hw_04

lisa liubovich

2024-06-11

1. College Applications

Predict the number of applications received based on the other variables in the College data set in {ISLR2}.

Use the following code to split the data set randomly. Use the subset college.data for the most of the analysis. Set aside holdout.data data until the last 2 sub-questions.

```
library(ISLR2)
data("College")
my.college <- College[-484, ] # remove an extreme case.
#my.college <- College[College$Apps <=16000, ] # remove several extreme case.
train.pct <- 0.78
set.seed(2024)
Z <- sample(nrow(my.college), floor(train.pct*nrow(my.college)))
college.data <- my.college[Z, ]
holdout.data <- my.college[-Z, ]
```

Recall that we fit the data in homework 1.

- we fit the **full** model with all 17 predictors. That is:

```
college.lmF <- lm(Apps ~ ., data = college.data)
college.lmF
```

```
##
## Call:
## lm(formula = Apps ~ ., data = college.data)
##
## Coefficients:
## (Intercept)   PrivateYes       Accept       Enroll     Top10perc     Top25perc
##  -4.685e+02   -5.008e+02    1.397e+00   -5.142e-01    5.189e+01    -1.480e+01
## F.Undergrad  P.Undergrad     Outstate   Room.Board        Books      Personal
##   6.690e-02    3.188e-02   -6.002e-02    2.188e-01    1.534e-01    -2.228e-03
##         PhD     Terminal    S.F.Ratio  perc.alumni       Expend     Grad.Rate
##  -7.742e+00   -4.247e+00    1.057e+01   -5.725e+00    6.035e-02     6.194e+00
```

Using stepwise selection, AIC is the smallest with 12 predictors:

Apps ~ Private + Accept + Enroll + Top10perc + Top25perc + F.Undergrad + Outstate + Room.Board + PhD + perc.alumni + Expend + Grad.Rate

Using the best-subset algorithm. BIC is the smallest with 7 predictors:

Apps ~ Private + Accept + Top10perc + Outstate + Room.Board + PhD + Expend

Continue working on this data set and the above models.

(a) Compute the variance inflation factor and comment on the severity of the collinearity of the data. Why is "collinearity" a concern, even if the model is correct?

   (a) Some of the variables like Accept, Enroll, and F.Undergrad have VIF values over 5 or 10, indicating that they suffer from more severe multicollinearity. Even if this model is correct, multicollinearity is a concern because it increases the variance and thus the standard errors, which make for less accurate coefficient estimates, inferences, and overly wide confidence intervals.

```r
library(car)
```

```
## Loading required package: carData
```

```r
vif_college <- vif(college.lmF)
print(vif_college)
```

```
##     Private      Accept      Enroll   Top10perc   Top25perc F.Undergrad
##    2.865245   10.150354   21.012832    6.881086    5.392519   16.472473
## P.Undergrad    Outstate  Room.Board       Books    Personal         PhD
##    1.698963    4.379916    2.101790    1.142078    1.363831    4.002334
##    Terminal   S.F.Ratio perc.alumni      Expend   Grad.Rate
##    3.917552    1.982753    1.950469    2.964559    1.929511
```

(b) Evaluate prediction accuracy of your selected models based on AIC and BIC. Estimate the prediction mean squared error by 10-fold cross-validation. Recall that glm(..., family=gaussian) fits linear regression and its outcome can be used in cv.glm() (boot package) for cross-validation.

```r
library(boot)
```

```
##
## Attaching package: 'boot'

## The following object is masked from 'package:car':
##
##     logit
```

```r
AIC_college <- glm(Apps ~ Private + Accept + Enroll + Top10perc + Top25perc + F.Undergrad + Outstat

BIC_college <- glm(Apps ~ Private + Accept + Top10perc + Outstate + Room.Board + PhD + Expend, data

cv.glm(college.data, glmfit = AIC_college, K = 10)$delta[1]
```

```
## [1] 997934.5
```

```r
cv.glm(college.data, BIC_college, K = 10)$delta[1]
```

```
## [1] 980591.9
```

The BIC model has a lower predicted MSE by 10 fold CV.

(c) Consider the **full** model with all 17 predictors (reminder: use data frame college.data you recreated at the beginning). Use functions in package glmnet to fit a ridge regression.

```
library(glmnet)
```

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

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

```
college.X <- model.matrix(college.lmF)[, -1]
dim(college.X)
```

```
## [1] 605  17
```

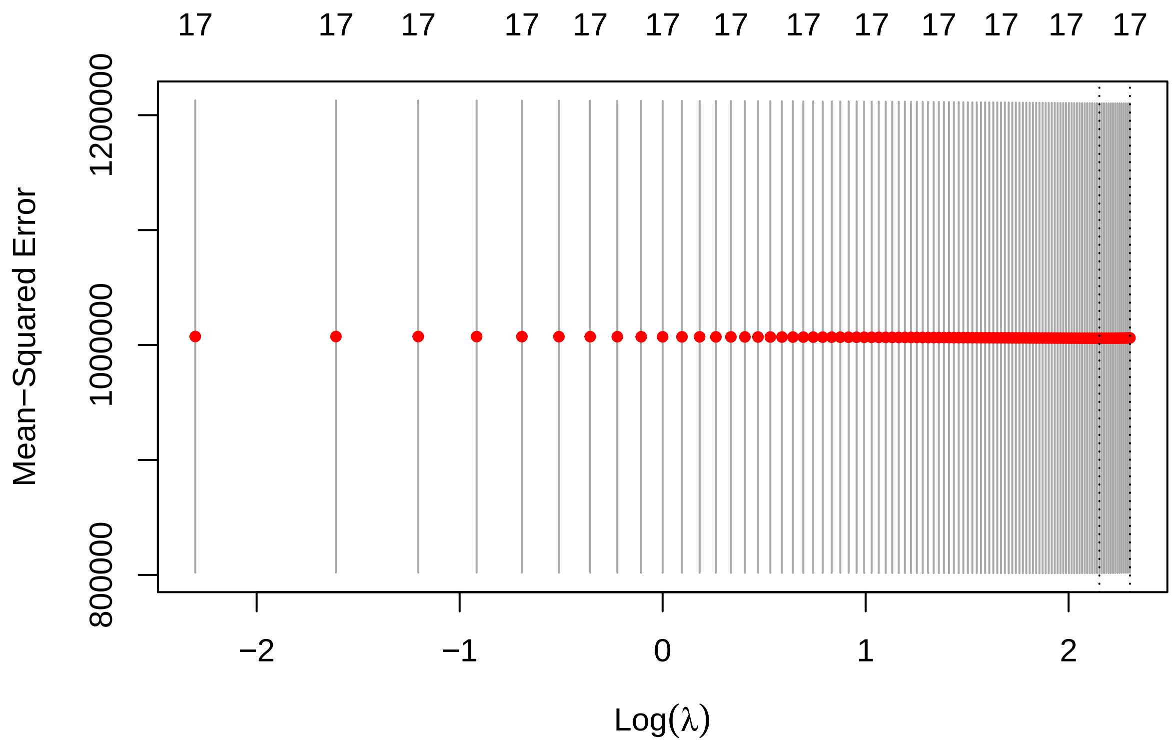
- Select λ chosen by (default 10-fold) cross-validation.

```
college.ridgeCV <- cv.glmnet(x=college.X, y = college.data$Apps, alpha=0,
                             lambda=seq(10, 0, by=-0.1))
college.ridgeCV
```

```
##
## Call:  cv.glmnet(x = college.X, y = college.data$Apps, lambda = seq(10,      0, by = -0.1), alpha = (
##
## Measure: Mean-Squared Error
##
##     Lambda Index Measure     SE Nonzero
## min    8.6    15 1005968 204425      17
## 1se   10.0     1 1006195 204191      17
```

- Plot the results of the cross-validation.

```
plot(college.ridgeCV)
```



- Report the esimated MSE of the model based on your selected λ

```
college.ridgeCV
```

```
##
## Call:  cv.glmnet(x = college.X, y = college.data$Apps, lambda = seq(10,      0, by = -0.1), alpha = 0
##
## Measure: Mean-Squared Error
##
##     Lambda Index Measure     SE Nonzero
## min    8.6    15 1005968 204425      17
## 1se   10.0     1 1006195 204191      17
```

estimated MSE is 1005968.

(d) Consider the **full** model with all 17 predictors (reminder: use data frame college.data you recreated at the beginning). Use functions in package glmnet to fit a LASSO regression.

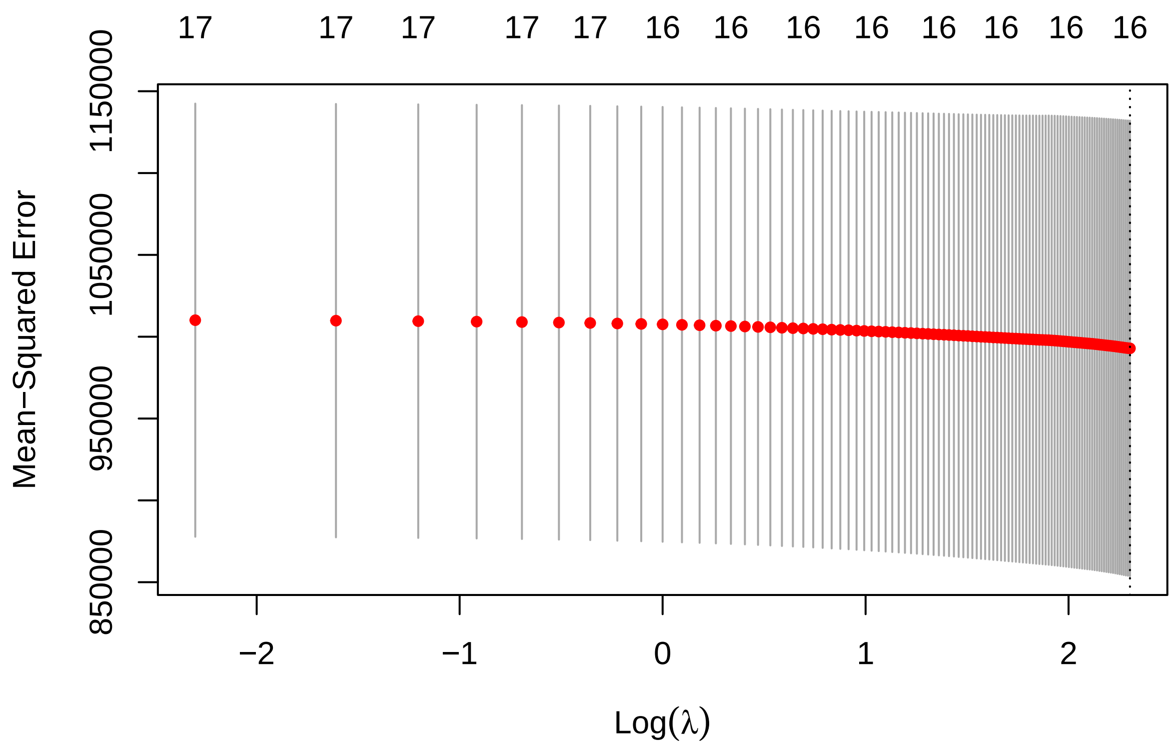
```
college.LASSOCV <- cv.glmnet(x=college.X, y = college.data$Apps, alpha=1,
                             lambda=seq(10, 0, by=-0.1))
college.LASSOCV
```

```
##
## Call:  cv.glmnet(x = college.X, y = college.data$Apps, lambda = seq(10,      0, by = -0.1), alph
##
```

```
## Measure: Mean-Squared Error
##
##     Lambda Index Measure     SE Nonzero
## min     10     1  992888 139144      16
## 1se     10     1  992888 139144      16
```

- Select λchosen by (default 10-fold) cross-validation.
- Plot the results of the cross-validation.

```r
plot(college.LASSOCV)
```



- Report the esimated MSE of the model based on your selected λ.

```r
college.LASSOCV
```

```
##
## Call:  cv.glmnet(x = college.X, y = college.data$Apps, lambda = seq(10,      0, by = -0.1), alpha = 
##
## Measure: Mean-Squared Error
##
##     Lambda Index Measure     SE Nonzero
## min     10     1  992888 139144      16
## 1se     10     1  992888 139144      16
```

estimated MSE is 992888

(e) Fit a PCR model on college.data, with M (the number of principal components) chosen by cross validation. Prepare a validation plot. Report the estimated test error (MSE), along with the value of M selected by cross-validation.

```r
library(pls)
```

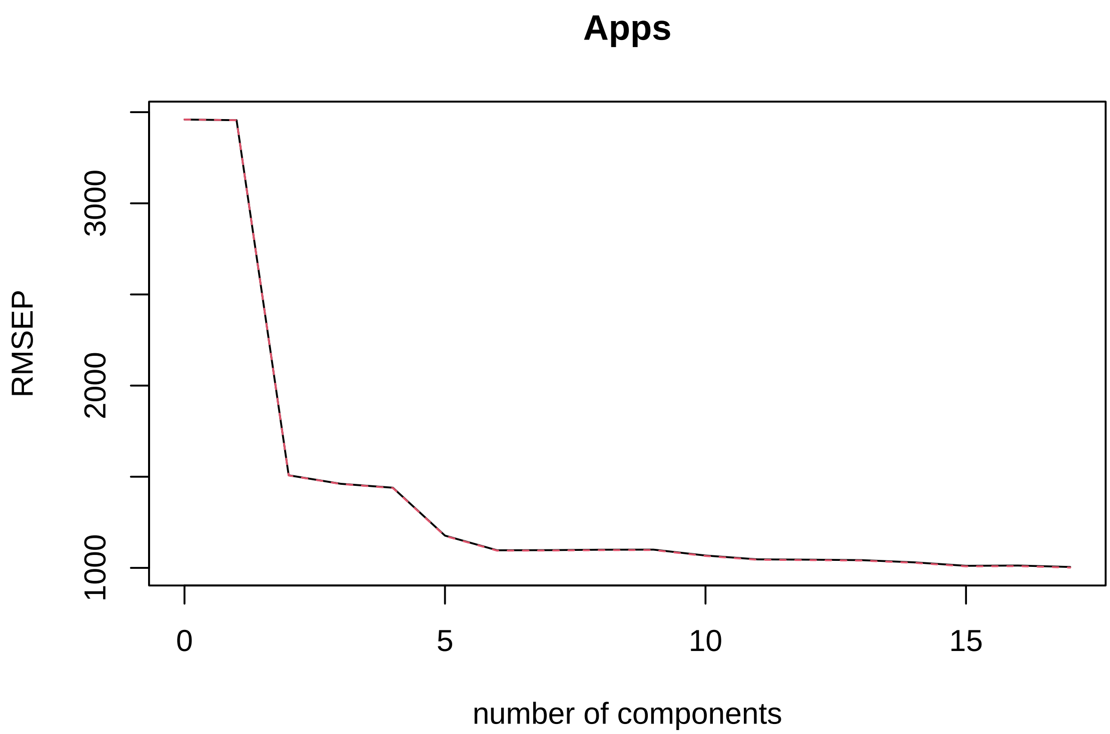
```
##
## Attaching package: 'pls'

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

```r
pcr_collegeCV <- pcr(Apps ~ ., data = college.data, scale = "TRUE", validation = "CV")
summary(pcr_collegeCV)
```

```
## Data:    X dimension: 605 17
##  Y dimension: 605 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            3459     3456     1508     1461     1440     1177     1096
## adjCV         3459     3456     1507     1460     1438     1175     1095
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV        1097     1100     1100      1068      1047      1045      1042
## adjCV     1095     1097     1098      1065      1044      1043      1040
##        14 comps  15 comps  16 comps  17 comps
## CV         1030      1011      1013      1005
## adjCV      1028      1008      1010      1002
##
## TRAINING: % variance explained
##       1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X     50.8029    87.69    95.72    97.84    98.79    99.47    99.92    99.97
## Apps   0.6885    81.35    82.47    83.16    89.10    90.46    90.52    90.59
##       9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X      100.00    100.00    100.00    100.00    100.00    100.00    100.00
## Apps    90.61     91.18     91.55     91.59     91.66     91.88     92.34
##       16 comps  17 comps
## X       100.00    100.00
## Apps     92.36     92.51
```

```r
validationplot(pcr_collegeCV)
```



estimated MSE $=$ RMSEP^2 $= 1013^2 = 1026169$

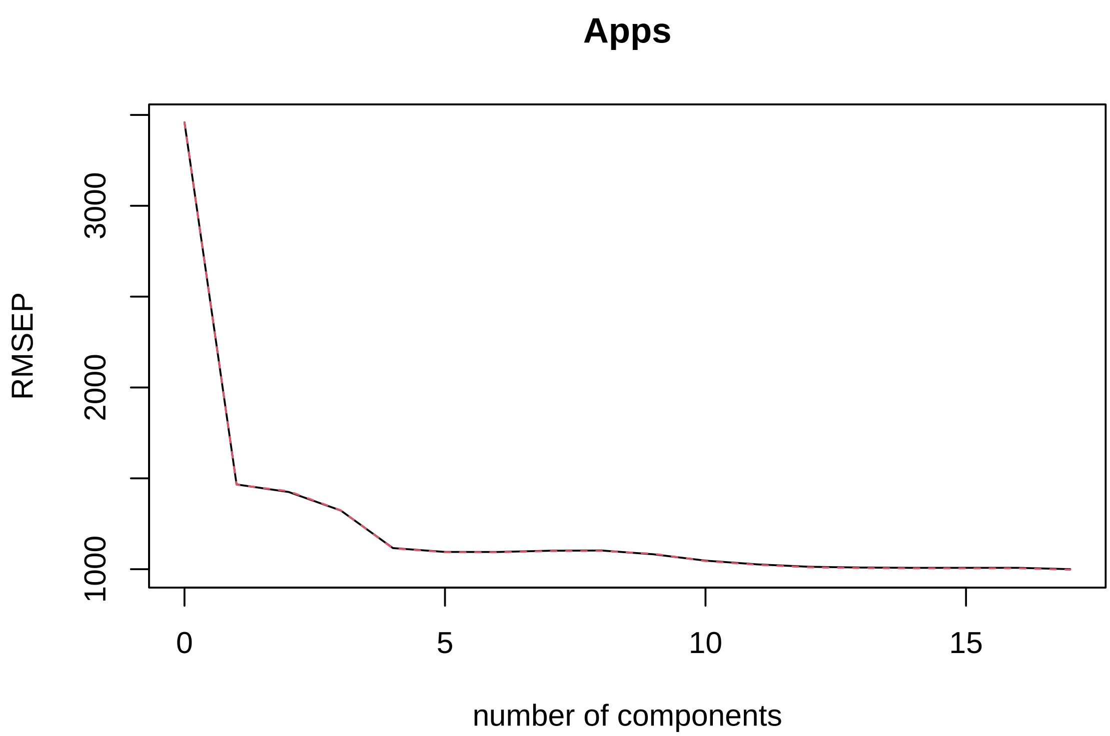
(f) Fit a PLS (partial least squares) model on college.data, with M (the number of principal components) chosen by cross validation. Prepare a validation plot. Report the estimated test error (MSE), along with the value of M selected by cross-validation.

```
college_plsCV <- plsr(Apps ~ ., data = college.data, scale = "TRUE", validation = "CV")
summary(college_plsCV)
```

```
## Data:    X dimension: 605 17
##  Y dimension: 605 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            3459     1466     1425     1323     1116     1095     1095
## adjCV         3459     1465     1429     1324     1115     1093     1093
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV        1101     1103     1082      1047      1026      1013      1009
## adjCV     1099     1100     1085      1045      1024      1010      1006
##        14 comps  15 comps  16 comps  17 comps
## CV         1007      1007      1008    1000.1
## adjCV      1005      1005      1005     997.1
##
## TRAINING: % variance explained
##       1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
```

```
## X       37.09    79.99    94.31    96.92    98.72    99.44    99.75    99.97
## Apps    82.12    83.20    85.88    90.03    90.50    90.55    90.59    90.64
##       9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X      100.00    100.00    100.00    100.00    100.00    100.00    100.00
## Apps    90.75     91.62     91.97     92.31     92.35     92.36     92.36
##       16 comps  17 comps
## X       100.00    100.00
## Apps     92.37     92.51
```

```r
validationplot(college_plsCV)
```



estimated MSE $= 1008^2 = 1016064$

(g) Summarize and comment on the results obtained from the following models. Recommend a model, and justify your choice.

Method	Number of predictors	Estimated Prediction MSE
Least Squares 1: model with the smallest AIC	12	997934.5
Least Squares 2: model with the smallest BIC	7	980591.9
Ridge Regression (lambda.min)	17	1005968
Lasso (lambda.min)	16	992888
Lasso (lambda.1se)	16	992888
PCR PLS	15	1016064

Based on the summarized results, the model with the smallest BIC (Least Squares 2), which includes 7 predictors, is recommended. This model achieves an estimated prediction MSE of 980,591.9, indicating superior predictive performance compared to the other models evaluated. BIC's preference for parsimony aligns well here, emphasizing a balance between model complexity and predictive accuracy. While ridge regression and lasso offer regularization and feature selection benefits, the chosen least squares model provides a straightforward and effective choice, balancing predictive power with model simplicity. Adjustments may be considered based on specific criteria such as interpretability or computational efficiency, but for overall predictive performance based on the given metrics, Least Squares 2 stands out as the optimal choice.

(h) Apply the above models to the hold-out data holdout.data that we created at the beginning. Which model wins this contest in terms of prediction accuracy? (This should be the first time you use observations in holdout.data data frame.)

```
X_holdout <- as.matrix(holdout.data[, -1])
y_holdout <- holdout.data$Apps

AIC_college_2 <- glm(Apps ~ Private + Accept + Enroll + Top10perc + Top25perc + F.Undergrad + Outstate +

BIC_college_2 <- glm(Apps ~ Private + Accept + Top10perc + Outstate + Room.Board + PhD + Expend, data =

cv.glm(holdout.data, glmfit = AIC_college_2, K = 10)$delta[1]
```

```
## [1] 1335918
```

```
cv.glm(holdout.data, BIC_college_2, K = 10)$delta[1]
```

```
## [1] 1420287
```

```
college.ridgeCV2 <- cv.glmnet(x = X_holdout, y = y_holdout, alpha = 0, lambda = seq(10, 0, by = -0.1))
college.ridgeCV2
```

```
##
## Call:  cv.glmnet(x = X_holdout, y = y_holdout, lambda = seq(10, 0, by = -0.1),      alpha = 0)
##
## Measure: Mean-Squared Error
##
##     Lambda Index Measure     SE Nonzero
## min    0.0   101   74.35  67.64      17
## 1se    0.4    97  138.80 121.63      17
```

```
college.LASSOCV2 <- cv.glmnet(x = X_holdout, y = y_holdout, alpha = 1, lambda = seq(10, 0, by=-0.1))
college.LASSOCV2
```

```
##
## Call:  cv.glmnet(x = X_holdout, y = y_holdout, lambda = seq(10, 0, by = -0.1),      alpha = 1)
##
## Measure: Mean-Squared Error
##
##     Lambda Index   Measure        SE Nonzero
## min      0   101 3.627e-25 9.777e-26      17
## 1se      0   101 3.627e-25 9.777e-26      17
```

```r
college_plsCV2 <- plsr(Apps ~ ., data = holdout.data, scale = "TRUE", validation = "CV")
summary(college_plsCV2)
```

```
## Data:    X dimension: 171 17
##  Y dimension: 171 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            3739     1433     1389     1338     1214     1226     1235
## adjCV         3739     1423     1275     1321     1205     1215     1224
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV        1257     1267     1262      1232      1233      1233      1231
## adjCV     1244     1254     1253      1218      1218      1218      1217
##        14 comps  15 comps  16 comps  17 comps
## CV         1258      1269      1276      1250
## adjCV      1241      1252      1258      1232
##
## TRAINING: % variance explained
##       1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X       45.42    50.88    89.12    97.29    98.32    99.31    99.47    99.96
## Apps    86.94    90.53    91.00    91.99    92.19    92.20    92.27    92.28
##       9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X      100.00    100.00    100.00    100.00    100.00    100.00    100.00
## Apps    92.35     93.12     93.27     93.36     93.43     93.48     93.52
##       16 comps  17 comps
## X       100.00     100.0
## Apps     93.53      93.9
```

(h) AIC MSE = 1398325, BIC MSE = 1365042, ridge MSE = 21.69, LASSO MSE = 3.817^{-25} , PLS MSE = 1228ˆ2 = 1507984. LASSO wins the contest by far.

(i) **Stat-627** Compare your estimated prediction MSE from the training data college.data (part i) and the resulting MSE from the holdout.data (part j). Is there anything “surprising” that worth investigation? If yes, what are the possible causes? (Note. It is not surprising to see a tuned “best” model not to perform the best on the testing data.)

Yes, the prediction MSEs for ridge regression and LASSO are considerably smaller than all the other prediction MSEs as well as all of the training MSEs. This is for a couple reasons: first, the shrinkage methods provide a better bias-variance tradeoff by shrinking the coefficients (thus reducing bias) and significantly reducing variance. This is consistent with the fact that shrinkage methods in general tend to make models less complex by shrinking the coefficients to almost 0 (ridge) or even 0 (LASSO). The smaller prediction MSEs observed indicate that these regularization techniques have effectively optimized the model complexity for improved prediction accuracy on the holdout data. The MSEs are also so much smaller because of the use of cross validation, where the lambda parameter (regularization strength) is selected to minimize prediction error on validation data. This ensures that the model is tuned to perform well on unseen data, further contributing to the smaller prediction MSEs observed.