EX9

1. Split the data set into a training set and a test set.

library(ISLR)

## Warning: package 'ISLR' was built under R version 3.4.4

data("College")  
college <- College  
sum(is.na(College))

## [1] 0

set.seed(123)  
attach(college)  
indexes=sample(1:nrow(college),size=0.3\*nrow(college))  
# Split data, 70% training & 30% test  
train=college[-indexes,]  
test=college[indexes,]

1. Fit a linear model using least squares on the training set, and report the test error obtained.

model=lm(Apps~.,data=train)  
summary(model)

##   
## Call:  
## lm(formula = Apps ~ ., data = train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2818.7 -498.6 -44.0 340.5 6464.0   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -233.97009 491.79329 -0.476 0.634451   
## PrivateYes -859.99870 169.01676 -5.088 5.04e-07 \*\*\*  
## Accept 1.22978 0.06047 20.336 < 2e-16 \*\*\*  
## Enroll -0.06919 0.25206 -0.275 0.783806   
## Top10perc 55.57605 6.35642 8.743 < 2e-16 \*\*\*  
## Top25perc -17.86627 5.04564 -3.541 0.000434 \*\*\*  
## F.Undergrad 0.03436 0.04453 0.772 0.440698   
## P.Undergrad 0.03569 0.03563 1.002 0.316872   
## Outstate -0.06382 0.02238 -2.852 0.004522 \*\*   
## Room.Board 0.21330 0.05598 3.810 0.000155 \*\*\*  
## Books 0.16288 0.32559 0.500 0.617107   
## Personal -0.05011 0.07316 -0.685 0.493741   
## PhD -6.14646 5.25532 -1.170 0.242704   
## Terminal -6.10063 5.76071 -1.059 0.290083   
## S.F.Ratio 5.63435 16.10093 0.350 0.726524   
## perc.alumni -6.52568 4.70008 -1.388 0.165597   
## Expend 0.06970 0.01410 4.944 1.03e-06 \*\*\*  
## Grad.Rate 11.85922 3.42604 3.461 0.000581 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1003 on 526 degrees of freedom  
## Multiple R-squared: 0.9179, Adjusted R-squared: 0.9152   
## F-statistic: 345.8 on 17 and 526 DF, p-value: < 2.2e-16

pred=predict(model,newdata=test)  
MSE=mean((test$Apps-pred)^2)  
MSE

## [1] 1707004

1. Fit a ridge regression model on the training set, with λ chosen by cross-validation. Report the test error obtained

require(glmnet)

## Loading required package: glmnet

## Warning: package 'glmnet' was built under R version 3.4.4

## Loading required package: Matrix

## Loading required package: foreach

## Warning: package 'foreach' was built under R version 3.4.4

## Loaded glmnet 2.0-16

set.seed(1)  
xtrain=model.matrix (Apps~.,train)[,-1]  
ytrain=train$Apps  
xtest=model.matrix (Apps~.,test)[,-1]  
ytest=test$Apps  
cv.out=cv.glmnet(xtrain,ytrain,alpha =0)  
bestlam=cv.out$lambda.min  
pred.ridge <- predict(cv.out, s = bestlam,newx = xtest)  
mean((pred.ridge - ytest)^2)

## [1] 3043159

1. Fit a lasso model on the training set, with λ chosen by crossvalidation. Report the test error obtained, along with the number of non-zero coefficient estimates.

set.seed(1)  
cv.out=cv.glmnet (xtrain,ytrain,alpha =1)  
bestlam=cv.out$lambda.min  
pred.lasso <- predict(cv.out, s = bestlam,newx = xtest)  
mean((pred.ridge - ytest)^2)

## [1] 3043159

1. Fit a PCR model on the training set, with M chosen by crossvalidation. Report the test error obtained, along with the value of M selected by cross-validation.

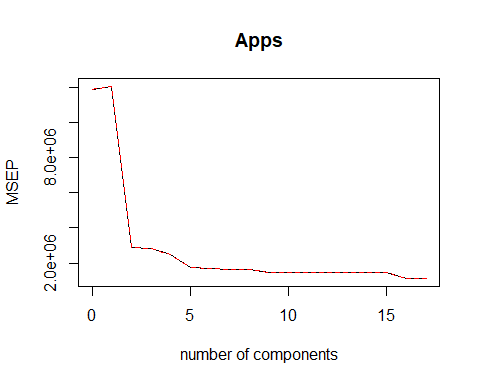
library(pls)

## Warning: package 'pls' was built under R version 3.4.4

##   
## Attaching package: 'pls'

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

fit.pcr <- pcr(Apps ~ ., data = train, scale = TRUE, validation = "CV")  
validationplot(fit.pcr, val.type = "MSEP")

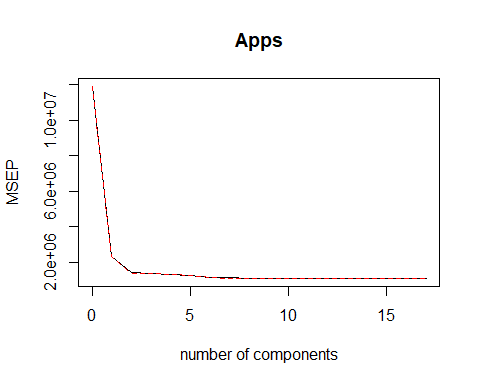


pred.pcr <- predict(fit.pcr, test, ncomp = 10)  
mean((pred.pcr - ytest)^2)

## [1] 3873203

1. Fit a PLS model on the training set, with M chosen by cross-validation. Report the test error obtained, along with the value of M selected by cross-validation.

fit.pls <- plsr(Apps ~ ., data =train, scale = TRUE, validation = "CV")  
validationplot(fit.pls, val.type = "MSEP")

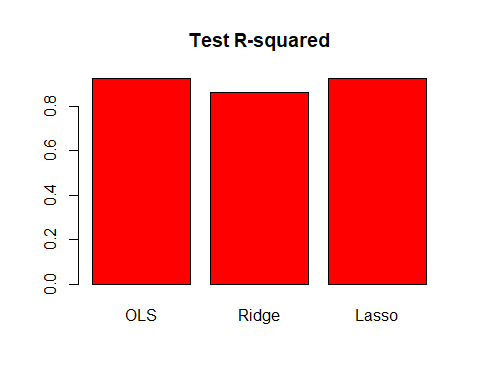


pred.pls <- predict(fit.pls, test, ncomp = 10)  
mean((pred.pls - ytest)^2)

## [1] 1759007

1. Comment on the results obtained. How accurately can we predict the number of college applications received? Is there much difference among the test errors resulting from these five approaches?

test.avg <- mean(test[, "Apps"])  
  
lm.test.r2 <- 1 - mean((test[, "Apps"] - pred)^2) /  
 mean((test[, "Apps"] - test.avg)^2)  
  
ridge.test.r2 <- 1 - mean((test[, "Apps"] - pred.ridge)^2)/  
 mean((test[, "Apps"] - test.avg)^2)  
  
lasso.test.r2 <- 1 - mean((test[, "Apps"] - pred.lasso)^2) /  
 mean((test[, "Apps"] - test.avg)^2)  
  
barplot(c(lm.test.r2, ridge.test.r2, lasso.test.r2),  
 col = "red", names.arg = c("OLS", "Ridge", "Lasso"),  
 main = "Test R-squared")

 lasso and OLS R^2 are better