STATS415hw7

Yunguo Cai

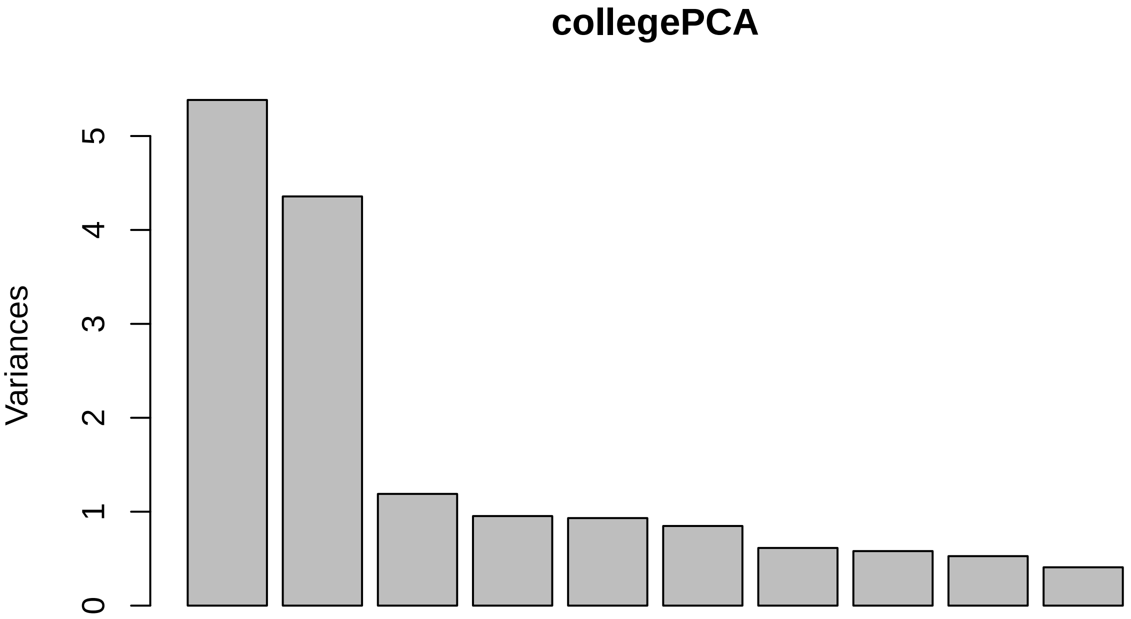
3/14/2018

1.

```
X <- model.matrix(Apps ~ ., data = College)[, -1]
collegePCA <- prcomp(x = X, center = T, scale = T)
summary(collegePCA)
```

```
## Importance of components:
##                           PC1    PC2     PC3    PC4     PC5     PC6
## Standard deviation     2.3203 2.0873 1.09067 0.9766 0.96542 0.92090
## Proportion of Variance 0.3167 0.2563 0.06997 0.0561 0.05483 0.04989
## Cumulative Proportion  0.3167 0.5730 0.64295 0.6990 0.75387 0.80376
##                            PC7     PC8     PC9    PC10    PC11    PC12
## Standard deviation     0.78375 0.76191 0.72584 0.63900 0.59815 0.55366
## Proportion of Variance 0.03613 0.03415 0.03099 0.02402 0.02105 0.01803
## Cumulative Proportion  0.83989 0.87404 0.90503 0.92905 0.95010 0.96813
##                           PC13    PC14    PC15    PC16    PC17
## Standard deviation     0.43069 0.37981 0.32170 0.28448 0.16631
## Proportion of Variance 0.01091 0.00849 0.00609 0.00476 0.00163
## Cumulative Proportion  0.97904 0.98752 0.99361 0.99837 1.00000
```

```
plot(collegePCA)
```

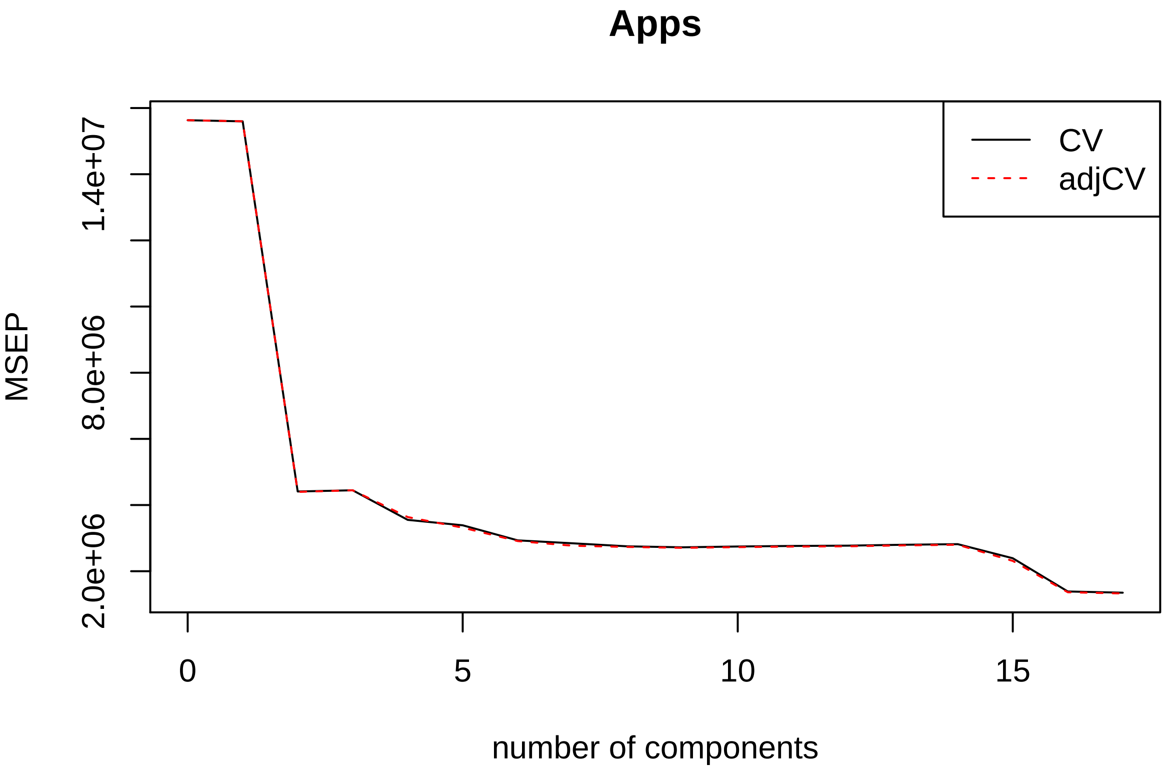


```
collegePCA$rotation[,1:2]
```

```
##                    PC1          PC2
## PrivateYes -0.20281185  0.319558922
## Accept     -0.01314987 -0.419033332
## Enroll      0.02870458 -0.442951639
## Top10perc  -0.34473625 -0.130412436
## Top25perc   -0.31867477 -0.161422543
## F.Undergrad  0.05444324 -0.447617431
## P.Undergrad  0.11699395 -0.296842509
## Outstate    -0.37412907  0.064735802
## Room.Board  -0.28258496 -0.007119393
## Books       -0.03425741 -0.083671688
## Personal     0.13338760 -0.174034867
## PhD         -0.24681294 -0.254606301
## Terminal    -0.25157455 -0.242940493
## S.F.Ratio    0.26821705 -0.124625202
## perc.alumni -0.29005763  0.098787317
## Expend      -0.33716672 -0.060092077
## Grad.Rate   -0.29676237  0.016405705
```

We need 9 eigenvalues to explain 90% of the variance in the data. The loadings w1j, w2j mean the weights of each (original) variable in the new linear combination variable Z1, Z2.Zi = wi1x1+wi2x2+…wi17x17. 2.

```
set.seed(23456)
collegePCR = pcr(Apps ~ ., data = College_train, scale = TRUE, validation = "CV")
#summary(collegePCR)
validationplot(collegePCR, val.type = "MSEP", legendpos = "topright")
```



```
validationMSE1 = numeric(collegePCR$ncomp)
for (i in 1:collegePCR$ncomp) {
  validationMSE1[i] = mean((collegePCR$validation$pred[,,i] - College_train$Apps)^2)
}
K_PCR = which.min(validationMSE1)
K_PCR
```

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

```
collegePCR_train = predict(collegePCR, College_train, ncomp = K_PCR)
PCRTrainMSE = mean((collegePCR_train - College_train$Apps)^2)
PCRTrainMSE
```

```
## [1] 993164.6
```

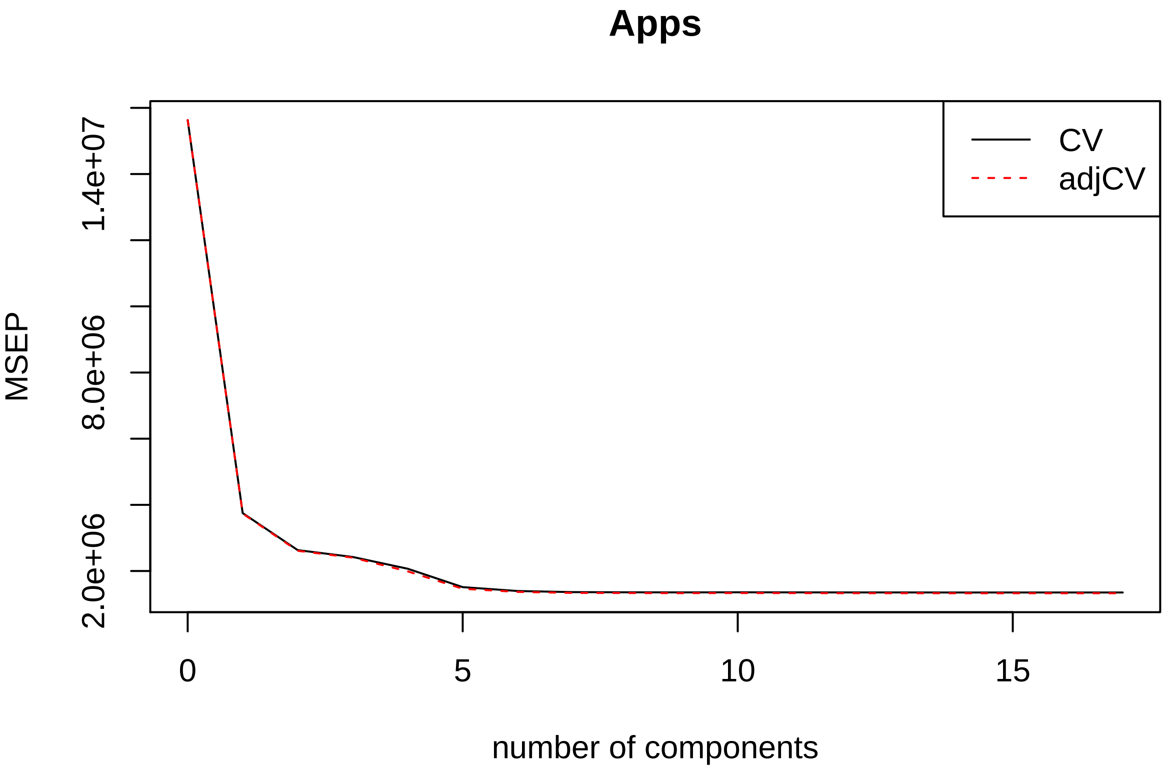
```
collegePCR_test = predict(collegePCR, College_test, ncomp = K_PCR)
PCRTestMSE = mean((collegePCR_test - College_test$Apps)^2)
PCRTestMSE
```

```
## [1] 1300431
```

The training error is 993164.6 and test error is 1300431, along with the value of K selected is 17.

3.

```
set.seed(23456)
collegePLS = plsr(Apps ~ ., data = College_train, scale = TRUE, validation = "CV")
#summary(collegePLS)
validationplot(collegePLS, val.type = "MSEP", legendpos = "topright")
```



```
validationMSE2 = numeric(collegePLS$ncomp)
for (i in 1:collegePLS$ncomp) {
  validationMSE2[i] = mean((collegePLS$validation$pred[,,i] - College_train$Apps)^2)
}
K_PLS = which.min(validationMSE2)
K_PLS
```

```
## [1] 14
```

```
collegePLS_train = predict(collegePLS, College_train, ncomp = K_PLS)
PLSTrainMSE = mean((collegePLS_train - College_train$Apps)^2)
PLSTrainMSE
```

```
## [1] 993169.5
```

```
collegePLS_test = predict(collegePLS, College_test, ncomp = K_PLS)
PLSTestMSE = mean((collegePLS_test - College_test$Apps)^2)
PLSTestMSE
```

```
## [1] 1300759
```

The training error is 993169.5 and test error is 1300759, along with the value of K selected is 14.

4.

```
test_errOLS = 1300431
test_errFwd = 1334782
test_errBwd = 1355206
test_errAIC = 1282321
test_errBIC = 1380054
test_errRidge = 1223126
test_errLasso = 1292839
d = data.frame("TestMSE" = c(PCRTestMSE, PLSTestMSE, test_errOLS, test_errFwd, test_errBwd,
                             test_errAIC, test_errBIC, test_errRidge, test_errLasso))
rownames(d) = c("PCR", "PLS", "OLS", "Fwd", "Bwd", "AIC", "BIC", "Ridge", "Lasso")
knitr::kable(d)
```

PCR	
PLS	
OLS	
Fwd	
Bwd	
AIC	
BIC	
Ridge	
Lasso	
The test	error is smallest for the ridge regression, followed by AIC method and Lasso. This suggests that ridge regression