hw3

Mao Xin

Math

（科目：　　）数　学　作　业　纸

编号：　　班级：　　姓名：　　第　　页

1.

a. $\hat{e}_i = y_i - x_i^t \beta \sim N(0, \sigma^2)$

$$L(0, \sigma^2) = \prod_{i=1}^{n} \left(\frac{1}{\sqrt{2\pi}\sigma} \exp\left\{ -\frac{e_i^2}{2\sigma^2} \right\} \right)$$

$$= (2\pi\sigma^2)^{-\frac{n}{2}} \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^{n} e_i^2\right)$$

$$\ln L = -\frac{1}{2\sigma^2} \sum_{i=1}^{n} e_i^2 - \frac{n}{2} \ln \sigma^2 - \frac{n}{2} \ln(2\pi)$$

要求得一组 $\hat{\beta}_1, \cdots, \hat{\beta}_k$ 使 $\ln L$ 最大

即求 $\frac{\partial \sum_{i=1}^{n} e_i^2}{\partial \beta_1} = 0, \cdots, \frac{\partial \sum_{i=1}^{n} e_i^2}{\partial \beta_k} = 0$

与 OLS 目标一致，两者等价.

求得 $\hat{\beta} = (X^T X)^{-1} X^T y$.

b. 对单变量.

$\sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y - \beta_i x_i)^2$，使 $\frac{\partial \sum_{i=1}^{n} e_i^2}{\partial \beta_i^*} = 0$.

β_i 只与 y 及 x_i 有关

而 a 中 $\hat{\beta}$ 与 X 整体有关

Programing

Import libarary

```
if (!require("glmnet")) {
  install.packages("glmnet", dependencies = TRUE)
  library(glmnet)
}
```

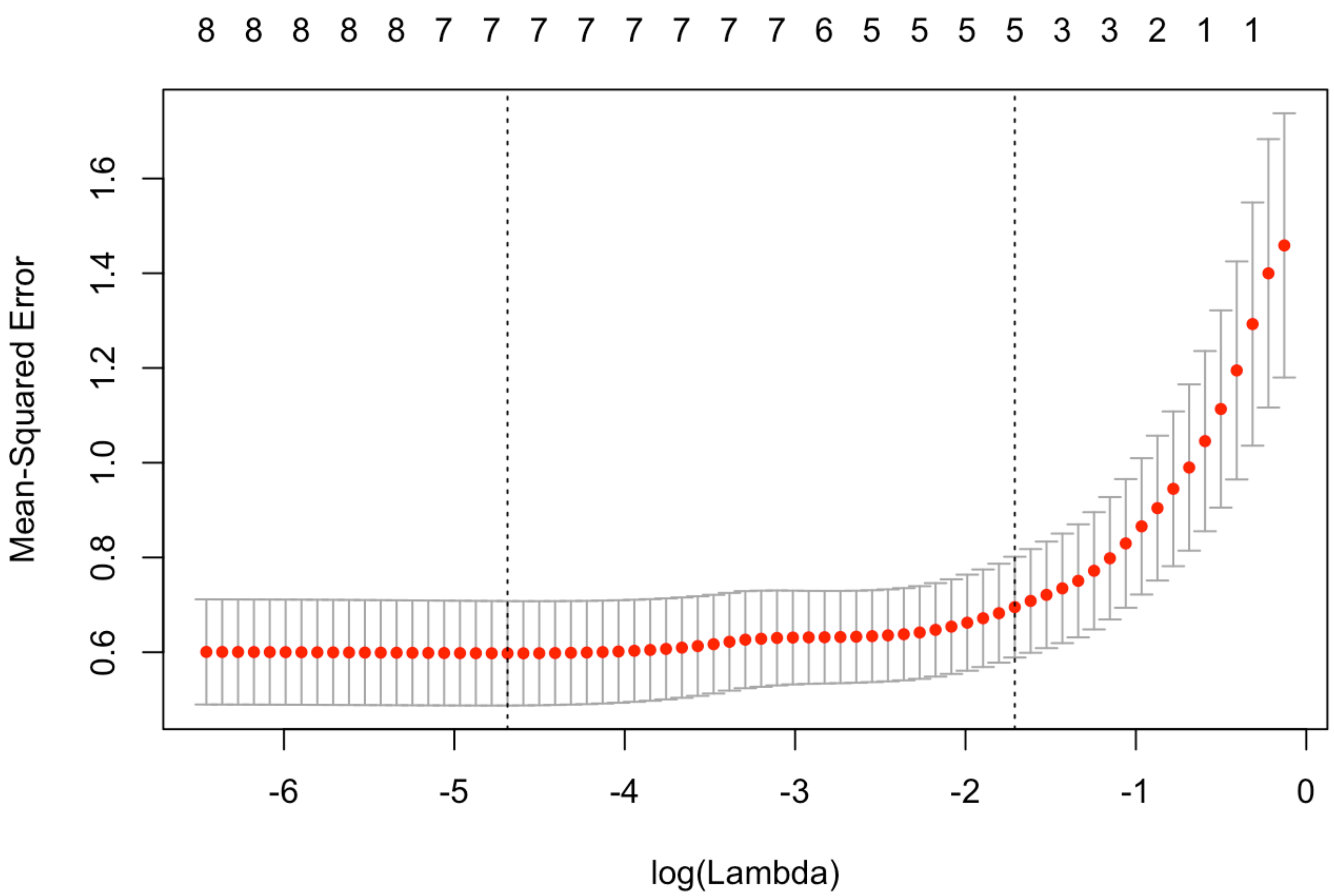
```
## Loading required package: glmnet
## Loading required package: Matrix
## Loading required package: foreach
## Loaded glmnet 2.0-2
```

LASSO using glmnet Libarary

Method

```
# Load Data
prostate = read.table('/Users/maoxin/Desktop/prostate.txt')
prostate_train = prostate[which(prostate$train==TRUE),]
prostate_test = prostate[which(prostate$train==FALSE),]
x_train <- as.matrix(prostate_train[,1:8])
y_train <- as.matrix(prostate_train[,9])
x_test <- as.matrix(prostate_test[,1:8])
y_test <- as.matrix(prostate_test[,9])

model <- cv.glmnet(x_train, y_train)
# lambda.1se corresponds to the simplest model that has comparable error to the bes
t model given the uncertainty
lambda = model$lambda.1se
plot(model)
```



Regression Parameters And Model Estimation

```
param <- coef(model$glmnet.fit, s=lambda)
print(param)
```

```
## 9 x 1 sparse Matrix of class "dgCMatrix"
##                        1
## (Intercept) 0.2602328543
## lcavol      0.4549883478
## lweight     0.4183048099
## age         .
## lbph        0.0199337242
## svi         0.2745819509
## lcp         .
## gleason     .
## pgg45       0.0005623797
```

```
y_predict <- predict(model, newx=x_test, s=lambda)
RSS <- sum((y_predict - y_test)^2)
print(RSS)
```

```
## [1] 13.96641
```

LASSO using Subgradient

Method

```
# eliminate beta_0
x_train<-scale(prostate_train[,1:8])
sigma<-attr(x_train,"scaled:scale")
y_train<-prostate_train[,9]
y_train<-y_train-mean(y_train)
x_train<-as.matrix(x_train)
y_train<-as.matrix(y_train)

w <- solve(t(x_train) %*% x_train + lambda) %*% t(x_train) %*% y_train

while(TRUE) {
  w_old <- w
  for(j in 1:8) {
    a_j <- sum(x_train[,1]^2)
    c_j <- sum(x_train[,j] * (y_train - x_train %*% w +  x_train[,j] * w[j]))
    # soft threshold
    w[j] <- sign(c_j/a_j) * max(abs(c_j/a_j) - lambda, 0)
  }

  # converged
  if(max(abs(w - w_old)) <= 1e-10) {
    break
  }
}

w <- w / sigma
```

Regression Parameters

```
print(w)
```

```
##                [,1]
## lcavol  0.4551345228
## lweight 0.4193542010
## age     0.0000000000
## lbph    0.0208146345
## svi     0.2767985067
## lcp     0.0000000000
## gleason 0.0000000000
## pgg45   0.0005907177
```

Discussion

The result of LASSO using subgradient method is simmilar to the result using glmnet libarary. They set the same coefficients (age, lcp, gleason) to be zero and the difference of the rest coefficients are negligible.

The LASSO using subgradient, however, can't return the intercept of model directly. So we didn't calculate the RSS as we did in the LASSO using glmnet.