**LECTURE 6 - BREAKOUTS**

**Breakout 1:**

Using the grid of lambdas, x and y (from the Hitters dataset) already created:

1- create a sample set, 2/3 for train, 1/3 for test

2- Build a ridge model with 5-fold cross validation using the training samples

3- Decide which is the best lambda

4- Get the MSE for the training set and testing set using the best lambda

5- List the coefficiencts for the best lambda

grid=10^seq(10,-2,length=100) #10 to -2, 100 elements

train=sample(1:nrow(x), 2/3\*nrow(x))

test=(-train)

cv.out=cv.glmnet(x[train,],y[train],alpha=0, nfolds=5)

bestlam=cv.out$lambda.min

bestlam

ridge.pred=predict(ridge.mod,s=bestlam,newx=x[train,])

mean((ridge.pred-y[train])^2) # Calculate its MSE

ridge.pred=predict(ridge.mod,s=bestlam,newx=x[test,])

mean((ridge.pred-y[test])^2) # Calculate its MSE

c3=predict(cv.out,type="coefficients",s=bestlam)

c1=coef(cv.out, s = bestlam)

ridge.mod=glmnet(x[train,], y[train],alpha=0,lambda=bestlam)

c2=coef(ridge.mod)

cbind(c1,c2, c3)

**Breakout 2:**

Using the Hitters dataset (with NAs omitted):

1- create a sample set, 2/3 for train, 1/3 for test

2- Fit a PCR model using the training data

3- Decide the best number of components using a validation plot (or other method if you like)

4- Find the MSE on the test set for the best

5- Now build a loop to collect the MSE on the test set for each possible number of components (1:19) and plot this.

6- Does the min choice still look right?

train=sample(1:nrow(Hitters), 2/3\*nrow(Hitters))

test=(-train)

pcr.fit=pcr(Salary~., data=Hitters,subset=train,scale=TRUE,

validation="CV")

validationplot(pcr.fit,val.type="MSEP")

pcr.pred=predict(pcr.fit,Hitters[test,],ncomp=6)

pcr.pred # Take a look at the predicted values

mean((pcr.pred-Hitters$Salary[test])^2) # Compute the MSE

testMSEs=NULL

for(i in 1:19) {

pcr.pred=predict(pcr.fit,Hitters[test,],ncomp=i)

mse=mean((pcr.pred-Hitters$Salary[test])^2) # Compute the MSE

testMSEs=c(testMSEs,mse)

}

plot(testMSEs, type="l")