Chapter 6 Problem 8

Andira Putri

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

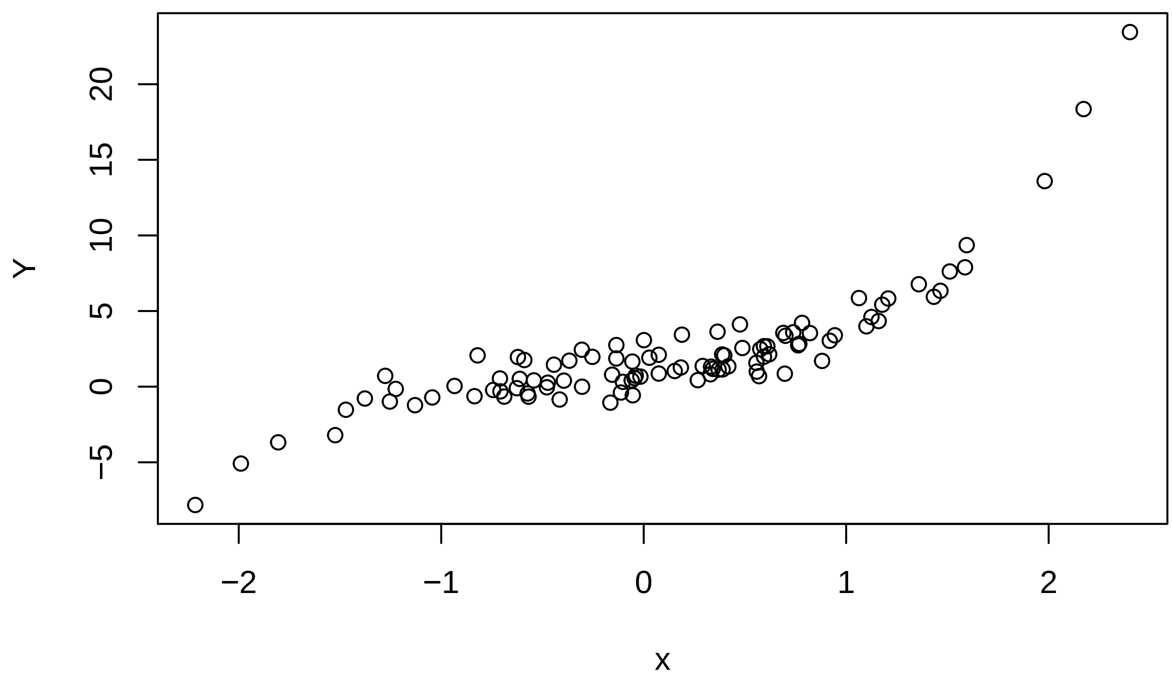
a. Use the `rnorm()` function to generate a predictor `X` of length n=100, as well as a noise vector `eps` of length n=100.

```
set.seed(1)
x=rnorm(100,0,1)
eps=rnorm(100)
```

b. Generate a response vector `Y` of length n=100 according to the model:

$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \epsilon$

```
# coefficients are constants of my choice...they are all 1 :-)
Y=1+1*x+1*x^2+1*x^3+eps
plot(x,Y)
```



The model has an apparent cubic form, which makes sense given the equation of Y.

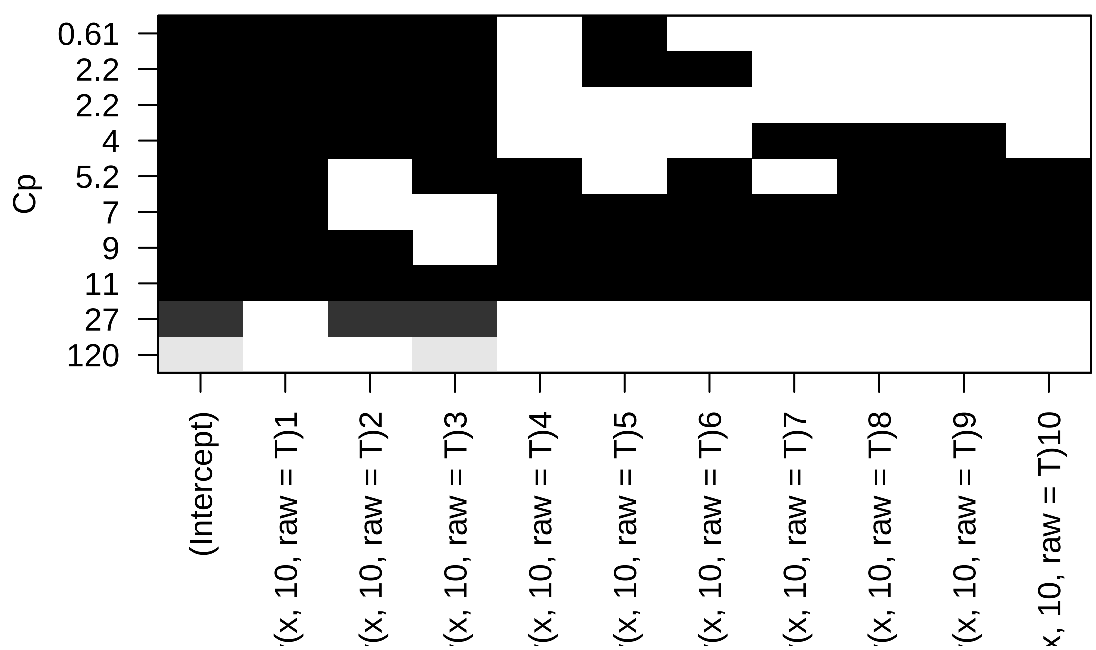
c. Use the `regsubsets()` function to perform best subset selection in order to choose the best model containing 10 variables. What is the best model obtained using Cp, BIC, and adjusted

R-squared? Show some plots to provide evidence, and report the coefficients of the best model obtained.

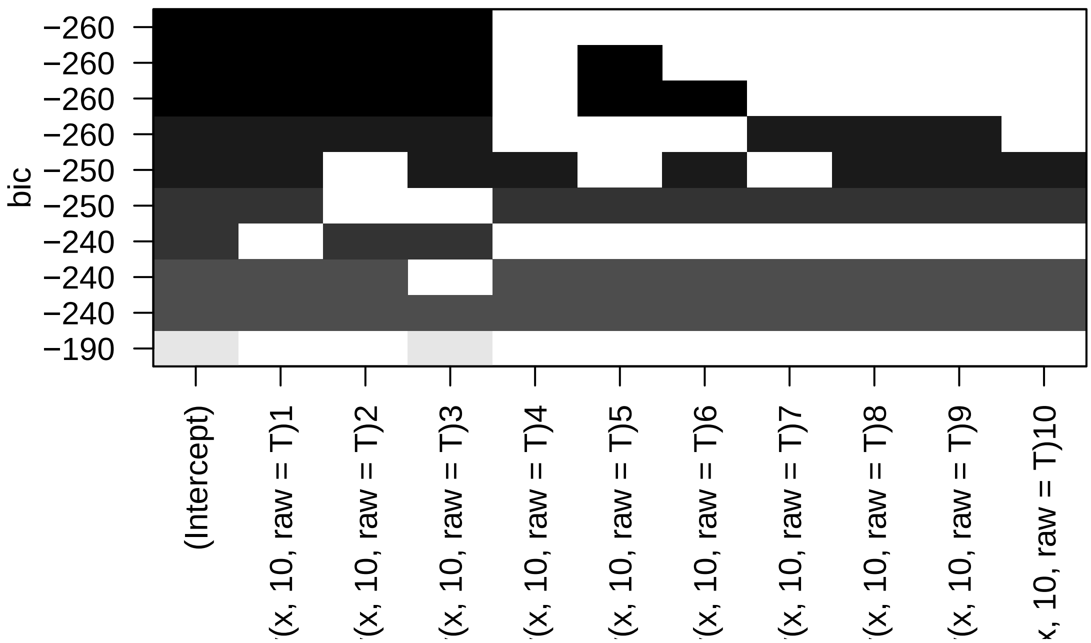
For my benefit, I used this as reference on how to use this function:

http://www2.hawaii.edu/~taylor/z632/Rbestsubsets.pdf

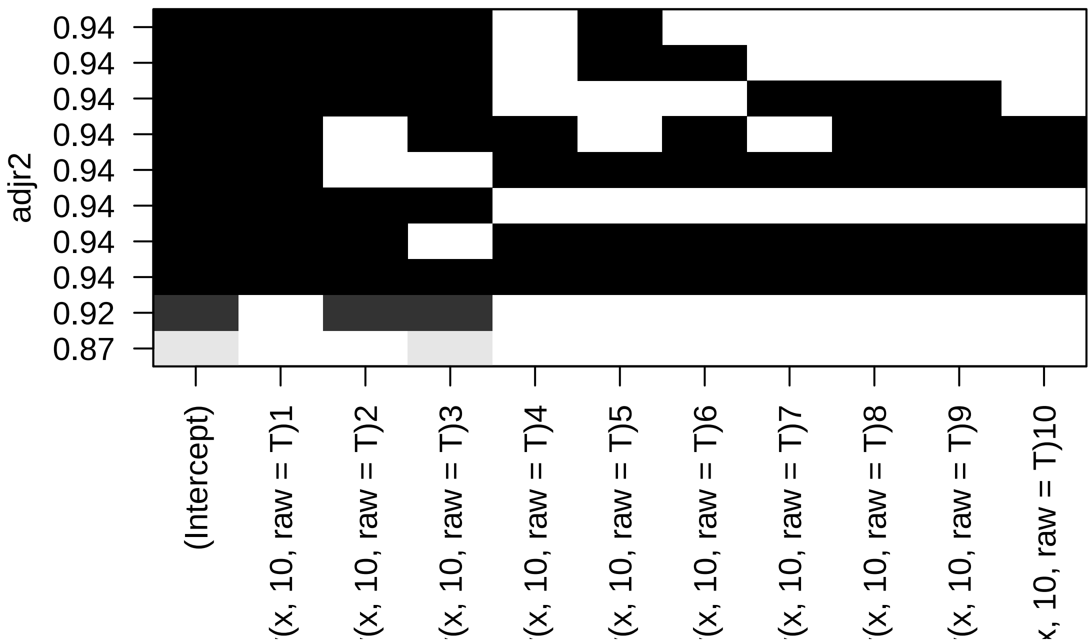
```
set.seed(1)
df=data.frame(x,Y)
library(leaps)
bss=regsubsets(Y~poly(x,10,raw=T),data=df,nvmax=10)
sum=summary(bss)

#We use these plots to evaluate Cp, BIC, and adjusted R-sq
plot(bss, scale = "Cp")
```




```
plot(bss, scale = "bic")
```




```
plot(bss, scale = "adjr2")
```




```r
Cp=which.min(sum$cp) #Choose model w/ lowest Cp
bic=which.min(sum$bic) #Choose model w/ lowest BIC
r2=which.max(sum$adjr2) #Choose model w/ highest adjR-sq

Cp
```

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

```r
bic
```

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

```r
r2
```

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

The model of degree 3 is the best according to analysis using BIC, but C_p and adjusted R^2 suggest that the model up to degree 4 is the best.

Combining the results and my intuition, the best model would be the one with degree 3. The coefficients for that model are here:

```r
coef(bss,bic)
```

```
##          (Intercept) poly(x, 10, raw = T)1 poly(x, 10, raw = T)2
##            1.0615072             0.9752803             0.8762090
## poly(x, 10, raw = T)3
##            1.0176386
```

d. Repeat (c) using forward and backwards stepwise selection. How does your answer compare to (c)?

```
fwd=regsubsets(Y~poly(x,10,raw=T),data=df,nvmax=10,method="forward")
sumf=summary(fwd)
which.min(sumf$cp) #Choose model w/ lowest Cp
```

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

```
which.min(sumf$bic) #Choose model w/ lowest BIC
```

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

```
which.max(sumf$adjr2) #Choose model w/ highest adjR-sq
```

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

```
bwd=regsubsets(Y~poly(x,10),data=df,nvmax=10,method="backward")
sumb=summary(bwd)
which.min(sumb$cp) #Choose model w/ lowest Cp
```

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

```
which.min(sumb$bic) #Choose model w/ lowest BIC
```

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

```
which.max(sumb$adjr2) #Choose model w/ highest adjR-sq
```

```
## [1] 5
```

In terms of forward selection, the results are the exact same as (c) in terms of C_p, BIC, and adjusted-R^2. With backwards selection, C_p and BIC results remain the same, but adjusted-R^2 is highest in the model of degree 5.

e. Now fit a lasso model to the simulated data, again using 10 predictors. Use cross-validation to select the optimal value of tuning parameter. Report the resulting coefficient estimates, and discuss the results obtained.

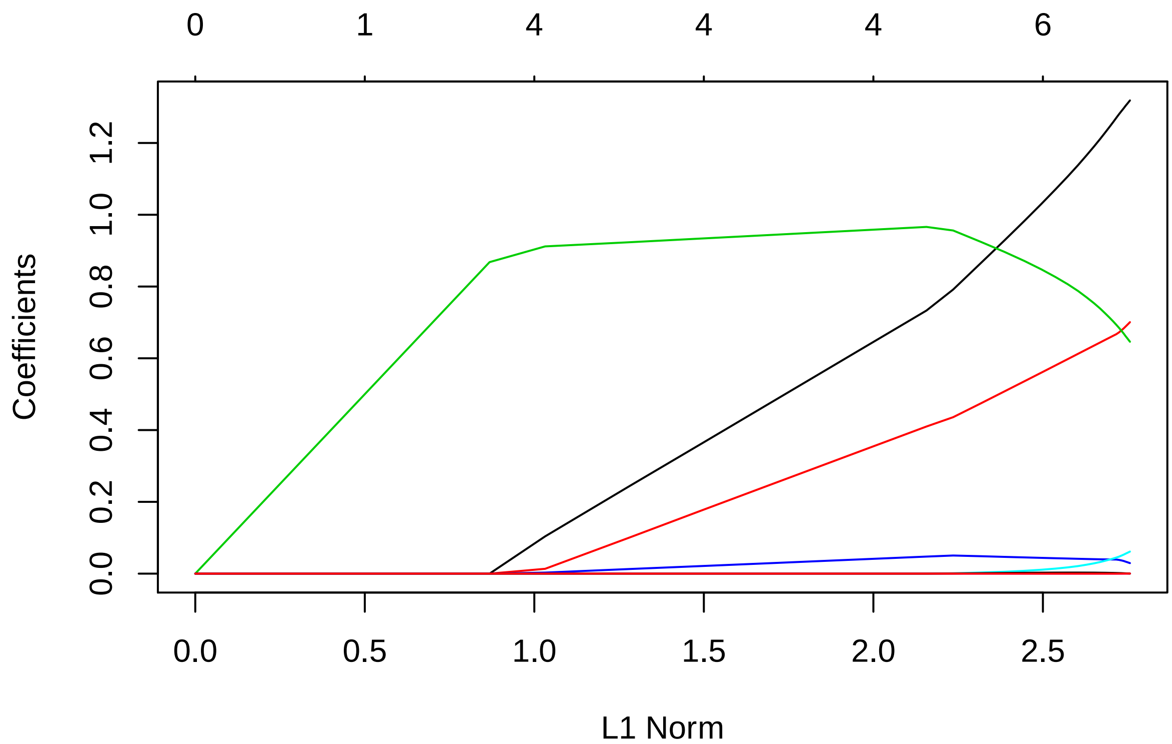
```
set.seed(1)
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loading required package: foreach
```

```
## Loaded glmnet 2.0-16
```

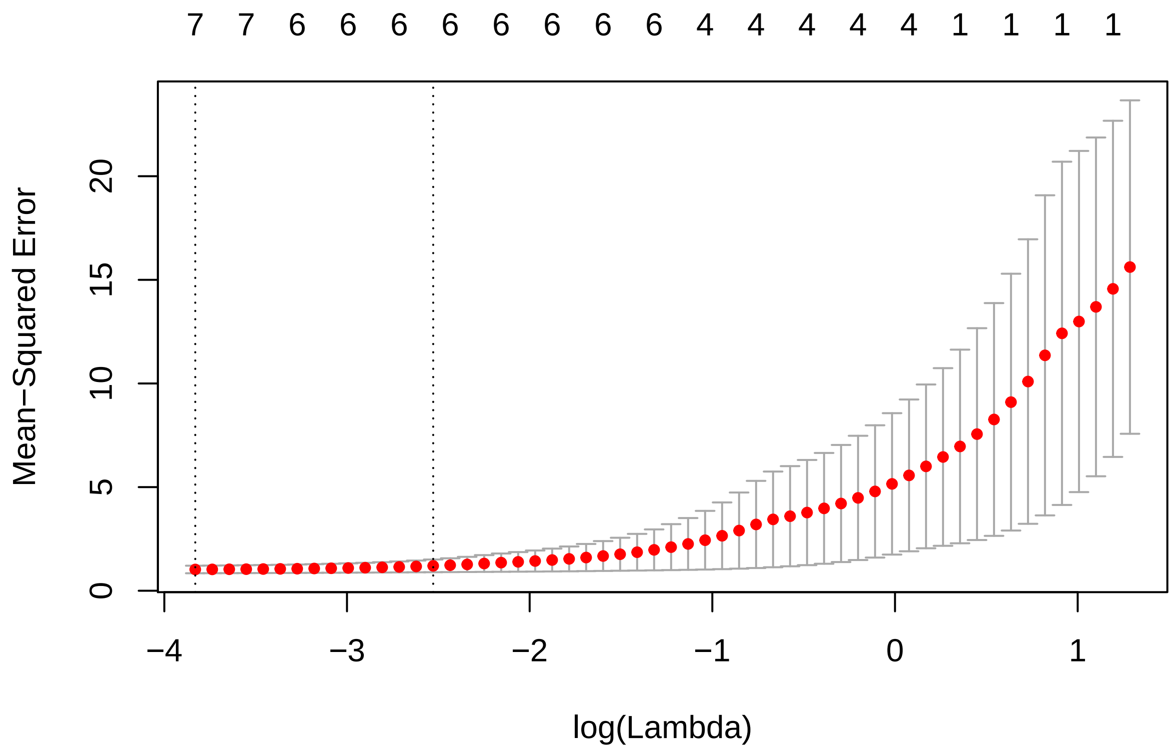
```
#grid of lambdas from 10^10 to 10^-2
grid=10^seq(5,-2,length=100)
#initialize 10 variables
xtrain=model.matrix(Y~poly(x,10,raw=T))[,-1]
#for lasso, alpha=1 (0 for ridge reg.)
lasso=glmnet(xtrain,Y,alpha=1,lambda=grid)
plot(lasso)
```




```
#CV
lasso.cv = cv.glmnet(xtrain, Y, alpha = 1)
opt.lambda = lasso.cv$lambda.min
opt.lambda
```

```
## [1] 0.02169634
```

```
plot(lasso.cv)
```




```
#get coefficients using best lambda
predict(lasso, s = opt.lambda, type = "coefficients")
```

```
## 11 x 1 sparse Matrix of class "dgCMatrix"
##                                 1
## (Intercept)            1.138582e+00
## poly(x, 10, raw = T)1  1.291249e+00
## poly(x, 10, raw = T)2  6.795564e-01
## poly(x, 10, raw = T)3  6.742959e-01
## poly(x, 10, raw = T)4  3.627185e-02
## poly(x, 10, raw = T)5  5.101526e-02
## poly(x, 10, raw = T)6  .
## poly(x, 10, raw = T)7  1.211388e-03
## poly(x, 10, raw = T)8  8.186005e-05
## poly(x, 10, raw = T)9  .
## poly(x, 10, raw = T)10 .
```

f. Now generate a response vector Y according to the model below. Perform best subset selection and the lasso. Discuss the results obtained.

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

```
#like the previous model, the coefficients are all 1
#Best Subset Selection
Y1=1+1*x^7+eps
df=data.frame(x,Y1)
bss1=regsubsets(Y1~poly(x,10,raw=T),data=df,nvmax=10)
```

```r
sum=summary(bss1)
which.min(sum$cp) #Choose model w/ lowest Cp
```

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

```r
which.min(sum$bic) #Choose model w/ lowest BIC
```

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

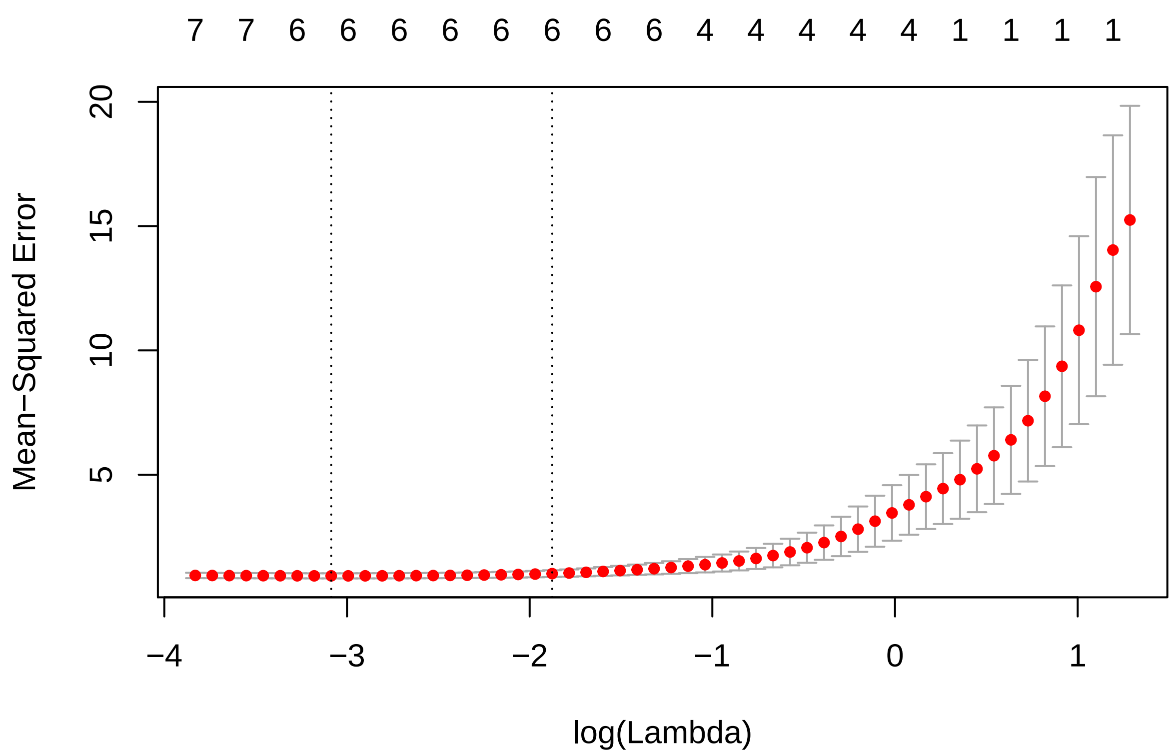
```r
which.max(sum$adjr2) #Choose model w/ highest adjR-sq
```

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

```r
#Lasso
#grid of lambdas from 10^10 to 10^-2
grid=10^seq(5,-2,length=100)
#initialize 10 variables
xtrain=model.matrix(Y1~poly(x,10,raw=T))[,-1]
#for lasso, alpha=1 (0 for ridge reg.)
lasso=glmnet(xtrain,Y,alpha=1,lambda=grid)
#CV
lasso.cv = cv.glmnet(xtrain, Y, alpha = 1)
opt.lambda = lasso.cv$lambda.min
opt.lambda
```

```
## [1] 0.04566871
```

```r
plot(lasso.cv)
```



```r
predict(lasso, s = opt.lambda, type = "coefficients")
```

```
## 11 x 1 sparse Matrix of class "dgCMatrix"
##                                   1
## (Intercept)            1.156943735
## poly(x, 10, raw = T)1  1.240197135
## poly(x, 10, raw = T)2  0.655788812
## poly(x, 10, raw = T)3  0.716801250
## poly(x, 10, raw = T)4  0.039546029
## poly(x, 10, raw = T)5  0.038166920
## poly(x, 10, raw = T)6  .
## poly(x, 10, raw = T)7  0.002190266
## poly(x, 10, raw = T)8  .
## poly(x, 10, raw = T)9  .
## poly(x, 10, raw = T)10 .
```