ANLY512_HW6

Hongyang Zheng

2019/3/25

```r
library(ISLR)
library(glmnet)
```

```
## Loading required package: Matrix
## Loading required package: foreach
## Loaded glmnet 2.0-16
```

```r
library(MASS)
library(biglars)
```

```
## Loading required package: ff
## Loading required package: bit
## Attaching package bit
## package:bit (c) 2008-2012 Jens Oehlschlaegel (GPL-2)
## creators: bit bitwhich
## coercion: as.logical as.integer as.bit as.bitwhich which
## operator: ! & | xor != ==
## querying: print length any all min max range sum summary
## bit access: length<- [ [<- [[ [[<-
## for more help type ?bit
##
## Attaching package: 'bit'
## The following object is masked from 'package:base':
##
##     xor
## Attaching package ff
## - getOption("fftempdir")=="/var/folders/g6/p8xqtmjs7qv1t2j96d414fh80000gn/T//RtmpoIN3J8"
## - getOption("ffextension")=="ff"
## - getOption("ffdrop")==TRUE
## - getOption("fffinonexit")==TRUE
## - getOption("ffpagesize")==65536
## - getOption("ffcaching")=="mmnoflush"  -- consider "ffeachflush" if your system stalls on large writ
## - getOption("ffbatchbytes")==16777216 -- consider a different value for tuning your system
## - getOption("ffmaxbytes")==536870912 -- consider a different value for tuning your system
##
## Attaching package: 'ff'
## The following objects are masked from 'package:bit':
##
##     clone, clone.default, clone.list

## The following objects are masked from 'package:utils':
##
##     write.csv, write.csv2

## The following objects are masked from 'package:base':
##
##     is.factor, is.ordered
```

Problem 2ab

a)

i. False, the lasso is more inflexible.
ii. False, the lasso is more inflexible.
iii. True, less flexible model will have less variance, more bias. Since the decrease in variance is bigger than the increase in bias, the prediction accuracy is improved.
iv. False, less flexible model will decrease in variance

b)

i. False, the ridge is more inflexible.
ii. False, the ridge is more inflexible.
iii. True, less flexible model will have less variance, more bias. Since the decrease in variance is bigger than the increase in bias, the prediction accuracy is improved.
iv. False, less flexible model will decrease in variance

Problem 4abc

a)

As we increase λ, the model will become less flexible and all β decrease from their least square estimate values to 0. Therefore, the model will have more bias, and the training RSS will increase steadily. The correct answer is iii.

b)

When $\lambda = 0$, all β have their least square estimate values. In this case, the model tries to fit hard to training data and hence test RSS is high. As we increase λ, β start reducing to zero and some of the overfitting is reduced. Thus, test RSS initially decreases. Eventually, as β approach 0, the model becomes too simple and test RSS increases, so test RSS decreases initially, and then eventually starts increasing in a U shape. The correct answer is ii.

c)

As we increase λ, the model will become less flexible and all β decrease from their least square estimate values to 0. Therefore, the model will become more simple and contain less variance. So the variance will decrease steadily. The correct answer is iv.

Problem 9abcd

a)

```
set.seed(1234)
# Number of observations
n=dim(College)[1]

# Split the data set: train 80%, test 20%
train = sample(1:n, n*0.8)
College.train = College[train, ]
College.test = College[-train, ]
```

b)

```
# Linear model
model.1=lm(Apps~., data=College.train)

# Predict
model.pred.1 = predict(model.1, College.test)

# test RSS
mean((College.test$Apps - model.pred.1)^2)
```

```
## [1] 904977.9
```

Test error obtained is 904977.9.

c)

```
# Model matrix
train.mat = model.matrix(Apps~., data=College.train)
test.mat = model.matrix(Apps~., data=College.test)

# Using cross validation to select lambda: ridge
grid = 10 ^ seq(4, -2, length=100)
mod.ridge = cv.glmnet(train.mat, College.train$Apps, alpha=0, lambda=grid)
lambda.best = mod.ridge$lambda.min
lambda.best
```

```
## [1] 0.01
```

```
# Use the best lambda to build model and calculate test RSS
ridge.pred = predict(mod.ridge, newx=test.mat, s=lambda.best)
mean((College.test$Apps - ridge.pred)^2)
```

```
## [1] 904933.5
```

The test error obtained with $\lambda = 0.01$ is slightly lower than the one in b)

d)

```r
# Using cross validation to select lambda: lasso
grid = 10 ^ seq(4, -2, length=100)
mod.lasso = cv.glmnet(train.mat, College.train$Apps, alpha=1, lambda=grid)
lambda.best = mod.lasso$lambda.min
lambda.best
```

```
## [1] 0.01
```

```r
# Use the best lambda to build model and calculate test RSS
lasso.pred = predict(mod.lasso, newx=test.mat, s=lambda.best)
mean((College.test$Apps -lasso.pred)^2)
```

```
## [1] 904771.2
```

```r
# Coefficient
predict(mod.lasso, s=lambda.best, type="coefficients")
```

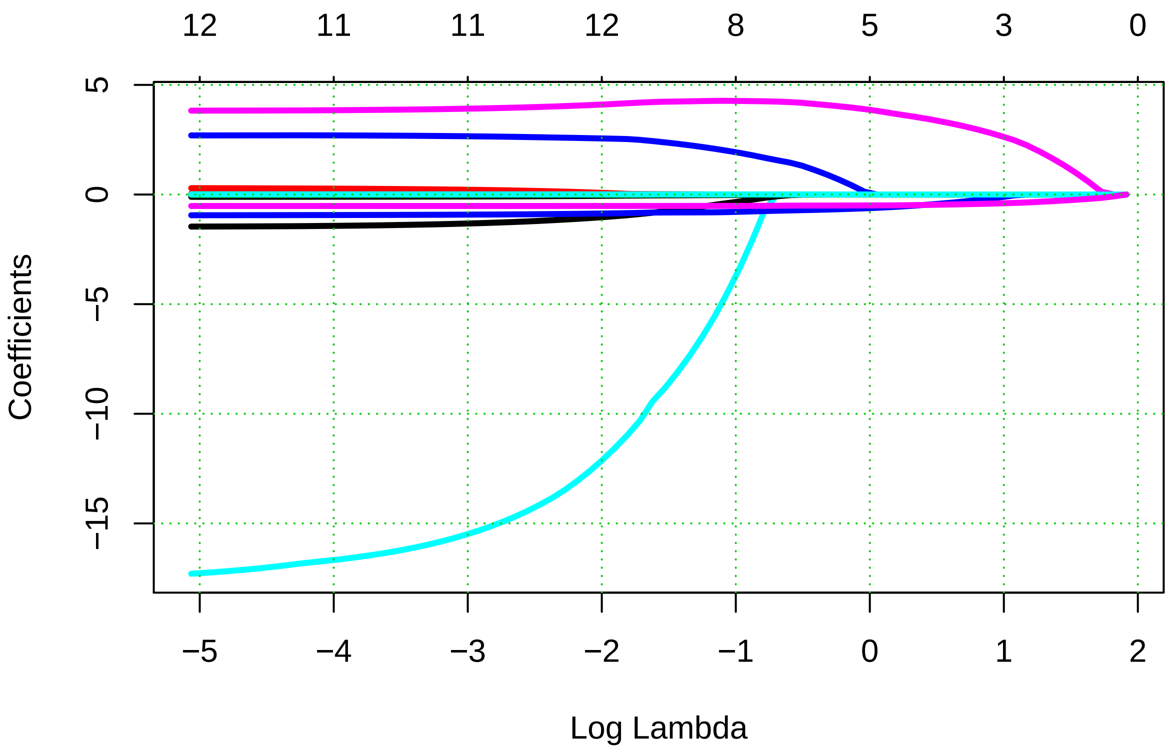
```
## 19 x 1 sparse Matrix of class "dgCMatrix"
##                         1
## (Intercept) -8.008221e+02
## (Intercept)  .
## PrivateYes  -4.134652e+02
## Accept       1.596546e+00
## Enroll      -9.957905e-01
## Top10perc    4.994765e+01
## Top25perc   -1.424032e+01
## F.Undergrad  7.531225e-02
## P.Undergrad  5.642903e-02
## Outstate    -1.004955e-01
## Room.Board   1.647538e-01
## Books        8.283267e-03
## Personal     7.814131e-02
## PhD         -8.750279e+00
## Terminal    -2.850293e+00
## S.F.Ratio    2.654884e+01
## perc.alumni  4.316996e+00
## Expend       9.028622e-02
## Grad.Rate    7.561167e+00
```

The test error obtained with lasso is lower than the test errors in both a) and b). There are 17 non-zero coefficients.

Problem Xtra #49

a)

```r
set.seed(1234)
# Build lasso model
X.model <- model.matrix(lm(medv ~ ., data = Boston))
fit.lasso = glmnet(X.model, Boston$medv, alpha = 1)

# Plot trajectories of coefficients
plot(fit.lasso, xvar = "lambda", lwd = 3)
grid(col = 3)
```



```r
# The last five variables
max(which(fit.lasso$df == 5))
```

```
## [1] 25
```

```r
as.matrix(fit.lasso$beta[,25])
```

```
##                      [,1]
## (Intercept)  0.000000000
## crim         0.000000000
## zn           0.000000000
## indus        0.000000000
## chas         0.906034673
## nox          0.000000000
## rm           4.079788153
## age          0.000000000
## dis          0.000000000
## rad          0.000000000
## tax          0.000000000
## ptratio     -0.685961897
## black        0.004202317
## lstat       -0.502594040
```

The last five variables remained in the model are chas, rm, ptratio, black and lstat.

b)

```r
# 10-fold cross validation
cv.lasso = cv.glmnet(X.model,Boston$medv,alpha = 1, nfolds=10)

# Choose the 1-SE lambda:
cv.lasso$lambda.1se
```

```
## [1] 0.4564174
```

```
lambda.1se=cv.lasso$lambda.1se

# Build and predict model
lasso.1se = glmnet(X.model, Boston$medv, alpha =1, lambda = lambda.1se)
lasso.pred=predict(lasso.1se, newx=X.model)

# Calculate RSE
n = dim(X.model)[1]
p = lasso.1se$df
sqrt(mean((lasso.pred-Boston$medv)^2)*n/(n-p-1))
```

```
## [1] 5.114997
```

The 1SE value of is 0.4564174. The cross validation estimate for the residual standard error is 5.114997.

c)

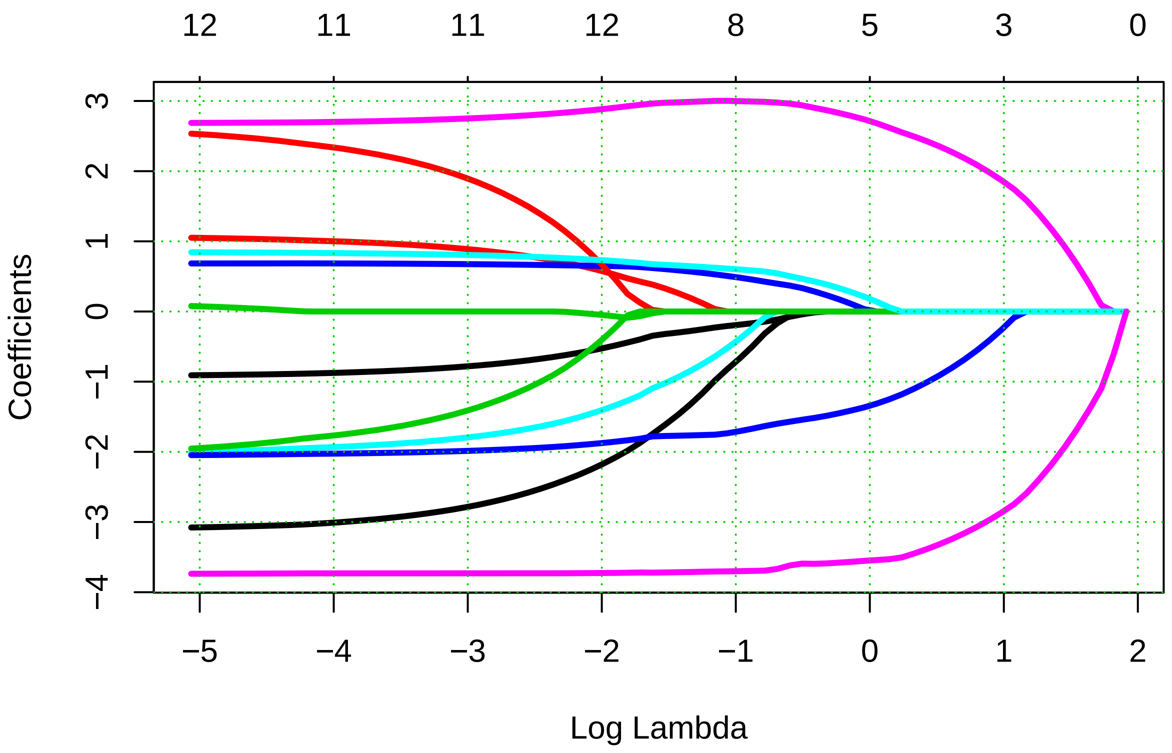
```
# Rescale all predictors
scaled.boston <- scale(Boston)
apply(scaled.boston, 2, mean)
```

```
##          crim            zn         indus          chas           nox
## -7.202981e-18  2.282481e-17  1.595296e-17 -3.544441e-18 -2.150022e-16
##            rm           age           dis           rad           tax
## -1.056462e-16 -1.643357e-16  1.153079e-16  4.799652e-17  2.024415e-17
##       ptratio         black         lstat          medv
## -3.924246e-16 -1.151679e-16 -7.052778e-17 -1.374631e-16
```

```
apply(scaled.boston, 2, sd)
```

```
##    crim      zn   indus    chas     nox      rm     age     dis     rad
##       1       1       1       1       1       1       1       1       1
##     tax ptratio   black   lstat    medv
##       1       1       1       1       1
```

```
scaled.boston <- as.data.frame(scaled.boston)

# Build lasso model
X.model.2 <- model.matrix(lm(Boston$medv ~ ., data = scaled.boston))
fit.lasso.2 = glmnet(X.model.2, Boston$medv, alpha = 1)

# Plot trajectories of coefficients
plot(fit.lasso.2, xvar = "lambda", lwd = 3)
grid(col = 3)
```



```
# The last five variables
max(which(fit.lasso.2$df == 5))
```

```
## [1] 25
```

```
as.matrix(fit.lasso.2$beta[,25])
```

```
##                    [,1]
## (Intercept)  0.0000000
## crim         0.0000000
## zn           0.0000000
## indus        0.0000000
## chas         0.2301274
## nox          0.0000000
## rm           2.8665291
## age          0.0000000
## dis          0.0000000
## rad          0.0000000
## tax          0.0000000
## ptratio     -1.4850701
## black        0.3836499
## lstat       -3.5890550
```

The last five variables remained in the model are chas, rm, ptratio, black and lstat. This answer is same as the answer in part a).

d)

```
# 10-fold cross validation
cv.lasso.2 = cv.glmnet(X.model.2, Boston$medv, alpha = 1, nfolds=10)

# Choose the 1-SE lambda:
```

```
cv.lasso.2$lambda.1se
```

```
## [1] 0.4564174
```

```
lambda.1se.2=cv.lasso.2$lambda.1se

# Build and predict model
lasso.1se.2 = glmnet(X.model.2, Boston$medv, alpha =1, lambda = lambda.1se.2)
lasso.pred.2=predict(lasso.1se.2, newx=X.model.2)

# Calculate RSE
n = dim(X.model.2)[1]
p = lasso.1se.2$df
sqrt(mean((lasso.pred.2-Boston$medv)^2)*n/(n-p-1))
```

```
## [1] 5.114997
```

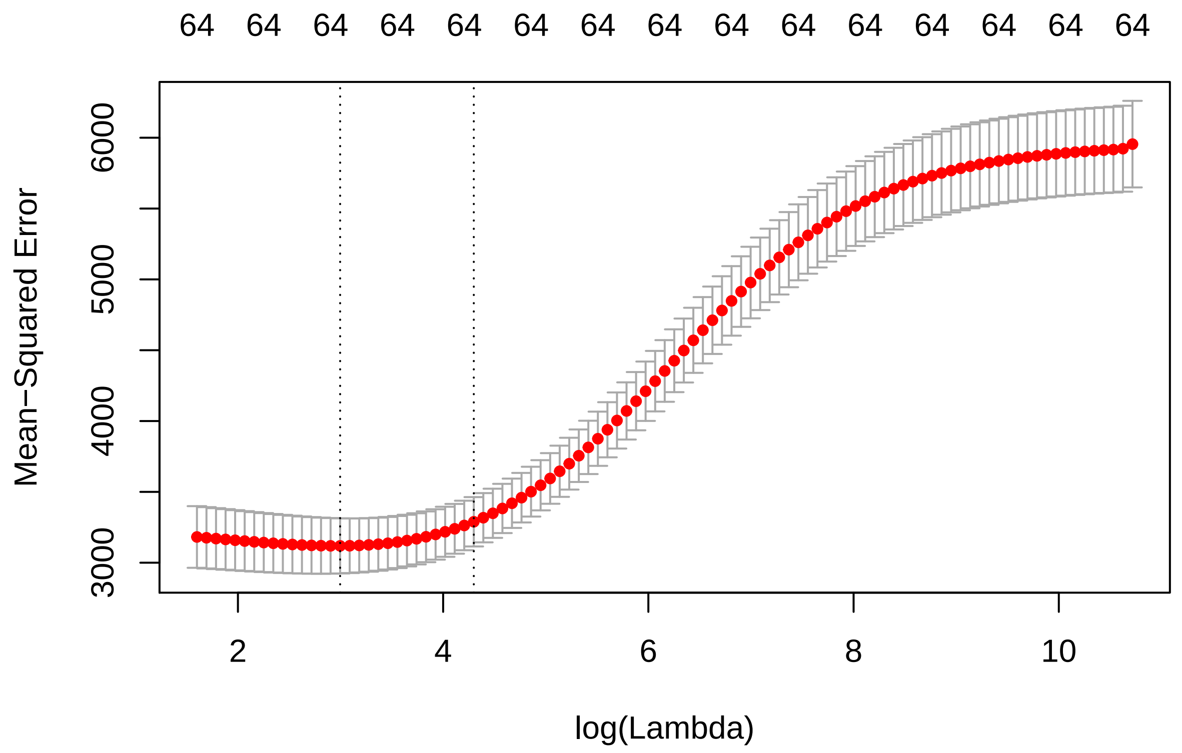
The cross validation estimate for the residual standard error is 5.114997, which is same as the RSE in part b). Rescaling does not lead to a better performing model.

Problem Xtra #50

```
# Load data
data(diabetes)

# Create the dataframe from x2 and y
Dia <- as.data.frame.matrix(diabetes$x2)
Dia$y <- diabetes$y
```

a)

```
set.seed(1234)

# Ridge model using 10-fold cross validation
X2.model = model.matrix(lm(y ~ ., data = Dia))
cv.ridge = cv.glmnet(X2.model, Dia$y, alpha = 0, nfolds = 10)

# Plot the mean sqared error estimates
plot(cv.ridge)
```



```
# Lambda 1SE
lambda.1se=cv.ridge$lambda.1se
lambda.1se
```

```
## [1] 73.5996
```

The Lambda 1SE is 73.5995971.

b)

```
# Predict with the 1se lambda
fit.ridge.1se = glmnet(X2.model, Dia$y, alpha = 0, lambda = lambda.1se)
predict.ridge.1se = predict(fit.ridge.1se, newx = X2.model)

# RMS prediction error
sqrt(mean((predict.ridge.1se-Dia$y)^2))
```

```
## [1] 54.42877
```

The RMS prediction error according to cross validation for this λ_{1se} is 54.42877.

Problem Xtra #52

```
set.seed(123)
# Load data
mnist=load('~/Desktop/other/data/mnist_all.RData')
```

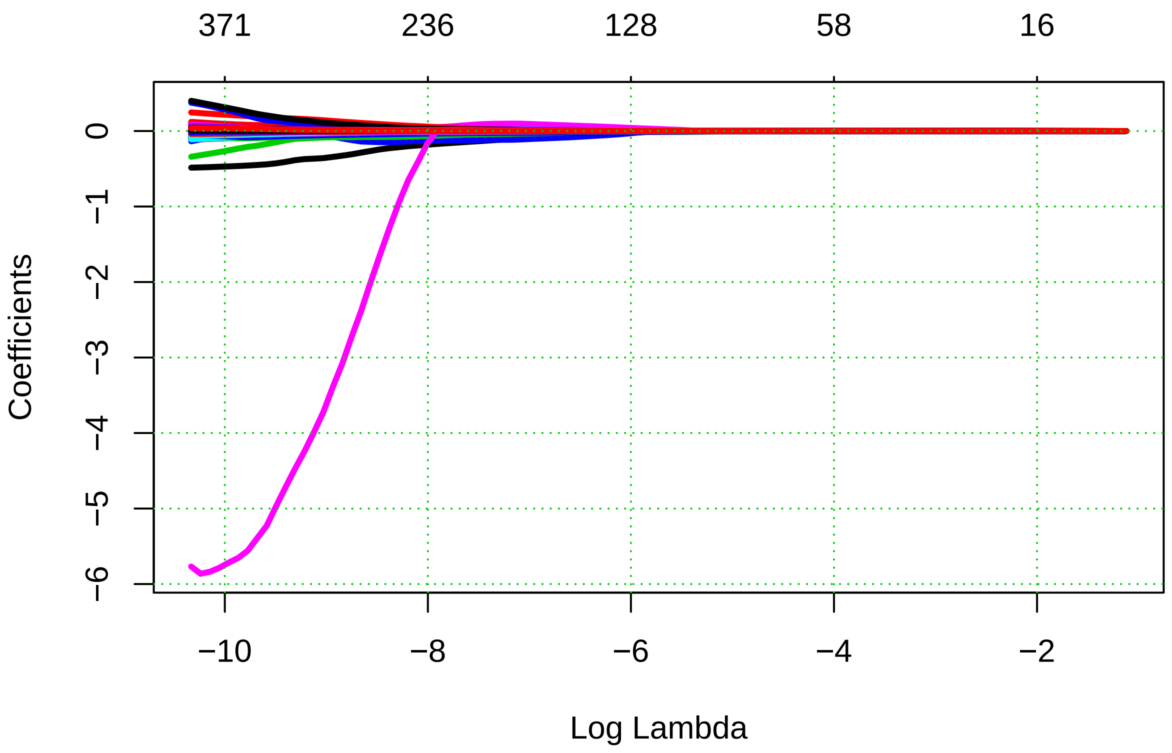
```
# Generate dataframe
# Extract data for digit1 and digit8
y.nist = train$y
```

```r
index <- (y.nist == 1 | y.nist == 8)
x.nist <- train$x[index,]
x.train <- as.data.frame(x.nist)

y.nist1 = test$y
index1 <- (y.nist1 == 1 | y.nist1 == 8)
x.nist1 <- test$x[index1,]
x.test <- as.data.frame(x.nist1)

# y=1 if it is digit8, y=0 if it is digit1
x.train$y <- 0
x.train$y[y.nist[index] == 8] <- 1
x.test$y <- 0
x.test$y[y.nist1[index1] == 8] <- 1
```

```r
# Calculate variance for each pixel for train data
variance.train=rep(0,784)
for (i in 1:784)
{
  variance.train[i]=var(x.train[,i])
}
variance.train=as.data.frame(variance.train)
train=x.train[,variance.train != 0]
train$y=x.train$y

# Calculate variance for each pixel for test data
variance.test=rep(0,784)
for (i in 1:784)
{
  variance.test[i]=var(x.test[,i])
}
variance.test=as.data.frame(variance.test)
test=x.test[,variance.test != 0]
test$y=x.test$y
```

a)

```r
# Build lasso model
lasso.model <- model.matrix(glm(y ~ ., data = train))
fit.lasso = glmnet(lasso.model, train$y, alpha = 1, family = 'binomial')

# Plot trajectories of coefficients
plot(fit.lasso, xvar = "lambda", lwd = 3)
grid(col = 3)
```



b)

```r
# The last ten variables
max(which(fit.lasso$df == 10))
```

```
## [1] 7
```

```r
c=as.matrix(fit.lasso$beta[,7])
print(c[c!=0,])
```

```
##          V236          V263          V264          V292          V296 
##  1.700242e-03  5.975194e-05  5.964574e-05  2.720889e-04 -1.933823e-03 
##          V320          V348          V355          V376          V404 
##  1.024413e-03  7.928993e-04  1.566265e-03  1.133928e-03  1.014616e-03
```

```r
# The last nine variables
max(which(fit.lasso$df == 9))
```

```
## [1] 6
```

```r
c=as.matrix(fit.lasso$beta[,6])
print(c[c!=0,])
```

```
##          V236          V264          V292          V296          V320 
##  1.495988e-03  1.695259e-04  6.439989e-05 -1.619874e-03  8.798873e-04 
##          V348          V355          V376          V404 
##  6.690818e-04  1.317205e-03  1.205437e-03  6.102932e-04
```

```r
# The last eight variables
max(which(fit.lasso$df == 8))
```

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

```
c=as.matrix(fit.lasso$beta[,5])
print(c[c!=0,])
```

```
##          V236          V264          V296          V320          V348
##  0.0012718642  0.0001939409 -0.0012884081  0.0005777368  0.0006325599
##          V355          V376          V404
##  0.0010631302  0.0012427090  0.0001922937
```

```
# The last seven variables
max(which(fit.lasso$df == 7))
```

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

```
c=as.matrix(fit.lasso$beta[,4])
print(c[c!=0,])
```

```
##          V236          V264          V296          V320          V348
##  0.0010491040  0.0001462934 -0.0009419291  0.0002048476  0.0007242939
##          V355          V376
##  0.0007663709  0.0010331016
```

```
# The last six variables-- not exits
# The last five variables
min(which(fit.lasso$df == 5))
```

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

```
c=as.matrix(fit.lasso$beta[,2])
print(c[c!=0,])
```

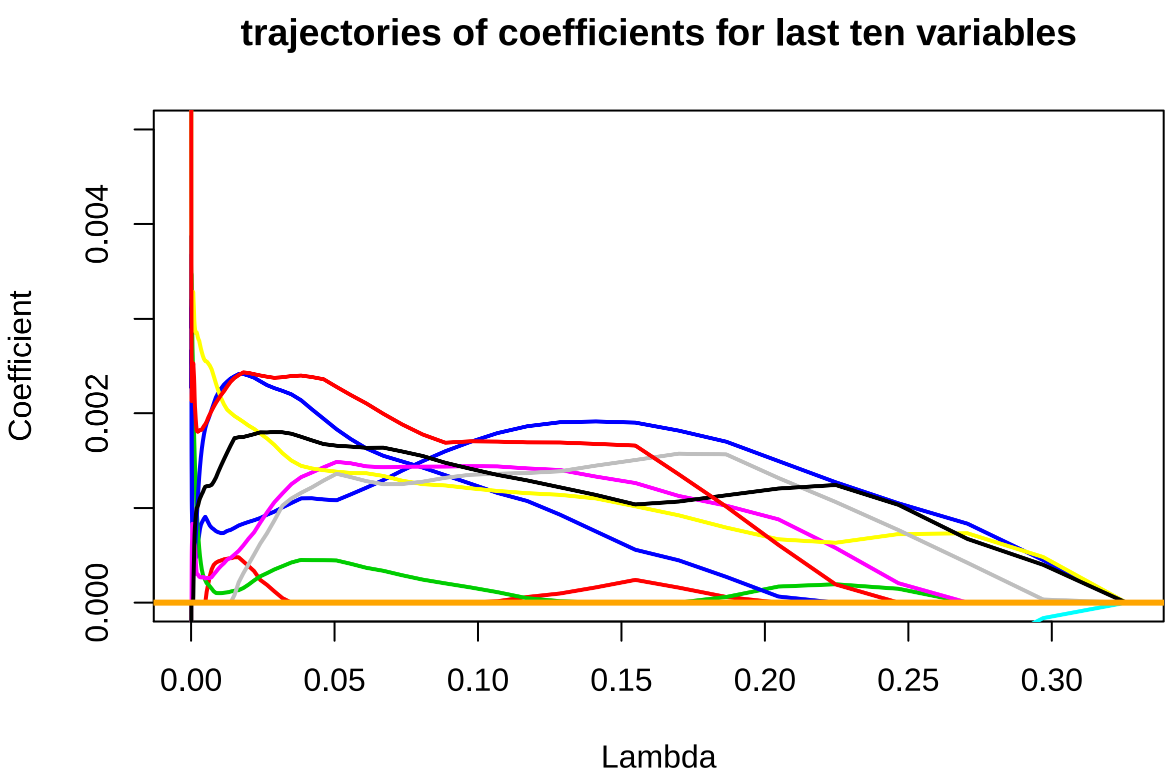
```
##          V236          V296          V348          V355          V376
##  4.493623e-04 -1.636806e-04  4.814813e-04  3.254483e-05  3.993704e-04
```

The last ten variables are V236, V263, V264, V292, V296, V320, V348, V355, V376, V404.

The final model contains five variables which are V236, V296, V348, V355, V376. The leaving order is V263, V292, V404, V320 and V264 together.

c)

```
# Plot trajectories of coefficients for last ten variables
plot(x = fit.lasso$lambda, y = fit.lasso$beta[(fit.lasso$beta[ , 7]!=0),][1,], ylim = c(0,0.005), lwd =
     main = 'trajectories of coefficients for last ten variables', ylab = 'Coefficient', xlab = 'Lambda
     type = 'l', col = 20)

for (i in 2:10){
  lines(x = fit.lasso$lambda, y = fit.lasso$beta[(fit.lasso$beta[,7]!=0),][i,], col = i,lwd = 2)
}

# Add a horizontal line for coefficient=0
abline(h = 0, col = 'orange', lwd = '3')
```


trajectories of coefficients for last ten variables
Coefficient
0.000
0.002
0.004
0.00
0.05
0.10
0.15
0.20
0.25
0.30
Lambda