sum \beta_j x_{ij})^2 \right) + \frac{2\sigma^2}{b} \sum_{j=1}^{n} |\beta_j|$$

Since $\lambda = \frac{2\sigma^2}{b}$, so

$$\arg\min \left(\sum (y_i - \beta_0 + \sum \beta_j x_{ij})^2 \right) + \lambda \sum_{j=1}^{n} |\beta_j|$$

$$= \arg\min RSS + \lambda \sum_{j=1}^{n} |\beta_j|$$

Figure 1: Problem 3 Part c solution

part d.

$$\prod_{i=1}^{p} p(\beta_i) = \prod_{i=1}^{p} \frac{1}{\sqrt{2\pi c}} \exp\left(-\frac{\beta_i^2}{2c}\right) = \left(\frac{1}{\sqrt{2\pi c}}\right) \exp\left(-\frac{1}{2c}\sum_{i=1}^{p}\beta_i^2\right)$$

portier for β:

$$= \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^n \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\left(y-\beta_0 + \sum x_{ij}\beta_j\right)^2\right)\cdot\left(\frac{1}{\sqrt{2c\pi}}\right)^p \exp\Big($$

$$-\frac{1}{2c}\sum_{i=1}^{p}(\beta_i^2)\Big)$$

Figure 2: Problem 3 Part d solution

part e.

$$\ln\left[\left(\frac{1}{\sqrt{2\pi}\sigma}\right)^n\left(\frac{1}{\sqrt{2\pi c}}\right)^p\right] - \frac{1}{2\sigma^2}\sum_{i=1}^{n}\left(y-\beta_0+\sum x_{ij}\beta_j\right)^2 + \frac{1}{2c}$$

$$\sum \beta_i^2$$

$$\arg\max\left(\ln\left(\frac{1}{\sigma\sqrt{2\pi}}\,\frac{1}{2b}\right)\right) - \frac{1}{2\sigma^2}\ \cdots$$

$$= \arg\min\left(\frac{1}{2\sigma^2}\sum\left(y_i - \left(\beta_0 + \sum \beta_j x_{ij}\right)^2\right)\right)$$

$$= \arg\min\left(\sum\left(y_i - \beta_0 + \sum \beta_j x_{ij}\right)^2\right) + \frac{2\sigma^2}{b}\sum_{j=1}^{n}|\beta_j|$$

Since $\lambda = \frac{2\sigma^2}{b}$, so

$$\arg\min\left(\sum\left(y_i - \beta_0 + \sum \beta_j x_{ij}\right)^2\right) + \lambda\sum_{j=1}^{n}|\beta_j|$$

$$= \arg\min \text{RSS} + \lambda\sum_{j=1}^{n}|\beta_j|$$

Figure 3: Problem 3 Part e solution

Question 4

Part a

```
set.seed(1)
x <- rnorm(100)
epsilon <- rnorm(100)
```

Part b

```
beta0 <- 11
beta1 <- 4
beta2 <- 5
beta3 <- 14
y <- beta0 + beta1*x + beta2*x^2 + beta3*x^3 + epsilon
```

Part c

```
# produce data set
data_set1 <- data.frame(y=y, x=x)
# import leaps lib for regsubsets
library(leaps)
# to produce model for 15 predicted points
best_model1 <- regsubsets(y ~ poly(x, 10, raw=TRUE), data = data_set1,
                          nvmax = 15)
summary1 <- summary(best_model1)
which.min(summary1$bic)
```

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

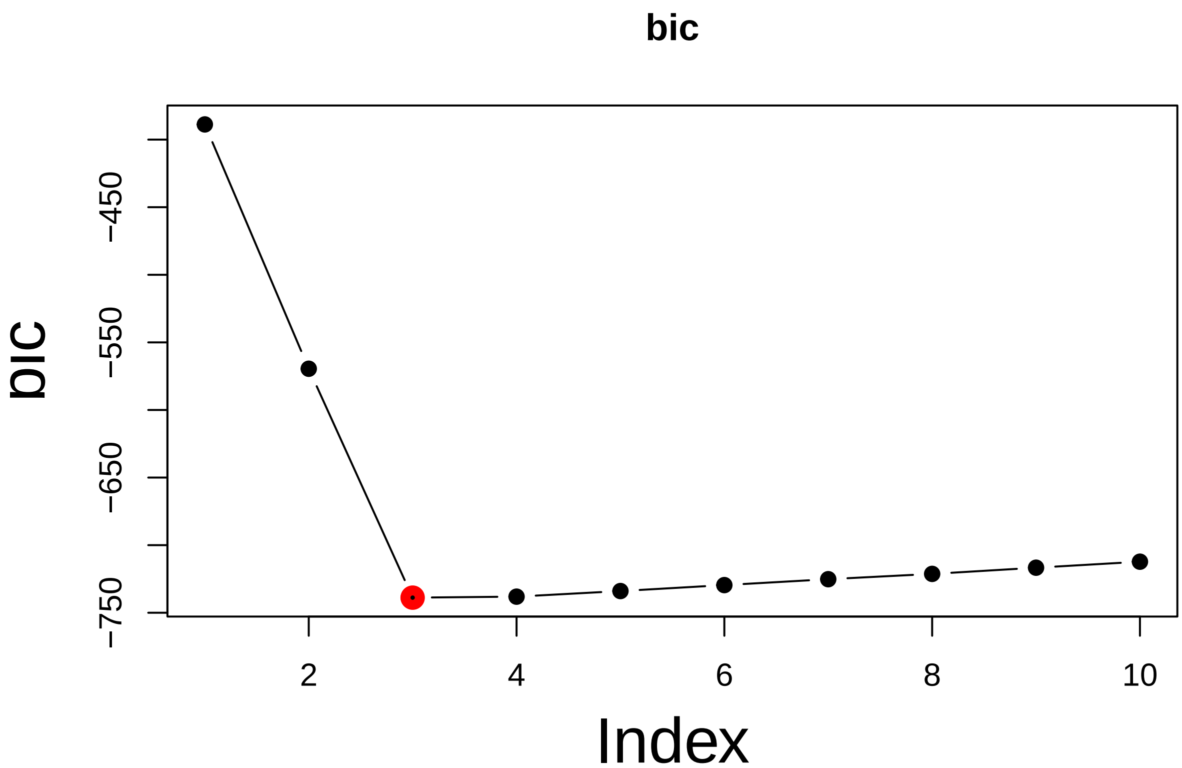
```
which.min(summary1$cp)
```

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

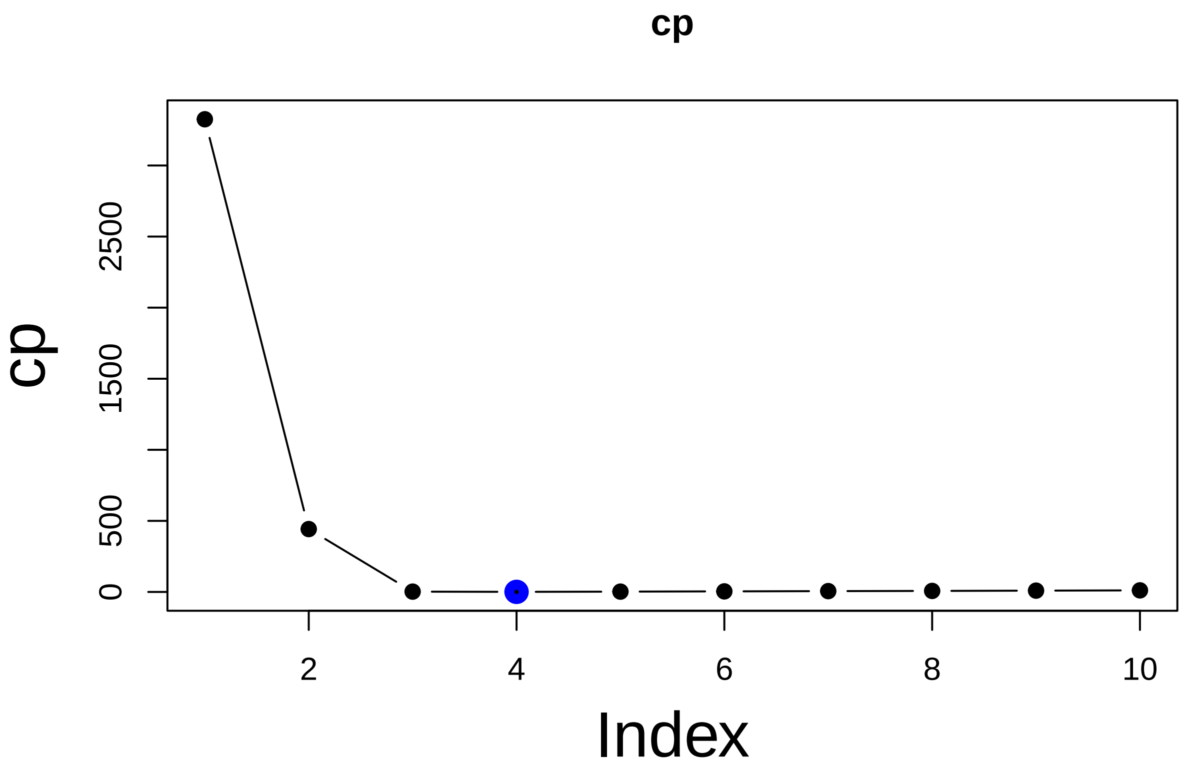
```
which.min(summary1$adjr2)
```

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

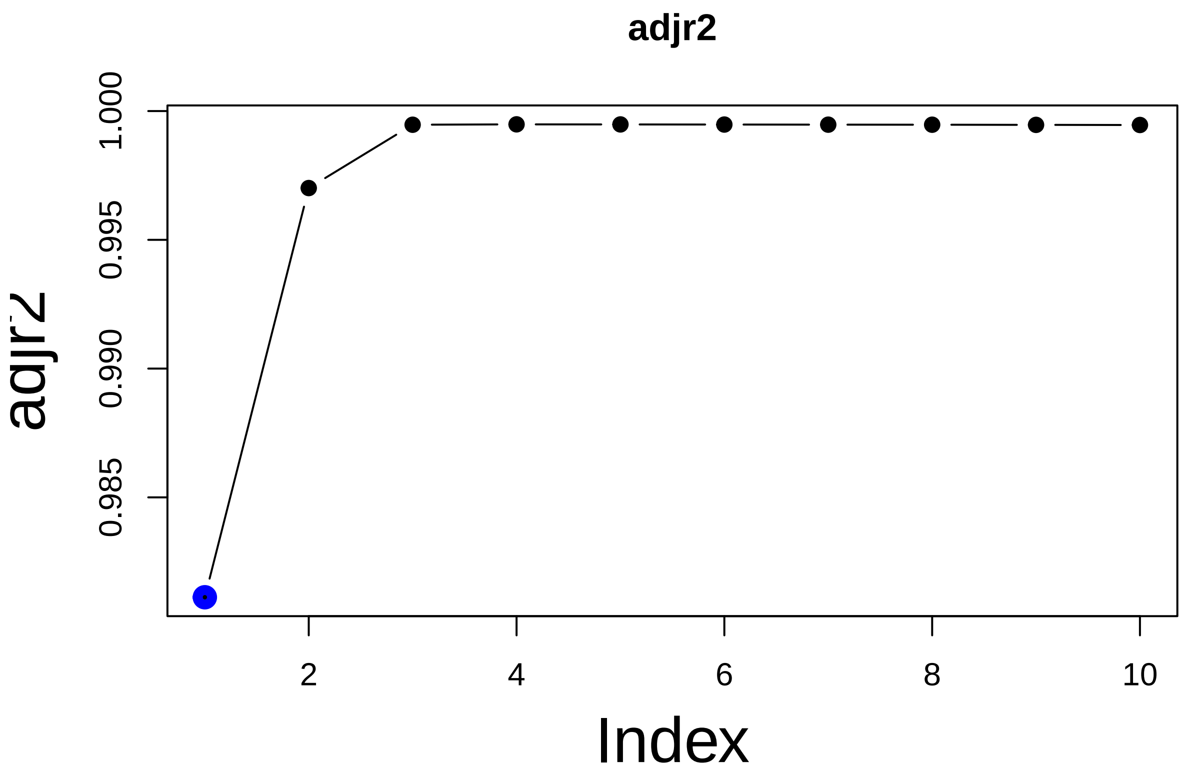
```
# show diagrams for bic, cp and adjr2
plot(summary1$bic, type = "b", pch=19, cex.lab=2, ylab="bic")
points(3, summary1$bic[3], col = "red",lwd=5)
title("bic")
```




```
plot(summary1$cp, type = "b", pch=19, cex.lab=2, ylab="cp")
points(4, summary1$cp[4], col = "blue",lwd=5)
title("cp")
```




```
plot(summary1$adjr2, type = "b", pch=19, cex.lab=2, ylab = "adjr2")
points(1, summary1$adjr2[1], col = "blue",lwd=5)
title("adjr2")
```




```
# generate hat beta 1 to 3
coefficients(best_model1, id=3)
```

```
##              (Intercept) poly(x, 10, raw = TRUE)1 poly(x, 10, raw = TRUE)2
##                11.061507                 3.975280                 4.876209
## poly(x, 10, raw = TRUE)3
##                14.017639
```

Where $\hat{\beta}_0 = 11.061507, \quad \hat{\beta}_1 = 3.975280, \quad \hat{\beta}_2 = 4.876209, \quad \hat{\beta}_3 = 14.017639$. According to the assumption $\beta_0 = 11, \quad \beta_1 = 4, \quad \beta_2 = 5, \quad \beta_3 = 14$, can be found that:

$$|\hat{\beta}_0 - \beta_0| = 0.061507, \quad |\hat{\beta}_1 - beta_1| = 0.02472, \quad |\hat{\beta}_2 - beta_2| = 0.123791, \quad |\hat{\beta}_3 - \beta_3| = 0.017639$$

Part d

```
# For forward stepwise selection
best_model1_forward <- regsubsets(y ~ poly(x, 10, raw=TRUE), data = data_set1,
                                  nvmax = 15, method="forward")
summary1_forward <- summary(best_model1_forward)
which.min(summary1_forward$bic)
```

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

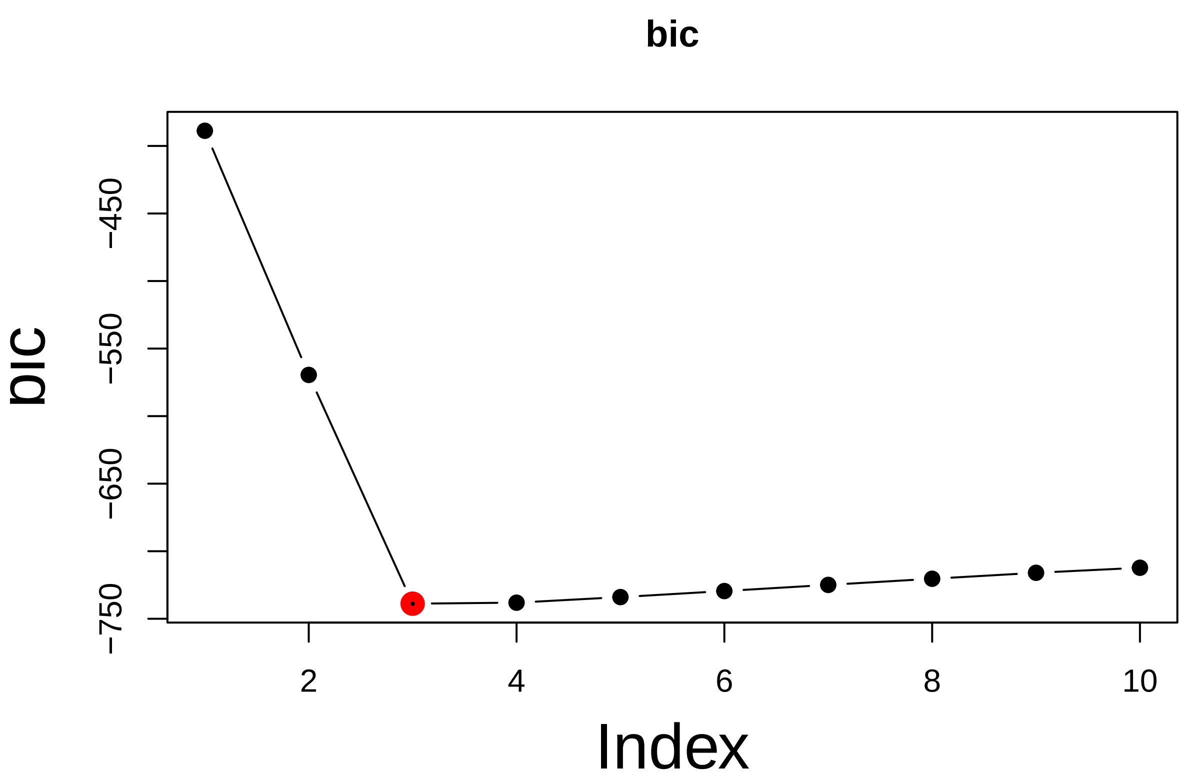
```
which.min(summary1_forward$cp)
```

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

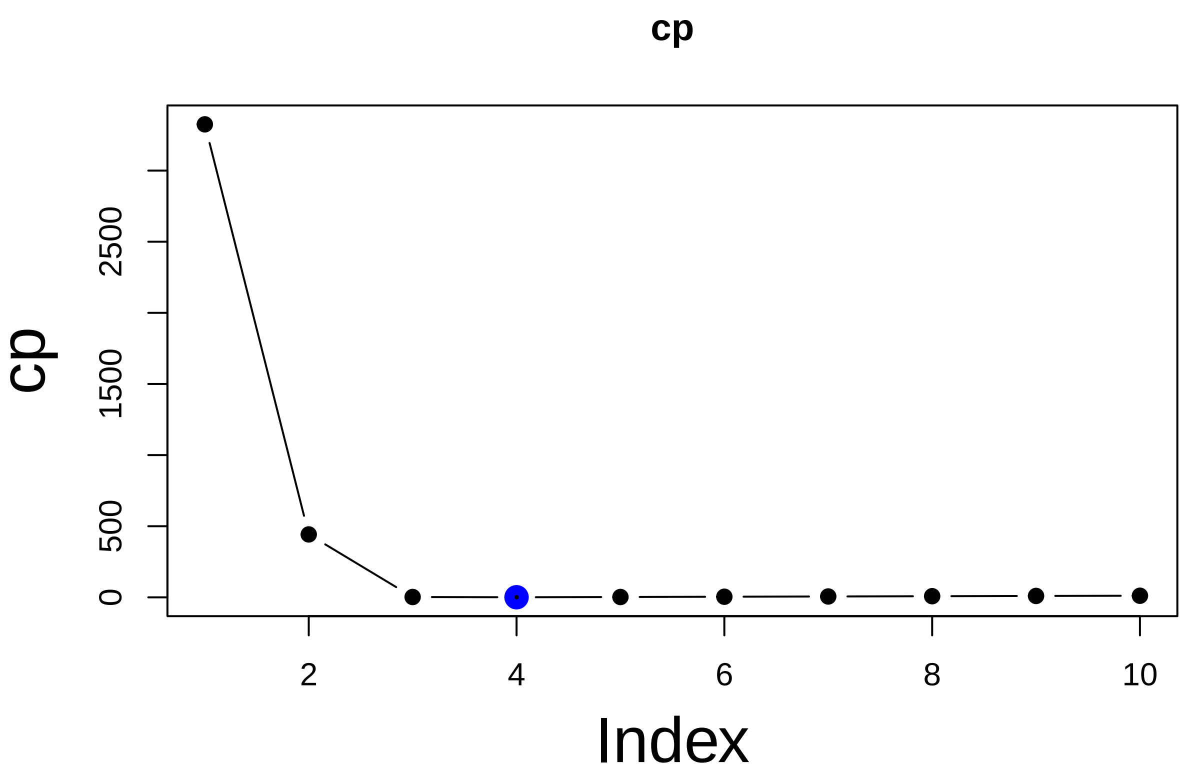
```
which.min(summary1_forward$adjr2)
```

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

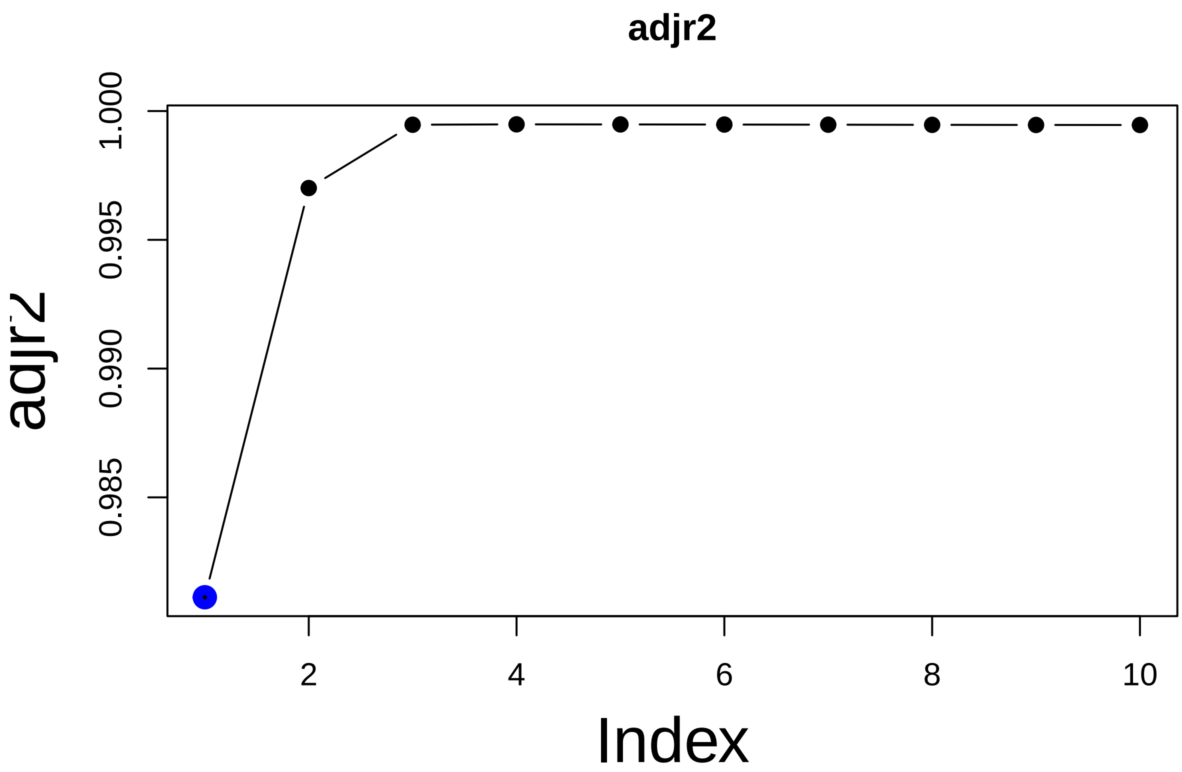
```
# show diagrams for bic, cp and adjr2
plot(summary1_forward$bic, type = "b", pch=19, cex.lab=2, ylab="bic")
points(3, summary1_forward$bic[3], col = "red",lwd=5)
title("bic")
```




```
plot(summary1_forward$cp, type = "b", pch=19, cex.lab=2, ylab="cp")
points(4, summary1_forward$cp[4], col = "blue",lwd=5)
title("cp")
```




```
plot(summary1_forward$adjr2, type = "b", pch=19, cex.lab=2, ylab = "adjr2")
points(1, summary1_forward$adjr2[1], col = "blue",lwd=5)
title("adjr2")
```




```
# generate hat beta 1 to 3
coefficients(best_model1_forward, id=3)
```

```
##              (Intercept) poly(x, 10, raw = TRUE)1 poly(x, 10, raw = TRUE)2
##                11.061507                3.975280                4.876209
## poly(x, 10, raw = TRUE)3
##                14.017639
```

The output for the forward stepwise selection method, the coefficient for $\hat{\beta}_0$, $\hat{\beta}_1$, $\hat{\beta}_2$, $\hat{\beta}_3$ are nearly the same.

```
# For backward stepwise selection
best_model1_backward <- regsubsets(y ~ poly(x, 10, raw=TRUE), data = data_set1,
                                   nvmax = 15, method="backward")
summary1_backward <- summary(best_model1_backward)
which.min(summary1_backward$bic)
```

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

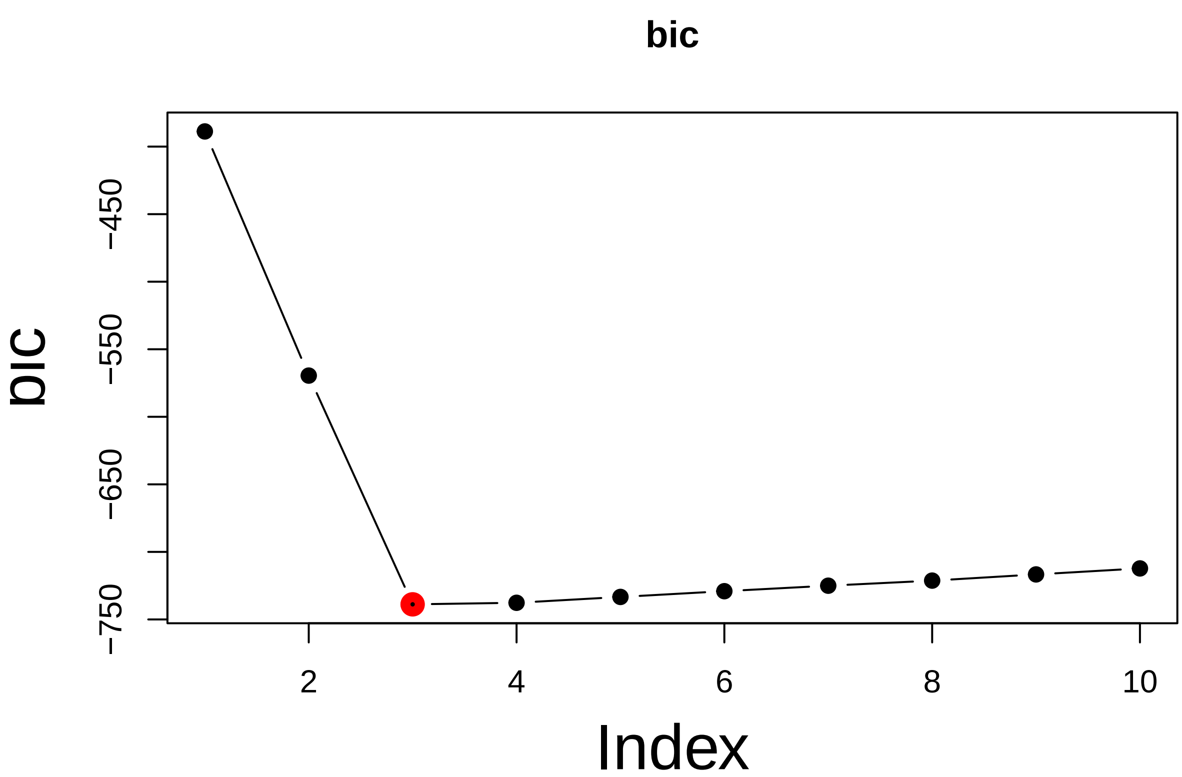
```
which.min(summary1_backward$cp)
```

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

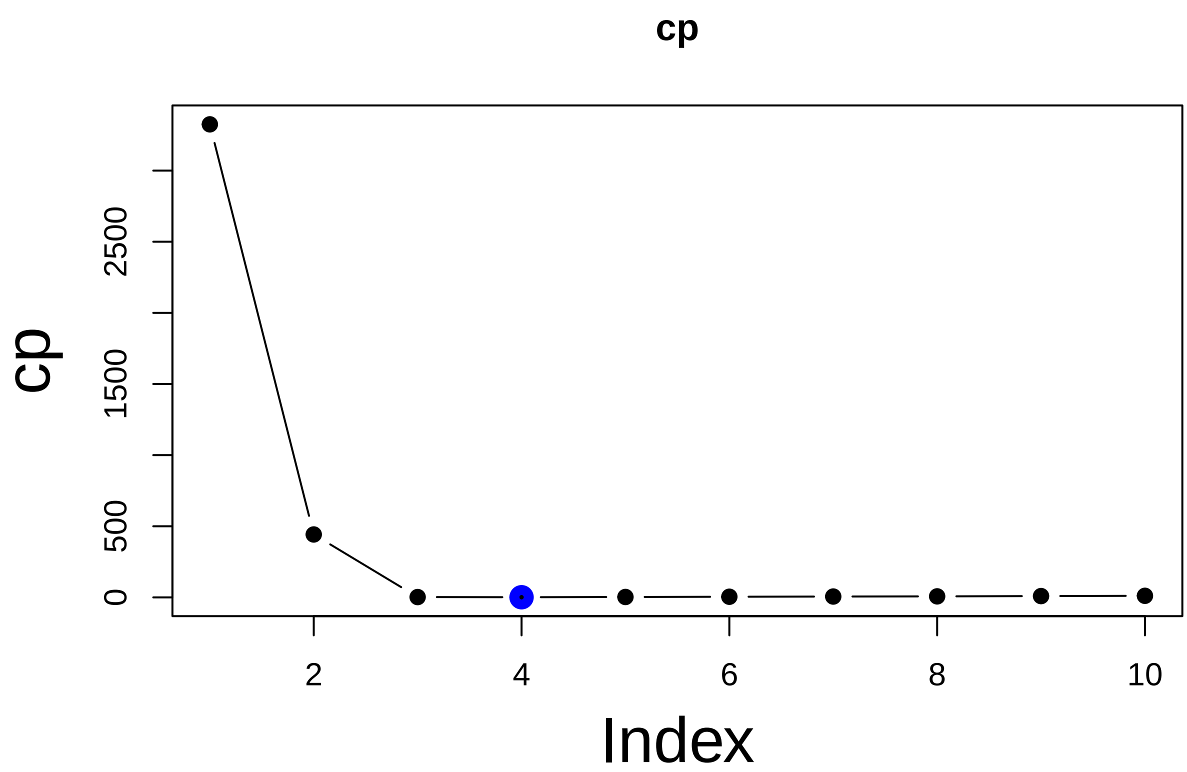
```
which.min(summary1_backward$adjr2)
```

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

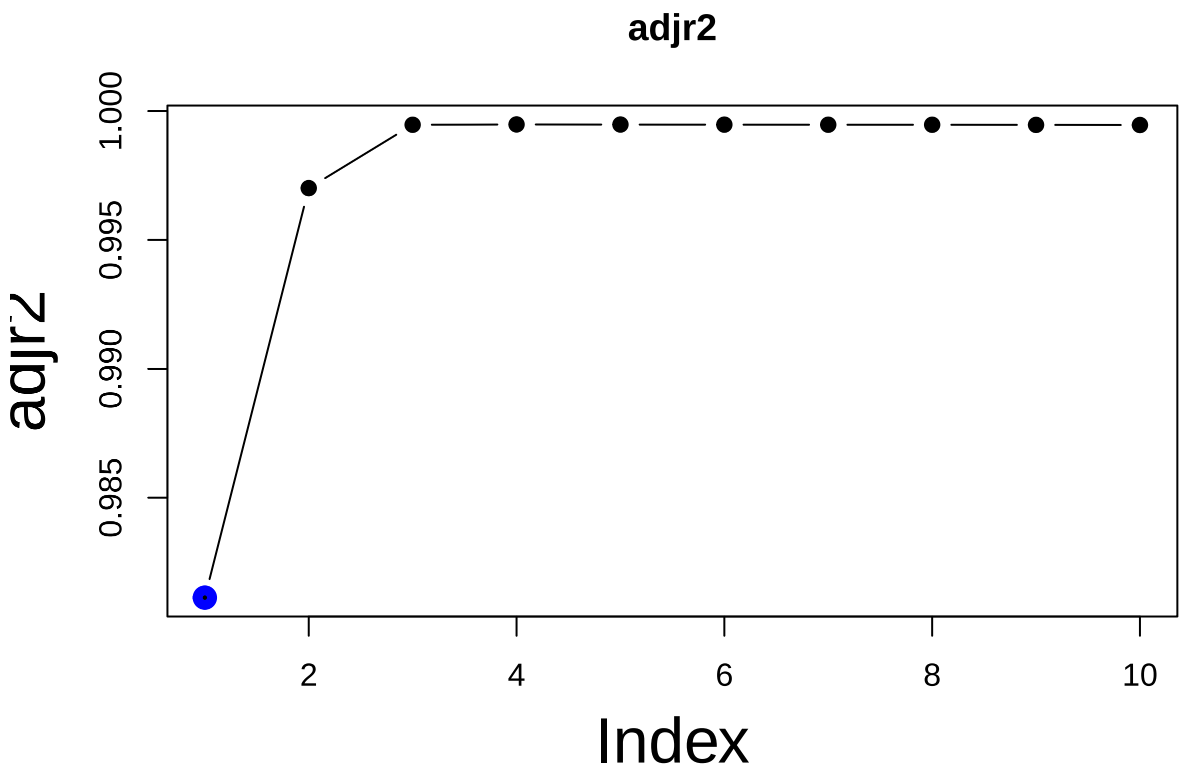
```
# show diagrams for bic, cp and adjr2
plot(summary1_backward$bic, type = "b", pch=19, cex.lab=2, ylab="bic")
points(3, summary1_backward$bic[3], col = "red",lwd=5)
title("bic")
```




```
plot(summary1_backward$cp, type = "b", pch=19, cex.lab=2, ylab="cp")
points(4, summary1_backward$cp[4], col = "blue",lwd=5)
title("cp")
```




```
plot(summary1_backward$adjr2, type = "b", pch=19, cex.lab=2, ylab = "adjr2")
points(1, summary1_backward$adjr2[1], col = "blue",lwd=5)
title("adjr2")
```




```
# generate hat beta 1 to 3
coefficients(best_model1_backward, id=3)
```

```
##              (Intercept) poly(x, 10, raw = TRUE)1 poly(x, 10, raw = TRUE)2
##                11.061507                 3.975280                 4.876209
## poly(x, 10, raw = TRUE)3
##                14.017639
```

The output for the backward stepwise selection method, the coefficient for $\hat{\beta}_0$, $\hat{\beta}_1$, $\hat{\beta}_2$, $\hat{\beta}_3$ are also nearly the same.

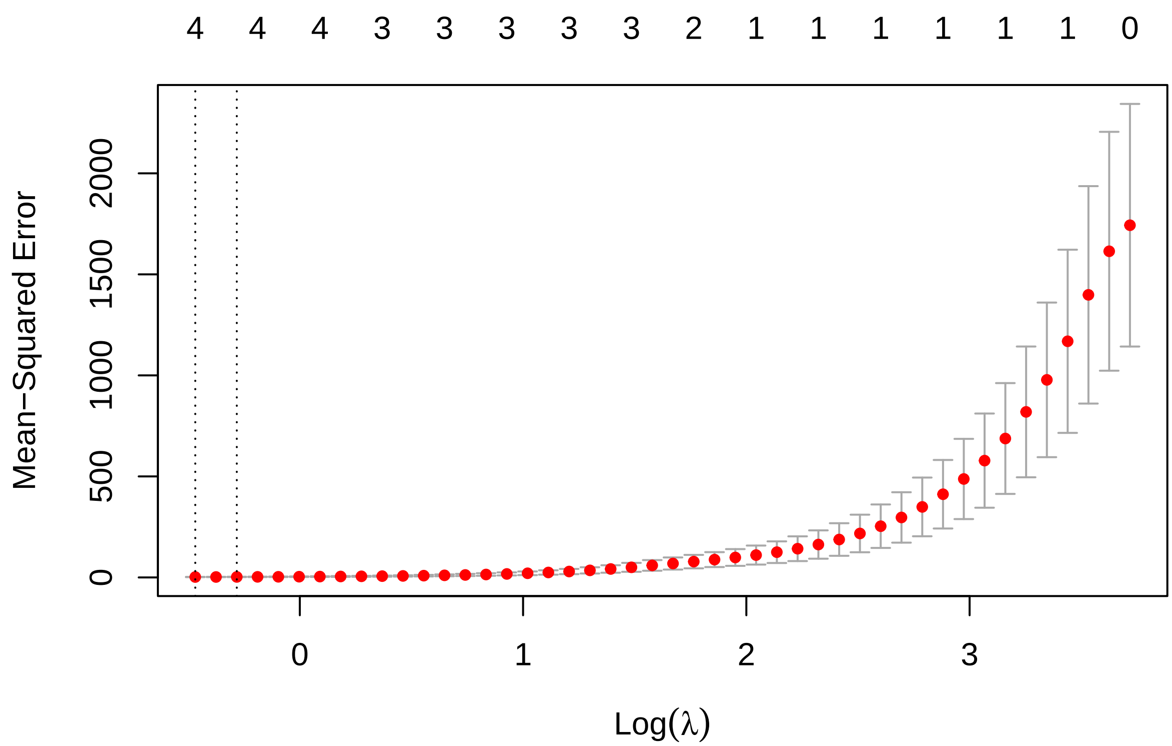
Part e

```
x4e <- model.matrix(y ~ poly(x, 10, raw = TRUE), data = data_set1)[, -1]
library(glmnet)
```

```
##      Matrix
```

```
## Loaded glmnet 4.1-4
```

```r
lasso_model2 <- cv.glmnet(x4e, y)
plot(lasso_model2)
```




```r
model_4e <- glmnet(x4e, y)
lasso_model2$lambda.min
```

```
## [1] 0.6260777
```

```r
predict(model_4e, s = lasso_model2$lambda.min, type = "coefficients")
```

```
## 11 x 1 sparse Matrix of class "dgCMatrix"
##                                  s1
## (Intercept)             11.54371736
## poly(x, 10, raw = TRUE)1   3.49067649
## poly(x, 10, raw = TRUE)2   4.30272660
## poly(x, 10, raw = TRUE)3  13.95485504
## poly(x, 10, raw = TRUE)4   0.02489254
## poly(x, 10, raw = TRUE)5   .
## poly(x, 10, raw = TRUE)6   .
## poly(x, 10, raw = TRUE)7   .
## poly(x, 10, raw = TRUE)8   .
## poly(x, 10, raw = TRUE)9   .
## poly(x, 10, raw = TRUE)10  .
```

In this approach, β_1 to β_4 will be selected.

Part f

```
# Y=beta0+beta7*x^7+epsilon
beta7 <- 15
y2 <- beta0+beta7*x^7+epsilon
dataset_4f <- data.frame(y = y2, x = x)
model_4f <- regsubsets(y ~ poly(x, 10, raw = TRUE), data = dataset_4f, nvmax = 15)
model_4f_summary <- summary(model_4f)
which.min(model_4f_summary$bic)
```

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

```
coefficients(model_4f, id=1)
```

```
##              (Intercept) poly(x, 10, raw = TRUE)7
##                 10.95894                 15.00077
```

```
which.min(model_4f_summary$cp)
```

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

```
coefficients(model_4f, id=2)
```

```
##              (Intercept) poly(x, 10, raw = TRUE)2 poly(x, 10, raw = TRUE)7
##               11.0704904               -0.1417084               15.0015552
```

```
which.min(model_4f_summary$adjr2)
```

```
## [1] 10
```

```
coefficients(model_4f, id=10)
```

```
##               (Intercept)  poly(x, 10, raw = TRUE)1  poly(x, 10, raw = TRUE)2
##               11.17282867                0.51409233               -1.13146007
##  poly(x, 10, raw = TRUE)3  poly(x, 10, raw = TRUE)4  poly(x, 10, raw = TRUE)5
##               -0.93113515                1.90382807                0.55109577
##  poly(x, 10, raw = TRUE)6  poly(x, 10, raw = TRUE)7  poly(x, 10, raw = TRUE)8
##               -1.26499408               14.84430680                0.31986888
##  poly(x, 10, raw = TRUE)9 poly(x, 10, raw = TRUE)10
##                0.01627747               -0.02690171
```

```
x_4f <- model.matrix(y ~ poly(x, 10, raw = TRUE), data = dataset_4f)[, -1]
lasso_model_4f <-cv.glmnet(x_4f, y2, alpha=1)
best_model_4f <- glmnet(x_4f, y2)
predict(best_model_4f, s=lasso_model_4f$lambda.min, type="coefficients")
```

```
## 11 x 1 sparse Matrix of class "dgCMatrix"
##                                     s1
## (Intercept)                   12.80442
## poly(x, 10, raw = TRUE)1      .
## poly(x, 10, raw = TRUE)2      .
## poly(x, 10, raw = TRUE)3      .
## poly(x, 10, raw = TRUE)4      .
## poly(x, 10, raw = TRUE)5      .
## poly(x, 10, raw = TRUE)6      .
## poly(x, 10, raw = TRUE)7  14.56349
## poly(x, 10, raw = TRUE)8      .
## poly(x, 10, raw = TRUE)9      .
## poly(x, 10, raw = TRUE)10     .
```

Question 5

Part a

```
library(ISLR)
set.seed(20)
training_set <-sample(1:nrow(College), nrow(College) / 1.25)
test_set <- -training_set
```

Part b

```
# Applications fitting
lm.fit <- lm(Apps~., data=College[training_set, ])
lm.pred <- predict(lm.fit, College[test_set, ])
mean((College[test_set, "Apps"] - lm.pred)^2)
```

```
## [1] 862173.9
```

Part c

```
training_matrix <- model.matrix(Apps~., data=College[training_set, ])
test_matrix <- model.matrix(Apps~., data=College[test_set, ])
grid <-10 ^ seq(4, -2, length=100)
ridge_5c <- cv.glmnet(training_matrix, College[training_set, "Apps"])
mean((College[test_set, "Apps"] - predict(ridge_5c, newx = test_matrix, s=ridge_5c$lambda.min))^2)
```

```
## [1] 869476.4
```

Where the test error here is higher.

Part d

```r
lasso_5c <- cv.glmnet(training_matrix, College[training_set, "Apps"])
lambda_5c_min <-lasso_5c$lambda.min
mean((College[test_set, "Apps"]-predict(lasso_5c, newx = test_matrix, s=lambda_5c_min))^2)
```

```
## [1] 869383.3
```

```r
model_5c <- glmnet(model.matrix(Apps~., data=College), College[, "Apps"], alpha=1)
predict(model_5c, s=lambda_5c_min, type="coefficients")
```

```
## 19 x 1 sparse Matrix of class "dgCMatrix"
##                        s1
## (Intercept) -617.31880531
## (Intercept)    .
## PrivateYes  -416.97045207
## Accept         1.44708222
## Enroll        -0.17427400
## Top10perc     32.67059035
## Top25perc     -1.70837845
## F.Undergrad    .
## P.Undergrad    0.01828366
## Outstate      -0.05584952
## Room.Board     0.12282498
## Books          .
## Personal       .
## PhD           -5.39587417
## Terminal      -3.33838812
## S.F.Ratio      3.73845191
## perc.alumni   -1.01151800
## Expend         0.06912593
## Grad.Rate      4.95916199
```

Where the test error has no has slightly decrease than part c.

Question 6

Part a

```r
set.seed(38)
p <- 20
n <- 1000
x <- matrix(rnorm(p*n), n, p)
# random for beta has 20 features
beta <- rnorm(20)
# random zero for random beta[]
beta[4] <- 0
beta[7] <- 0
```

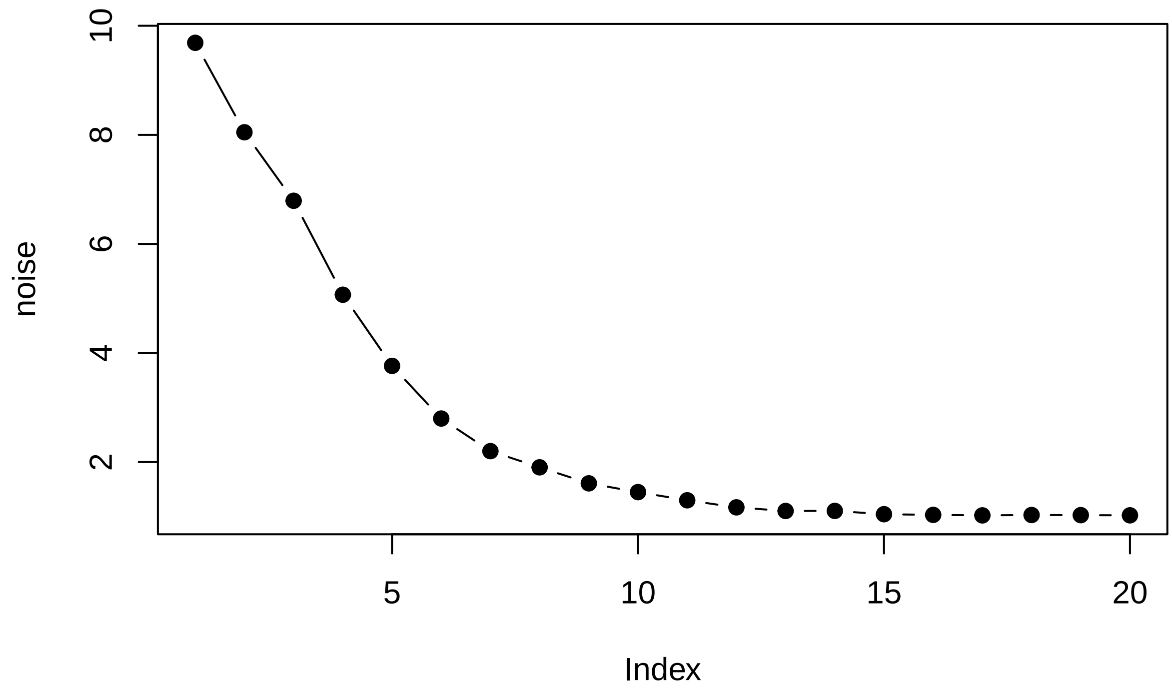
```
beta[11] <- 0
beta[18] <- 0
# random epsilon has 20 features
epsilon <- rnorm(20)
y <- x%*%beta+epsilon
```

Part b

```
# 100 training set
training_set <- sample(seq(1000), 100)
test_set <- -training_set
x_training <- x[training_set, ]
y_training <- y[training_set, ]
x_test <- x[test_set, ]
y_test <- y[test_set, ]
```

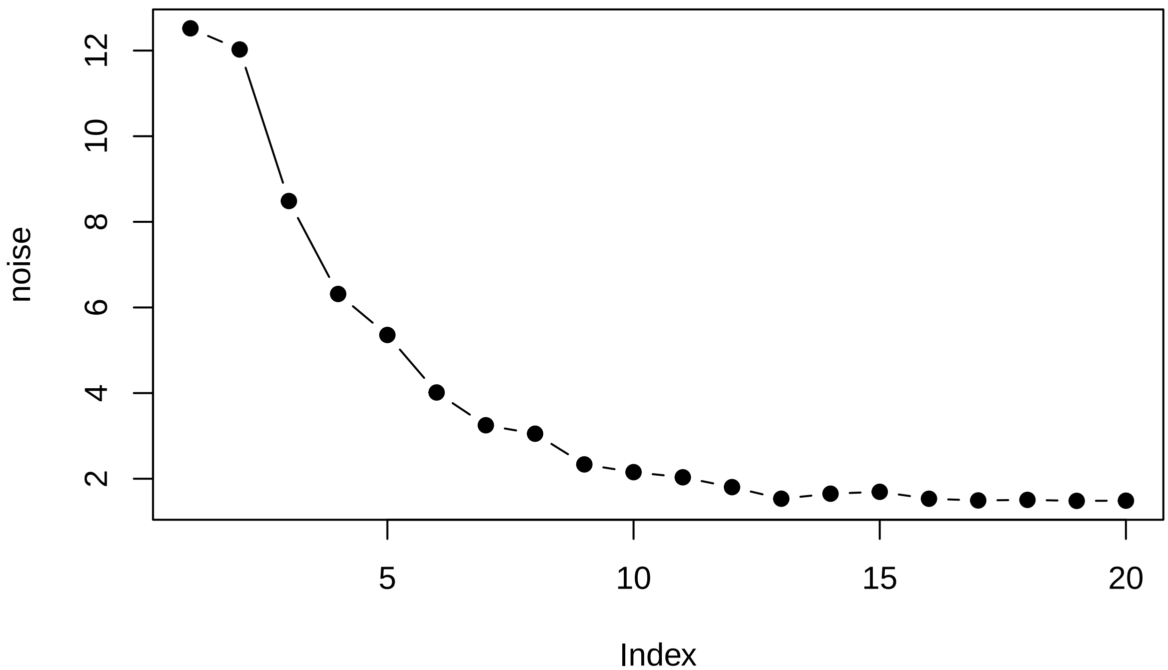
Part c

```
a <- regsubsets(y~., data = data.frame(x=x_training, y=y_training), nvmax=20)
noise <- rep(NA, p)
x_ma <- colnames(x, prefix = "x.", do.NULL = FALSE)
# fit
coeff <- coef(a, id = 1)
prediction <- as.matrix(x_training[, (x_ma %in% names(coeff)) ]) %*% (coeff[names(coeff) %in% x_ma])
noise[1] <- mean((y_training - prediction)^2)
for (i in 2:20) {
  coeff <- coef(a, id = i)
  prediction <- as.matrix(x_training[, (x_ma %in% names(coeff)) ]) %*% (coeff[names(coeff) %in% x_ma])
  noise[i] <- mean((y_training - prediction)^2)
}
plot(noise, type = "b",pch=19)
```



Part d

```
a <- regsubsets(y~., data = data.frame(x=x_training, y=y_training), nvmax=20)
noise <- rep(NA, p)
x_ma <- colnames(x, prefix = "x.", do.NULL = FALSE)
# fit
for (i in 1:20) {
  coeff <- coef(a, id = i)
  prediction <- as.matrix(x_test[, (x_ma %in% names(coeff)) ]) %*% (coeff[names(coeff) %in% x_ma])
  noise[i] <- mean((y_test - prediction)^2)
}
plot(noise, type = "b",pch=19)
```



Part e

```r
noise
```

```
##  [1] 12.519491 12.025774  8.486354  6.315274  5.357798  4.014332  3.248884
##  [8]  3.051739  2.334890  2.153535  2.033621  1.803766  1.533372  1.650507
## [15]  1.694321  1.532701  1.492700  1.506103  1.483087  1.487239
```

```r
which.min(noise)
```

```
## [1] 19
```

```r
noise[which.min(noise)]
```

```
## [1] 1.483087
```

Where the 19th model size is on the minimum value. Since is not an intercept or containing all features, so no need to repeat part a.

Part f

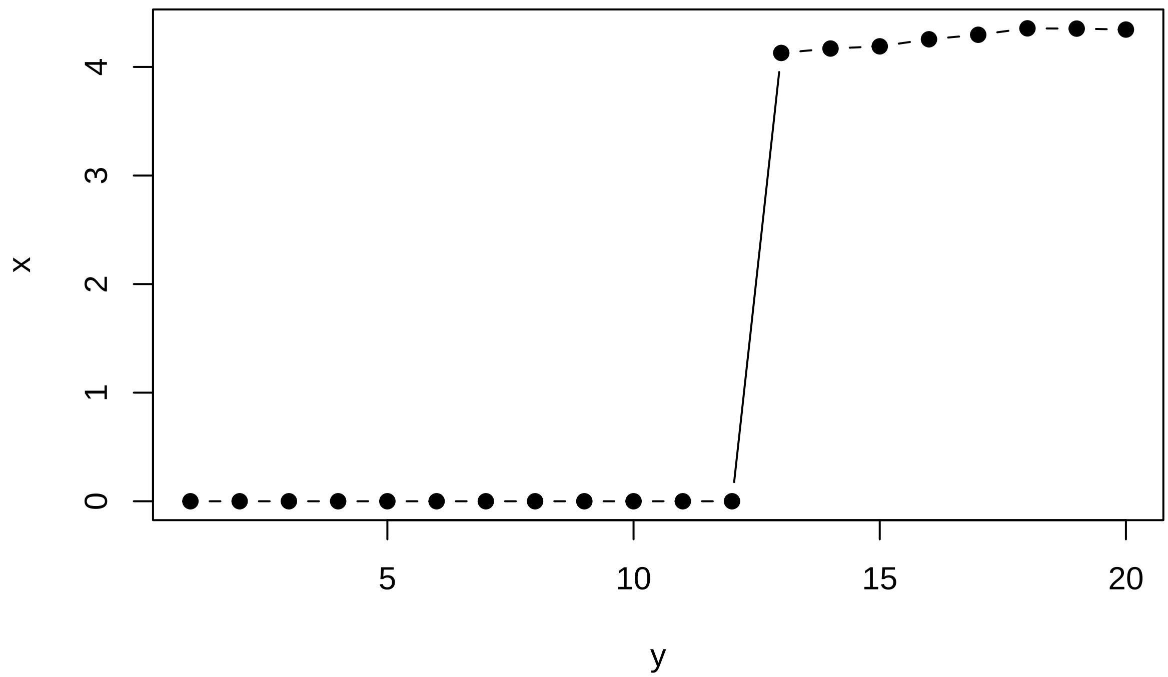
```r
coef(a, id=19)
```

```
## (Intercept)         x.1         x.2         x.3         x.4         x.5
## -0.39303149 -0.40074129  0.22650904 -0.75593406  0.08579493  1.80240193
##         x.6         x.7         x.8         x.9        x.10        x.12
## -1.39724738  0.24051604  0.96107252 -0.16141556 -1.57810058  0.38120788
##        x.13        x.14        x.15        x.16        x.17        x.18
##  1.11553122 -0.32218466  0.05228901  1.09364215  0.78965349  0.15460586
##        x.19        x.20
##  0.42554814  0.49562110
```

The coefficient values are all different, where the biggest is x.13, and the smallest is x.10

Part g

```r
a <- regsubsets(y~., data = data.frame(x=x_training, y=y_training), nvmax=20)
noise <- rep(NA, p)
x <- 0
y <- 0
x_ma <- colnames(x, prefix = "x.", do.NULL = FALSE)
for (i in 1:20) {
  coeff <- coef(a, id = i)
  x[i] <- sqrt(sum((beta[x_ma %in% names(coeff)] - coeff[names(coeff) %in% x_ma])^2))
  y[i] <- length(coeff)-1
  noise[i] <- mean((y_test - prediction)^2)
}
plot(x=y, y=x, type = "b",pch=19)
```



From the diagram, the error value is increase to non-zero since the 13th, then it continuesly increase, which has the reverse trend as diagram in part d.