OLS, Ridge, Lasso, and PCR of College Applications

Greg Johnson

8/14/2017

Our data are acceptance and enrollment counts for 777 colleges and universities. We want to predict acceptances from the other variables. Because there are so many variables, we will entertain regularization/penalization methods.

First we designate a training set and a test set.

```
set.seed(1)
n<-nrow(college)
ntrain<-round(.80*n) #80-20 split
index<-sample(1:n,ntrain,replace=FALSE)
college_train<-college[index,]
college_test<-college[-index,]
```

Linear Model

```
fit1 <- lm(paste("Apps ~", (paste(names(college_train)[-c(1, 3)], collapse = " + "))),
    data = college_train)
y <- college_test[["Apps"]]
yhat <- predict(fit1, newdata = college_test[, -c(1, 3)])
MSE_lm <- mean((y - yhat)^2)
```

Ridge Regression

```
grid <- 10^seq(10, -2, length = 100)  #range of possible lambdas; use cv to select an optimal lambda

nfold <- 10
CVE <- numeric(length(grid))  #cross-validation error for multiple lambdas
for (l in 1:length(grid)) {

    # 10-fold CV
    folds_i <- sample(rep(1:nfold, length.out = ntrain))
    MSE <- numeric(nfold)
    for (k in 1:nfold) {
        test_index <- which(folds_i == k)
        train_fold <- college_train[-test_index, ]
        test_fold <- college_train[test_index, ]

        ridge.mod <- glmnet(as.matrix(train_fold[, c(2, 4:19)]), as.numeric(train_fold[,
            3]), alpha = 0, lambda = grid[l], thresh = 1e-12)
        yhat <- predict(ridge.mod, newx = as.matrix(test_fold[, c(2, 4:19)]))
        y <- test_fold[, 3]
        MSE[k] <- mean((y - yhat)^2)
    }
    CVE[l] <- mean(MSE)
}
```

```
# our cross-validated lambda is:
(lambda <- grid[which.min(CVE)])
```

```
## [1] 0.09326033
```

```
ridge.mod <- glmnet(as.matrix(college_train[, c(2, 4:19)]), as.numeric(college_train[,
    3]), alpha = 0, lambda = lambda, thresh = 1e-12)
yhat <- predict(ridge.mod, newx = as.matrix(college_test[, c(2, 4:19)]))
y <- college_test[, 3]
MSE_ridge <- mean((y - yhat)^2)
```

Lasso

```
nfold <- 10
CVE <- numeric(length(grid))  #cross-validation error for multiple lambdas
for (l in 1:length(grid)) {

    # 10-fold CV
    folds_i <- sample(rep(1:nfold, length.out = ntrain))
    MSE <- numeric(nfold)
    for (k in 1:nfold) {
        test_index <- which(folds_i == k)
        train_fold <- college_train[-test_index, ]
        test_fold <- college_train[test_index, ]

        lasso.mod <- glmnet(as.matrix(train_fold[, c(2, 4:19)]), as.numeric(train_fold[,
            3]), alpha = 1, lambda = grid[l], thresh = 1e-12)
        yhat <- predict(lasso.mod, newx = as.matrix(test_fold[, c(2, 4:19)]))
        y <- test_fold[, 3]
        MSE[k] <- mean((y - yhat)^2)
    }
    CVE[l] <- mean(MSE)
}

# our cross-validated lambda is:
(lambda <- grid[which.min(CVE)])
```

```
## [1] 0.1232847
```

```
lasso.mod <- glmnet(as.matrix(college_train[, c(2, 4:19)]), as.numeric(college_train[,
    3]), alpha = 1, lambda = lambda, thresh = 1e-12)
yhat <- predict(lasso.mod, newx = as.matrix(college_test[, c(2, 4:19)]))
y <- college_test[, 3]
MSE_lasso <- mean((y - yhat)^2)
```

Principal Component Regression

```
require(pls)
```

```
## Loading required package: pls
```

```
##
```

```
## Attaching package: 'pls'
```

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

```
grid<-1:16 #range of M
nfold<-10 #number of folds

CVE<-numeric(length(grid)) #cross-validation error for multiple M's
for(l in 1:length(grid)){

  #10-fold CV
  set.seed(1)
  folds_i<-sample(rep(1:nfold,length.out=ntrain))
  MSE<-numeric(nfold)
  for(k in 1:nfold){
    test_index<-which(folds_i==k)
    train_fold<-college_train[-test_index,]
    test_fold<-college_train[test_index,]

    pcr.mod<-pcr(formula(paste("Apps~",(paste(names(college_train)[-c(1,3)],collapse=" + ")))),ncomp=gri
    yhat<-predict(pcr.mod,ncomp=grid[l],newdata=as.matrix(test_fold[,c(2,4:19)]))
    y<-test_fold[,3]
    MSE[k]<-mean((y-yhat)^2)
  }
  CVE[l]<-mean(MSE)
}
```

```
M<-grid[which.min(CVE)]

#fit principal components regression
PCRfit<-pcr(formula(paste("Apps~",(paste(names(college_train)[-c(1,3)],collapse=" + ")))),ncomp=M,data=
PCRfit[["loadings"]]
```

```
##
## Loadings:
##             Comp 1 Comp 2 Comp 3 Comp 4 Comp 5 Comp 6 Comp 7 Comp 8 Comp 9
## Private      0.216 -0.306  0.158         0.165         0.184        -0.216
## Accept             0.414                0.377                       0.186
## Enroll             0.436                0.294 -0.100        -0.114  0.106
## Top10perc    0.341  0.147         0.382                        0.333 -0.196
## Top25perc    0.313  0.173 -0.128  0.400                        0.363 -0.278
## F.Undergrad        0.441                0.231
## P.Undergrad -0.142  0.302  0.160 -0.275        -0.114        -0.105 -0.781
## Outstate     0.376         0.113 -0.208  0.136
## Room.Board   0.274         0.279 -0.504  0.182  0.165  0.207  0.292
## Books                     0.608  0.309         0.650 -0.246 -0.162
## Personal    -0.137  0.168  0.435  0.250 -0.371 -0.277  0.646 -0.108  0.131
## PhD          0.240  0.267 -0.163 -0.207 -0.454  0.138                0.139
## Terminal     0.246  0.255        -0.266 -0.441  0.187        -0.130  0.134
```

```
## S.F.Ratio   -0.267  0.133 -0.310                0.485  0.231  0.167 -0.116
## perc.alumni  0.291        -0.242  0.175                      -0.733 -0.237
## Expend       0.337         0.228                -0.305 -0.284         0.183
## Grad.Rate    0.299        -0.188  0.101  0.295  0.214  0.542         0.140
##             Comp 10 Comp 11 Comp 12 Comp 13 Comp 14 Comp 15 Comp 16
## Private      0.226  -0.702   0.110  -0.407
## Accept       0.107  -0.179           0.113  -0.128  -0.662   0.330
## Enroll               -0.115          -0.104           0.308  -0.115
## Top10perc                                             0.292   0.671
## Top25perc                    -0.193   0.141          -0.317  -0.552
## F.Undergrad                          -0.115   0.110   0.432  -0.255
## P.Undergrad -0.311            0.199
## Outstate     0.182  -0.119   0.250   0.779           0.219
## Room.Board   0.226   0.363  -0.427  -0.189
## Books
## Personal     0.181
## PhD                 -0.187                  -0.701
## Terminal            -0.198                   0.677           0.130
## S.F.Ratio    0.503   0.119   0.462
## perc.alumni  0.274   0.310  -0.183
## Expend               0.323   0.603  -0.310          -0.149  -0.160
## Grad.Rate   -0.592   0.108   0.206  -0.102
##
##                Comp 1 Comp 2 Comp 3 Comp 4 Comp 5 Comp 6 Comp 7 Comp 8
## SS loadings     1.000  1.000  1.000  1.000  1.000  1.000  1.000  1.000
## Proportion Var  0.059  0.059  0.059  0.059  0.059  0.059  0.059  0.059
## Cumulative Var  0.059  0.118  0.176  0.235  0.294  0.353  0.412  0.471
##                Comp 9 Comp 10 Comp 11 Comp 12 Comp 13 Comp 14 Comp 15
## SS loadings     1.000   1.000   1.000   1.000   1.000   1.000   1.000
## Proportion Var  0.059   0.059   0.059   0.059   0.059   0.059   0.059
## Cumulative Var  0.529   0.588   0.647   0.706   0.765   0.824   0.882
##                Comp 16
## SS loadings      1.000
## Proportion Var   0.059
## Cumulative Var   0.941
```

```
#estimate test error with test set
yhat<-predict(PCRfit,newdata=as.matrix(college_test[,c(2,4:19)]))
y<-college_test[,3]
MSEpcr<-mean((y-yhat)^2)
MSEpcr
```

```
## [1] 2442098
```

Comparison of Approaches

Compared to the scale of our response variable, our model MSE's are gigantic:

Model	MSE
Linear Model	1082005
Ridge	1081995
Lasso	1108837
PCR	2442098

There isn't much variation in performance with the exception of PCR. Since the Ridge model and Lasso model didn't perform much better than the linear model, it appears that shrinkage wasn't really necessary. Looking at just the correlation matrix between predictors there isn't really a multicollinearity that necessitates the shrinkage that ridge or lasso offers.