STAT4001 Homework 2

Chan Sze Yuen Syngiene 1155127616

13/11/2021

Q1

(a) Recall the OLS estimation for $\hat{\beta}_0$ and $\hat{\beta}_1$:

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1\bar{x}$$
$$\hat{\beta}_1 = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2} = \frac{S_{XY}}{S_{XX}}$$

We first show that $\hat{\beta}_1$ is unbiased by finding the expectation of the estimator:

$$\begin{aligned}
E(\hat{\beta}_1) &= E(\frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}) \\
&= \frac{1}{\sum_{i=1}^n (x_i - \bar{x})^2} \sum_{i=1}^n (x_i - \bar{x}) E(y_i - \bar{y}) \\
&= \frac{1}{\sum_{i=1}^n (x_i - \bar{x})^2} \sum_{i=1}^n (x_i - \bar{x}) E(\beta_0 + \beta_1 x_i + \varepsilon - \beta_0 - \beta_1 \bar{x}_i - 0) \text{ as sum of the error term=0} \\
&= \frac{1}{\sum_{i=1}^n (x_i - \bar{x})^2} \sum_{i=1}^n (x_i - \bar{x})^2 \beta_1 \\
&= \beta_1
\end{aligned}$$

Hence, with the result of the unbiased estimator $\hat{\beta}_1$,

$$\begin{aligned}
E(\hat{\beta}_0) &= E(\bar{y} - \hat{\beta}_1\bar{x}) \\
&= E(\beta_0 + \beta_1\bar{x} - \beta_1\bar{x}) \\
&= \beta_0
\end{aligned}$$

from this we can also see that $\hat{\beta}_0$ is unbiased if and only if $\hat{\beta}_1$ is unbiased

(b) Recall the Ridge regression estimator, $\hat{\beta}_0$ and $\hat{\beta}_1$:

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1\bar{x}$$
$$\hat{\beta}_1 = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2 + \lambda} = \frac{S_{XY}}{S_{XX} + \lambda}$$

Same with the OLS part, we start with $\hat{\beta}_1$:

$$
\begin{aligned}
E(\hat{\beta}_1) &= E(\frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sum_{i=1}^{n}(x_i-\bar{x})^2+\lambda}) \\
&= \frac{1}{\sum_{i=1}^{n}(x_i-\bar{x})^2+\lambda}\sum_{i=1}^{n}(x_i-\bar{x})E(y_i-\bar{y}) \\
&= \frac{1}{\sum_{i=1}^{n}(x_i-\bar{x})^2+\lambda}\sum_{i=1}^{n}(x_i-\bar{x})^2\beta_1 \\
&\neq \beta_1
\end{aligned}
$$

And

$$
\begin{aligned}
E(\hat{\beta}_0) &= E(\bar{y}-\hat{\beta}_1\bar{x}) \\
&= E(\beta_0+\beta_1\bar{x}-\frac{1}{\sum_{i=1}^{n}(x_i-\bar{x})^2+\lambda}\sum_{i=1}^{n}(x_i-\bar{x})^2\beta_1\bar{x}) \\
&\neq \beta_0
\end{aligned}
$$

Hence $\hat{\beta}_1$ is biased for ridge regression setting, and by the OLS $\hat{\beta}_0$ part, we can also conclude the $\hat{\beta}_0$ is also biased as $\hat{\beta}_1$ is biased.

Q2

Consider the objective function $f(\beta_j) = a\beta_j^2 - 2b\beta_j + \lambda|\beta_j|$

If $\beta_j < 0$, $f'(\beta_j) = 2a\beta_j - 2b - \lambda$, set it to 0 we have $\beta_j = \frac{2b+\lambda}{2a}$

Since $\beta_j < 0$ and we are given a>0, we must have $2b + \lambda < 0$ and hence $b < -\frac{-\lambda}{2} < 0$

Now we let β_j be the minimizer for the f(β_j),

If $\beta_j > 0$, $f(-\beta_j) = a\beta_j^2 + 2b\beta_j + \lambda|\beta_j| < f(\beta_j)$ as we know when $\beta_j < 0$,$f(\beta_j) = a\beta_j^2 - 2b\beta_j + \lambda|\beta_j| > f(-\beta_j) = a\beta_j^2 + 2b\beta_j + \lambda|\beta_j|$ as $b < -\frac{-\lambda}{2} < 0$ for $\beta_j < 0$

Hence if β_j is the minimizer for the f(β_j), $\beta_j < 0$ must to be true.

now if $\beta_j = 0$ such that f(β_j) minimized, we first find f(0)=0 and

$$
\begin{aligned}
f(\beta_j) &= a\beta_j^2 - 2b\beta_j + \lambda|\beta_j| \\
&= a\beta_j^2 - 2b\beta_j - \lambda\beta_j \text{ , as we proved } \beta_j < 0 \\
&= a\beta_j(\beta_j - \frac{2b+\lambda}{a}) \\
f(\frac{2b+\lambda}{2a}) &= a(\frac{2b+\lambda}{2a})(\frac{2b+\lambda}{2a} - \frac{2b+\lambda}{a}) \\
&= -a(\frac{2b+\lambda}{2a})^2 < f(0) = 0
\end{aligned}
$$

Hence we have β_j is not 0 as 0 will not minimize objective function Hence β_j<0 and $\hat{\beta}_j = \frac{2b+\lambda}{2a}$ minimize objective function.

Q3

(a) For OLS estimator:

$$
\begin{aligned}
\text{Bias} &= \beta_0 + \beta_1 - E(\hat{y_0}) \\
&= \beta_0 + \beta_1 x_0 - E(\hat{\beta}_0 + \hat{\beta}_1 x_0) \\
&= \beta_0 + \beta_1 x_0 - \beta_0 - \beta_1 x_0 = 0 \text{ as proved in Q1} \\
\text{variance} &= \text{Var}(\hat{\beta}_0 + \hat{\beta}_1 x_0 + \varepsilon) \\
&= \text{Var}(\hat{\beta}_0) + \text{Var}(\hat{\beta_1 x_0}) + 2\text{Cov}(\hat{\beta}_0, \hat{\beta_1 x_0}) + \sigma^2 \text{since noise and x are indepentant} \\
&= \sigma^2\left(\frac{1}{n} + \frac{\bar{x}^2}{\sum_{i=1}^n (x_i - \bar{x})^2}\right) + \frac{x_0^2 \sigma^2}{\sum_{i=1}^n (x_i - \bar{x})^2} - \frac{2\bar{x}\sigma^2}{\sum_{i=1}^n (x_i - \bar{x})^2} + \sigma^2 \\
&= \sigma^2\left(\frac{1}{n} + 1 + \frac{(x_0 - \bar{x})^2}{\sum_{i=1}^n (x_i - \bar{x})^2}\right)
\end{aligned}
$$

x_0 -5

(b) For the Ridge regression setting:

$$
\begin{aligned}
\text{Bias} &= \beta_0 + \beta_1 x_0 - E(\hat{y_0}) \\
&= \beta_0 + \beta_1 x_0 - E(\hat{\beta}_0 + \hat{\beta}_1 x_0) \\
&= \beta_0 + \beta_1 x_0 - \beta_0 - \beta_1 \bar{x} + \frac{1}{\sum_{i=1}^n (x_i - \bar{x})^2 + \lambda} \sum_{i=1}^n (x_i - \bar{x})^2 \beta_1 \bar{x} - \frac{x_0}{\sum_{i=1}^n (x_i - \bar{x})^2 + \lambda} \sum_{i=1}^n (x_i - \bar{x})^2 \beta_1 \text{ from Q1} \\
&= -\beta_1\left(\frac{\sum_{i=1}^n (x_i - \bar{x})^2 (\bar{x} - x_0)}{\sum_{i=1}^n (x_i - \bar{x})^2 + \lambda}\right) \\
\text{Bias}^2 &= \left[\beta_1\left(\frac{\sum_{i=1}^n (x_i - \bar{x})^2 (\bar{x} - x_0)}{\sum_{i=1}^n (x_i - \bar{x})^2 + \lambda}\right)\right]^2 > 0 \\
\text{variance} &= \text{Var}(\hat{\beta}_0 + \hat{\beta}_1 x_0 + \varepsilon) \\
&= \text{Var}(\hat{\beta}_0) + \text{Var}(\hat{\beta_1 x_0}) + 2\text{Cov}(\hat{\beta}_0, \hat{\beta_1 x_0}) + \sigma^2 \text{since noise and x are indepentant} \\
&= \sigma^2\left(\frac{1}{n} + 1 + \frac{(x_0 - \bar{x})^2}{\sum_{i=1}^n (x_i - \bar{x})^2 + \lambda}\right)
\end{aligned}
$$

-3

-5

Q4

(a) Consider the Lagrange multiplier applied to the second from equation:

$$
\begin{aligned}
L(\beta_j, \lambda) &= \sum_{l=1}^m (\tilde{y} - \sum_{j=1}^p \beta_j \tilde{x_{lj}})^2 - \lambda g(\beta_j) \\
&= \sum_{l=1}^m (\tilde{y} - \sum_{j=1}^p \beta_j \tilde{x_{lj}})^2 + \lambda g(\beta_j) \text{ subject to g=0}
\end{aligned}
$$

Which is same as the fisrt form of the objective function. Also recall that in the Lagrange multiplier setting we max/min L subject to g(β_j)=0. So here the sign of g is not important if we subject to g to be 0. Consider the following $g(\beta_j^*)$ for some constant c:

$$
g(\beta_j) = (\beta_j - c)^2 \text{ subject to g=0}
$$

The square here come from the property of the ridge regression that wont force the parameters to 0. From here we can see that we must have the condition that $\beta_j \leq c$ whcih is because $(\beta_j - c)^2 \geq 0$ for all constant c. So we can simply discard g in the argmin statment once we have the new condition of the beta.

Hence the function L become:

we want to retain the origin objective function form

$$L(\beta_j,\lambda) = \sum_{l=1}^{m}(\tilde{y} - \sum_{j=1}^{p}\beta_j \tilde{x_{lj}})^2 - \lambda(\beta_j - c)^2$$
$$= \sum_{l=1}^{m}(\tilde{y} - \sum_{j=1}^{p}\beta_j \tilde{x_{lj}})^2 \text{ subject to } \beta_j \le c$$

-8

Hence two form of the objective function is same with subject to $\beta_j \le c$. Where $(m, \tilde{y}, \tilde{x_{lj}}) = (n, y, x_{ij})$

(b) The objective function of this question become:

$$g(\hat{\beta_1}, \hat{\beta_2}) = \sum_{i=1}^{n}(y - \hat{\beta_1}x_{1i} - \hat{\beta_2}x_{2i})^2 - \lambda(\hat{\beta_1} + \hat{\beta_2})$$
$$= \sum_{i=1}^{n}(y - x_{1i}(\hat{\beta_1} + \hat{\beta_2}))^2 - \lambda(\hat{\beta_1}^2 + \hat{\beta_2}^2)$$

Partial differentiate g with respect to $\hat{\beta_1}$ and $\hat{\beta_2}$ withspectively, we will have two same differential equation for $\hat{\beta_1}$ and $\hat{\beta_2}$:

$$\frac{\partial g}{\partial \hat{\beta_1}} = 2\sum_{i=1}^{n}(y - x_{1i}(\hat{\beta_1} + \hat{\beta_2}))(x_{1i}) - 2\lambda\hat{\beta_1}$$
$$\frac{\partial g}{\partial \hat{\beta_2}} = 2\sum_{i=1}^{n}(y - x_{1i}(\hat{\beta_1} + \hat{\beta_2}))(x_{1i}) - 2\lambda\hat{\beta_2}$$

two x_i are different: x_i1 and x_i2

Which proved that $\hat{\beta_1} = \hat{\beta_2}$

-10

Q5

(a)

```
X=rnorm(100,0,1) ; noise=rnorm(100,0,0.1)
```

(b)

```
Y=1+X+X^2+X^3+noise
```

(c) We first start with setting the number of the folds and its labels, here we simply choose 5 fold.

```
set.seed(1155127616)
X_Q3=matrix(c(X,X^2,X^3,X^4,X^5,X^6,X^7,X^8,X^9,X^10),ncol=10,byrow=F)
Y_Q3=Y
nFold=5
cv_fold_label=sample(nFold, length(X), replace=T)
cv_fold_label
```

```
##   [1] 5 2 5 3 2 1 4 5 1 5 1 1 1 4 4 4 1 5 1 2 1 4 5 4 1 3 2 5 4 3 1 3 2 5 2 5 4
##  [38] 1 5 3 1 1 1 3 3 2 1 1 1 1 1 2 2 5 5 4 2 5 4 1 2 4 2 5 1 3 5 3 4 1 3 5 3 4
##  [75] 2 3 3 2 1 3 1 1 4 2 3 1 4 5 3 5 2 4 1 5 2 2 2 2 4 3
```

Use fold 1 as test data, fold 2-5 as training data

```r
library(MASS)
```

```
## Warning: package 'MASS' was built under R version 3.6.3
```

```r
library(ISLR)
```

```
## Warning: package 'ISLR' was built under R version 3.6.3
```

```r
library(glmnet)
```

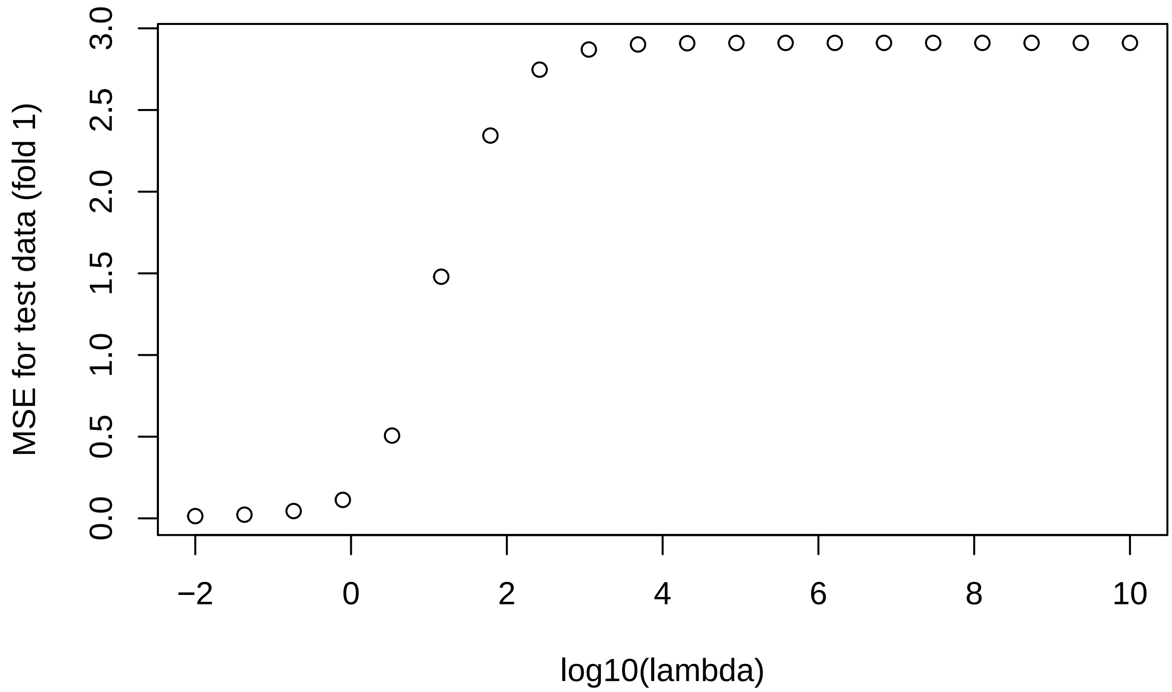
```
## Warning: package 'glmnet' was built under R version 3.6.3

## Loading required package: Matrix

## Warning: package 'Matrix' was built under R version 3.6.2

## Loaded glmnet 4.0-2
```

```r
fold=1
train_label <- which(cv_fold_label!=fold)
test_label <- which(cv_fold_label==fold)
allLambda=10^seq(10,-2,length=20)
allLambda
```

```
##  [1] 1.000000e+10 2.335721e+09 5.455595e+08 1.274275e+08 2.976351e+07
##  [6] 6.951928e+06 1.623777e+06 3.792690e+05 8.858668e+04 2.069138e+04
## [11] 4.832930e+03 1.128838e+03 2.636651e+02 6.158482e+01 1.438450e+01
## [16] 3.359818e+00 7.847600e-01 1.832981e-01 4.281332e-02 1.000000e-02
```

-5

```r
## sample size for test data
n_k <- length(test_label)
mse_test <- rep(NA, length(allLambda))
for (i in 1:length(allLambda)){
lambda <- allLambda[i]
ridge.mod <- glmnet(X_Q3[train_label,], Y_Q3[train_label],alpha=[illegible],lambda=lambda)
ridge.pred <- predict(ridge.mod,s=lambda,newx=X_Q3[test_label,])
mse_test[i] <- mean((ridge.pred-Y_Q3[test_label])^2)
}
plot(log10(allLambda), mse_test, xlab="log10(lambda)", ylab="MSE for test data (fold 1)")
```



We now Perform cross-validation

```
mse_allFold <- matrix(nrow=nFold, ncol=length(allLambda))
n_allFold <- rep(NA, nFold)
for (fold in 1:nFold){
train_label <- which(cv_fold_label!=fold)
test_label <- which(cv_fold_label==fold)
## sample size for test data
n_allFold[fold] <- length(test_label)
for (i in 1:length(allLambda)){
lambda <- allLambda[i]
ridge.mod <- glmnet(X_Q3[train_label,], Y_Q3[train_label],alpha=0,lambda=lambda)
ridge.pred <- predict(ridge.mod,s=lambda,newx=X_Q3[test_label,])
mse_allFold[fold, i] <- mean((ridge.pred-Y_Q3[test_label])^2)
}
}
n_allFold
```

```
## [1] 27 20 17 17 19
```

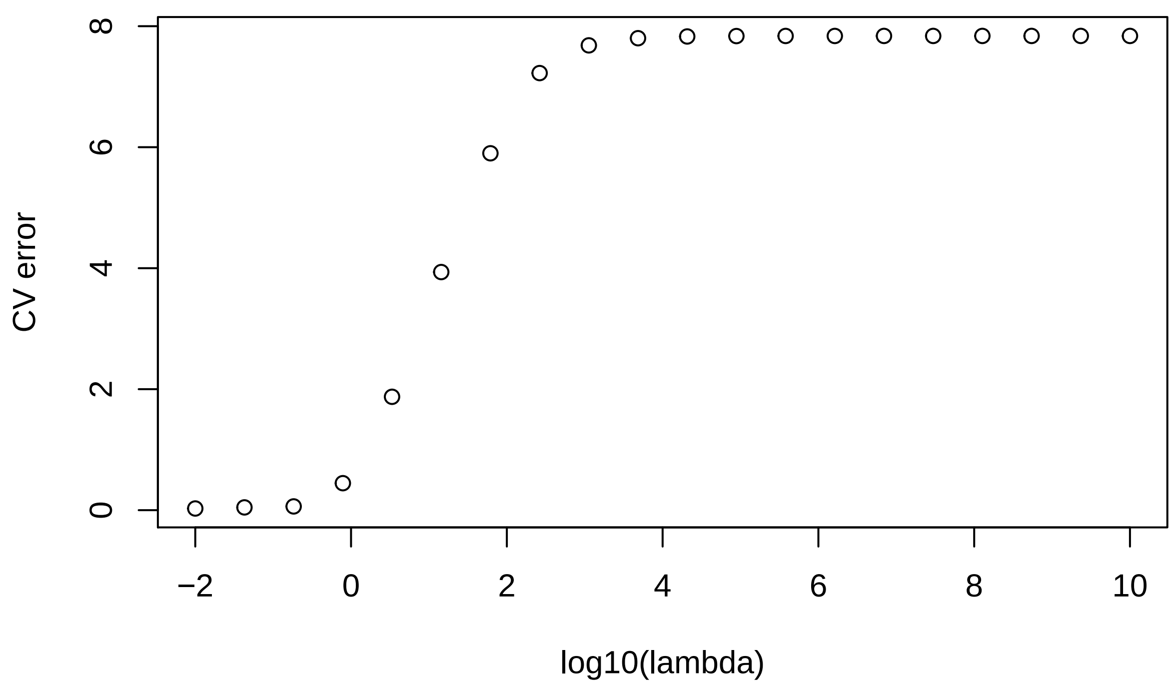
```
mse_allFold
```

```
##          [,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7]
## [1,] 2.910960 2.910960 2.910960 2.910960 2.910959 2.910954 2.910932
## [2,] 5.961949 5.961949 5.961949 5.961948 5.961947 5.961940 5.961910
## [3,] 5.863697 5.863696 5.863696 5.863695 5.863692 5.863677 5.863612
```

```
## [4,] 15.427140 15.427140 15.427139 15.427135 15.427117 15.427041 15.426714
## [5,] 11.794624 11.794624 11.794624 11.794623 11.794621 11.794613 11.794578
##           [,8]      [,9]     [,10]     [,11]     [,12]     [,13]     [,14]
## [1,]  2.910837  2.910432  2.908699  2.901317  2.870287  2.747081  2.343606
## [2,]  5.961781  5.961229  5.958869  5.948801  5.906247  5.733443  5.122392
## [3,]  5.863334  5.862147  5.857064  5.835433  5.744797  5.389875  4.295271
## [4,] 15.425315 15.419328 15.393734 15.284946 14.832762 13.122760  8.627514
## [5,] 11.794427 11.793783 11.791026 11.779254 11.729345 11.524103 10.763795
##         [,15]     [,16]     [,17]      [,18]       [,19]       [,20]
## [1,] 1.479257 0.5067345 0.1130800 0.04474375 0.022446537 0.013724742
## [2,] 3.567834 1.3707394 0.2461154 0.04490852 0.012348888 0.006075175
## [3,] 2.385451 0.8028509 0.1991804 0.05618344 0.009828093 0.004839453
## [4,] 4.662414 2.9152960 0.8615691 0.09925486 0.155020731 0.062348128
## [5,] 8.551733 4.3738901 0.9805412 0.07830419 0.049588754 0.061282999
```

Summarize and obtain cross-validation error:

```
cv_error <- matrix(n_allFold/sum(n_allFold), nrow=1)%*%mse_allFold
plot(log10(allLambda), cv_error, xlab="log10(lambda)", ylab="CV error")
```




```
bestLambda <- which.min(cv_error) #lambda with lowerest CV error
bestLambda
```

```
## [1] 20
```

now we fit the ridge regression using this lambda

```
final_ridge_mod <- glmnet(X_Q3, Y_Q3,alpha=0,lambda=bestLambda)
final_ridge_mod$beta #coeff we want
```

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##                s0
## V1   2.761663e-01
## V2   1.259971e-02
## V3   1.012256e-01
## V4  -6.613814e-03
## V5   1.626607e-02
## V6  -2.150539e-03
## V7   2.162037e-03
## V8  -4.440864e-04
## V9   2.989636e-04
## V10 -8.098156e-05
```

(d) We just follow (c) with different Y:

```
set.seed(1155127616)
X_Q4=matrix(c(X,X^2,X^3,X^4,X^5,X^6,X^7,X^8,X^9,X^10),ncol=10,byrow=F)
Y_Q4=1+X^7+noise
nFold=5
cv_fold_label=sample(nFold, length(X), replace=T)
cv_fold_label
```

```
##   [1] 5 2 5 3 2 1 4 5 1 5 1 1 1 4 4 4 1 5 1 2 1 4 5 4 1 3 2 5 4 3 1 3 2 5 2 5 4
##  [38] 1 5 3 1 1 1 3 3 2 1 1 1 1 1 2 2 5 5 4 2 5 4 1 2 4 2 5 1 3 5 3 4 1 3 5 3 4
##  [75] 2 3 3 2 1 3 1 1 4 2 3 1 4 5 3 5 2 4 1 5 2 2 2 2 4 3
```

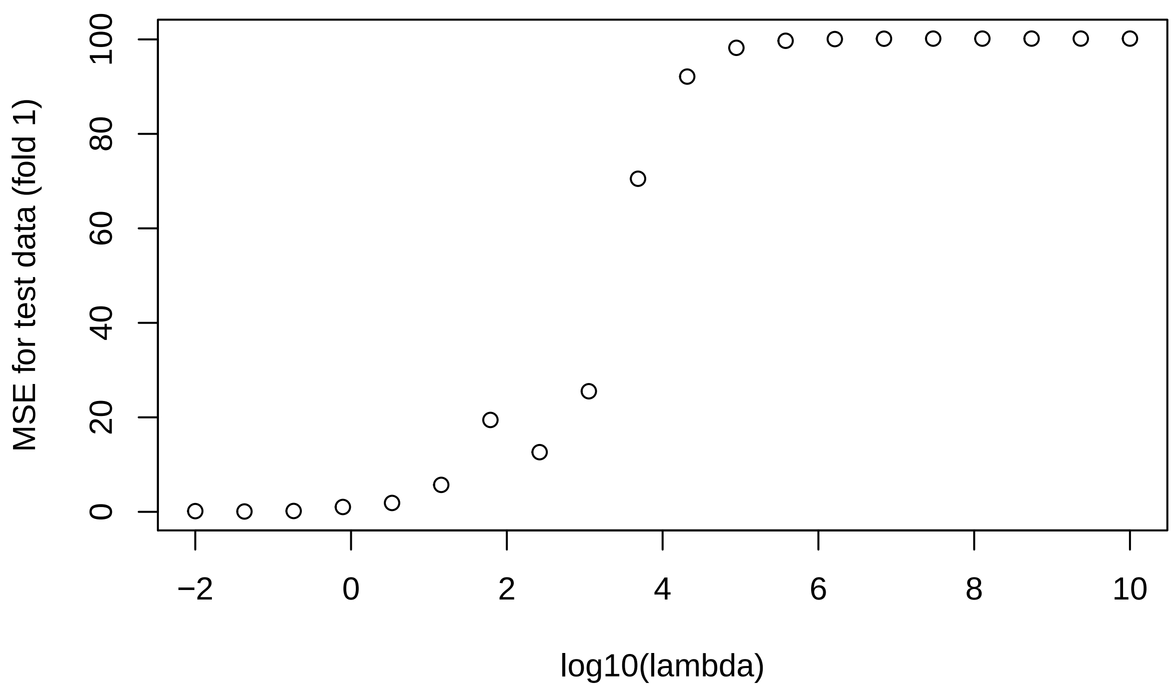
Use fold 1 as test data, fold 2-5 as training data

```
library(MASS)
library(ISLR)
library(glmnet)
fold=1
train_label <- which(cv_fold_label!=fold)
test_label <- which(cv_fold_label==fold)
allLambda=10^seq(10,-2,length=20)
allLambda
```

```
##  [1] 1.000000e+10 2.335721e+09 5.455595e+08 1.274275e+08 2.976351e+07
##  [6] 6.951928e+06 1.623777e+06 3.792690e+05 8.858668e+04 2.069138e+04
## [11] 4.832930e+03 1.128838e+03 2.636651e+02 6.158482e+01 1.438450e+01
## [16] 3.359818e+00 7.847600e-01 1.832981e-01 4.281332e-02 1.000000e-02
```

```
## sample size for test data
n_k <- length(test_label)
mse_test <- rep(NA, length(allLambda))
for (i in 1:length(allLambda)){
```

```r
lambda <- allLambda[i]
ridge.mod <- glmnet(X_Q4[train_label,], Y_Q4[train_label],alpha=0,lambda=lambda)
ridge.pred <- predict(ridge.mod,s=lambda,newx=X_Q4[test_label,])
mse_test[i] <- mean((ridge.pred-Y_Q4[test_label])^2)
}
plot(log10(allLambda), mse_test, xlab="log10(lambda)", ylab="MSE for test data (fold 1)")
```



We now Perform cross-validation

```r
mse_allFold <- matrix(nrow=nFold, ncol=length(allLambda))
n_allFold <- rep(NA, nFold)
for (fold in 1:nFold){
train_label <- which(cv_fold_label!=fold)
test_label <- which(cv_fold_label==fold)
## sample size for test data
n_allFold[fold] <- length(test_label)
for (i in 1:length(allLambda)){
lambda <- allLambda[i]
ridge.mod <- glmnet(X_Q4[train_label,], Y_Q4[train_label],alpha=0,lambda=lambda)
ridge.pred <- predict(ridge.mod,s=lambda,newx=X_Q4[test_label,])
mse_allFold[fold, i] <- mean((ridge.pred-Y_Q4[test_label])^2)
}
}
n_allFold
```

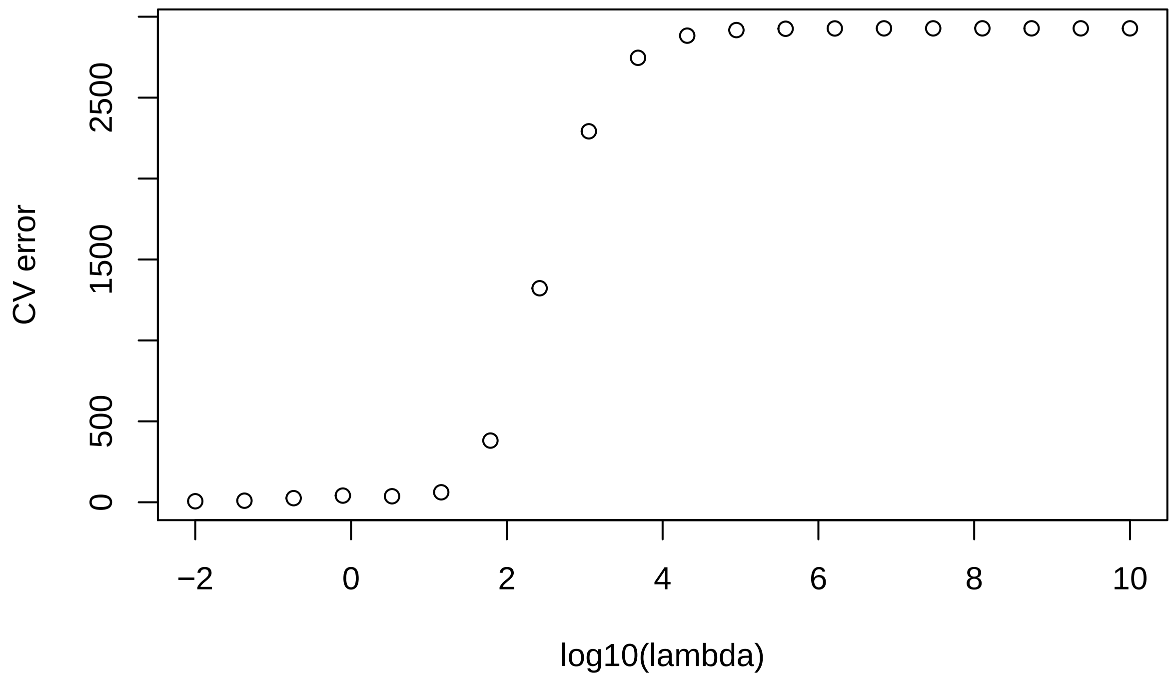
```
## [1] 27 20 17 17 19
```

```r
mse_allFold
```

```
##             [,1]        [,2]        [,3]        [,4]        [,5]        [,6]
## [1,]   100.16485   100.16479   100.16455   100.16350   100.15900   100.13976
## [2,]    88.43757    88.43753    88.43739    88.43679    88.43423    88.42325
## [3,]  1288.68213  1288.68168  1288.67975  1288.67149  1288.63614  1288.48482
## [4,] 15088.07532 15088.07412 15088.06899 15088.04700 15087.95289 15087.54996
## [5,]   522.95737   522.95732   522.95712   522.95624   522.95249   522.93642
##             [,7]        [,8]        [,9]       [,10]       [,11]       [,12]
## [1,]   100.05741    99.70545    98.21648    92.12241    70.50082    25.52393
## [2,]    88.37626    88.17540    87.32454    83.82631    71.16774    42.37148
## [3,]  1287.83713  1285.06742  1273.31282  1224.60198  1042.47014   572.34333
## [4,] 15085.82506 15078.44300 15046.88961 14913.03881 14360.38663 12320.39483
## [5,]   522.86767   522.57368   521.32628   516.16151   496.89716   446.80610
##           [,13]      [,14]      [,15]      [,16]      [,17]       [,18]
## [1,]   12.63366   19.44717   5.704804   1.878552   1.024262   0.1793433
## [2,]   23.99668   17.53961   8.714659   5.403025   2.505977   0.5459149
## [3,]  113.08893   23.24046   6.737452   2.762650   3.775238   2.7035021
## [4,] 7184.93270 1805.29636  82.717365  49.383184 153.850001 118.0576641
## [5,]  386.61846  323.74822 227.293204 140.372746  71.889885  24.6493321
##            [,19]      [,20]
## [1,]  0.07438211  0.1503327
## [2,]  0.06042782  0.1095662
## [3,]  0.09429324  0.9489481
## [4,] 52.48516092 33.7615152
## [5,]  3.83085800  1.7545692
```

Summarize and obtain cross-validation error:

```r
cv_error <- matrix(n_allFold/sum(n_allFold), nrow=1)%*%mse_allFold
plot(log10(allLambda), cv_error, xlab="log10(lambda)", ylab="CV error")
```




```
bestLambda <- which.min(cv_error) #lambda with lowerest CV error
bestLambda
```

```
## [1] 20
```

now we fit the ridge regression using this lambda

```
final_ridge_mod <- glmnet(X_Q4, Y_Q4,alpha=0,lambda=bestLambda)
final_ridge_mod$beta #coeff we want
```

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##               s0
## V1  -0.132740185
## V2   1.277092476
## V3   2.544079164
## V4  -0.479377235
## V5   0.888058141
## V6  -0.215256504
## V7   0.174522535
## V8  -0.050553288
## V9   0.029692561
## V10 -0.009736758
```