HW 7 solutions

Homework 7

Chapter 6, Exercise 9

```
library(ISLR)
set.seed(123)

# recommended (but optional):
# check if there are any missing observations:
print(sum(is.na(College)))
```

Previous commands (a)-(d):

```
## [1] 0
```

```
n = dim(College)[1]
train.size = n / 2
train = sample(1:n, train.size)
test = -train
College.train = College[train, ]
College.test = College[test, ]
```

```
lm.fit = lm(Apps~., data=College.train)
lm.pred = predict(lm.fit, College.test)
mean((College.test[, "Apps"] - lm.pred)^2)
```

Linear model using least squares on the training set, and report the test error obtained.

```
## [1] 1373995
```

Ridge regression model on the training set, with λ chosen by cross-validation. Report the test error obtained. (Pick λ using College.train and report error on College.test)

```
library(glmnet)
```

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

```
## Loaded glmnet 4.1-4
```

```
train.mat = model.matrix(Apps~., data=College.train)
test.mat = model.matrix(Apps~., data=College.test)
grid = 10 ^ seq(4, -2, length=100)
mod.ridge = cv.glmnet(train.mat, College.train[, "Apps"], alpha=0, lambda=grid, thresh=1e-12)
lambda.best = mod.ridge$lambda.min
lambda.best
```

```
## [1] 18.73817
```

```r
ridge.pred = predict(mod.ridge, newx=test.mat, s=lambda.best)
mean((College.test[, "Apps"] - ridge.pred)^2)
```

```
## [1] 1431537
```

Lasso model on the training set, with λ chosen by cross-validation. Report the test error obtained, along with the number of non-zero coefficient estimates. (Pick λ using College.train and report error on College.test)

```r
mod.lasso = cv.glmnet(train.mat, College.train[, "Apps"], alpha=1, lambda=grid, thresh=1e-12)

lambda.best = mod.lasso$lambda.min
lambda.best
```

```
## [1] 21.54435
```

```r
lasso.pred = predict(mod.lasso, newx=test.mat, s=lambda.best)
mean((College.test[, "Apps"] - lasso.pred)^2)
```

```
## [1] 1397303
```

The coefficients look like

```r
mod.lasso = glmnet(model.matrix(Apps~., data=College), College[, "Apps"], alpha=1)
lasso.coefs = predict(mod.lasso, s=lambda.best, type="coefficients")

print(lasso.coefs)
```

```
## 19 x 1 sparse Matrix of class "dgCMatrix"
##                         s1
## (Intercept) -6.038452e+02
## (Intercept)  .
## PrivateYes  -4.235413e+02
## Accept       1.455236e+00
## Enroll      -2.003696e-01
## Top10perc    3.367640e+01
## Top25perc   -2.403036e+00
## F.Undergrad  .
## P.Undergrad  2.086035e-02
## Outstate    -5.781855e-02
## Room.Board   1.246462e-01
## Books        .
## Personal     1.832912e-05
## PhD         -5.601313e+00
## Terminal    -3.313824e+00
## S.F.Ratio    4.478684e+00
## perc.alumni -9.796600e-01
## Expend       6.967693e-02
## Grad.Rate    5.159652e+00
```

```r
sum(lasso.coefs != 0)
```

```
## [1] 16
```

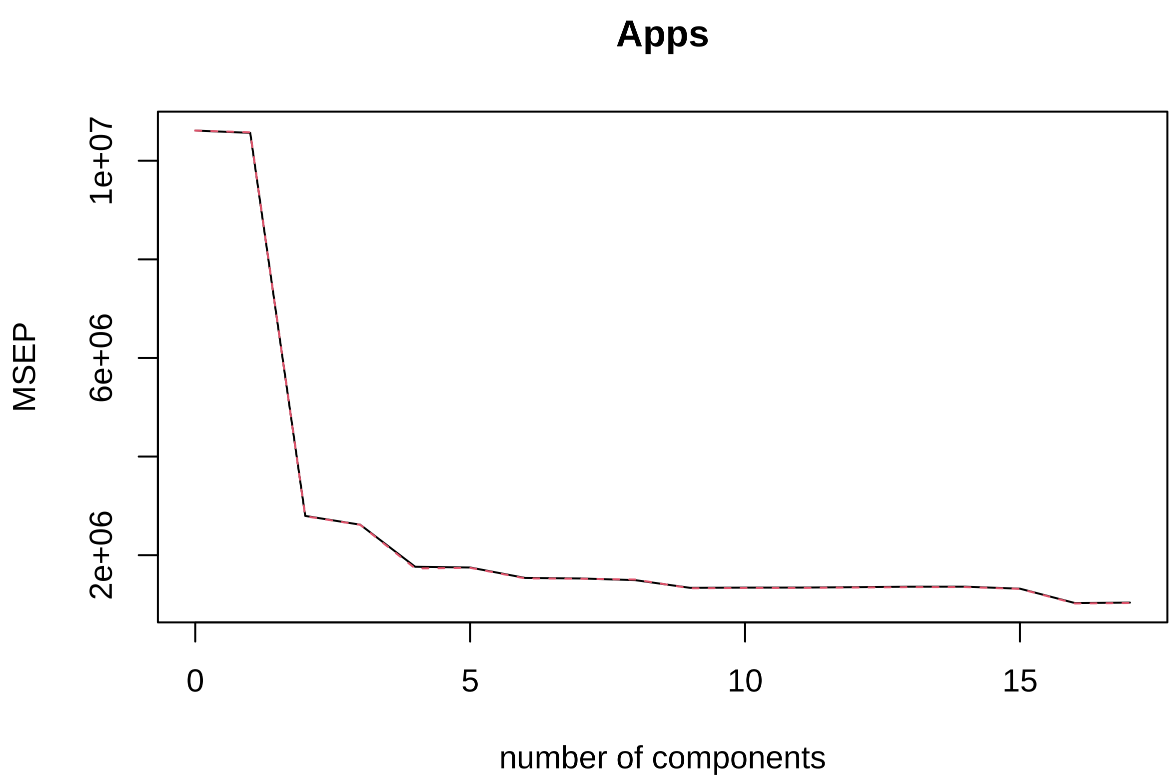
There are 16 non-zero lasso coefficients.

(e) Fit a PCR model on the training set, with number of principal components M chosen by cross-validation. Report the test error obtained, along with the value of M selected by cross-validation.

```
library(pls)
```

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

```
pcr.fit = pcr(Apps~., data=College.train, scale=T, validation="CV")
validationplot(pcr.fit, val.type="MSEP")
```



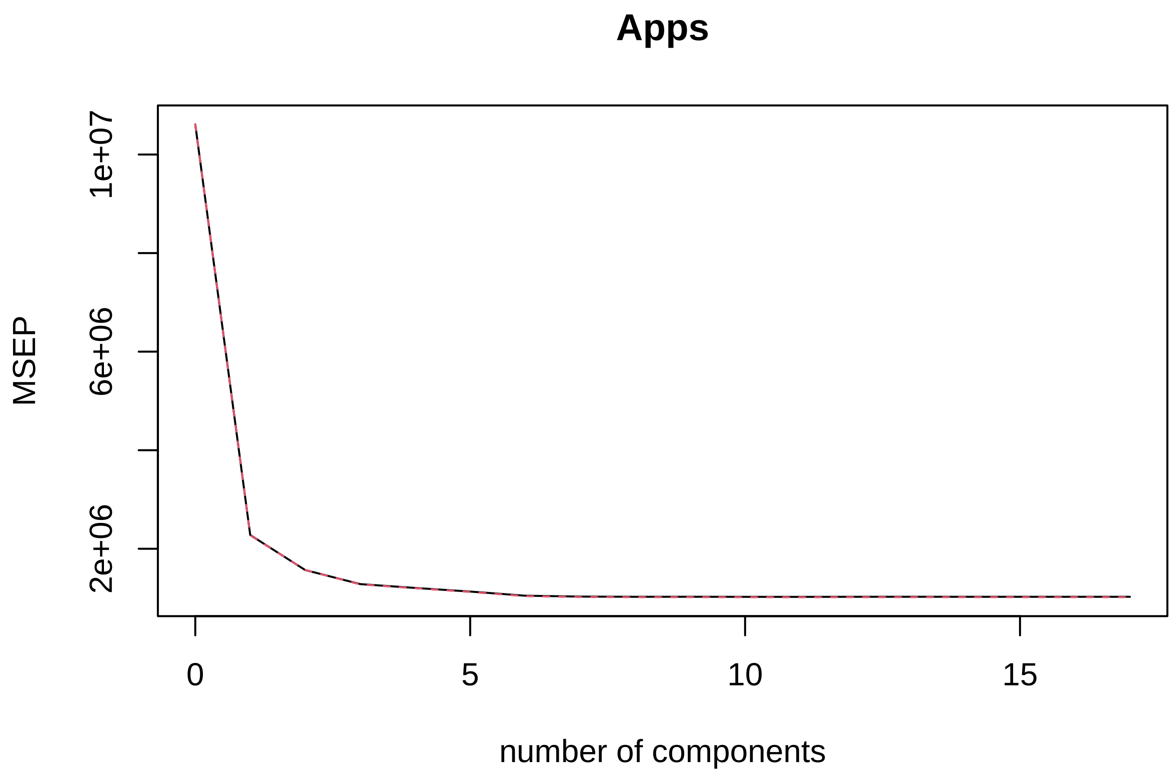

```
pcr.pred = predict(pcr.fit, College.test, ncomp=10)
mean((College.test$Apps - pcr.pred)^2)
```

```
## [1] 2887472
```

Test RSS for PCR is about $2,887,472$.

(f) 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.

```
pls.fit = plsr(Apps~., data=College.train, scale=T, validation="CV")
validationplot(pls.fit, val.type="MSEP")
```




```
pls.pred = predict(pls.fit, College.test, ncomp=10)
mean((College.test$Apps - pls.pred)^2)
```

```
## [1] 1384151
```

Test RSS for PLS is about $1,384,151$.

g 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?

Results for OLS, Lasso, Ridge are comparable. Lasso reduces the `F.Undergrad` and `Books` variables to zero and shrinks coefficients of other variables.

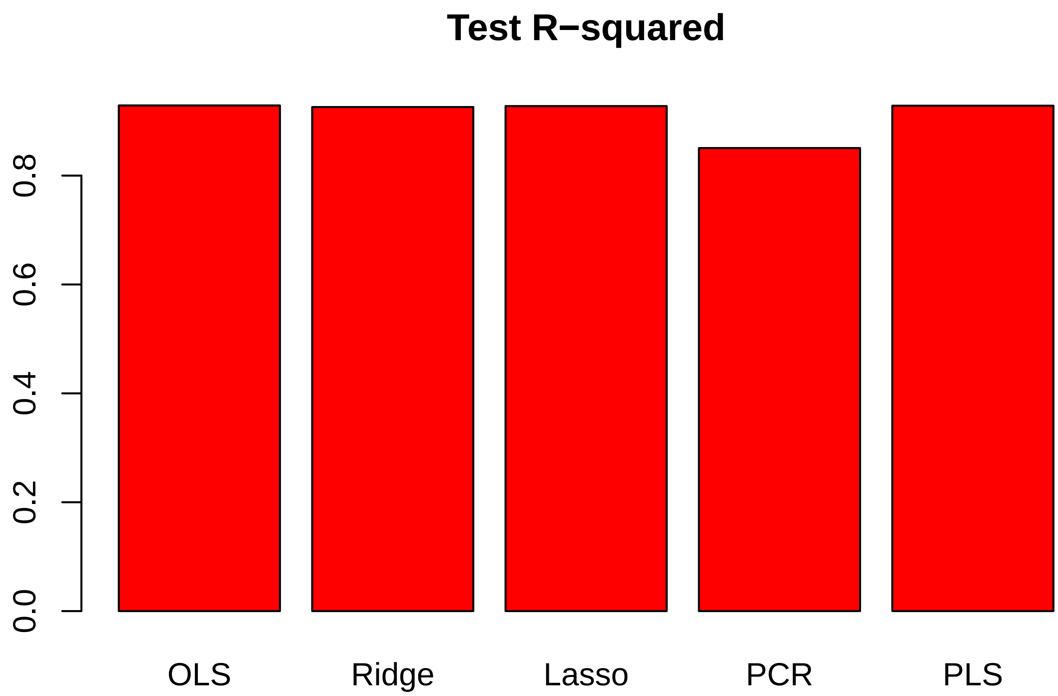
Evidence for this can be presented in many ways.

One is to compute the test R^2 for all models, and visualize the difference via a bar plot:

```
test.avg = mean(College.test$Apps)

lm.test.r2 = 1 - mean((College.test$Apps - lm.pred)^2) /mean((College.test$Apps - test.avg)^2)
ridge.test.r2 = 1 - mean((College.test$Apps - ridge.pred)^2) /mean((College.test$Apps - test.avg)^2)
lasso.test.r2 = 1 - mean((College.test$Apps - lasso.pred)^2) /mean((College.test$Apps - test.avg)^2)

pcr.test.r2 = 1 - mean((College.test$Apps - pcr.pred)^2) /mean((College.test$Apps - test.avg)^2)
pls.test.r2 = 1 - mean((College.test$Apps - pls.pred)^2) /mean((College.test$Apps - test.avg)^2)
barplot(c(lm.test.r2, ridge.test.r2, lasso.test.r2, pcr.test.r2, pls.test.r2),
        col="red", names.arg=c("OLS", "Ridge", "Lasso", "PCR", "PLS"), main="Test R-squared")
```



The plot shows that test R^2 for all models except PCR are around 0.9, with PLS having slightly higher test R^2 than others. PCR has a smaller test R^2 of less than 0.8. All models except PCR predict college applications with high accuracy.