Assignment 3

Keiryn Hart, 300428418

29/05/2020

Question 1:

a)

```
set.seed(123)
x <- matrix(rnorm(1000 * 20), 1000, 20)
X <- cbind(rep(1,1000),x)
b <- c(2,3,3,3,3,3,3,3,3,3,0,0,0,0,0,0,0,0,0,0,0)
e <- rnorm(1000)
eps <- rnorm(1000)
y <- X %*% b + e
dat = data.frame(y,x)
```

b)

```
nrow(dat)
```

```
## [1] 1000
```

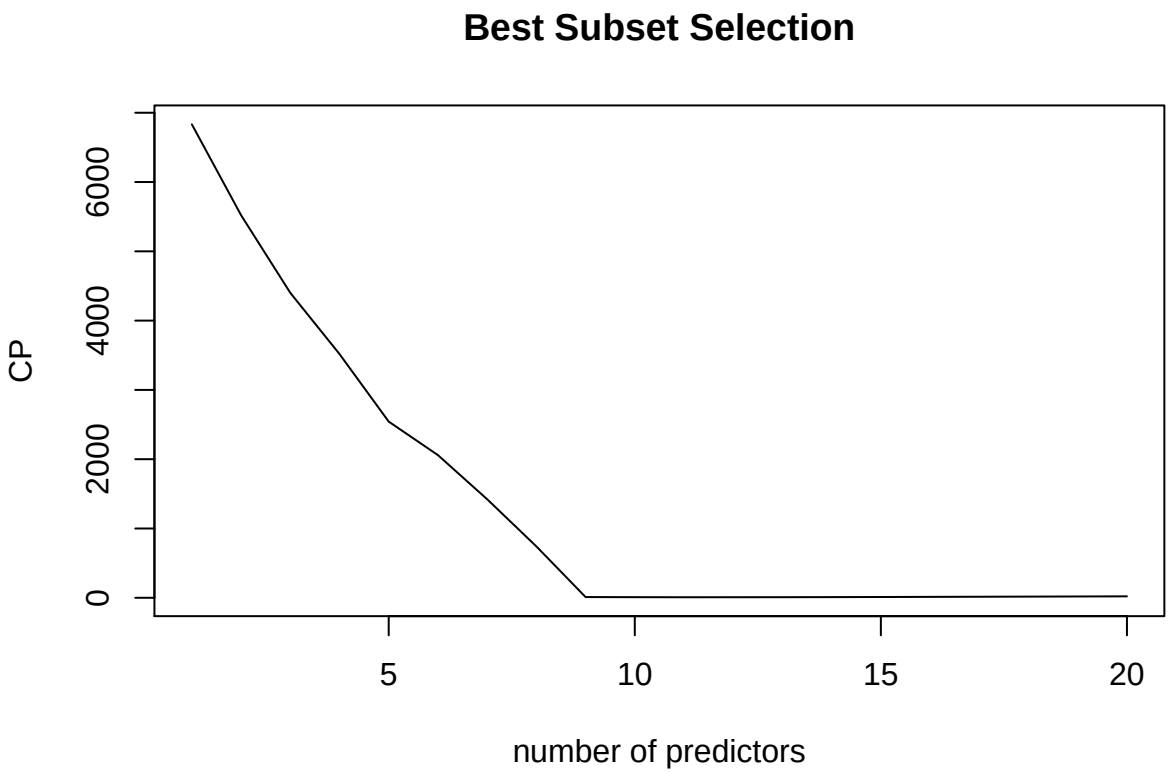
```
train <- sample(c(TRUE, FALSE), nrow(dat), rep=TRUE, prob = c(0.1 ,0.9))
test <- (!train)
```

c)

```
library(leaps)
```

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

```
best_sub <- regsubsets(y~., data = dat[train,], nvmax = 20)
plot(summary(best_sub)$cp, main = "Best Subset Selection",xlab = "number of predictors", ylab = "CP" ,
```




```
which.min(summary(best_sub)$cp)
```

```
## [1] 11
```

```
coef(best_sub, 11)
```

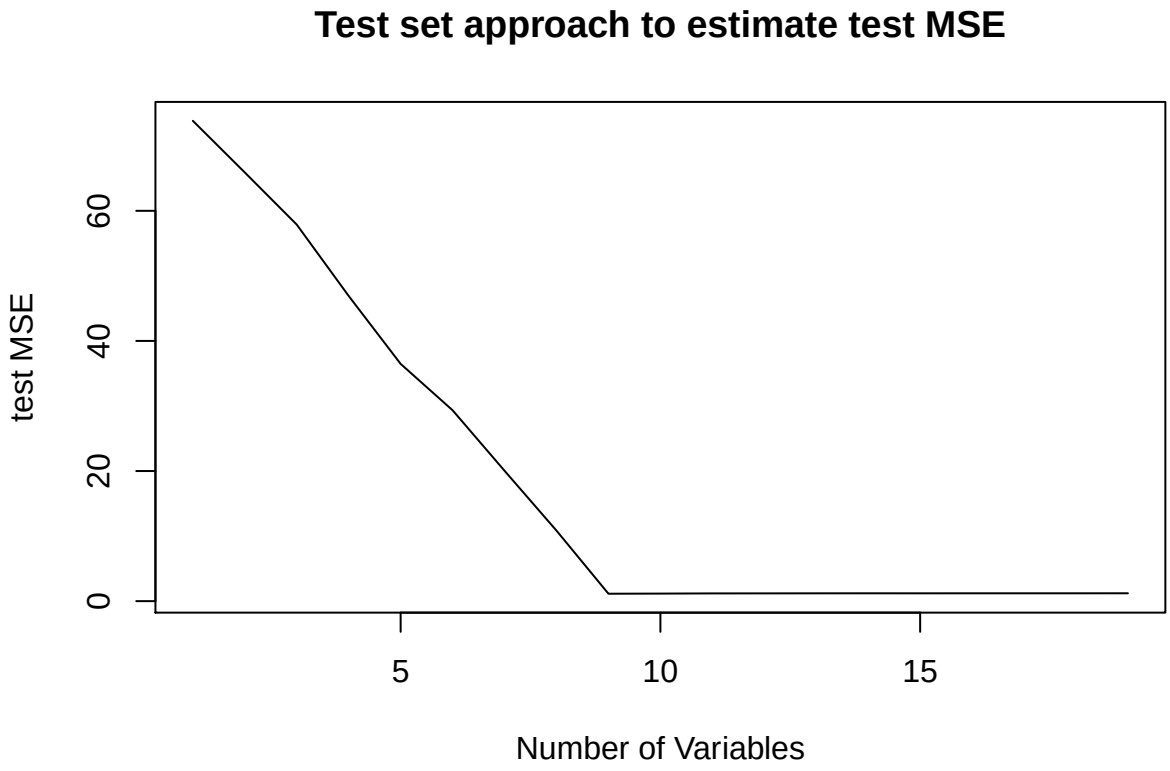
```
## (Intercept)          X1          X2          X3          X4          X5
##   1.8798857   3.0394455   2.9556285   3.0701137   2.9404479   3.1265043
##          X6          X7          X8          X9         X13         X18
##   3.0641333   2.8699228   2.9037091   3.0311166   0.1844532   0.1572677
```

d)

```
test_matrix <- model.matrix(y~., data = dat[test,])

val.errors = rep(NA,19)
for (i in 1:19){
  coefi=coef(best_sub, id=i)
  pred=test_matrix[,names(coefi)]%*%coefi
  val.errors[i]=mean((dat$y[test]-pred)^2)
}

val.errors
```

```
##  [1] 73.833676 65.894840 57.881922 46.912945 36.471318 29.343231 20.035036
##  [8] 10.834905  1.141021  1.161813  1.189277  1.191739  1.203009  1.205631
## [15]  1.200087  1.205736  1.207288  1.207240  1.208145
```

```
plot(val.errors, ,main="Test set approach to estimate test MSE", xlab="Number of Variables",ylab="test M
```



e)

```
best_i = which.min(val.errors)
best_i
```

```
## [1] 9
```

```
coef(best_sub, best_i)
```

```
## (Intercept)          X1          X2          X3          X4          X5
##    1.868101    3.004555    2.911359    3.037909    2.942693    3.152356
##          X6          X7          X8          X9
##    3.023974    2.881382    2.949666    3.057339
```

```
best_subset <- regsubsets(y~., data = dat, nvmax = 20)
coef(best_subset, best_i)
```

```
## (Intercept)          X1          X2          X3          X4          X5
```

```
##    1.980577    3.041514    2.982038    3.006169    3.035737    2.948883
##          X6          X7          X8          X9
##    2.999030    2.982434    3.010707    2.933027
```

Question 2:

a)

```
k <- 10

folds <- sample(1:k, nrow(dat), replace = T)
table(folds)
```

```
## folds
##   1   2   3   4   5   6   7   8   9  10
## 104  88 111  90 104  90 104 115 103  91
```

```
cv.errors <- matrix(NA, k, 20, dimnames = list(NULL, paste(1:20)))

x <- model.matrix(y~., data =dat)
```

```
for(j in 1:k){
  best <- regsubsets(y~., data = dat[folds!=j,], nvmax = 20)
  for(i in 1:20){
    coefi=coef(best, id = i)
    pred=x[folds==j, names(coefi)]%*%coefi
    cv.errors[j,i]=mean((dat$y[folds==j]-pred)^2)
  }
}
cv.errors
```

```
##              1        2        3        4        5        6        7
##  [1,] 56.50259 44.16374 37.79948 32.18366 27.27652 21.47717 17.67583
##  [2,] 88.58473 60.93054 44.70608 39.63688 33.28974 29.45043 20.46584
##  [3,] 80.31967 70.90855 57.86508 46.44843 44.77754 35.82388 18.25673
##  [4,] 92.76036 71.55104 59.80299 56.12459 42.16727 26.52787 17.07132
##  [5,] 73.29580 54.12054 49.84354 43.12096 36.85963 20.79456 14.86443
##  [6,] 74.51995 57.70667 46.90127 41.68717 34.08570 24.55697 16.96159
##  [7,] 87.67630 73.61548 60.42108 54.69262 39.78885 30.78191 21.15339
##  [8,] 70.54854 63.26918 44.86933 41.64777 31.77035 23.68442 15.01502
##  [9,] 58.31748 54.07130 51.89692 45.91230 35.00486 22.77105 17.80524
## [10,] 62.35170 55.85960 48.74076 46.76356 38.93972 31.37928 17.05044
##               8         9        10        11        12        13
##  [1,] 10.490296 1.2178480 1.2403076 1.2477089 1.2581584 1.2619578
##  [2,]  9.876265 1.0329114 1.0614331 1.0617997 1.0584968 1.0570730
##  [3,] 10.644637 1.0173028 1.0176948 1.0244661 1.0235519 1.0217188
##  [4,] 10.990813 1.0526059 1.0607988 1.0603663 1.0626449 1.0607150
##  [5,]  9.789804 0.8570737 0.8558737 0.8550937 0.8554157 0.8569282
##  [6,] 10.871892 0.8987136 0.9185188 0.9199512 0.9207474 0.9198345
##  [7,] 11.456640 1.0989293 1.1158546 1.1368308 1.1378361 1.1347923
##  [8,] 11.151077 0.8530097 0.8623357 0.8678735 0.8691663 0.8732667
##  [9,] 11.921746 1.1690016 1.1713498 1.1772009 1.1801915 1.1764890
```

```
## [10,] 11.636319 1.0314072 1.0624837 1.0734640 1.0761573 1.0844380
##              14        15        16        17        18        19
##  [1,] 1.2579187 1.2518996 1.2507676 1.2494120 1.2531180 1.2533117
##  [2,] 1.0532406 1.0539618 1.0599727 1.0628411 1.0668030 1.0643306
##  [3,] 1.0246849 1.0215514 1.0194621 1.0198593 1.0193062 1.0186747
##  [4,] 1.0701495 1.0751327 1.0746319 1.0757569 1.0784659 1.0770520
##  [5,] 0.8578079 0.8587847 0.8587611 0.8685918 0.8647402 0.8650624
##  [6,] 0.9225905 0.9206623 0.9200604 0.9199369 0.9192513 0.9181695
##  [7,] 1.1363994 1.1320655 1.1295500 1.1280398 1.1305581 1.1299942
##  [8,] 0.8724540 0.8779844 0.8804243 0.8803457 0.8796895 0.8799480
##  [9,] 1.1725005 1.1736578 1.1681391 1.1681700 1.1673194 1.1667996
## [10,] 1.0870394 1.0864445 1.0949349 1.0958724 1.0904304 1.0938214
##              20
##  [1,] 1.2550849
##  [2,] 1.0632665
##  [3,] 1.0181831
##  [4,] 1.0759884
##  [5,] 0.8649634
##  [6,] 0.9198381
##  [7,] 1.1301900
##  [8,] 0.8794881
##  [9,] 1.1667717
## [10,] 1.0933528
```

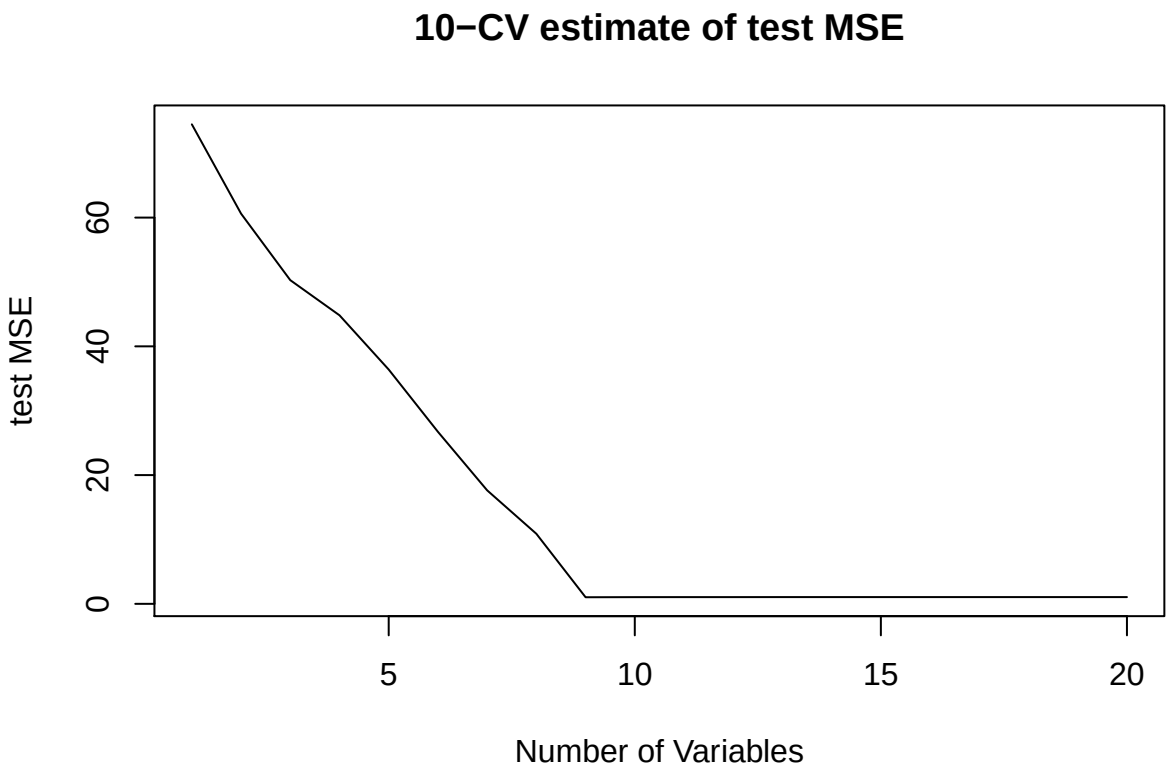
```
mean_cv_errors <- apply(cv.errors, 2, mean)
mean_cv_errors
```

```
##         1         2         3         4         5         6         7
## 74.487713 60.619663 50.284655 44.821794 36.396017 26.724755 17.631984
##         8         9        10        11        12        13        14
## 10.882949  1.022880  1.036665  1.042475  1.044237  1.044721  1.045479
##        15        16        17        18        19        20
##  1.045214  1.045670  1.046883  1.046968  1.046716  1.046713
```

```
#test mse
```

b)

```
plot(mean_cv_errors, ,main="10-CV estimate of test MSE", xlab="Number of Variables",ylab="test MSE",typ
```




```
best_mod <- which.min(mean_cv_errors)
best_mod
```

```
## 9
## 9
```

```
reg.best=regsubsets(y~., data=dat, nvmax=20)
coef(reg.best, best_mod)
```

```
## (Intercept)          X1          X2          X3          X4          X5
##    1.980577    3.041514    2.982038    3.006169    3.035737    2.948883
##          X6          X7          X8          X9
##    2.999030    2.982434    3.010707    2.933027
```

Question 3:

a)

```
library(ISLR)
```

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

```
data(College)
College <- na.omit(College)

train1 <- sample(c(TRUE, FALSE), nrow(College), rep=TRUE)

test1 <-(!train1)

training <- College[train1,]
testing <- College[-train1,]
```

b)

```
#still need this..
library(glmnet)
```

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

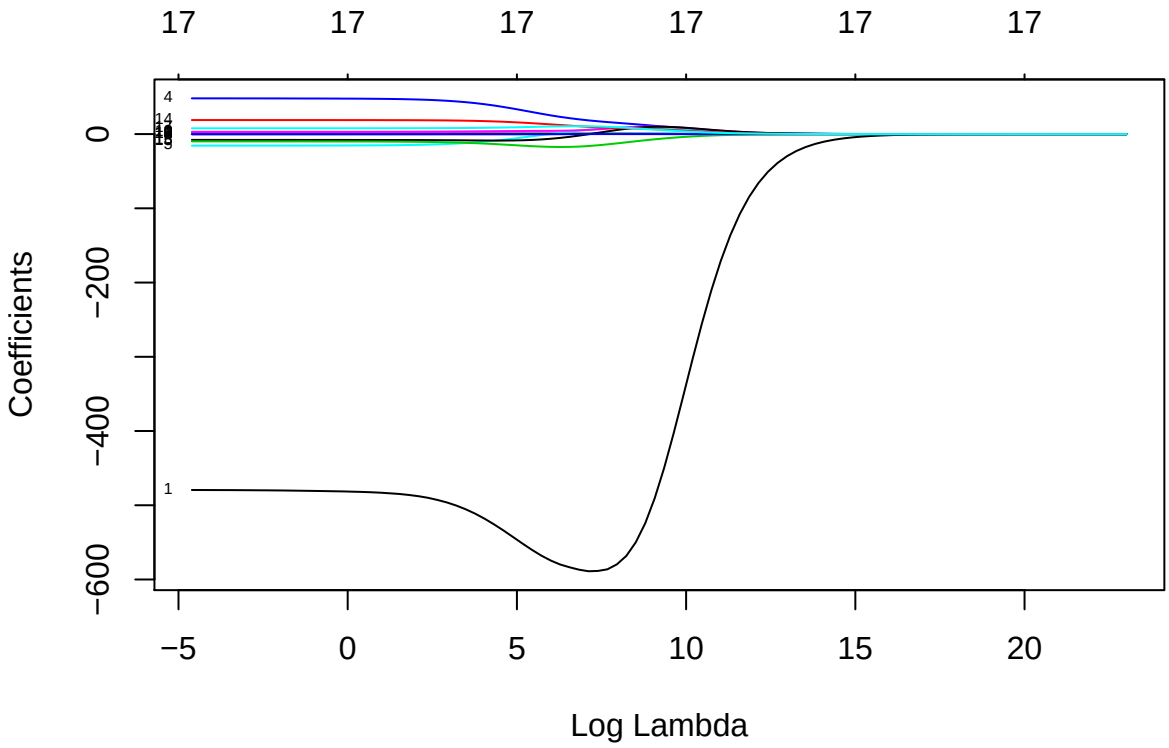
```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.0
```

```
x.train=x[train,]
y.train=y[train]

x.test=x[test,]
y.test=y[test]

model1 <- lm(y.train ~ x.train)
yfit <- model1$fitted.values
```

c)

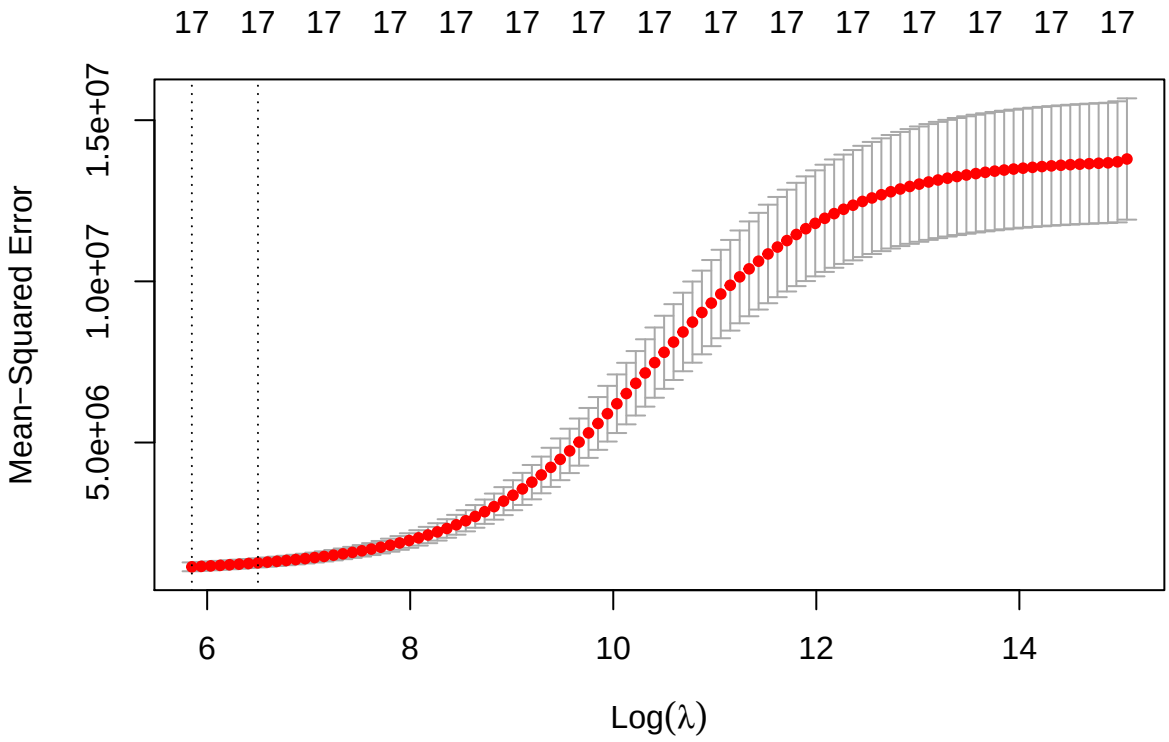
```
#x
ridge.x <- model.matrix(Apps~., training)[,-1]
testx <- model.matrix(Apps~., testing)[,-1]

#y
ridge.y <- training$Apps
testy <- testing$Apps


grid <- 10^seq(10,-2, length = 100)
ridge <- glmnet(ridge.x, ridge.y, alpha = 0, lambda = grid)


plot(ridge, xvar = "lambda", label = TRUE)
```




```
cv.out <- cv.glmnet(ridge.x, ridge.y, alpha = 0)
cv.out
```

```
##
## Call:  cv.glmnet(x = ridge.x, y = ridge.y, alpha = 0)
##
## Measure: Mean-Squared Error
##
##     Lambda Measure     SE Nonzero
## min  347.0 1144113 137290      17
## 1se  665.5 1267264 141662      17
```

```
plot(cv.out)
```




```
bestlam <- cv.out$lambda.min
bestlam
```

```
## [1] 346.9805
```

```
log(bestlam)
```

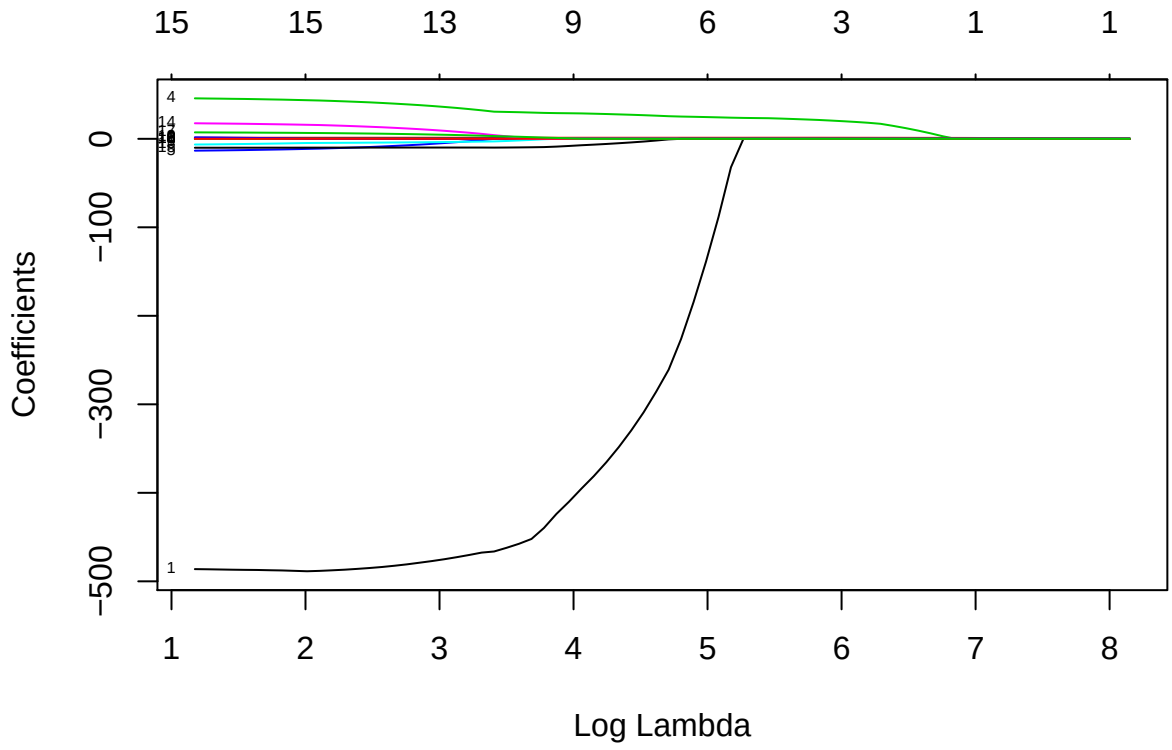
```
## [1] 5.849268
```

```
ridge.best <- predict(cv.out, type = "coefficients", s = bestlam)
mse.ridge <- predict(cv.out, s = 5.795785, newx=testx)
meanmse <- mean((mse.ridge-testy)^2)
meanmse
```

```
## [1] 1735406
```

c)

```
lasso <- glmnet(ridge.x, ridge.y, alpha =1)

plot(lasso, xvar = "lambda", label = TRUE)
```



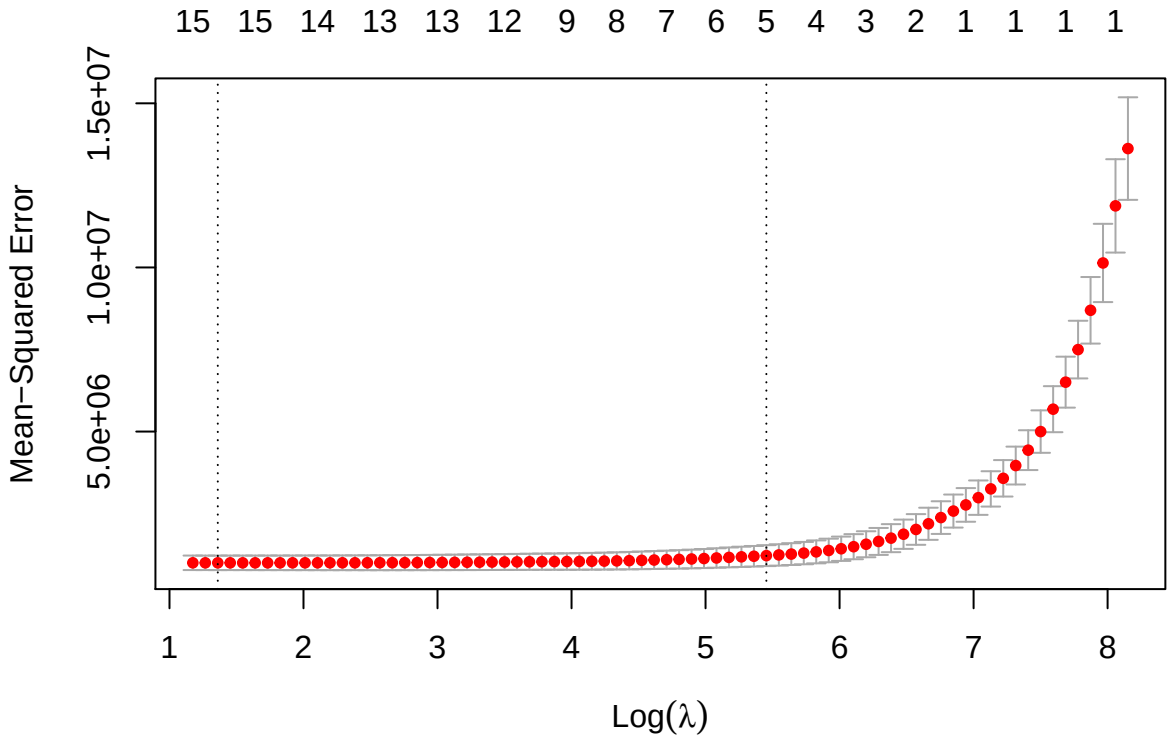

```
#least squared:
predict(lasso, s=exp(0), type = "coefficients")
```

```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                           1
## (Intercept) -1.252138e+03
## PrivateYes  -4.862218e+02
## Accept       1.200725e+00
## Enroll       .
## Top10perc    4.571868e+01
## Top25perc   -1.340104e+01
## F.Undergrad  2.873842e-02
## P.Undergrad  .
## Outstate    -4.447023e-02
## Room.Board   2.697469e-01
## Books       -7.914952e-02
## Personal     1.201735e-01
## PhD          1.689859e+00
## Terminal    -6.472798e+00
## S.F.Ratio    1.755161e+01
## perc.alumni -1.006998e+01
## Expend       6.410243e-02
## Grad.Rate    7.273544e+00
```

```
cv.lasso <- cv.glmnet(ridge.x, ridge.y, alpha = 1)
cv.lasso
```

```
##
## Call:  cv.glmnet(x = ridge.x, y = ridge.y, alpha = 1)
##
## Measure: Mean-Squared Error
##
##     Lambda Measure     SE Nonzero
## min    3.9 1000799 220967      15
## 1se  233.7 1220391 318967       5
```

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




```
lam1se <- cv.lasso$lambda.1se
bestlambda <- predict(cv.lasso, type="coefficients", s = lam1se)

log(lam1se)
```

```
## [1] 5.453875
```

```
bestlambda
```

```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                          1
## (Intercept) -4.590477e+02
## PrivateYes   .
## Accept       1.213291e+00
## Enroll       .
## Top10perc    2.327357e+01
## Top25perc    .
## F.Undergrad  1.868528e-02
## P.Undergrad  .
## Outstate     .
## Room.Board   9.029226e-04
## Books        .
## Personal     .
## PhD          .
## Terminal     .
## S.F.Ratio    .
## perc.alumni  .
## Expend       3.156690e-02
## Grad.Rate    .
```

```
lasso.best <- predict(cv.lasso, type = "coefficients", s = lam1se)
mse.lasso <- predict(cv.lasso, s = 5.539633, newx=testx)
mean_lasso <- mean((mse.lasso-testy)^2)
mean_lasso
```

```
## [1] 1242815
```