stats 406 hw7

Li Hsuan Lin

Lab section 002

UM_ID: 49109112

(1)

(a):

(6). remains constant.

Since in Ridge regression, all the predictors are included in the model.

(b):

(5).steadily decreases.

Since increasing s means we increase the model complexity, the training error would decreases.

(c):

(3). decreases initially, and then eventually starts increasing.

This is the result of variance-bias trade off. When s increases initially, there is a relatively large decrease in variance and small increase in bias, which results in decreasing test error. However, as s continue to increases, the rate of bias increase is greater than that of variance decreases. Thus, we have increasing test error.

(d):

(4). steadily increases.

This is the result of increasing model complexity.

(e):

(5). steadily decreases.
Same reasoning as (d). This is the result of increasing model complexity.

(2)

```
library(ISLR)
library(tidyverse)
```

```
## — Attaching packages ——————————————————————— tidyverse 1.2.1 —
```

```
## ✔ ggplot2 3.2.1     ✔ purrr   0.3.3
## ✔ tibble  2.1.3     ✔ dplyr   0.8.3
## ✔ tidyr   1.0.0     ✔ stringr 1.4.0
## ✔ readr   1.3.1     ✔ forcats 0.4.0
```

```
## — Conflicts ———————————————————————— tidyverse_conflicts() —
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
```

```
library(glmnet)
```

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

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
## Loaded glmnet 3.0-2
```

(a):

reuse the code in hw6.

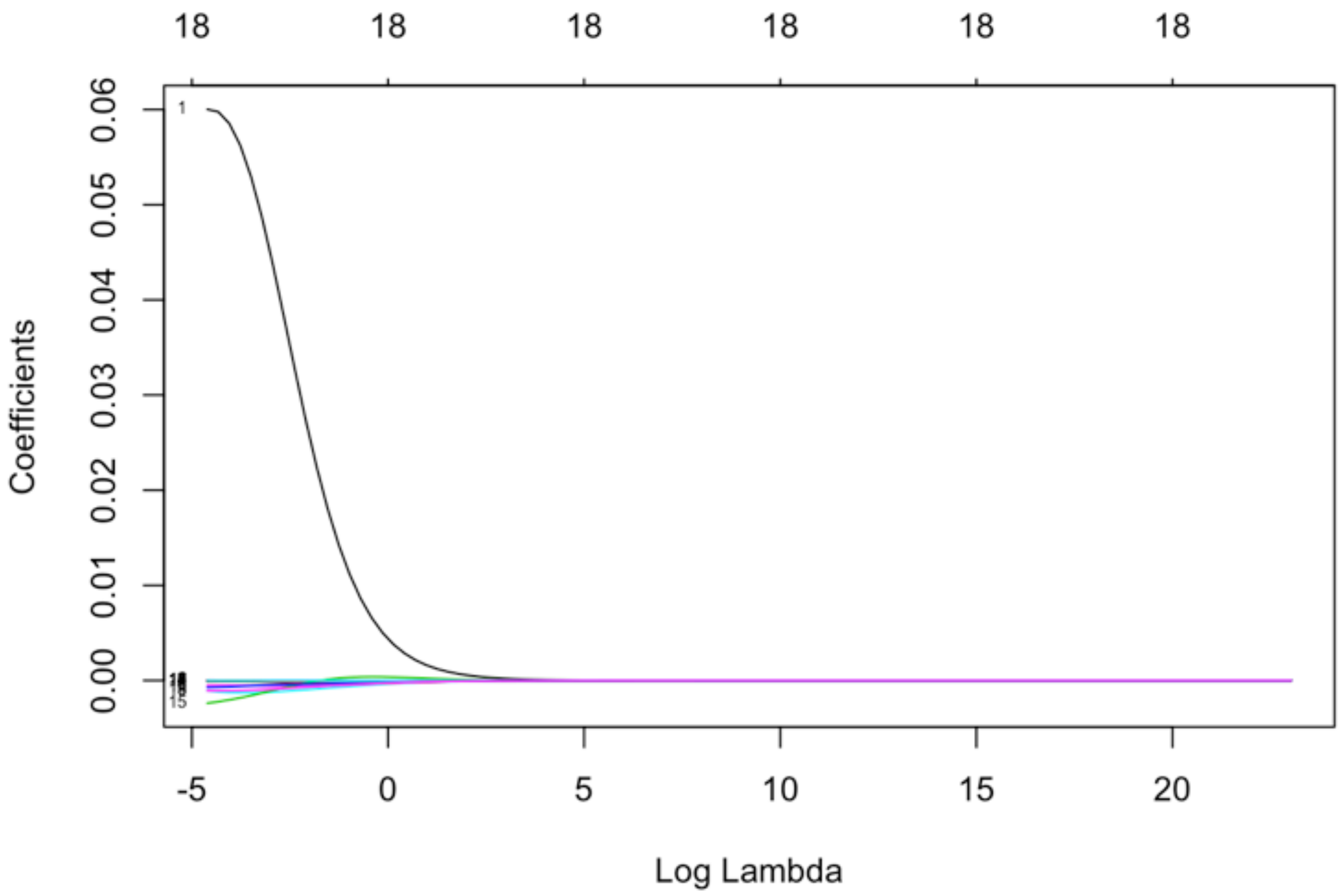
```
set.seed(234)
college_raw = College

#create new response variable
college = college_raw %>% mutate(accept_rate = Accept / Apps)


#split train/test dataset
RNGkind(sample.kind = "Rejection")
nrows_college = nrow(college)
train_id = sample(nrows_college, floor(0.7*nrows_college))
college_train = college[train_id,]
college_test = college[-train_id,]

#predictors
X = model.matrix(accept_rate ~ ., college)[, -1]
#response
Y <- college$accept_rate
```

Use the codes in the lab

(a)

Below are the output.
As we can see, the training error is 0.009790067.
The cross-validated error is 0.01021892.
The testing error is 0.01342292.

```
grid = 10^seq(10, -2, length=100)
ridge.mod = glmnet(X[train_id,], Y[train_id], alpha=0,lambda = grid)
plot(ridge.mod, xvar = "lambda", label = TRUE)
```



```
# 10 fold cross validation
cv.out = cv.glmnet(X[train_id,], Y[train_id], alpha = 0, nfolds = 10)
bestlam = cv.out$lambda.min
```

```
# training error
set.seed(234)
ridge.pred_train = predict(ridge.mod, s=bestlam, newx=X[train_id,])
training_error = mean((ridge.pred_train-Y[train_id])^2) %>% print()
```

```
## [1] 0.009790067
```

```
#cross validation error
set.seed(234)
lambda.grid = cv.out$lambda
mses = cv.out$cvm
cv_error = mses[which(lambda.grid == bestlam)]  %>% print()
```

```
## [1] 0.01021892
```

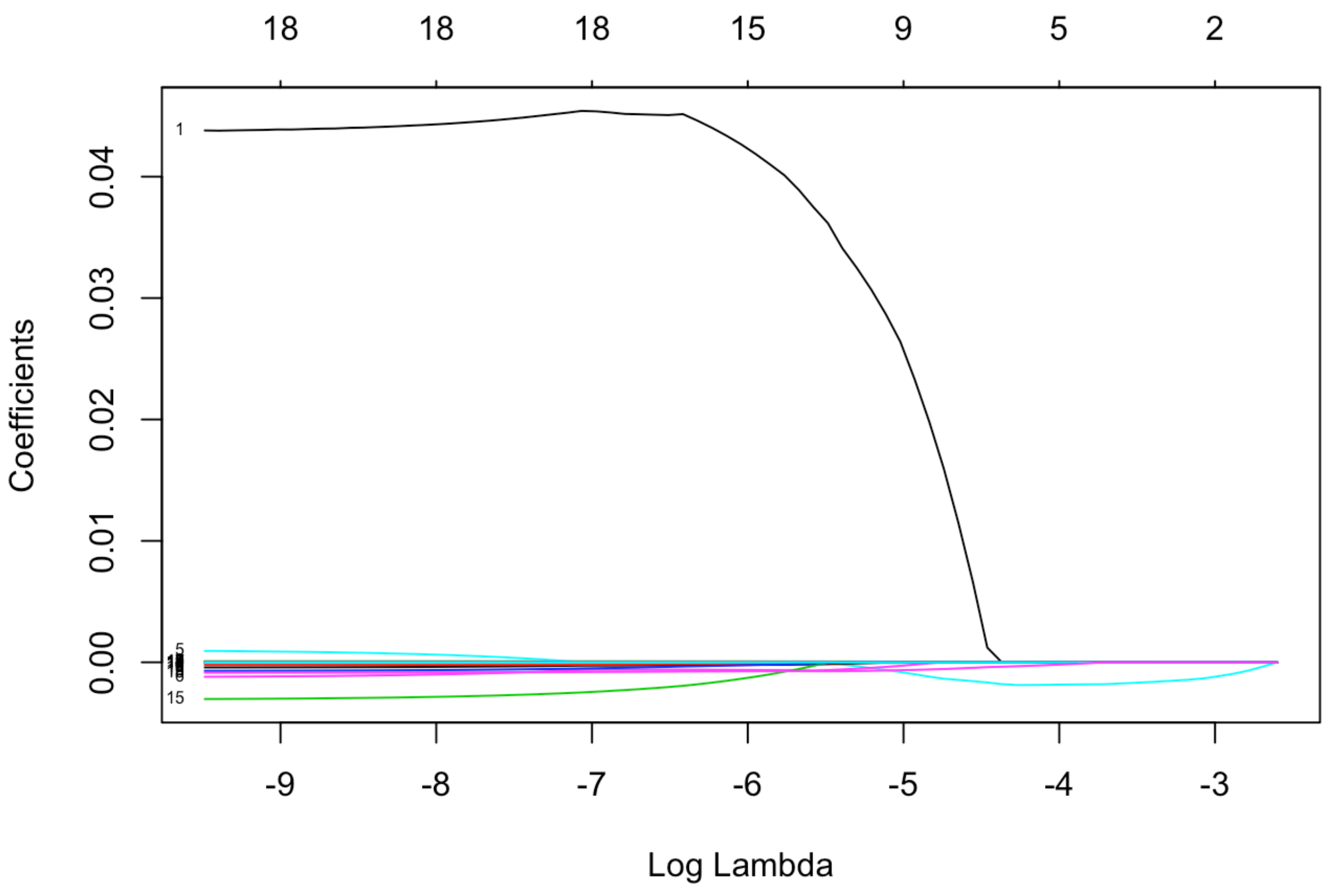
```
#testing error
set.seed(234)
ridge.pred_test = predict(ridge.mod, s=bestlam, newx=X[-train_id,])
test_error = mean((ridge.pred_test-Y[-train_id])^2) %>% print()
```

```
## [1] 0.01342292
```

(b):

Below are the output.
As we can see, the training error is 0.008288304.
The cross-validated error is 0.00904093.
The testing error is 0.0124019.

```
lasso.mod = glmnet(X[train_id,], Y[train_id], alpha=1)
plot(lasso.mod, xvar = "lambda", label = TRUE)
```



```
set.seed(234)
# 10 fold cross validation
cv.out = cv.glmnet(X[train_id,], Y[train_id], alpha = 1, nfolds = 10)
bestlam = cv.out$lambda.min
```

```
lasso.coef = predict(lasso.mod, type="coefficients", s=bestlam)
lasso.coef
```

```
## 19 x 1 sparse Matrix of class "dgCMatrix"
##                         1
## (Intercept)  9.749176e-01
## PrivateYes   4.510064e-02
## Apps        -6.522552e-05
## Accept       8.575292e-05
## Enroll       2.050891e-05
## Top10perc    .
## Top25perc   -6.062484e-04
## F.Undergrad  .
## P.Undergrad -3.291765e-06
## Outstate     4.160393e-07
## Room.Board  -9.724584e-06
## Books       -7.166188e-05
## Personal    -1.749210e-06
## PhD         -2.528644e-04
## Terminal    -1.700263e-04
## S.F.Ratio   -2.128099e-03
## perc.alumni -4.001136e-04
## Expend      -1.713219e-06
## Grad.Rate   -7.713032e-04
```

```
# training error
set.seed(234)
lasso.pred_train = predict(lasso.mod, s=bestlam, newx=X[train_id,])
training_error = mean((lasso.pred_train-Y[train_id])^2) %>% print()
```

```
## [1] 0.008288304
```

```
#cross validation error
set.seed(234)
lambda.grid = cv.out$lambda
mses = cv.out$cvm
cv_error = mses[which(lambda.grid == bestlam)]  %>% print()
```

```
## [1] 0.00904093
```

```
#testing error
set.seed(234)
lasso.pred_test = predict(lasso.mod, s=bestlam, newx=X[-train_id,])
test_error = mean((lasso.pred_test-Y[-train_id])^2) %>% print()
```

```
## [1] 0.0124019
```

(c):

In this dataset, Lasso regression perform slightly better than ridge regression as its training and testing error are slightly better than that of ridge regression. However, the differences of error is quite small. In hw6, the reduced model with the lowest test error is the forward selection model. (training error:0.008878469, testing error:0.009523646). Its training and testing error are smaller than that of Lasso/ridge regression.

In my opinion, I think the prediction the acceptance rate is done quite acurately as the test mse is around 0.01, meaning the accuracy rate is around 0.99.

Overall, the testing errors among 3 models are quite similar, with only 0.01 or 0.02 difference.

I would recommend using forward selection model as it is simpler model compared to ridge/Lasso regression and it gives smaller test mse.