Solution 2

STAT6306

Introduction

A major issue with antiretroviral drugs is the mutation of the virus' genes. Because of its high rate of replication (10^9 to 10^{10} virus per person per day) and error-prone polymerase[1], HIV can easily develop mutations that alter susceptibility to antiretroviral drugs. The emergence of resistance to one or more antiretroviral drugs is one of the more common reasons for therapeutic failure in the treatment of HIV.

In the paper 'Genotypic predictors of human immunodeficiency virus type 1 drug resistance'[2], a sample of in vitro[3] HIV viruses were grown and exposed to a particular antiretroviral therapy. The susceptibility of the virus to treatment and the number of genetic mutations of each virus were recorded.

Question 1

```
load("hiv.rda")

X = hiv.train$x
Y = hiv.train$y

geneLabels = colnames(X)
```

(a)

What are n and p in this problem? What are the features in this problem? What are the observations? What is the supervisor? **Note:** Attempt to answer this question before moving on to the rest of the questions.

```
#SOLUTION
(n = nrow(X))
```

```
## [1] 704
```

```
(p = ncol(X))
```

```
## [1] 208
```

SOLUTION

There are 208 features (p) and 704 observations (n). The features are indicators for whether or not there was a mutation in a gene. The supervisor is the log(susceptibility) of the HIV virus to a particular drug therapy

[1] An enzyme that 'stitches' back together DNA or RNA after replication
[2] The entire paper is on the website. Try to see what you can get out of it if you have the time.
[3] Latin for 'in glass', sometimes known colloquially as a test tube

Question 2

Consider the feature matrix $\mathbb{X}$. It is composed of 0's and 1's, with a 1 indicating a mutation in a particular gene. Look at the output for the following chunk of code.

```
table(X)
```

```
## X
##      0      1
## 135589  10843
```

What results do you see? What does this indicate?

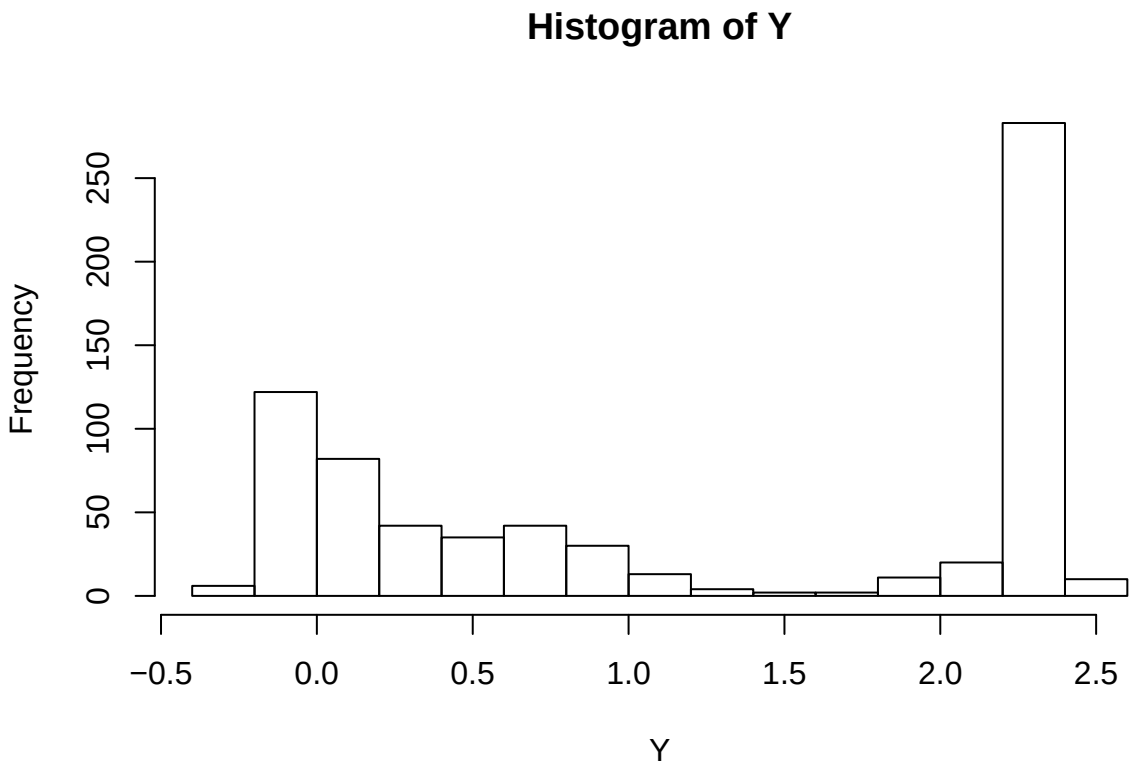
SOLUTION

Based on the feature matrix X, we see that there are 135589 unmutated/"normal" genes and 10843 genes that have mutations.

Question 3

The supervisor is the log transformed susceptibility of a virus to the considered treatment, with large values indicating the virus is relatively more resistant (that is, not susceptible). Run

```
hist(Y)
```



What plot did you just create? What does this indicate?

SOLUTION

This gives us a histogram of the frequency of the susceptibility of a virus to the considered treatment. The marginal distribution of the supervisor Y is bimodal, with a peak around 0 (higher susceptible) and around 2.4 (lower susceptible)

Question 4

We may have (at least) two goals with a data set such as this:

- inference: can we find some genes whose mutation seems to be most related to viral susceptibility?
- prediction: can we make a model that would predict whether this therapy would be efficacious, given a virus with a set of genetic mutations

(a)

Try to find the best subset solution for this problem. Discuss any problems or findings you discover. In particular, how many possible models are there?

SOLUTION

```
2^p
```

```
## [1] 4.113761e+62
```

There are 2^p possible different solutions, which, given the size of p (208), gives us 4.1137614e+62 possible solutions, which is way too large to compute all subsets.

(b) Inference

(i)

Find the selected model for:

- forward selection using BIC as the criterion
- lasso
- refitted/relaxed lasso (note: I added this in, it wasn't in original HW)

```
#Forward selection
#SOLUTION
if(!require(leaps)){install.packages('leaps',repos='http://cran.us.r-project.org');require(leaps)}
```

```
## Loading required package: leaps
```

```
outForward      = regsubsets(x=X,y=Y,nvmax=p,method='forward')
```

```
## Warning in leaps.setup(x, y, wt = weights, nbest = nbest, nvmax = nvmax, :
## 12 linear dependencies found
```

```
## Reordering variables and trying again:
```

```
## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: number of items to
## replace is not a multiple of replacement length
```

```
#  note this warning is that the feature matrix
#  isn't full rank.  This is, there are redundant
#  columns in it:
cat('The rank is: ',qr(X)$rank,' while the # of features is: ',p,'\n')
```

```
## The rank is:  196  while the # of features is:  208
```

```
sumForward      = summary(outForward)
model.forward   = sumForward$which[which.min(sumForward$bic),]
S.forward       = model.forward[-1]#get rid of the intercept entry
lm.forward      = lm(Y~X[,S.forward])#regsubsets only scores models, not fit them
betaHat.forward = coef(lm.forward)
betaHat.forward
```

```
##        (Intercept)  X[, S.forward]p33  X[, S.forward]p54
##         0.07426093         0.09489890         0.60889375
##  X[, S.forward]p58  X[, S.forward]p65  X[, S.forward]p67
##        -0.66464456         0.66361310         0.12187768
##  X[, S.forward]p69  X[, S.forward]p75  X[, S.forward]p90
##         0.09158451         0.08995627         0.15145415
## X[, S.forward]p102 X[, S.forward]p115 X[, S.forward]p117
##         0.07556800         0.29618456        -0.45272186
## X[, S.forward]p151 X[, S.forward]p172 X[, S.forward]p184
##         0.29833631         0.20732278         1.89449998
## X[, S.forward]p187 X[, S.forward]p210 X[, S.forward]p215
##        -0.45850428         0.07844976         0.21895472
```

```
#lasso
if(!require(glmnet)){install.packages('glmnet',repos='http://cran.us.r-project.org');require(glmnet)}
```

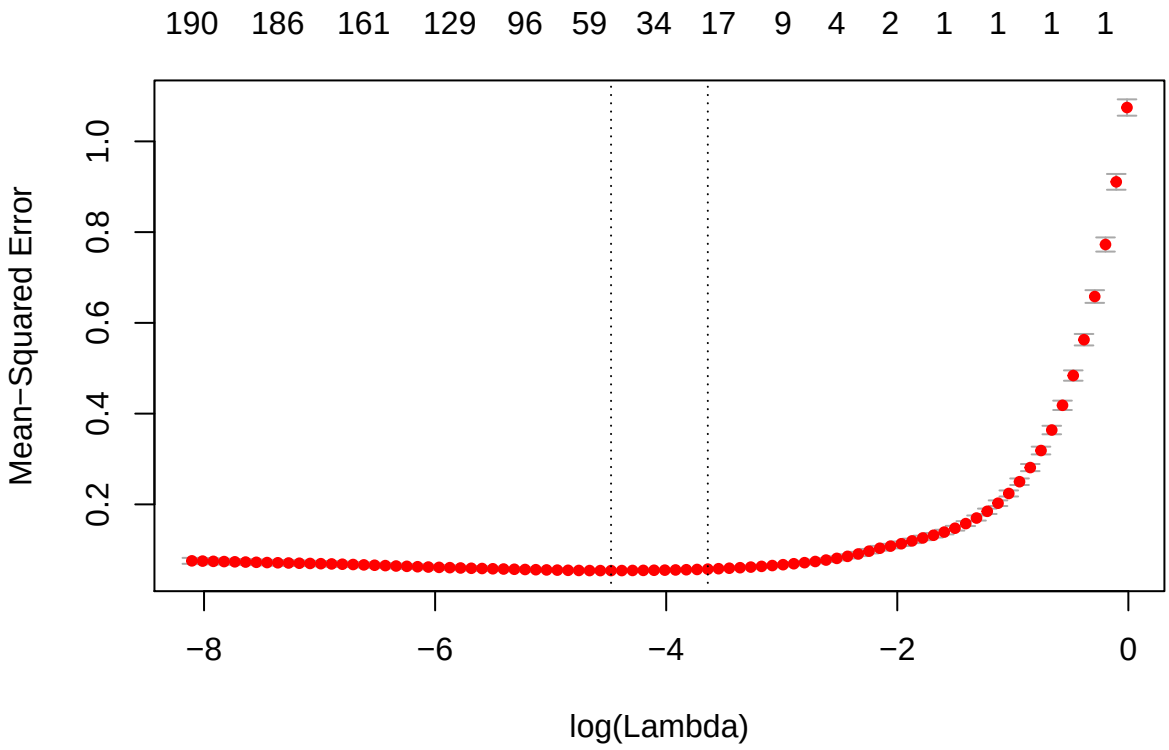
```
## Loading required package: glmnet
```

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

```
## Loading required package: foreach
```

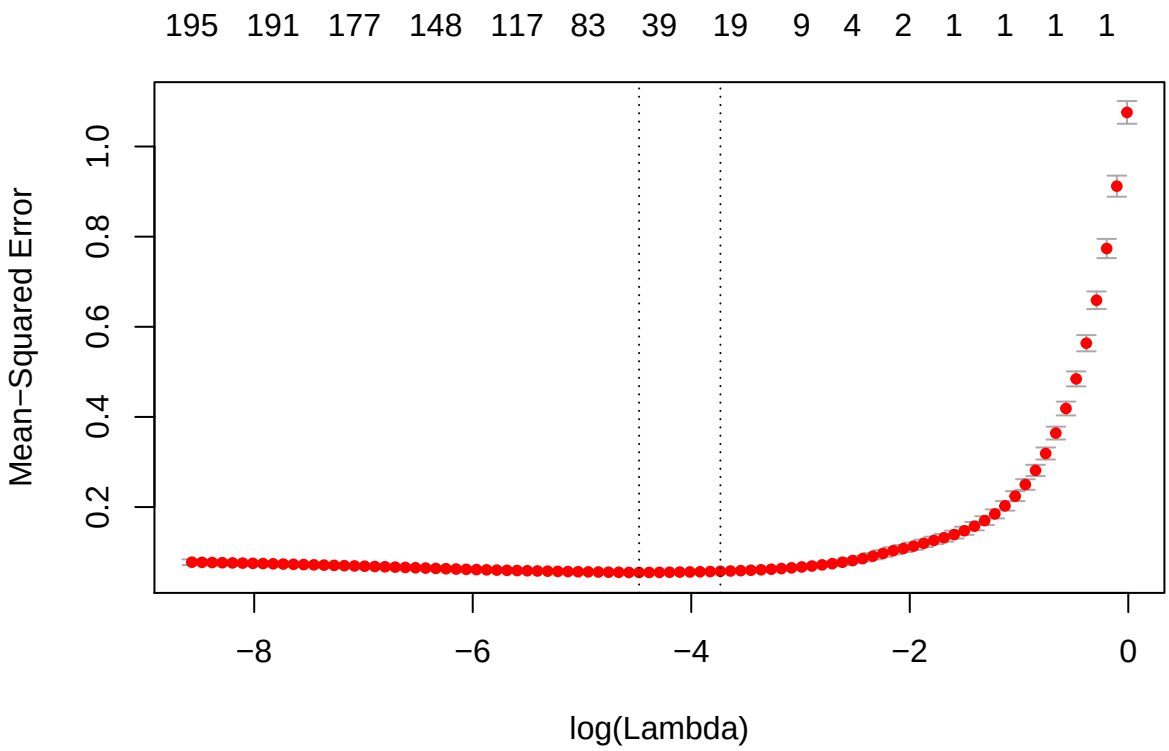
```
## Loaded glmnet 2.0-13
```

```
#SOLUTION
lasso.cv.glmnet = cv.glmnet(X,Y,alpha=1)#note: we are standardizing the features...
plot(lasso.cv.glmnet) #note that this output is random... why?
```




```
betaHat.lasso   = coef(lasso.cv.glmnet,s='lambda.min')[-1]
S.lasso         = which(abs(betaHat.lasso) > 1e-16)
```

```
#refitted lasso
#SOLUTION
lasso.cv.glmnet  = cv.glmnet(X,Y,alpha=1)
plot(lasso.cv.glmnet)
```



```
betaHat.temp     = coef(lasso.cv.glmnet,s='lambda.1se')[-1]
S.refitted       = which(abs(betaHat.temp) > 1e-16)
lm.refitted      = lm(Y ~ X[,S.refitted])
betaHat.refitted = coef(lm.refitted)
```

(ii)

Compare the selected models for each of the above methods. Which genes are selected by all the methods?

```
#SOLUTION
cat('The selected genes from forward selection + BIC are: \n',
     geneLabels[S.forward],'\n')
```

```
## The selected genes from forward selection + BIC are:
##  p33 p54 p58 p65 p67 p69 p75 p90 p102 p115 p117 p151 p172 p184 p187 p210 p215
```

```
cat('The selected genes from lasso are: \n',
     geneLabels[S.lasso],'\n')
```

```
## The selected genes from lasso are:
##  p21 p33 p41 p43 p54 p58 p60 p65 p66 p67 p69 p75 p77 p80 p83 p90 p102 p107 p110 p115 p116 p117 p118 p
```

```
cat('The selected genes from refitted lasso are: \n',
     geneLabels[S.refitted],'\n')
```

```
## The selected genes from refitted lasso are:
##  p33 p41 p43 p65 p67 p69 p75 p77 p90 p115 p116 p118 p151 p181 p184 p190 p210 p215 p219 p228
```

```
#Note that we can directly compare chosen models
geneLabels[S.forward]  %in% geneLabels[S.lasso]
```

```
##  [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [15] TRUE TRUE TRUE
```

```
geneLabels[S.lasso] %in% geneLabels[S.forward]
```

```
##  [1] FALSE  TRUE FALSE FALSE  TRUE  TRUE FALSE  TRUE FALSE  TRUE  TRUE
## [12]  TRUE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE  TRUE FALSE  TRUE
## [23] FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE  TRUE FALSE
## [34] FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE
## [45] FALSE FALSE
```

```
# and find which is in both:
intersect(geneLabels[S.lasso], geneLabels[S.forward])
```

```
##  [1] "p33"  "p54"  "p58"  "p65"  "p67"  "p69"  "p75"  "p90"  "p102" "p115"
## [11] "p117" "p151" "p172" "p184" "p187" "p210" "p215"
```

(iii)

At the genes selected by the lasso, which gene mutation sites are associated with a decrease in viral susceptibility to this particular drug? Hint: Consider the signs of the coefficients. What gene site has the largest estimated effect (positive or negative)?

```
#SOLUTION
## Gene mutation sites related to a decrease in viral susceptibility:
##  thought process: Y increase <-> decrease susceptibility (i.e. increase resistance)
```

```
##                   so \hat{beta}_j > 0 associated with decrease in susceptibility

geneLabels[betaHat.lasso > 1e-16]
```

```
##  [1] "p21"  "p33"  "p41"  "p43"  "p54"  "p65"  "p66"  "p67"  "p69"  "p75"
## [11] "p77"  "p90"  "p102" "p115" "p116" "p118" "p127" "p138" "p139" "p151"
## [21] "p162" "p171" "p172" "p181" "p184" "p188" "p190" "p210" "p211" "p215"
## [31] "p219" "p228"
```

```
geneLabels[which.max(abs(betaHat.lasso))]
```

```
## [1] "p184"
```

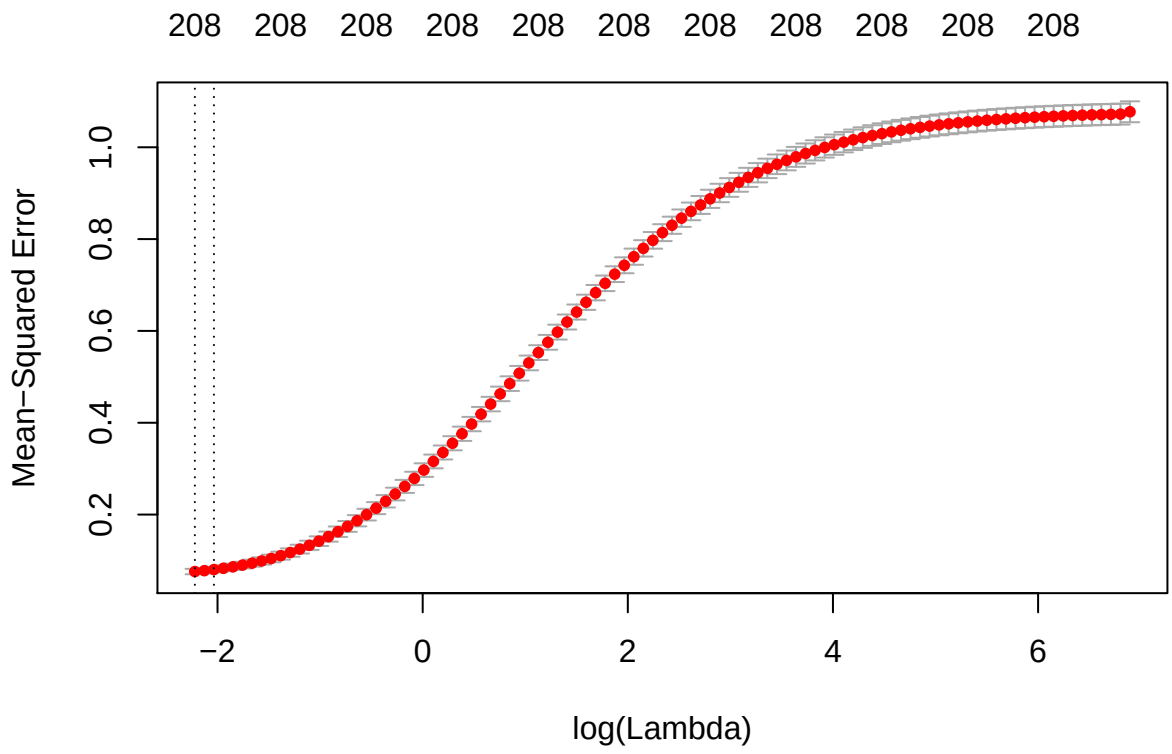
(c) Prediction

(i) Ridge regression

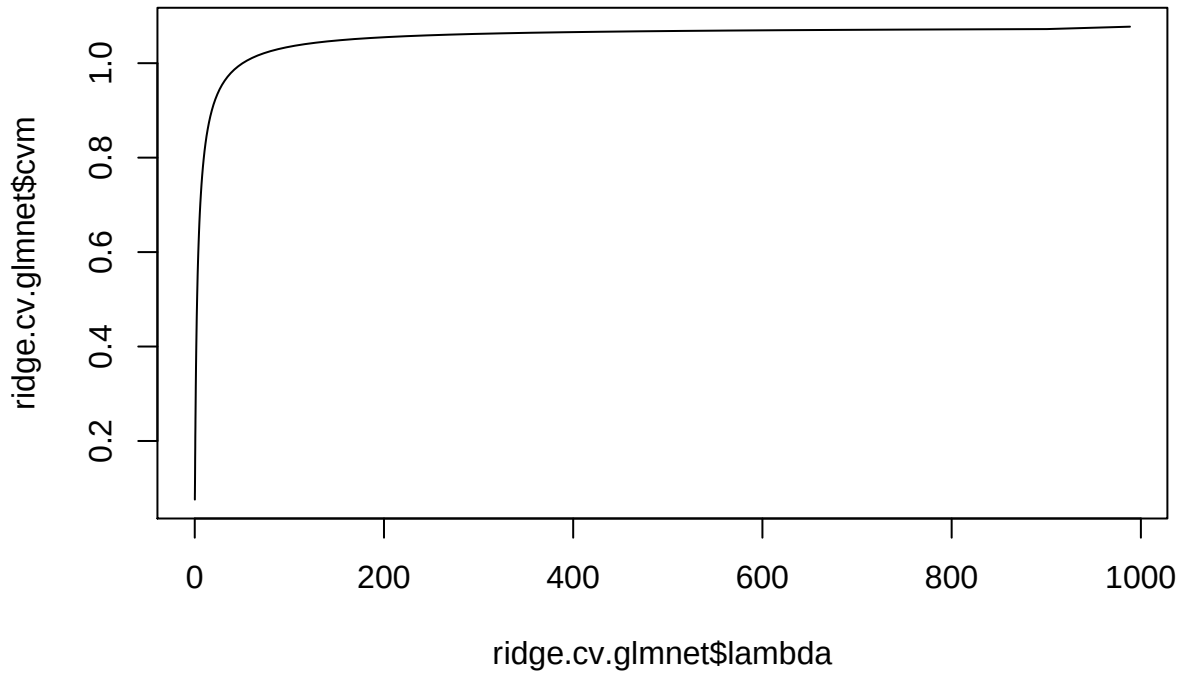
Now that are looking at prediction, we can use ridge regression (which only addresses prediction). Using the package glmnet, plot the CV curve over the grid of λ values and indicate the minimum, and finally report the CV estimate of the prediction risk for $\hat{\beta}_{\text{ridge}}(\hat{\lambda})$

Note: There is no need to report the p coefficient estimates from the ridge solution. Also, glmnet has a grid problem. Make two plots, one that shows the problem and one that shows it being corrected.

```
#SOLUTION
ridge.cv.glmnet = cv.glmnet(X,Y,alpha=0)
plot(ridge.cv.glmnet)
```



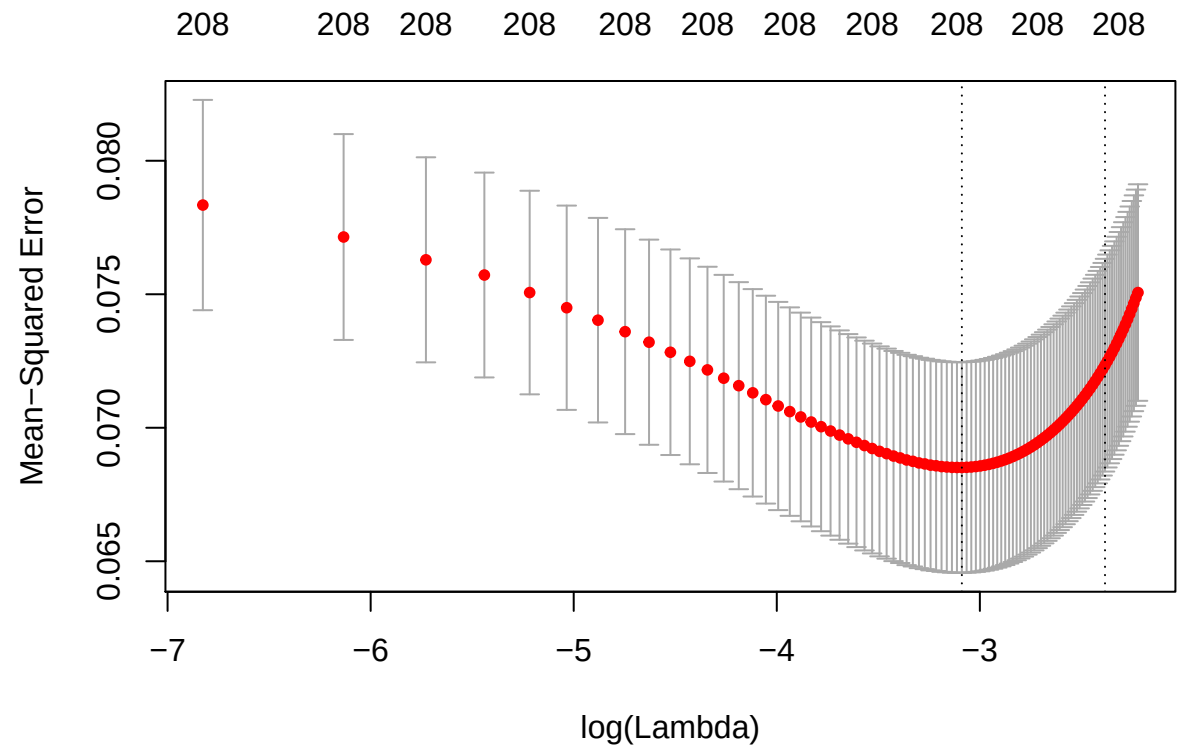

```
#or
plot(ridge.cv.glmnet$lambda,ridge.cv.glmnet$cvm,type='l')
```




```
#CV estimate of the prediction error:
min(ridge.cv.glmnet$cvm)
```

```
## [1] 0.07590684
```

```
min.lambda      = min(ridge.cv.glmnet$lambda)
lambda.new      = seq(min.lambda, min.lambda*0.01,length=100)
ridge.cv.glmnet = cv.glmnet(x = X, y = Y, alpha = 0,lambda = lambda.new)
plot(ridge.cv.glmnet) #now it is in middle
```



(ii) Prediction on a test set

Now, let's look at some predictions made by these methods. Use the following for the test set:

```
X_0 = hiv.test$x
Y_0 = hiv.test$y
```

Find an estimate of the risk using the test observations for

- forward selection using Cp as the criterion
- ridge
- lasso
- refitted lasso (note this wasn't in the original hw)

```
#SOLUTION
#### Get predictions on test set:
Yhat.test.forward  = X_0[,S.forward] %*% betaHat.forward[-1] + betaHat.forward[1]
Yhat.test.ridge    = predict(ridge.cv.glmnet,X_0,s='lambda.min')
Yhat.test.lasso    = predict(lasso.cv.glmnet,X_0,s='lambda.min')
Yhat.test.refitted = X_0[,S.refitted] %*% betaHat.refitted[-1] + betaHat.refitted[1]

# Get estimate of prediction risk via the test set error
Yhat.test.forward   = mean((Yhat.test.forward - Y_0)**2)
pred.error.ridge    = mean((Yhat.test.ridge - Y_0)**2)
pred.error.lasso    = mean((Yhat.test.lasso - Y_0)**2)
pred.error.refitted = mean((Yhat.test.refitted - Y_0)**2)

cat('The prediction error from forward selection + BIC are: \n',
     Yhat.test.forward,'\n')
```

```
## The prediction error from forward selection + BIC are:
##  0.07491952
```

```
cat('The prediction error from ridge is: \n',
     pred.error.ridge,'\n')
```

```
## The prediction error from ridge is:
##  0.09721523
```

```
cat('The prediction error from lasso is: \n',
     pred.error.lasso,'\n')
```

```
## The prediction error from lasso is:
##  0.06878929
```

```
cat('The prediction error from refitted lasso is: \n',
     pred.error.refitted,'\n')
```

```
## The prediction error from refitted lasso is:
##  0.06709472
```

(d)

Challenge Suppose we didn't have access to any test data. How could you provide an estimate of the risk? What are the pros and cons of your proposal?

```
#SOLUTION
cat('The CV estimate of risk of ridge(lambdaHat) = ',min(ridge.cv.glmnet$cvm),'\n')
```

```
## The CV estimate of risk of ridge(lambdaHat) =  0.06851163
```

Compare this with the test set estimate:

```
cat('The test set estimate of risk from ridge is: \n',
     pred.error.ridge,'\n')
```

```
## The test set estimate of risk from ridge is:
##  0.09721523
```

So, by minimizing CV as a function of lambda, we have produced a reasonable estimate of the risk.

Question 5

Building on the gradient descent code from lecture, implement (batch or stochastic) gradient descent to find the ridge regression solution (via the Lagrangian formulation) for the HIV data at lambda = 1. Compare your solution to the solution found by using the svd and verify that they are (approximately) equal.

Note that the form for the gradients we derived in class need to be augmented by the derivative of the penalty term.

```
##############################
##############################
##############################
lam = 1

bHatInitial   = rnorm(p)
bHat          = bHatInitial
learnRate     = .1

miniBatchParm = n/10
threshold     = .05/2

fF = function(X,b){
  return( X %*% b)
}
ellF = function(Y,f){
  return((Y - f)**2)
}
rF = function(ell){
  return( mean( ell ))
}

maxSweep  = 100
sweep     = 0
bHatSweep = matrix(0,nrow=maxSweep,ncol=p)
grad_j    = rep(2*threshold,p)

while(sqrt(sum(grad_j**2)) > threshold & sweep < maxSweep){
  sweep = sweep + 1
  batch = sample(1:n,miniBatchParm,replace=FALSE)
  #Step 1b: Iterate from 1 up to p
  for(j in 1:p){
    #Step 2a:Forward
    f   = fF(X[batch,],bHat)
```

```
    ell = ellF(Y[batch],f)
    r   = rF(ell)
    #Step 2b:Backward
    dell_df = -2*(Y[batch] - f)
    df_dbj  = X[batch,j]
    dR_dbj   = rF( dell_df*df_dbj ) + 2*lam*bHat[j]
    #Step 3: Update
    bHat[j]  = bHat[j] - learnRate * dR_dbj
    grad_j[j] = dR_dbj
  }
  bHatSweep[sweep,] = bHat
}
cat('total number or sweeps: ',sweep,'. Max number of sweeps is: ',maxSweep,'\n')
```

```
## total number or sweeps:  100 . Max number of sweeps is:  100
```

Let's compare the solutions:

```
library(glmnet)
ridge.glmnet = glmnet(x=X,y=Y,alpha=0,
                      standardize=FALSE,intercept=FALSE,
                      lambda = seq(lam*1.01,lam*.99,length.out = 1000))

rbind(coef(ridge.glmnet,s=lam)[-1][1:10],bHat[1:10])
```

```
##            [,1]       [,2]       [,3]       [,4]       [,5]       [,6]
## [1,] 0.01179384 0.01112260 0.01467873 0.01298910 0.01160466 0.01667763
## [2,] 0.01053424 0.01061902 0.01220603 0.01370475 0.01155801 0.01732104
##             [,7]        [,8]        [,9]       [,10]
## [1,] 0.010455649 0.009026895 0.011772387 0.009706876
## [2,] 0.009505938 0.011431345 0.009862109 0.008949871
```

At these settings, the solutions are quite close. Note that in the above implementations, both lambda sequences are implicitly divided by n as we are using the training error as the objective (instead of the RSS). If we want to compare to an RSS based solution (e.g. the SVD solution in the lecture notes) we would need to multiply the lambda by n. Here is what I mean by this:

```
svd.out = svd(X)
ridge.svd1 = svd.out$v %*%diag(svd.out$d/(svd.out$d**2 + lam))%*%t(svd.out$u) %*% Y

ridge.svd2 = svd.out$v %*%diag(svd.out$d/(svd.out$d**2 + lam*n))%*%t(svd.out$u) %*% Y

rbind(coef(ridge.glmnet,s=lam)[-1][1:10],bHat[1:10],ridge.svd1[1:10],ridge.svd2[1:10])
```

```
##             [,1]        [,2]        [,3]        [,4]        [,5]
## [1,]  0.01179384  0.01112260  0.01467873  0.01298910  0.01160466
## [2,]  0.01053424  0.01061902  0.01220603  0.01370475  0.01155801
## [3,] -0.08837528 -0.05424487 -0.01395821 -0.11271949 -0.04723696
## [4,]  0.01194714  0.01123536  0.01467547  0.01312337  0.01169851
##            [,6]        [,7]        [,8]         [,9]        [,10]
## [1,] 0.01667763 0.010455649 0.009026895  0.011772387  0.009706876
## [2,] 0.01732104 0.009505938 0.011431345  0.009862109  0.008949871
## [3,] 0.04156085 0.029231431 0.022403921 -0.095406042 -0.048909202
## [4,] 0.01688665 0.010592060 0.009184025  0.011850870  0.009818869
```

Additional challenge problems:

I don't want to overwhelm you with homework problems. However, there are additional topics that are relevant for an interested student. You don't need to do these/turn them in.

LARS vs. forward selection

The LARS algorithm is quite similar to forward selection. Run LARS using the option forward.stagewise and compare it to forward selection using Mallow's Cp.

GIC-based tuning parameter selection

Try and use a GIC-based method instead of K-fold CV for finding $\hat{\lambda}$ for the lasso using the HIV data.