STAT 388

Charles Hwang

Professor Matthews

STAT 388-001

4 October 2019

Exercise 6.4

```
rm(list=ls())
# a) iv. - As s increases, the bj's will increase until reaching the least squares estimate, decreasing
# b) ii. - As s increases, the bj's will increase until reaching the least squares estimate, initially
# c) iii. - As s increases, the bj's will increase until reaching the least squares estimate, increasin
# d) iv. - As s increases, the bj's will increase until reaching the least squares estimate, decreasing
# e) v. - By definition, irreducible error is inherent due to the nature of the model and thus independ
```

Exercise 6.9 (nice!)

```
rm(list=ls())
library(ISLR)
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-8
```

```
set.seed(69)                                                    # Exercise 6.9a
n <- floor(0.75*nrow(College))
trains <- sample(seq_len(nrow(College)),size=n)
trainvec <- College[trains,]
testvec <- College[-trains,]
lm <- lm(Apps~.,data=trainvec)                                  # Exercise 6.9b
summary(lm)
```

```
##
## Call:
## lm(formula = Apps ~ ., data = trainvec)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3330.0  -416.6   -59.9   331.9  6970.9
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -570.80840  426.77375  -1.337 0.181600
## PrivateYes  -576.44080  142.71188  -4.039 6.11e-05 ***
## Accept         1.35616    0.05457  24.853  < 2e-16 ***
```

```
## Enroll        -0.63749    0.21924  -2.908 0.003784 **
## Top10perc     51.91124    6.05470   8.574  < 2e-16 ***
## Top25perc    -16.52972    4.83082  -3.422 0.000667 ***
## F.Undergrad    0.10318    0.03624   2.847 0.004572 **
## P.Undergrad    0.03398    0.03205   1.060 0.289593
## Outstate      -0.05003    0.01982  -2.524 0.011866 *
## Room.Board     0.16593    0.04999   3.319 0.000960 ***
## Books          0.04829    0.25069   0.193 0.847310
## Personal       0.08350    0.07055   1.184 0.237084
## PhD           -7.06542    4.81444  -1.468 0.142784
## Terminal      -2.57922    5.27639  -0.489 0.625156
## S.F.Ratio      2.89297   13.01116   0.222 0.824125
## perc.alumni   -6.35528    4.20916  -1.510 0.131637
## Expend         0.06759    0.01299   5.204 2.73e-07 ***
## Grad.Rate      9.90684    3.19764   3.098 0.002044 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 941.7 on 564 degrees of freedom
## Multiple R-squared:  0.937,  Adjusted R-squared:  0.9351
## F-statistic: 493.3 on 17 and 564 DF,  p-value: < 2.2e-16
```

```
lmerr <- mean((predict(lm,testvec)-testvec$Apps)^2)
lmerr
```

```
## [1] 1897822
```

```
trainmat <- model.matrix(Apps~.,data=trainvec)                # Exercise 6.9c
testmat <- model.matrix(Apps~.,data=testvec)
grid <- 10^seq(10,-2,length=100)
ridge <- glmnet(trainmat,trainvec$Apps,alpha=0,lambda=grid)
cvridge <- cv.glmnet(trainmat,trainvec$Apps,alpha=0,lambda=grid)
bestridge <- cvridge$lambda.min
ridgeerr <- mean((predict(ridge,s=bestridge,newx=testmat)-testvec$Apps)^2)
c(bestridge,ridgeerr)
```

```
## [1]       0.01 1900340.64
```

```
lasso <- glmnet(trainmat,trainvec$Apps,alpha=1,lambda=grid)   # Exercise 6.9d
cvlasso <- cv.glmnet(trainmat,trainvec$Apps,alpha=1,lambda=grid)
bestlasso <- cvlasso$lambda.min
lassoerr <- mean((predict(lasso,s=bestlasso,newx=testmat)-testvec$Apps)^2)
c(bestlasso,lassoerr)
```

```
## [1] 1.072267e+01 2.039460e+06
```

```
predict(lasso,s=bestlasso,type="coefficients")
```

```
## 19 x 1 sparse Matrix of class "dgCMatrix"
##                         s1
## (Intercept) -732.15887549
## (Intercept)    .
## PrivateYes  -592.58278066
## Accept         1.25701176
## Enroll         .
## Top10perc     42.07158977
## Top25perc     -8.85443492
```

```
## F.Undergrad    0.02520215
## P.Undergrad    0.03250908
## Outstate      -0.03372018
## Room.Board     0.16618589
## Books          0.02412449
## Personal       0.07339227
## PhD           -6.32837806
## Terminal      -1.81590635
## S.F.Ratio      .
## perc.alumni   -7.36198674
## Expend         0.06332181
## Grad.Rate      7.62889965
```

```
# There are 10 nonzero coefficients.
library(pls)                                                    # Exercise 6.9e
```

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

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

```
pcr <- pcr(Apps~.,data=trainvec,scale=TRUE,validation="CV")
summary(pcr) # Lowest value is at M = 17
```

```
## Data:    X dimension: 582 17
##  Y dimension: 582 1
## Fit method: svdpc
## Number of components considered: 17
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV            3699     3693     1737     1739     1637     1341     1266
## adjCV         3699     3695     1733     1738     1730     1324     1260
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV        1261     1222     1175      1172      1179      1181      1184
## adjCV     1257     1214     1172      1170      1177      1179      1182
##        14 comps  15 comps  16 comps  17 comps
## CV         1185      1193      1009     991.7
## adjCV      1183      1191      1006     988.2
##
## TRAINING: % variance explained
##       1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X       30.52    56.49    63.39    69.06    74.65    79.81    83.76    87.28
## Apps     1.64    78.95    79.32    79.85    88.25    89.25    89.31    90.04
##       9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X       90.34     92.87     95.02     96.96     98.14     98.98     99.52
## Apps    90.41     90.55     90.57     90.59     90.59     90.59     90.60
##       16 comps  17 comps
## X        99.88     100.0
## Apps     93.30      93.7
```

```
pcrerr <- mean((predict(pcr,testvec,ncomp=17)-testvec$Apps)^2)
c(pcrerr,17)
```

```
## [1] 1897822      17
```

```
plsr <- plsr(Apps~.,data=trainvec,scale=TRUE,validation="CV") # Exercise 6.9f
summary(plsr) # Lowest value is at M = 9
```

```
## Data:    X dimension: 582 17
##  Y dimension: 582 1
## Fit method: kernelpls
## Number of components considered: 17
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV            3699     1579     1188     1181     1151     1110     1075
## adjCV         3699     1573     1168     1177     1145     1106     1067
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV        1067     1061     1051      1047      1053      1049      1054
## adjCV     1060     1055     1044      1040      1046      1043      1047
##        14 comps  15 comps  16 comps  17 comps
## CV         1054      1055      1055      1055
## adjCV      1047      1048      1049      1049
##
## TRAINING: % variance explained
##       1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X       25.97    36.02    61.86    65.53    69.44    72.64    76.35    80.15
## Apps    83.29    89.94    90.95    91.82    92.78    93.49    93.55    93.57
##       9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X       82.02     84.30     88.15     91.02     93.55      96.0     97.23
## Apps    93.63     93.68     93.69     93.70     93.70      93.7     93.70
##       16 comps  17 comps
## X        98.76     100.0
## Apps     93.70      93.7
```

```
plsrerr <- mean((predict(plsr,testvec,ncomp=9)-testvec$Apps)^2)
c(plsrerr,9)
```

```
## [1] 1939078       9
```

```
testavg <- mean(testvec$Apps)                                    # Exercise 6.9g
lmcorr <- 1-lmerr/mean((testavg-testvec$Apps)^2)
ridgecorr <- 1-ridgeerr/mean((testavg-testvec$Apps)^2)
lassocorr <- 1-lassoerr/mean((testavg-testvec$Apps)^2)
pcrcorr <- 1-pcrerr/mean((testavg-testvec$Apps)^2)
plsrcorr <- 1-plsrerr/mean((testavg-testvec$Apps)^2)
c(lmcorr,ridgecorr,lassocorr,pcrcorr,plsrcorr)
```

```
## [1] 0.8993459 0.8992123 0.8918339 0.8993459 0.8971577
```

```
# The r^2 for all models are very similar. We can predict the number of college applications received q
```

Exercise 6.10

```
rm(list=ls())
set.seed(1)
x <- matrix(rnorm(1000*20),1000,20)          # Exercise 6.10a
b <- rnorm(20)
```

```
b[2] <- 0
b[3] <- 0
b[5] <- 0
b[7] <- 0
b[11] <- 0
b[13] <- 0
b[17] <- 0
error <- rnorm(1000)
y <- x%*%b+error
train <- sample(seq(1000),100,replace=FALSE) # Exercise 6.10b
test <- (-train)
xtrain<-x[train,]
xtest<-x[test,]
ytrain<-y[train]
ytest<-y[test]
library(leaps)                               # Exercise 6.10c
data.train <- data.frame(y=ytrain,x=xtrain)

                                             # Exercise 6.10d

                                             # Exercise 6.10e

                                             # Exercise 6.10f

                                             # Exercise 6.10g
```