Individual_assignment6

Yuhan_Xu_474154

2019/10/4

Problem 8

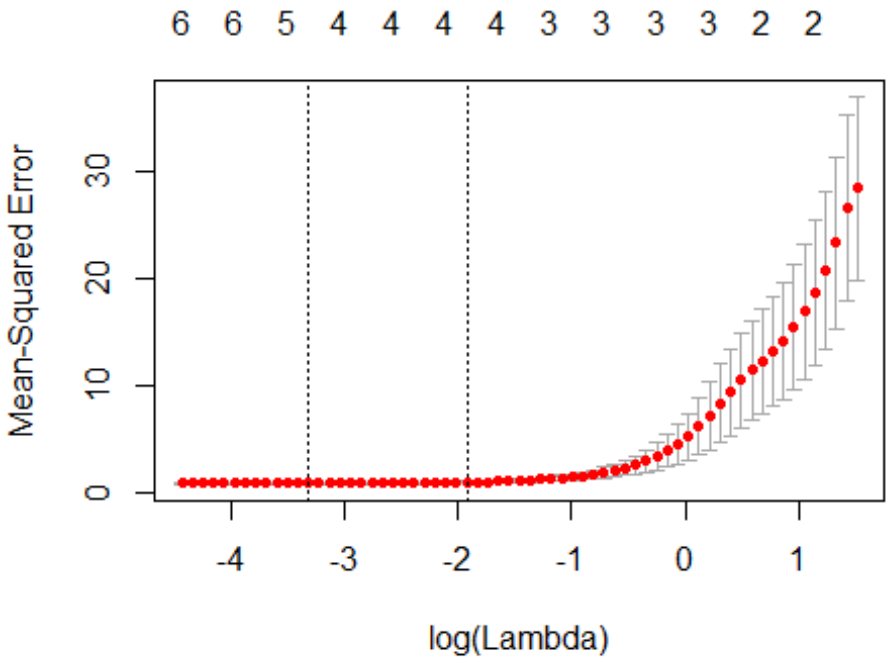
For Context, Refer to Problem 8 (parts a, b, c, & d).

```
set.seed(1)
X = rnorm(100)
noise = rnorm(100)
Y = 4 + 3*X - 2*X^2 + 1*X^3 + noise
xydata = data.frame(Y, X)
```

Q: (e)Now fit a lasso model to the simulated data, again using X, X^2,..., X^{10} as predictors. Use cross-validation to select the optimal value of λ. Create plots of the cross-validation error as a function of λ. Report the resulting coefficient estimates, and discuss the results obtained.

A:

```
library(glmnet)

## Loading required package: Matrix

## Loading required package: foreach

## Loaded glmnet 2.0-18

grid = 10 ^ seq(10, -2, length=100)

xdata = model.matrix(Y~poly(X, 10, raw = TRUE), data = xydata)[,-1]

cv.out = cv.glmnet(xdata, Y, alpha = 1)
plot(cv.out)
```




```
bestlam = cv.out$lambda.min
bestlam

## [1] 0.0361852

out = glmnet(xdata, Y, alpha = 1, lambda = grid)
lasso.coef = predict(out, type = "coefficients", s = bestlam, x = xdata,
 y = Y, exact = TRUE)
lasso.coef

## 11 x 1 sparse Matrix of class "dgCMatrix"
##                                       1
## (Intercept)                4.042291e+00
## poly(X, 10, raw = TRUE)1   3.277062e+00
## poly(X, 10, raw = TRUE)2  -2.108148e+00
## poly(X, 10, raw = TRUE)3   6.480788e-01
## poly(X, 10, raw = TRUE)4   .
## poly(X, 10, raw = TRUE)5   6.289128e-02
## poly(X, 10, raw = TRUE)6   .
## poly(X, 10, raw = TRUE)7   3.671476e-07
## poly(X, 10, raw = TRUE)8   .
## poly(X, 10, raw = TRUE)9   .
## poly(X, 10, raw = TRUE)10  .
```

LASSO picks x, x^2, x^3, x^5, x^7 as the predictors.

Q: (f) Now generate a response vector Y according to the model

$Y = \beta_0 + \beta_7 X^7 + \epsilon$,

and perform best subset selection and the lasso. Discuss the results obtained.

A: when $\beta_0 = 4$, $\beta_7 = 5$:

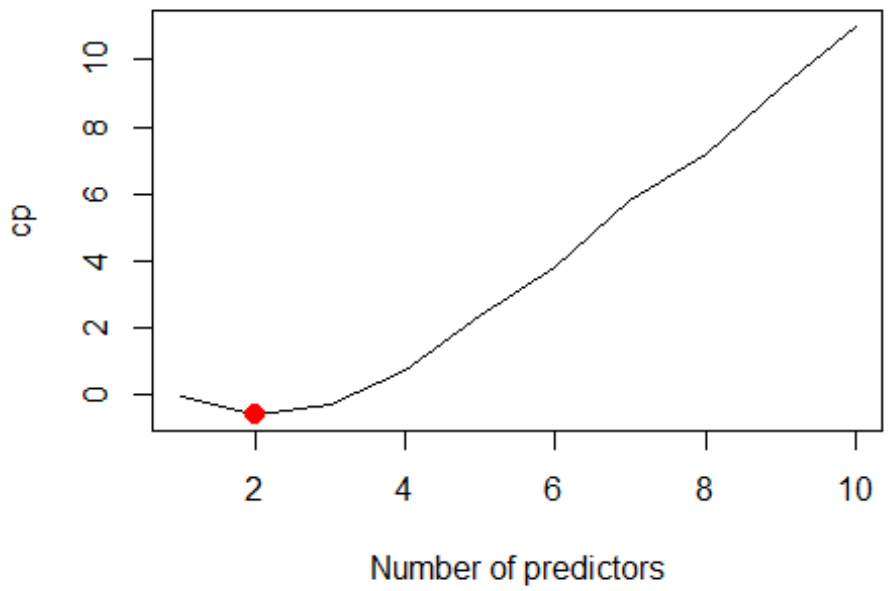
```
Y1 = 4 + 5*X^7 + noise
```

Best subset selection

```
xy.data = data.frame(Y1, X)

library("leaps")
regfit.full =regsubsets(Y1~poly(X, 10, raw = TRUE), data = xy.data, nvm
ax = 10)
reg.summary = summary(regfit.full)
names(reg.summary)

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

Cp

```
plot(reg.summary$cp, xlab = "Number of predictors", ylab = "cp", type =
 "l")
which.min(reg.summary$cp)

## [1] 2

points(2, reg.summary$cp[2], col = "red", cex = 2, pch = 20)
```



When there are 2 predictors, we can get the model with the smallest Cp. The predictor and coefficients of the predictor are:

```
coef(regfit.full, 2)
```

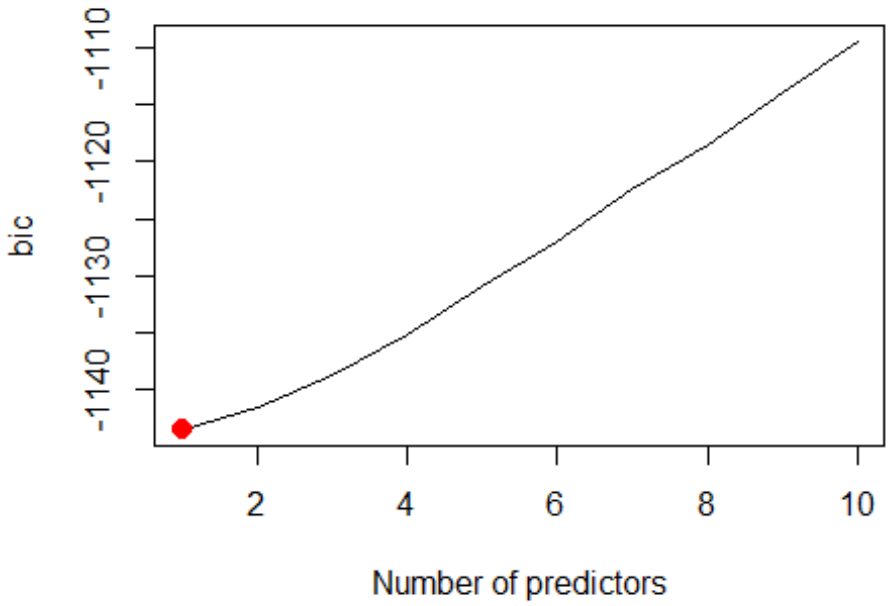
```
##               (Intercept) poly(X, 10, raw = TRUE)2 poly(X, 10, raw =
TRUE)7
##                 4.0704904               -0.1417084                5.0
015552
```

BIC

```
plot(reg.summary$bic, xlab = "Number of predictors", ylab = "bic", type
 = "l")
which.min(reg.summary$bic)
```

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

```
points(1, reg.summary$bic[1], col = "red", cex = 2, pch = 20)
```



When there is only 1 predictor, we can get the model with the smallest BIC. The predictor and coefficients of the predictor are:

```
coef(regfit.full, 1)
```

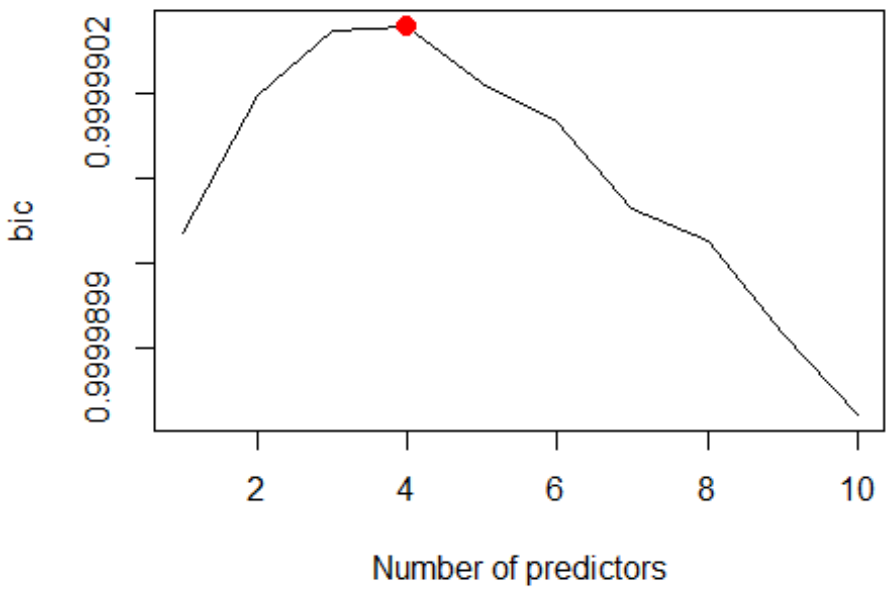
```
##               (Intercept) poly(X, 10, raw = TRUE)7
##                   3.95894                   5.00077
```

Adjusted R2

```
plot(reg.summary$adjr2, xlab = "Number of predictors", ylab = "bic", ty
pe = "l")
which.max(reg.summary$adjr2)
```

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

```
points(4, reg.summary$adjr2[4], col = "red", cex = 2, pch = 20)
```



When there are 4 predictors, we can get the model with the largest adjusted R2. The predictor and coefficients of the predictor are:

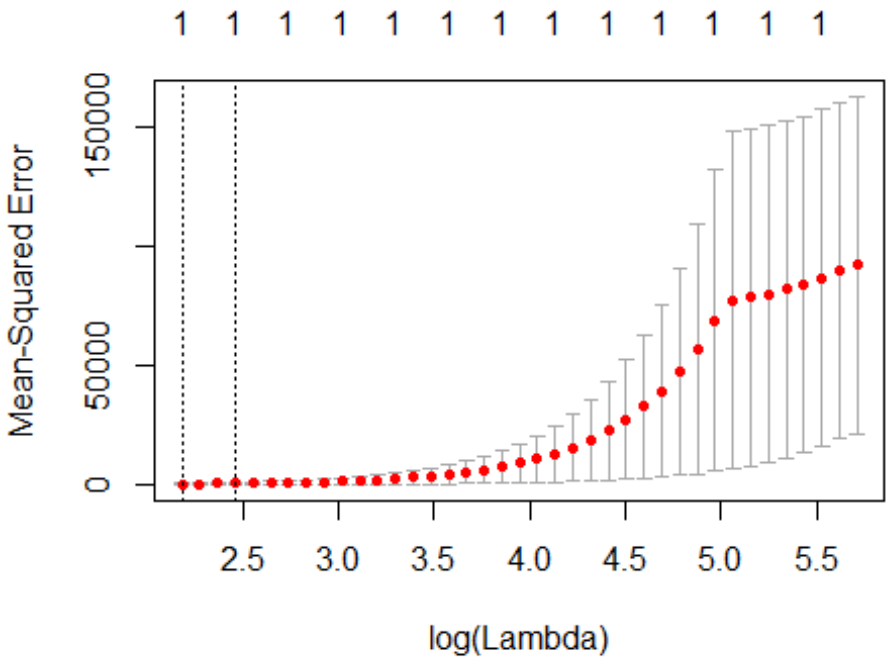
```
coef(regfit.full, 4)
```

```
##               (Intercept) poly(X, 10, raw = TRUE)1 poly(X, 10, raw =
TRUE)2
##                 4.0762524                 0.2914016                -0.1
617671
## poly(X, 10, raw = TRUE)3 poly(X, 10, raw = TRUE)7
##                -0.2526527                 5.0091338
```

In a conclusion, this time Cp, BIC and adjusted R^2 choosed model with different predictors. But these three model all included X and X^7, and the coefficients of these two predictors are similar to the betas in true function.

LASSO

```
x.data = model.matrix(Y1~poly(X, 10, raw = TRUE), data = xy.data)[,-1]

cv.out1 = cv.glmnet(x.data, Y1, alpha = 1)
plot(cv.out1)
```




```
bestlam1 = cv.out1$lambda.min
bestlam1

## [1] 8.835273

out1 = glmnet(x.data, Y1, alpha = 1, lambda = grid)
lasso.coef1 = predict(out1, type = "coefficients", s = bestlam1, x = x.
data, y = Y1, exact = TRUE)
lasso.coef1

## 11 x 1 sparse Matrix of class "dgCMatrix"
##                                      1
## (Intercept)                 4.574163
## 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    4.854995
## poly(X, 10, raw = TRUE)8    .
## poly(X, 10, raw = TRUE)9    .
## poly(X, 10, raw = TRUE)10   .
```

LASSO only picks x^7 as the predictor, which agrees with the true function. But the coefficients of intercept and predictor are somewhat different from the true function.