SDSC3006 Lab
6-Shrinkage method

Langming LIU langmiliu2-c@my.cityu.edu.hk

School of Data Science
City University of Hong Kong

Contents

- Ridge Regression
- LASSO Regression

Ridge Regression

Introduction

- Least squares: Minimize RSS

$$\mathrm{RSS} = \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 .$$

- Ridge Regression: Minimize RSS+Penalty(l2)

$$\sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 = \mathrm{RSS} + \lambda \sum_{j=1}^{p} \beta_j^2,$$

- Reason of shrinkage penalty: consider not only model fitting, but also shrinking the estimates of coefficients.(Trade-off)
- Tuning parameter λ: control the relative impact of two terms.

Initialize Data

- Hitters data set: predict a baseball player’s Salary based on various statistics associated with performance in the previous year.

```
library(ISLR)
names(Hitters)
dim(Hitters)
sum(is.na(Hitters$Salary))
Hitters=na.omit(Hitters)
dim(Hitters)
sum(is.na(Hitters))
```

Fitting using Ridge

```
##the function glmnet() in the glmnet package
install.packages("glmnet")
library(glmnet)

#create dataset for fitting
x=model.matrix(Salary~.,Hitters)[,-1]      #x: 19 predictors
y=Hitters$Salary
#why remove the 1st column?

##consider a vector of lambda values ranging from 10^10 to 10^-2
grid=10^seq(10,-2,length=100)    #length: points of grid
ridge.mod=glmnet(x,y,alpha=0,lambda=grid)
#alpha=0:Ridge       alpha=1: LASSO
dim(coef(ridge.mod))       #20*100
```

Results

```
##the 50th value of lambda
ridge.mod$lambda[50]
coef(ridge.mod)[,50]
##Norm of the estimates
sqrt(sum(coef(ridge.mod)[-1,50]^2))

##the 60th value of lambda
ridge.mod$lambda[60]
coef(ridge.mod)[,60]
sqrt(sum(coef(ridge.mod)[-1,60]^2))

##For new value of lambda
##for example, lambda=25
predict(ridge.mod,s=25,type="coefficients")[1:20,]
```

Cross validation for Ridge

```
##split the data into a training and a test set
set.seed(1)
train=sample(1:nrow(x), nrow(x)/2)
test=(-train)
y.test=y[test]
##fit ridge regression on training data
grid=10^seq(10,-2,length=100)
ridge.mod=glmnet(x[train,],y[train],alpha=0,lambda=grid)
##predict on test set using lamda=4, 1e10, 0
ridge.pred1=predict(ridge.mod,s=4,newx=x[test,])
mean((ridge.pred1-y.test)^2)      #test MSE
ridge.pred2=predict(ridge.mod,s=1e10,newx=x[test,])
mean((ridge.pred2-y.test)^2)
ridge.pred3=predict(ridge.mod,s=0,newx=x[test,])
mean((ridge.pred3-y.test)^2)
```

Select Tuning Parameter

```
##cross validation to get the best lambda
set.seed(1)
cv.out=cv.glmnet(x[train,],y[train],alpha=0)      #default is 10-folds CV
plot(cv.out)
bestlam=cv.out$lambda.min
bestlam

##now predict with the best lambda
ridge.pred=predict(ridge.mod,s=bestlam,newx=x[test,])
mean((ridge.pred-y.test)^2)

##refit ridge regression on the full dataset
out=glmnet(x,y,alpha=0)
predict(out,type="coefficients",s=bestlam)[1:20,]
```

LASSO Regression

Introduction

- Least squares: Minimize RSS

$$\mathrm{RSS} = \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 .$$

- LASSO Regression: Minimize RSS+Penalty(l1)

$$\sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j| = \mathrm{RSS} + \lambda \sum_{j=1}^{p} |\beta_j|.$$

- Advantage: can perform feature(variable) selection.
- Difference: Ridge reduces the coefficients by same proportion, while LASSO shrinks the coefficients by similar amount(some coefficients go to 0 if very small).

LASSO implemention

```
##for LASSO we use alpha=1
lasso.mod=glmnet(x[train,],y[train],alpha=1,lambda=grid)
plot(lasso.mod)

##use CV to find the optimal lambda
set.seed(1)
cv.out=cv.glmnet(x[train,],y[train],alpha=1)
plot(cv.out)
bestlam=cv.out$lambda.min
```

LASSO implemention

```
##use best lambda for prediction
lasso.pred=predict(lasso.mod,s=bestlam,newx=x[test,])
mean((lasso.pred-y.test)^2)
out=glmnet(x,y,alpha=1,lambda=grid)
lasso.coef=predict(out,type="coefficients",s=bestlam)[1:20,]

##check the estimated coefficients and the ‘0’ coefficients
lasso.coef
lasso.coef[lasso.coef!=0]
```