Homework 6

Anthony Anderson

April 7, 2019

Problem 1

```r
library(ISLR)
#a)
set.seed(42)
train=sample(777,388)
train.data <- College[train,]
test.data <- College[-train,]

#b)
linear.model <- lm(Apps~.,data=College, subset=train)
summary(linear.model)
```

```
##
## Call:
## lm(formula = Apps ~ ., data = College, subset = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4180.9  -396.6    18.5   370.3  6415.0
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -791.16155  655.59274  -1.207 0.228285
## PrivateYes  -216.85007  211.52484  -1.025 0.305950
## Accept         1.69400    0.05418  31.268  < 2e-16 ***
## Enroll        -1.10674    0.28620  -3.867 0.000130 ***
## Top10perc     58.31426    8.85605   6.585 1.57e-10 ***
## Top25perc    -21.05485    6.80832  -3.093 0.002135 **
## F.Undergrad    0.09158    0.04735   1.934 0.053870 .
## P.Undergrad    0.03189    0.06512   0.490 0.624627
## Outstate      -0.11118    0.03014  -3.688 0.000259 ***
## Room.Board     0.17742    0.07629   2.325 0.020586 *
## Books         -0.29869    0.36187  -0.825 0.409677
## Personal       0.05157    0.10936   0.472 0.637486
## PhD           -7.84830    6.69688  -1.172 0.241977
## Terminal      -6.10925    7.56561  -0.808 0.419896
## S.F.Ratio     29.78362   21.84336   1.364 0.173551
## perc.alumni    1.48008    6.06903   0.244 0.807463
## Expend         0.11852    0.02259   5.247 2.60e-07 ***
## Grad.Rate      9.30278    4.55793   2.041 0.041959 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1104 on 370 degrees of freedom
## Multiple R-squared:  0.933,  Adjusted R-squared:   0.93
## F-statistic: 303.3 on 17 and 370 DF,  p-value: < 2.2e-16
```

```r
linear.predict <- predict(linear.model, test.data)
linear.MSE <- mean((linear.predict-test.data$Apps)^2)
linear.MSE
```

```
## [1] 1128096
```

```r
#c)
y<- train.data$Apps
x<- model.matrix(Apps~.-1, data=train.data)
test.matrix<-model.matrix(Apps~.-1,data=test.data)

library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 3.5.3
## Loading required package: Matrix
## Loading required package: foreach
## Warning: package 'foreach' was built under R version 3.5.3
## Loaded glmnet 2.0-16
```

```r
grid <- 10^seq(6,-2,length=100)
CV.ridge.model<- cv.glmnet(x=x,y=y,alpha=0,lambda=grid,nfolds=10)
best.lambda<- CV.ridge.model$lambda.min

ridge.model<-glmnet(x=x,y=y,alpha=0,lambda=best.lambda)
ridge.predict<-predict(ridge.model,newx=test.matrix,s=best.lambda)
ridge.MSE<-mean((ridge.predict-test.data[,"Apps"])^2)
ridge.MSE
```

```
## [1] 1127337
```

```r
#d)
CV.lasso.model<- cv.glmnet(x=x,y=y,alpha=1,lambda=grid,nfolds=10)
best.lasso.lambda<- CV.lasso.model$lambda.min

lasso.model<-glmnet(x=x,y=y,alpha=1,lambda=best.lasso.lambda)
lasso.predict<-predict(lasso.model,newx=test.matrix,s=best.lasso.lambda)
lasso.MSE<-mean((lasso.predict-test.data[,"Apps"])^2)
lasso.MSE
```

```
## [1] 1127313
```

```r
coef(CV.lasso.model,s=best.lasso.lambda)
```

```
## 19 x 1 sparse Matrix of class "dgCMatrix"
##                          1
## (Intercept) -785.14471252
## PrivateNo     -8.01980435
## PrivateYes  -225.48385929
## Accept         1.69303839
## Enroll        -1.09801731
## Top10perc     58.22525565
## Top25perc    -20.98226938
## F.Undergrad    0.09040808
## P.Undergrad    0.03208491
## Outstate      -0.11103985
```

```
## Room.Board     0.17756067
## Books         -0.29858489
## Personal       0.05157563
## PhD           -7.84411539
## Terminal      -6.12961975
## S.F.Ratio     29.81823883
## perc.alumni    1.47112640
## Expend         0.11852962
## Grad.Rate      9.28908135
```

There are 15 coefficients that go to 0, leaving only 4 (including intercept) with non-zero estimates.

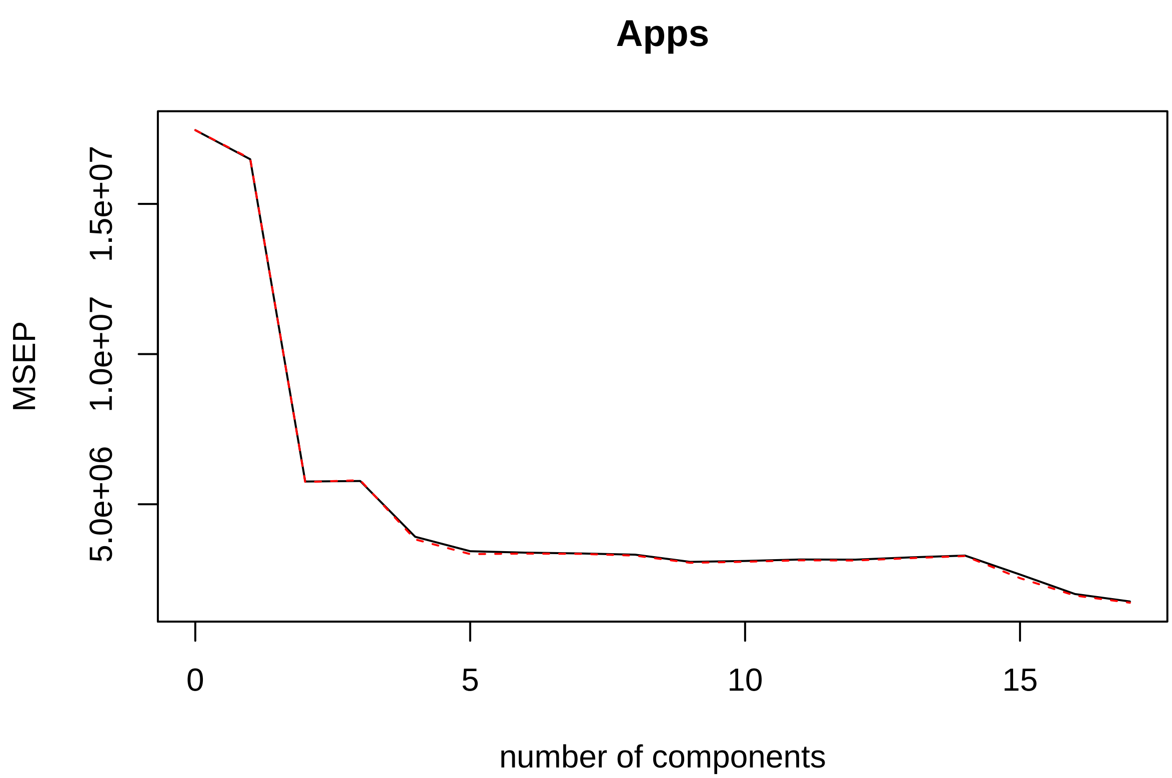
```
#e)
library(pls)
```

```
## Warning: package 'pls' was built under R version 3.5.3

##
## Attaching package: 'pls'

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

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



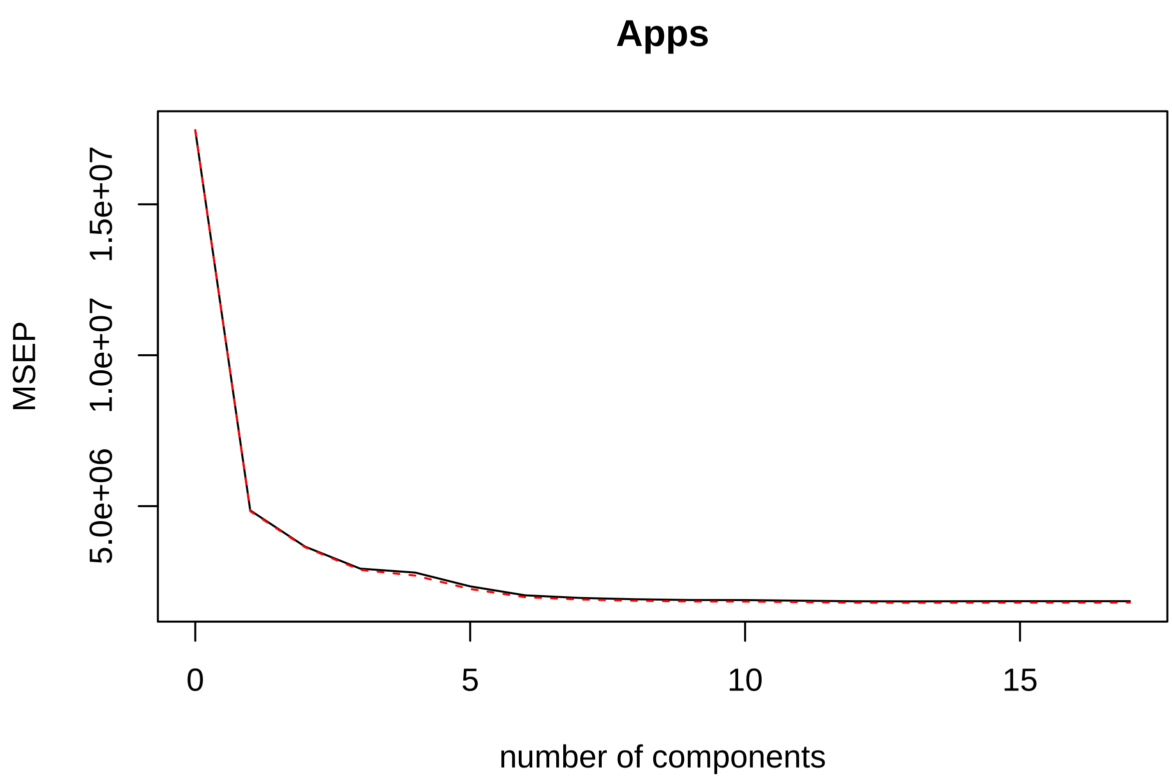

```
pcr.predict<-predict(pcr.model,test.data)
pcr.MSE<-mean((test.data[,"Apps"]-(pcr.predict))^2)
```

```
pcr.MSE
```

```
## [1] 2520407
```

The validation plot shows us that the best number of components to use is $M = 4$. MSE plateaus after this so to satisfy principle of parsimony we should use the smallest number of components that gives us the most information.

```
#e)
pls.model<-plsr(Apps~.,data=train.data,scale=T,validation="CV")
validationplot(pls.model,val.type="MSEP")
```




```
pls.predict<-predict(pls.model,test.data)
pls.MSE<-mean((test.data[,"Apps"]-pls.predict)^2)
pls.MSE
```

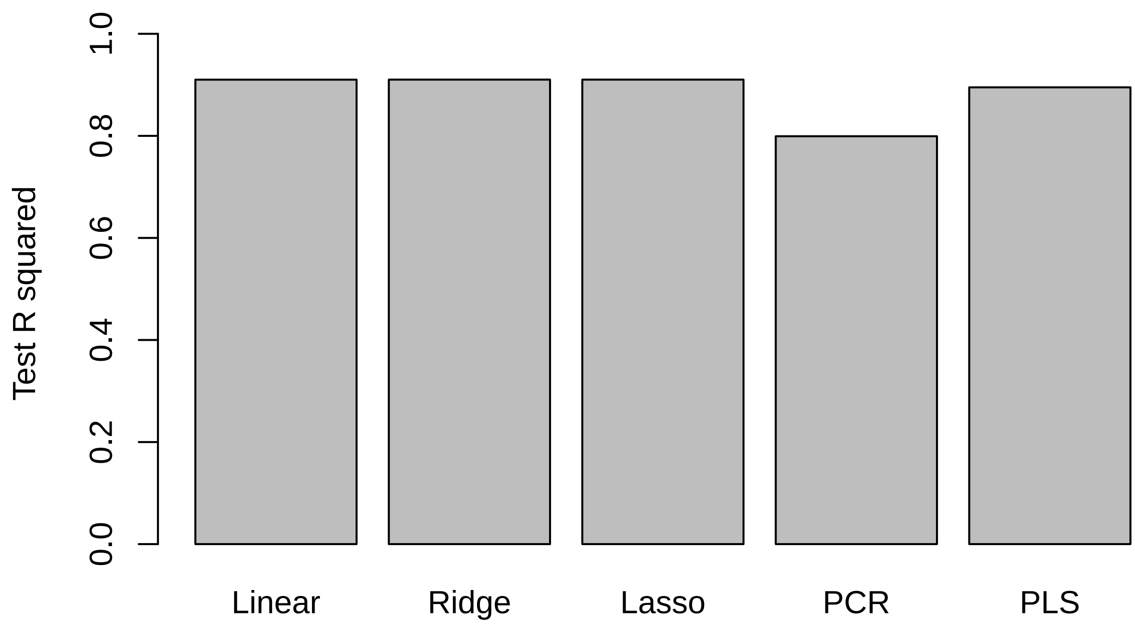
```
## [1] 1317110
```

M is roughly 6 according to the validation plot.

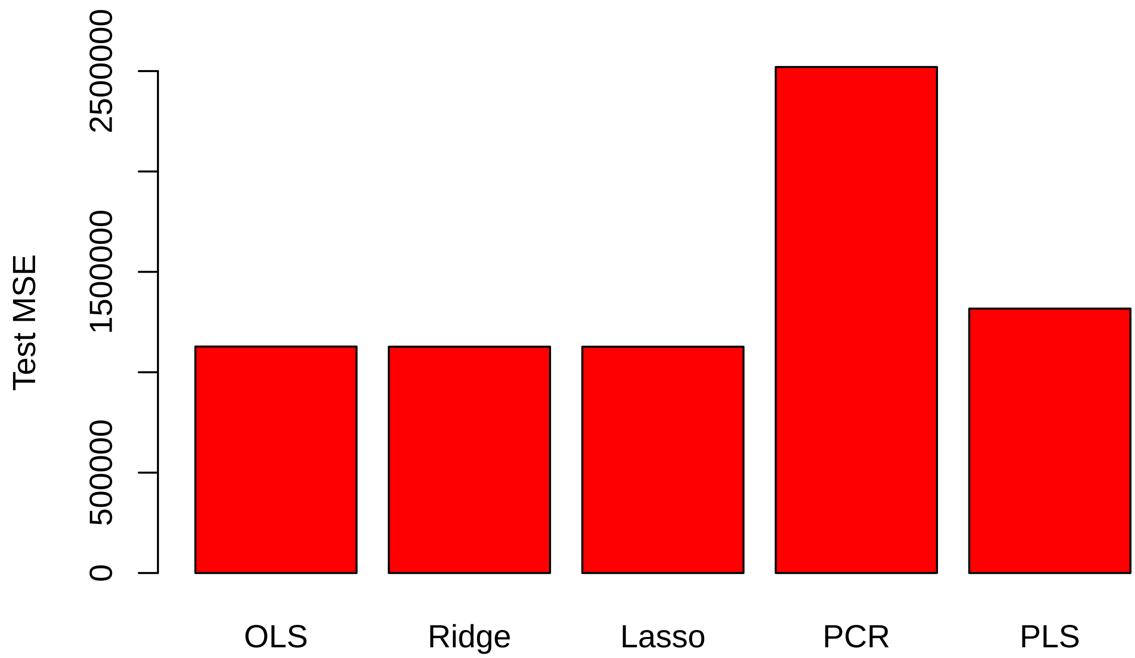
g)

```
average_test <- mean(test.data[, "Apps"])
linear_r2 = 1 - linear.MSE/mean((test.data[, "Apps"] - average_test)^2)
ridge_r2 = 1 - ridge.MSE /mean((test.data[, "Apps"] - average_test)^2)
lasso_r2 = 1 - lasso.MSE /mean((test.data[, "Apps"] - average_test)^2)
pcr_r2 = 1 - pcr.MSE /mean((test.data[, "Apps"] -average_test)^2)
pls_r2 = 1 - pls.MSE /mean((test.data[, "Apps"] -average_test)^2)
```

```r
barplot(c(linear_r2,ridge_r2,lasso_r2,pcr_r2,pls_r2),
        names.arg = c("Linear","Ridge","Lasso","PCR","PLS"),ylab="Test R squared ",ylim=c(0,1))
```




```r
barplot(c(linear.MSE, ridge.MSE, lasso.MSE, pcr.MSE, pls.MSE), col="red",
        names.arg=c("OLS","Ridge", "Lasso", "PCR", "PLS"), ylab = "Test MSE")
```



The above two graphs compare each model's R^2 value for the test data, as well as the Test MSE.
Each model has a very high R^2 value of around .9, except for PCR which has a value of .8
PCR also is the only model with a significant jump in Test MSE compared to the other models. Each model is comparable in terms of prediction power and variability, except for PCR, which should be avoided on this data set. PLS is also not the best choice of model for this data, but not as bad as PCR. Use either Linear, Ridge, or Lasso models for best results.