final.R

User

2020-06-26

library(MASS)  
library(glmnet)

## Loading required package: Matrix

## Loaded glmnet 4.0-2

library(assist)

## Loading required package: nlme

## Loading required package: lattice

##   
## Attaching package: 'assist'

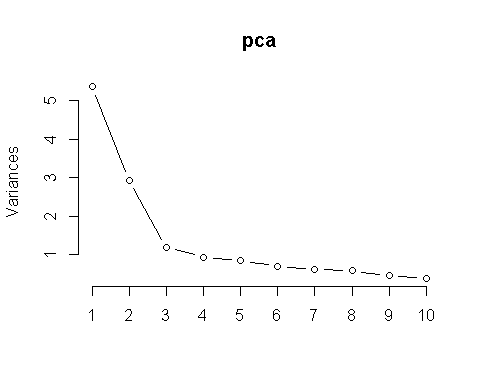
## The following object is masked from 'package:Matrix':  
##   
## bdiag

library(pls)

##   
## Attaching package: 'pls'

## The following object is masked from 'package:stats':  
##   
## loadings

setwd("C:\\Users\\User\\Desktop\\School\\Math\_537\\Final")  
  
data=read.csv("College.csv")  
  
data$exclu=100\*(data$Apps-data$Accept)/data$Apps+100\*(data$Enroll/data$Accept)  
Private=ifelse(data$Private=="Yes", 1,0)  
dataS=scale(cbind(Private,data[,7:dim(data)[2]-1]),center=T,scale=T)  
y=scale(data[,dim(data)[2]],center=T,scale=T)  
  
modelRes=matrix(0,30,1000)  
  
for(k in 1:1000)  
{  
 j=5  
 set.seed(k)  
 folds=cut(seq(1,nrow(data)),breaks=j,labels=F)  
 testIndexes=matrix(0,length(folds[folds==1]),j)  
 for (i in 1:j)  
 {  
 if(length(folds[folds==i])==156)  
 {  
 testIndexes[,i]=which(folds==i,arr.ind=T)  
 }  
 else  
 {  
 testIndexes[,i]=c(which(folds==i,arr.ind=T),0)  
 }  
 }  
   
 pca=prcomp(dataS)  
   
 plot(pca,type="l")  
   
 #Not a big gain for any addition components after 3, will pick 3.  
   
 i=1  
   
   
 for(i in 1:j)  
 {  
   
 testDatax=dataS[testIndexes[,i],]  
 trainDatax=dataS[-testIndexes[,i],]  
   
 testDatay=y[testIndexes[,i]]  
 trainDatay=y[-testIndexes[,i]]  
   
 trainData=data.frame(cbind(trainDatay,trainDatax))  
 testData=data.frame(cbind(testDatay,testDatax))  
   
 ols <- lm(trainData$trainDatay~ ., data=trainData)#trainData[,2:dim(trainData)[2]], data=trainData)  
 pred=predict.glm(ols,newdata=data.frame(testDatax),type="response")  
 modelRes[1,k]=modelRes[1,k]+sum((testDatay-pred)^2)  
   
 cvLam=cv.glmnet(data.matrix(trainDatax),y=trainDatay,alpha=0,nfolds=j)$lambda.min  
 ridge=glmnet(trainDatax,trainDatay,alpha=0,lambda=cvLam)  
 pred=predict(ridge,newx=data.matrix(testDatax),s=cvLam)  
 modelRes[2,k]=modelRes[2,k]+sum((testDatay-pred)^2)  
   
 cvLam=cv.glmnet(data.matrix(trainDatax),y=trainDatay,alpha=1,nfolds=j)$lambda.min  
 lasso=glmnet(data.matrix(trainDatax),trainDatay,alpha=1,lambda=cvLam)  
 pred=predict(lasso,newx=data.matrix(testDatax),s=cvLam)  
 modelRes[3,k]=modelRes[3,k]+sum((testDatay-pred)^2)  
   
 cvLam=cv.glmnet(data.matrix(trainDatax),y=trainDatay,alpha=.5,nfolds=j)$lambda.min  
 elast=glmnet(trainDatax,trainDatay,alpha=.5,lambda=cvLam)  
 pred=predict(elast,newx=data.matrix(testDatax),s=cvLam)  
 modelRes[4,k]=modelRes[4,k]+sum((testDatay-pred)^2)  
   
 for(j in 1:13)  
 {  
 mPCR=pcr(trainDatay~trainDatax ,ncomp=j)  
 pred=predict(mPCR,testDatax,ncomp=j)  
 modelRes[5+j-1,k]=modelRes[5+j-1,k]+sum((testDatay-pred)^2)  
   
 mPLSR = plsr(trainDatay~trainDatax,ncomp=j)  
 pred=predict(mPLSR,testDatax,ncomp=j)  
 modelRes[6+13+j-2,k]=modelRes[6+13+j-2,k]+sum((testDatay-pred)^2)  
 }  
   
 }  
 if(k%%100==0)  
 {  
 print(k)  
 }  
  
}



## [1] 100

## [1] 200

## [1] 300

## [1] 400

## [1] 500

## [1] 600

## [1] 700

## [1] 800

## [1] 900

## [1] 1000

name=c('LM','Ridge','Lasso','Elastic')  
for(i in 1:13)  
{  
 name=c(name,paste0("PCR: Lambda=",i))  
}  
for(i in 1:13)  
{  
 name=c(name,paste0("PLSR: Lambda=",i))  
}  
rownames(modelRes)=as.vector(name)  
  
winningModel=apply(modelRes,2,which.min)  
winningModel[winningModel==2]="Ridge"  
winningModel[winningModel==3]="Lasso"  
winningModel[winningModel==4]="Elastic"  
  
output=as.data.frame(table(winningModel))  
output=output[order(-output$Freq),]  
output

## winningModel Freq  
## 3 Ridge 566  
## 1 Elastic 275  
## 2 Lasso 159