homework3

zhuzhengda

2015年12月24日

# cyclical coordinate descent for lasso

The algorithm I use is coordinate descent for lasso (aka shooting algorithm). And the result shows that I got almost the same result compared with the coefficients calculated by glmnet.

library(glmnet)

## Warning: package 'glmnet' was built under R version 3.2.3

## Loading required package: Matrix  
## Loading required package: foreach

## Warning: package 'foreach' was built under R version 3.2.3

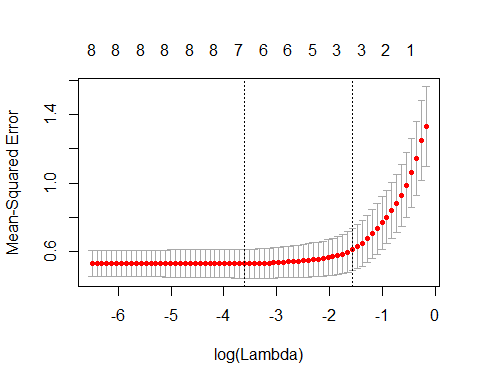
## Loaded glmnet 2.0-2

library(lars)

## Warning: package 'lars' was built under R version 3.2.3

## Loaded lars 1.2

## read in data  
 data<-read.table("data.txt",sep ='\t',header=TRUE)  
 x<-as.matrix(data[,1:8])  
 y<-as.matrix(data[,9])  
   
 ## R package glmnet for lasso  
 gla <- cv.glmnet(x, y, nfolds = 10)  
 plot(gla)



la<-gla$lambda.1se  
 para<-coef(gla$glmnet.fit, s = gla$lambda.1se)  
   
 ## my algorithm   
 x0<-scale(x) #Data standardization  
 sigma<-attr(x0,"scaled:scale")  
 muy<-mean(y)  
 y0<-y-muy  
 active<-array(1,c(1,8)) #predictors for variables  
 b<-array(0,c(8,1))  
 weight<-dim(x)[1]  
 threshold<-la #here I use the optimal lambda from previous results  
 while(TRUE){  
 bold<-b  
 r<-y0-x0[,which(active==1)]%\*%b[which(active==1)]  
 for(j in which(active==1)){ #Regress j-th partial residuals on j-th predictor  
 bjold<-b[j]  
 rj<-r+b[j]\*x0[,j]  
 bj<-(t(x0[,j])%\*%rj)/weight  
 b[j]<-sign(bj)\*max((abs(bj)-threshold),0) #Soft thresholding  
 if(b[j]==0)  
 active[j]<-0  
 r<-r-x0[,j]\*(b[j]-bjold)  
 }  
 if(max(abs(b-bold))<0.0001) #threshold test  
 break  
 }  
 b<-b/sigma  
 print("the result of glmnet:")

## [1] "the result of glmnet:"

print(para)

## 9 x 1 sparse Matrix of class "dgCMatrix"  
## 1  
## (Intercept) 0.7820637  
## lcavol 0.4485189  
## lweight 0.2804033  
## age .   
## lbph .   
## svi 0.3383490  
## lcp .   
## gleason .   
## pgg45 .

print("the result of my algorithm :")

## [1] "the result of my algorithm :"

print(b)

## [,1]  
## [1,] 0.4425012  
## [2,] 0.2791084  
## [3,] 0.0000000  
## [4,] 0.0000000  
## [5,] 0.3415953  
## [6,] 0.0000000  
## [7,] 0.0000000  
## [8,] 0.0000000