Chapter 6 Linear Model Selection and Regularization

Jishen Yin

2020/5/10

```
knitr::opts_chunk$set(echo = TRUE)
library(ISLR)
library(MASS)
library(leaps)
library(glmnet)
library(pls)
library(tidyverse)
```

Problem 9

In this exercise, we will predict the number of applications received using the other variables in the `College` data set.

(a) Split the data set into a training set and a test set.

```
data(College)
set.seed(1)
train <- sample(1:nrow(College), 500)
College_train <- College[train,]
College_val <- College[-train,]
```

(b) Fit a linear model using least square.

```
RMSE <- function(y_pred, y){
  return(sqrt(mean((y_pred-y)^2)))
}
```

```
ols <- lm(Apps ~ ., data = College_train)
RMSE(predict(ols, College_val), College_val$Apps)
```

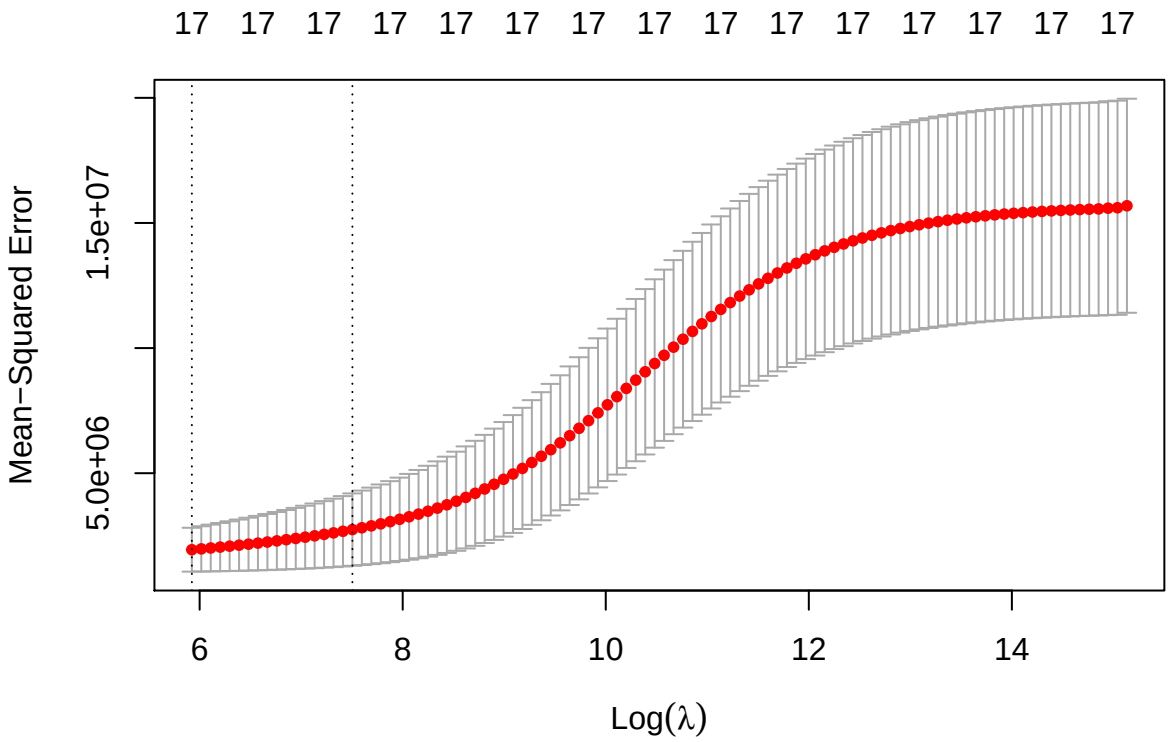
```
## [1] 1100.922
```

(c) Fit a ridge regression model.

```
x <- model.matrix(Apps~., data = College)[,-1]
y <- College$Apps

grid <- 10 ^ seq(5, -5, length = 100)
ridge <- glmnet(x[train,], y[train], alpha = 0, lambda = grid, thresh = 1e-12)
```

```
set.seed(1)
cv.out <- cv.glmnet(x[train,], y[train], alpha = 0)
plot(cv.out)
```




```
bestlam <- cv.out$lambda.min
bestlam
```

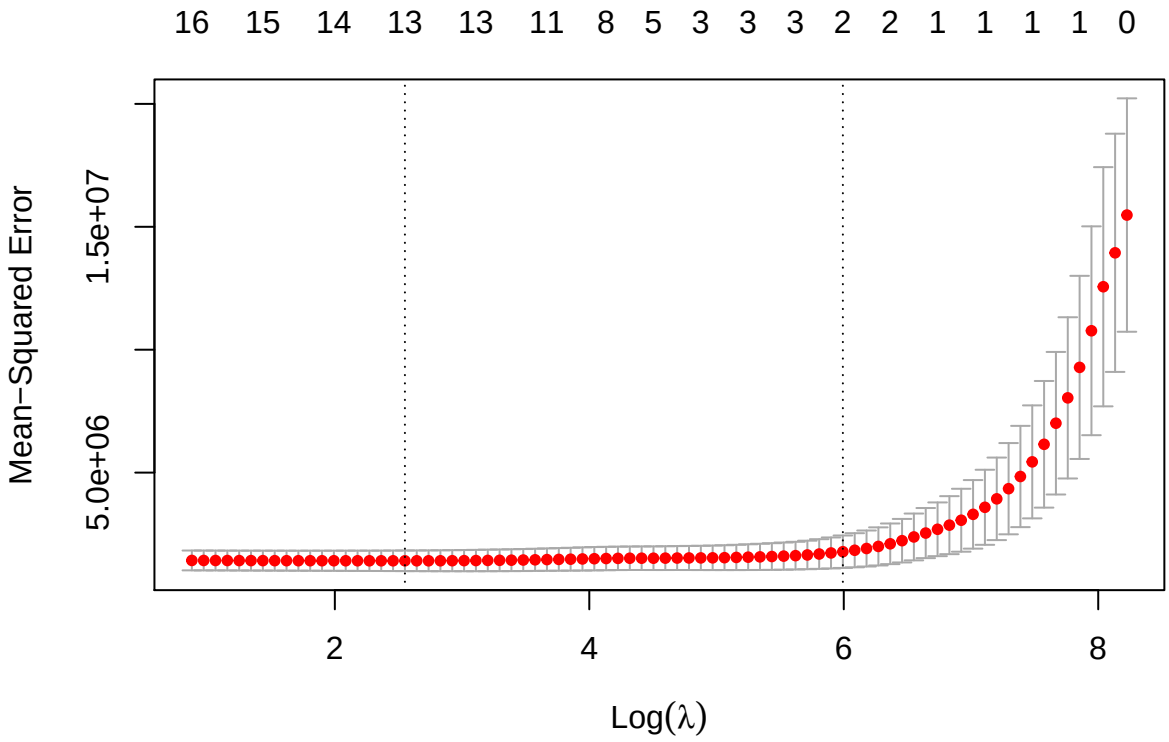
```
## [1] 373.6041
```

```
ridge.pred <- predict.glmnet(ridge, x[-train,], s = bestlam)
RMSE(ridge.pred, College_val$Apps)
```

```
## [1] 1053.561
```

(d) Fit a lasso model.

```
lasso <- glmnet(x[train,], y[train], alpha = 1, lambda = grid, thresh = 1e-12)
cv.out <- cv.glmnet(x[train,], y[train], alpha = 1)
plot(cv.out)
```




```
bestlam <- cv.out$lambda.min
bestlam
```

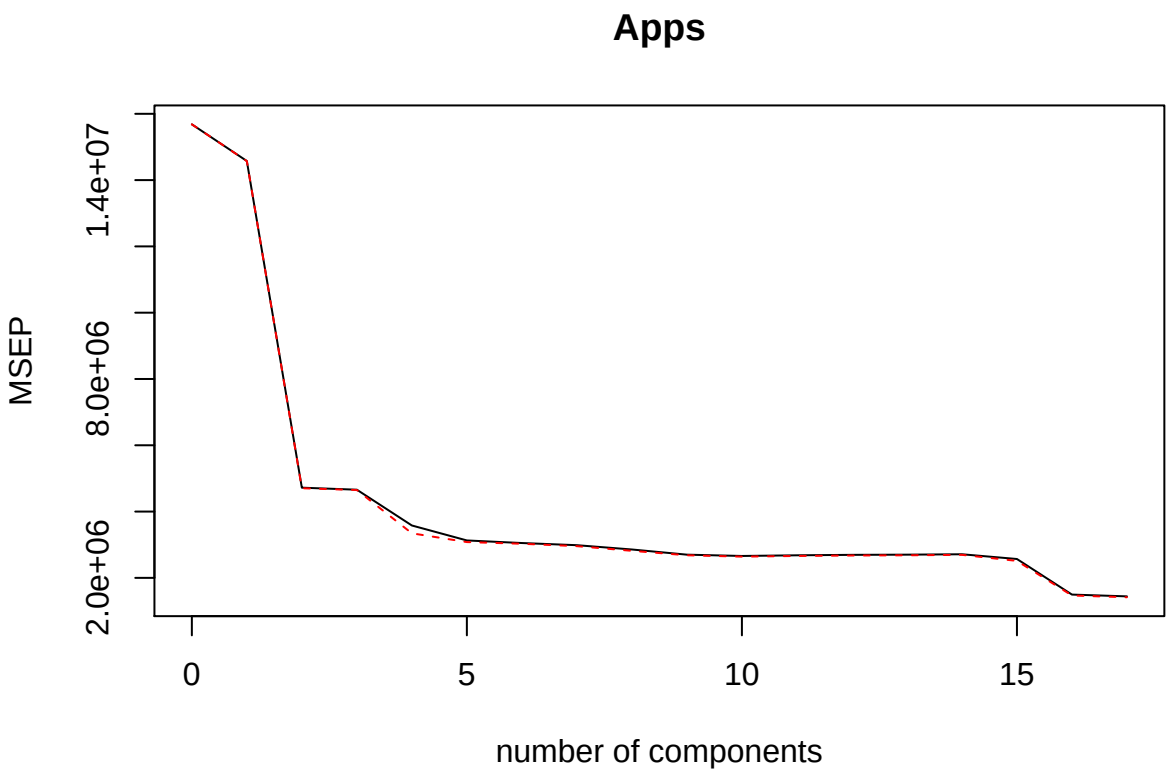
```
## [1] 12.81638
```

```
lasso.pred <- predict.glmnet(lasso, x[-train,], s = bestlam)
RMSE(lasso.pred, College_val$Apps)
```

```
## [1] 1081.334
```

(e) Fit a PCR model.

```
set.seed(5)
pcr.fit <- pcr(Apps~., data = College_train, scale = TRUE, validation = "CV")
validationplot(pcr.fit, val.type = "MSEP")
```

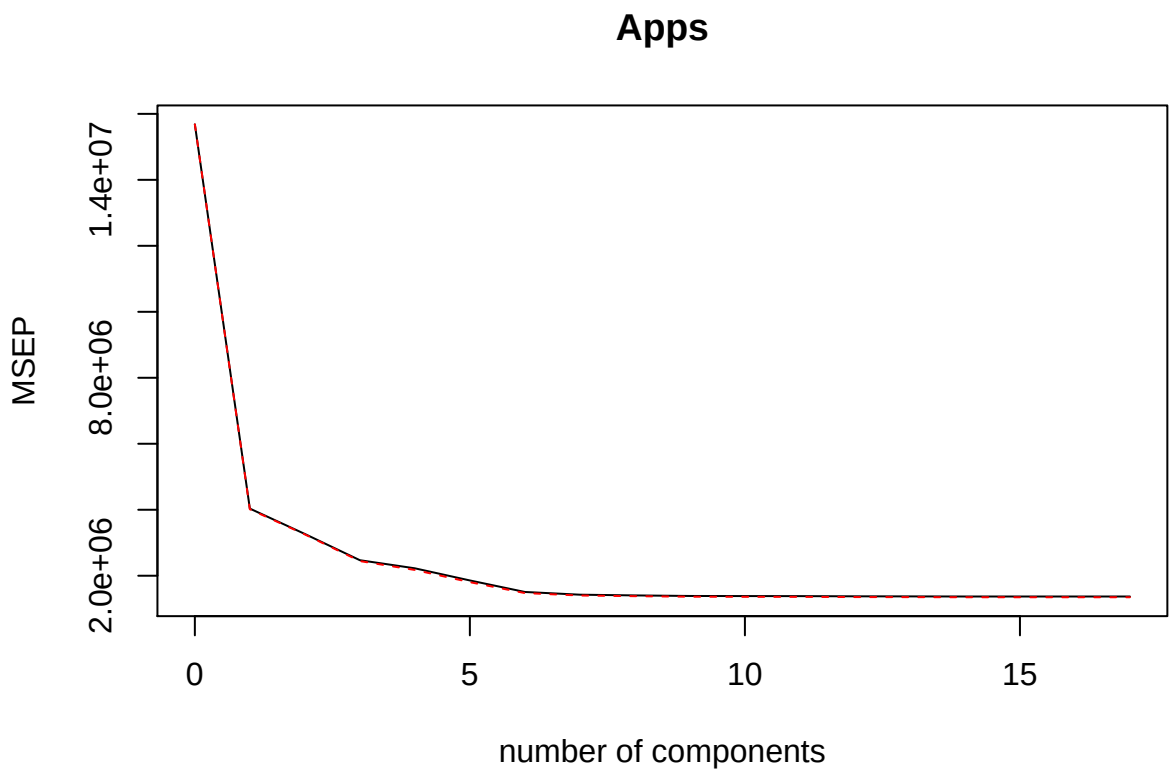



```
pcr.pred <- predict(pcr.fit, x[-train,], ncomp = 16)
RMSE(pcr.pred, College_val$Apps)
```

```
## [1] 1141.483
```

(f) Fit a PLS model.

```
pls.fit <- plsr(Apps~., data = College_train, scale = TRUE, validation = "CV")
validationplot(pls.fit, val.type = "MSEP")
```




```
summary(pls.fit)
```

```
## Data:    X dimension: 500 17
##  Y dimension: 500 1
## Fit method: kernelpls
## Number of components considered: 17
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV            3960     2009     1807     1572     1491     1363     1228
## adjCV         3960     2004     1803     1563     1475     1343     1214
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV        1194     1185     1177      1176      1175      1173      1172
## adjCV     1183     1175     1168      1166      1166      1163      1162
##        14 comps  15 comps  16 comps  17 comps
## CV         1170      1170      1170      1170
## adjCV      1161      1161      1161      1161
##
## TRAINING: % variance explained
##       1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X       26.20    49.93    62.77    66.14    70.56    74.22    78.00    80.92
## Apps    76.09    81.75    87.08    90.42    92.32    93.13    93.23    93.29
##       9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X       82.87     86.54     89.43     91.11     92.54     95.00     97.01
```

```
## Apps    93.34     93.35     93.36     93.38     93.39     93.39     93.39
##       16 comps  17 comps
## X       98.69    100.00
## Apps    93.39     93.39
```

```
pls.pred <- predict(pls.fit, x[-train,], ncomp = 14)
RMSE(pls.pred, College_val$Apps)
```

```
## [1] 1101.373
```