P8106 HW 1

xc2474 Xinlei Chen

Problem

```r
library(tidyverse)
library(ISLR)
library(glmnet)
library(caret)
library(corrplot)
library(plotmo)
library(pls)
```

```r
train = read.csv("./data/solubility_train.csv")
test= read.csv("./data/solubility_test.csv")
```

(a) Fit a linear model using least squares on the training data and calculate the mean square error using the test data.

1) Fit a linear model

```r
# Fit model using train data
fit1 = lm(Solubility ~ ., data = train)
# summary(fit1) see the details
```

2) MSE

```r
pred_va <- predict(fit1, test)
mean((pred_va-test$Solubility)^2)
```

```
## [1] 0.5558898
```

The test MSE is 0.5558898 for this linear model.

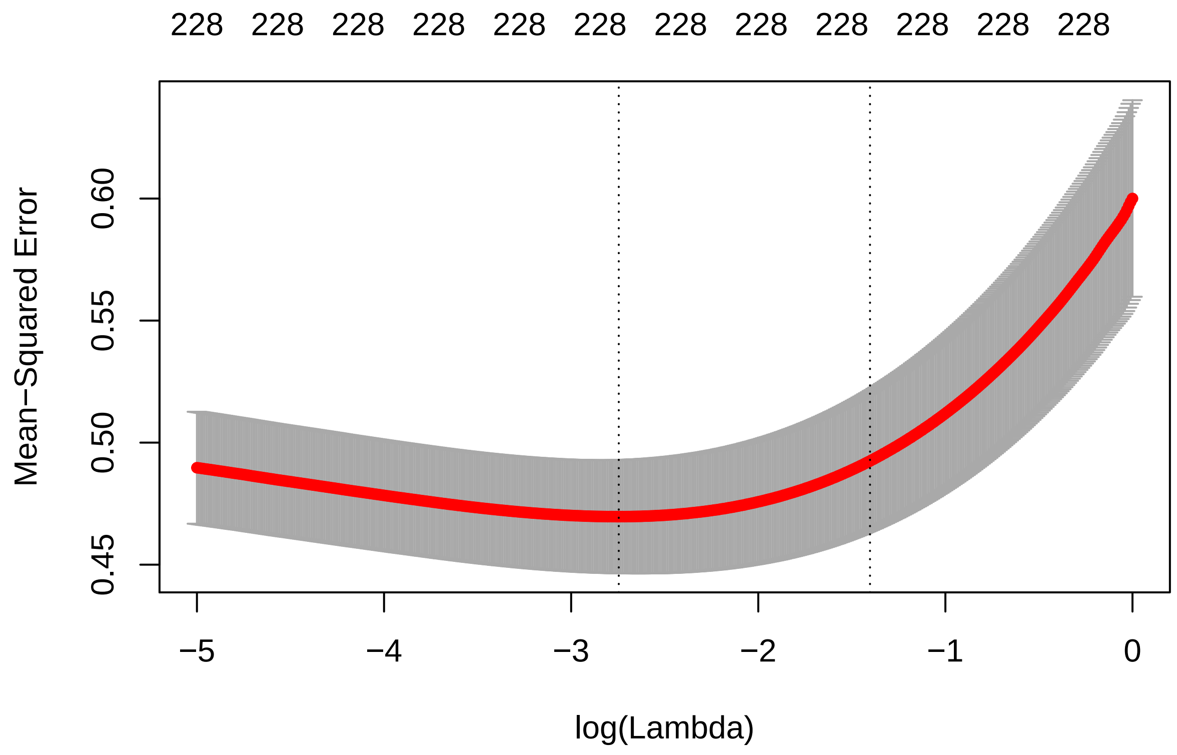
(b) Fit a ridge regression model on the training data, with λ chosen by cross-validation. Report the test error.

1) Fit a ridge regression model

```r
train <- na.omit(train)

x_train <- model.matrix(Solubility~., train)[,-1]
y_train <- train$Solubility

x_test <- model.matrix(Solubility~., test)[,-1]
y_test <- test$Solubility
```

```r
# fit the ridge regression (alpha = 0) with a sequence of lambdas
ridge.mod <- glmnet(x_train, y_train, alpha=0, lambda = exp(seq(-5, 0, length=500)))
```

```r
set.seed(1)
cv.ridge <- cv.glmnet(x_train, y_train,
                      alpha = 0,
                      lambda = exp(seq(-5, 0, length=500)),
                      type.measure = "mse")

plot(cv.ridge)
```



```r
best.lambda <- cv.ridge$lambda.min
best.lambda
```

```
## [1] 0.06421676
```

```r
#predict(ridge.mod, s = best.lambda, type="coefficients")
# all coefficients exist
```

The λ we choose is 0.0642168.

2) MSE

```r
ridge.pred = predict(ridge.mod, s = best.lambda, newx = x_test)
mean((ridge.pred - y_test)^2)
```

```
## [1] 0.5125536
```

The test MSE of this ridge model is 0.5125536.

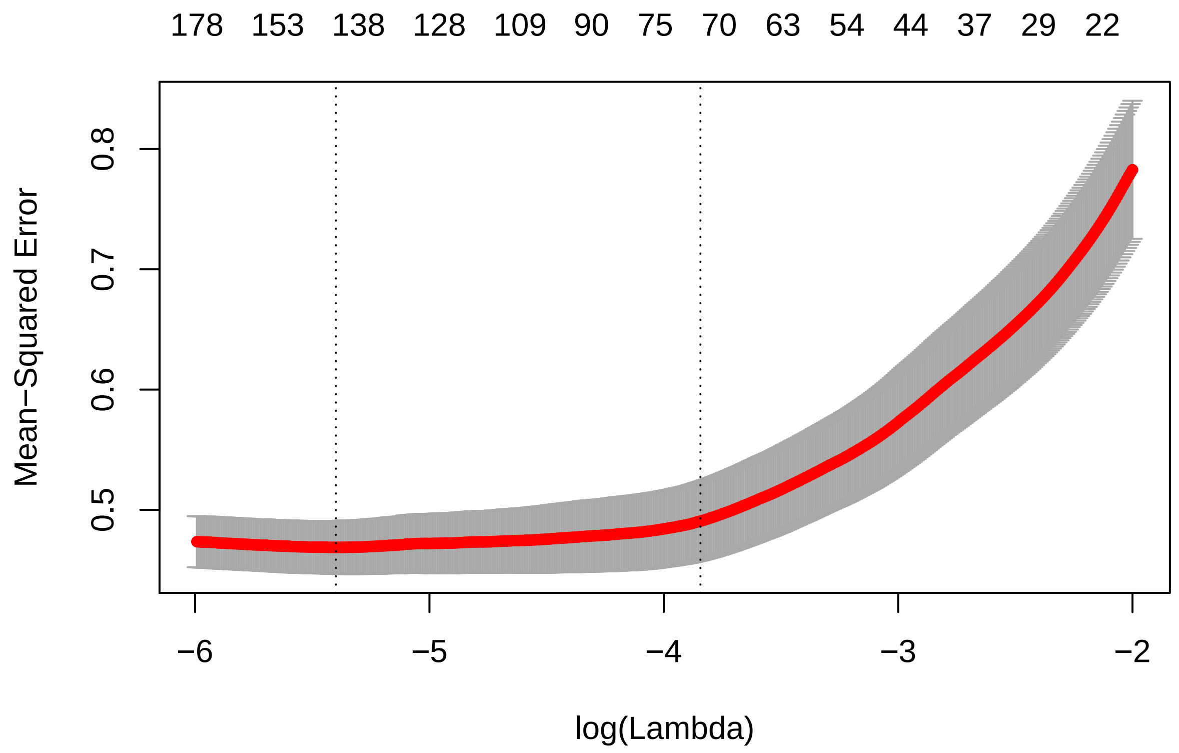
(c) Fit a lasso model on the training data, with λ chosen by cross-validation. Report the test error, along with the number of non-zero coefficient estimates.

1) Fit a lasso model

```
lasso.mod <- glmnet(x_train, y_train, alpha=1, lambda = exp(seq(-6, -2, length=500)))
```

```
set.seed(1)
cv.lasso <- cv.glmnet(x_train, y_train, alpha = 1, lambda = exp(seq(-6, -2, length=500)))
best.lambda <- cv.lasso$lambda.min
```

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



The λ we choose is 0.004522.

2) The number of non-zero coefficient estimates

```
lasso_coef = predict(cv.lasso, s="lambda.min", type="coefficients")
#lasso_coef some coefficients are shrunken to zero
length(lasso_coef[lasso_coef != 0]) # the number of non-zero coefficients
```

```
## <sparse>[ <logic> ] : .M.sub.i.logical() maybe inefficient

## [1] 144
```

The number of non-zero coefficient is 144.

3) MSE

```
lasso.pred = predict(lasso.mod, s=best.lambda, newx = x_test)
mean((lasso.pred - y_test)^2)
```

```
## [1] 0.4991321
```

The test MSE of this lasso model is 0.4991321.

(d) Fit a PCR model on the training data, with M chosen by cross-validation. Report the test error, along with the value of M selected by cross-validation.

1) Fit a PCR model

```r
set.seed(1)
ctrl1 <- trainControl(method = "repeatedcv", number = 10, repeats = 5)
pcr.fit <- train(x_train, y_train,
                 method = "pcr",
                 tuneLength = length(train),
                 trControl = ctrl1,
                 preProc = c("center", "scale"))
```

2) the number of components

```r
pcr.fit$bestTune
```

```
##     ncomp
## 158   158
```

The number of components is 158.

3) MSE

```r
pcr.pred = predict.train(pcr.fit, x_test)
mean((pcr.pred-y_test)^2)
```

```
## [1] 0.5490447
```

The test MSE of this PCR model is 0.5490447.

(e) Briefly discuss the results obtained in (a)~(d).

From the test MSEs of these four models, we can see that the test MSE of linear model and PCR model are pretty high which suggests that the models do not perform well. The test MSEs of ridge and lasso model are relatively low so the models perform well using these two methods. Among all these methods, lasso model's MSE is the lowest and linear model's MSE is the highest, which suggests that lasso model is the best and linear regression model is the worst in this case.