STAT425_Homework8

Giang Le

11/8/2021

Question 1

1a

(a) The Lasso, relative to Ordinary Least Squares, has (iii) Less flexible and hence will give improved prediction accuracy when its increase in bias is less than its decrease in variance.

The reason why this is so is because Lasso can shrink certain predictors to 0, so it produces a model with less variance (more stable) but more bias.

1b

Ridge regression relative to Ordinary Least Squares has (ii) More flexible and hence will give improved prediction accuracy when its increase in variance is less than its decrease in bias.

Ridge regression doesn't shrink predictors to 0. As λ increases, bias increases and variance decreases and vice versa, so Ridge regression is more flexible and has a larger variance.

Question 2

First I read in the data, and load the library.

```
library(ISLR)
head(College)
```

```
##                              Private Apps Accept Enroll Top10perc Top25perc
## Abilene Christian University     Yes 1660   1232    721        23        52
## Adelphi University               Yes 2186   1924    512        16        29
## Adrian College                   Yes 1428   1097    336        22        50
## Agnes Scott College              Yes  417    349    137        60        89
## Alaska Pacific University        Yes  193    146     55        16        44
## Albertson College                Yes  587    479    158        38        62
##                              F.Undergrad P.Undergrad Outstate Room.Board Books
## Abilene Christian University        2885         537     7440       3300   450
## Adelphi University                  2683        1227    12280       6450   750
## Adrian College                      1036          99    11250       3750   400
## Agnes Scott College                  510          63    12960       5450   450
## Alaska Pacific University            249         869     7560       4120   800
## Albertson College                    678          41    13500       3335   500
##                              Personal PhD Terminal S.F.Ratio perc.alumni Expend
## Abilene Christian University     2200  70       78      18.1          12   7041
## Adelphi University               1500  29       30      12.2          16  10527
## Adrian College                   1165  53       66      12.9          30   8735
## Agnes Scott College               875  92       97       7.7          37  19016
## Alaska Pacific University        1500  76       72      11.9           2  10922
```

```
## Albertson College                675  67       73       9.4          11   9727
##                               Grad.Rate
## Abilene Christian University         60
## Adelphi University                   56
## Adrian College                       54
## Agnes Scott College                  59
## Alaska Pacific University            15
## Albertson College                    55
```

2a

(a)Split the data set in two parts: a training and a testing data set. Choose 30% of the data at random for testing, and use the remaining for training. (Fix the seed to 425). Define the response rate as Rate = Accept/Apps. Plot this variable against every variable in the data set. In the remaining questions use the Rate variable as the response and the other variables as predictors. Do not forget to remove the Accept and Apps variables from the list of predictors.

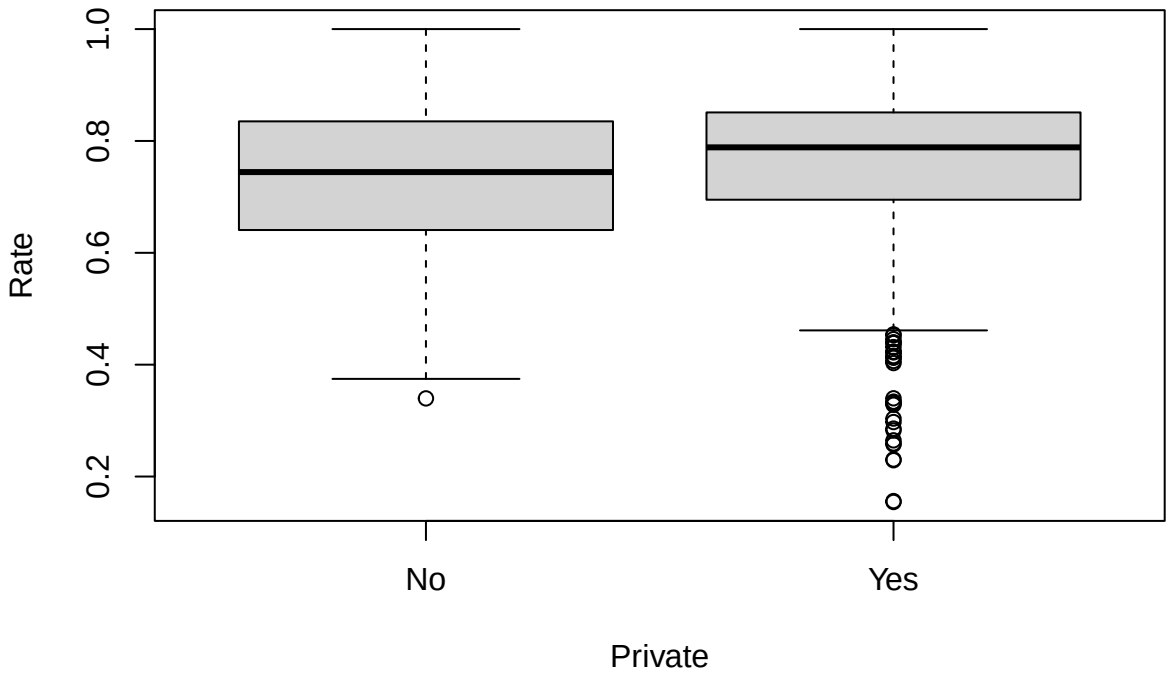
```
set.seed(425)
College$Rate <- College$Accept/College$Apps
n <- nrow(College)
ntrain <- round(n*0.70)  # 70% for training set
tindex <- sample(n, ntrain)
train <- College[tindex,-c(2,3)]
test <- College[-tindex,-c(2,3)]
dim(train)
```

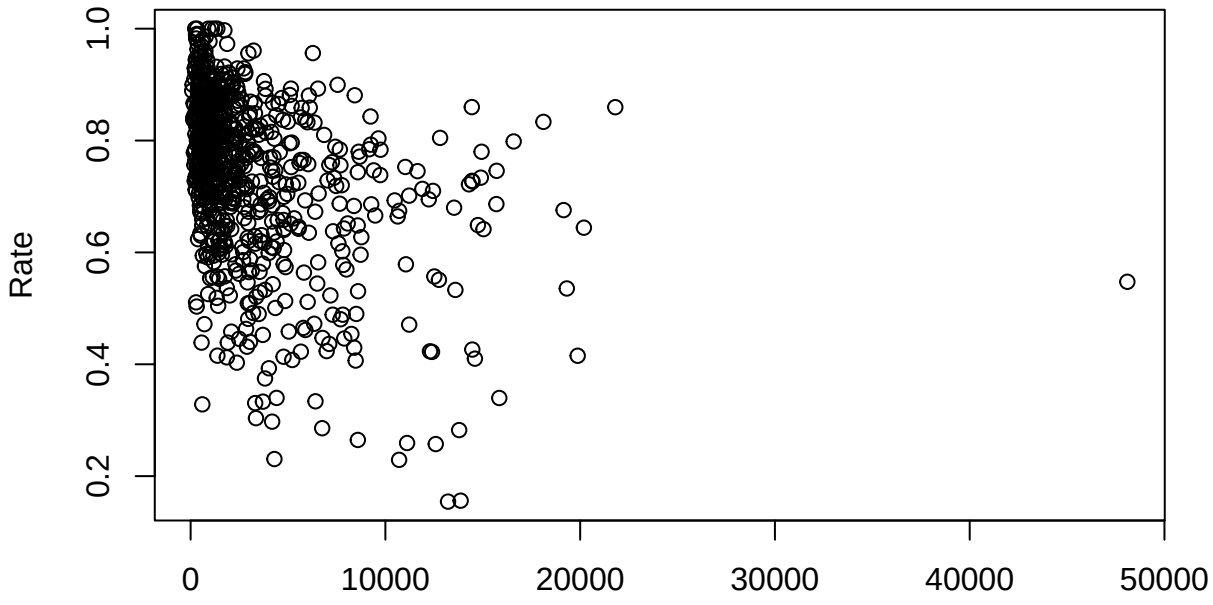
```
## [1] 544  17
```

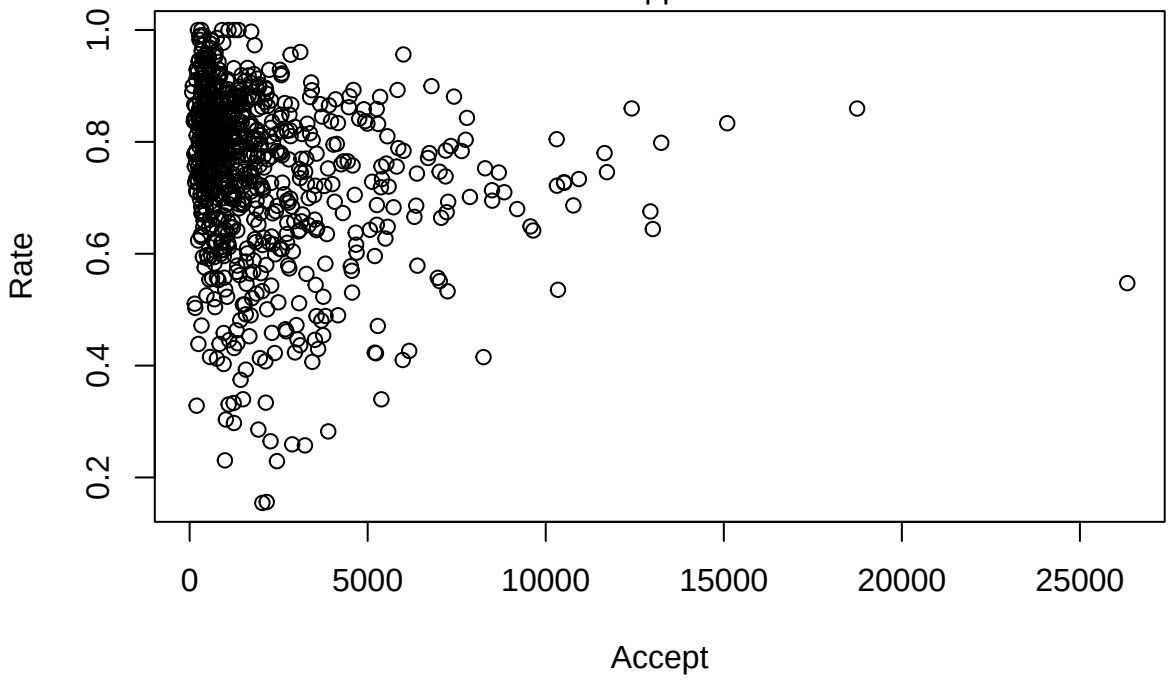
```
dim(test)
```

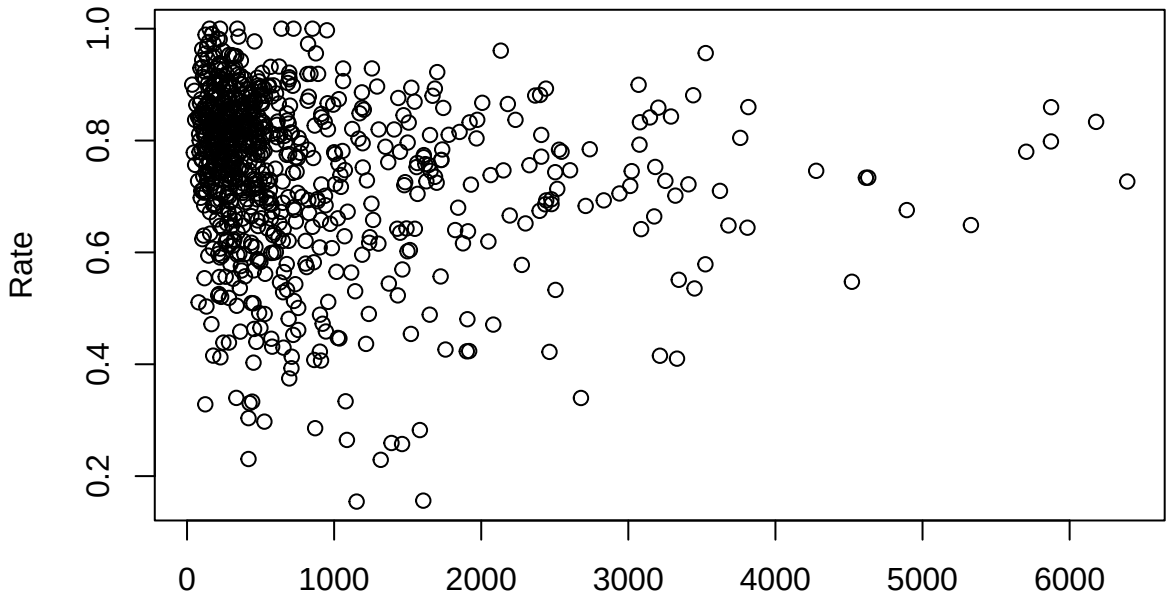
```
## [1] 233  17
```

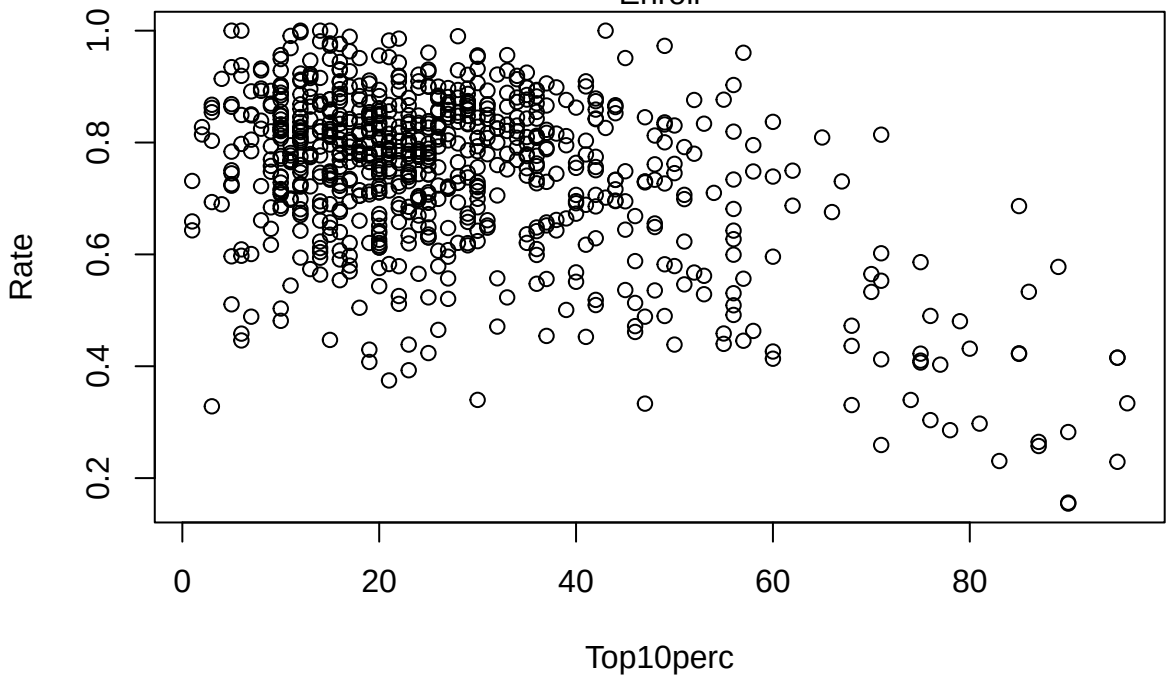
```
# Plot Rate against other variables in the data set.
plot(Rate ~., data=College)
```

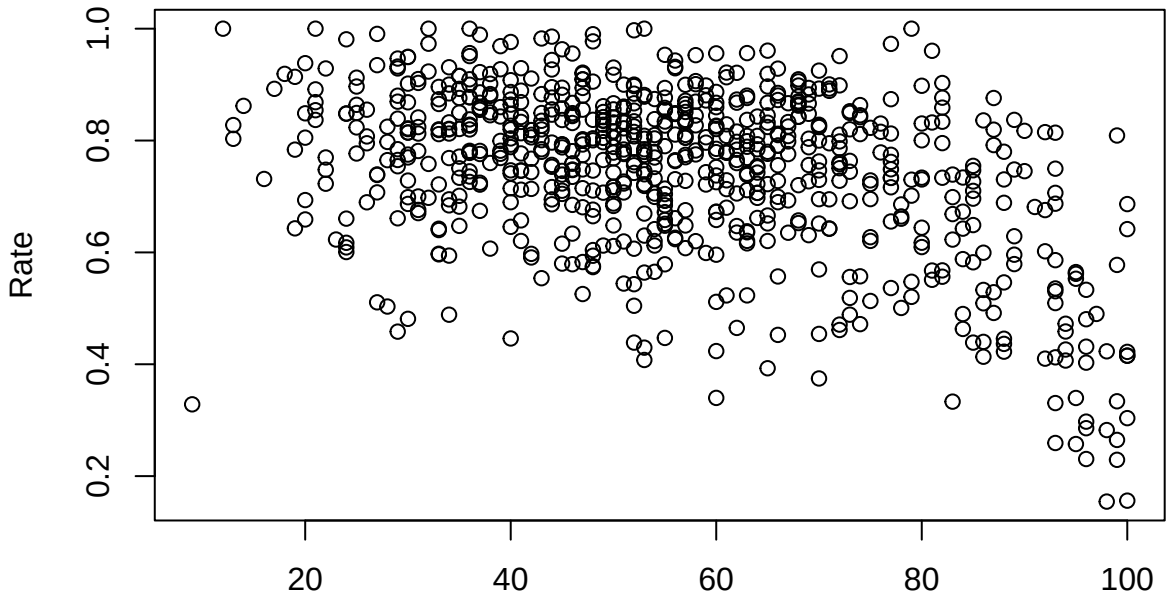


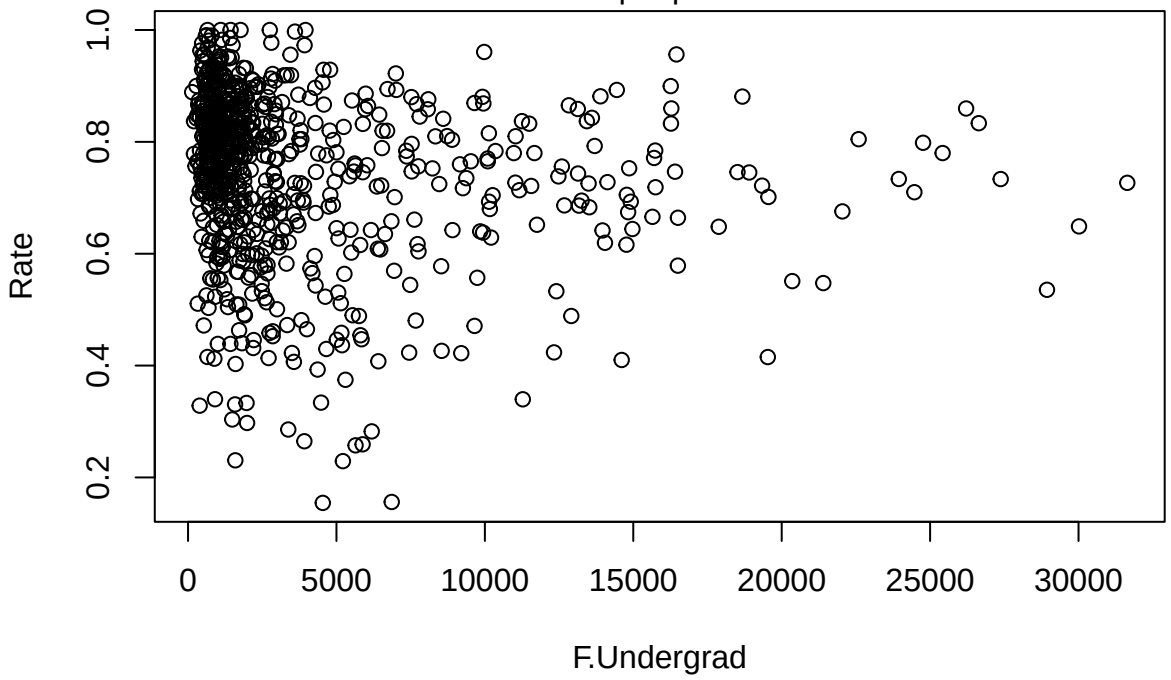

Rate
0.2
0.4
0.6
0.8
1.0
0
10000
20000
30000
40000
50000
Apps

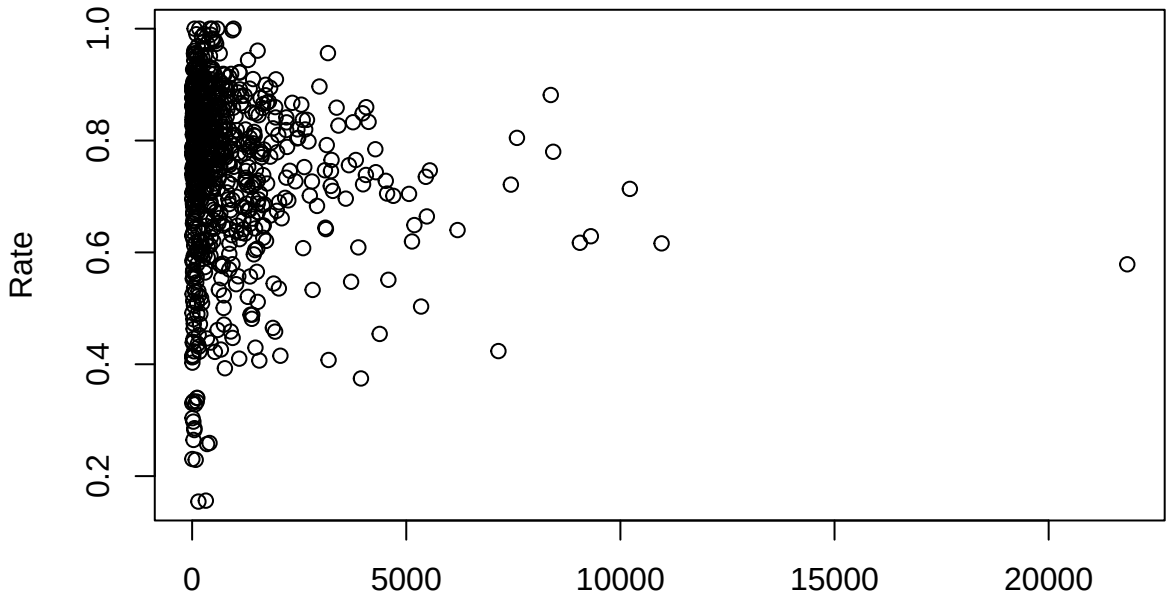

Rate
0.2
0.4
0.6
0.8
1.0
0
5000
10000
15000
20000
25000
Accept

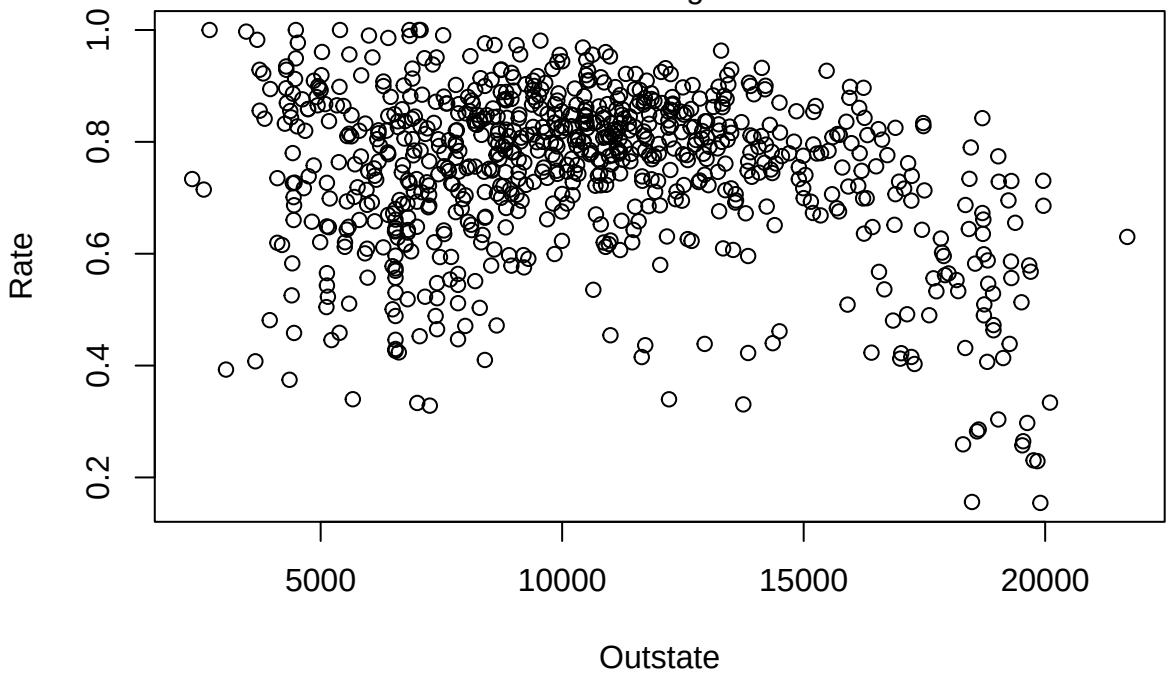

Rate
0.2
0.4
0.6
0.8
1.0
0
1000
2000
3000
4000
5000
6000
Enroll

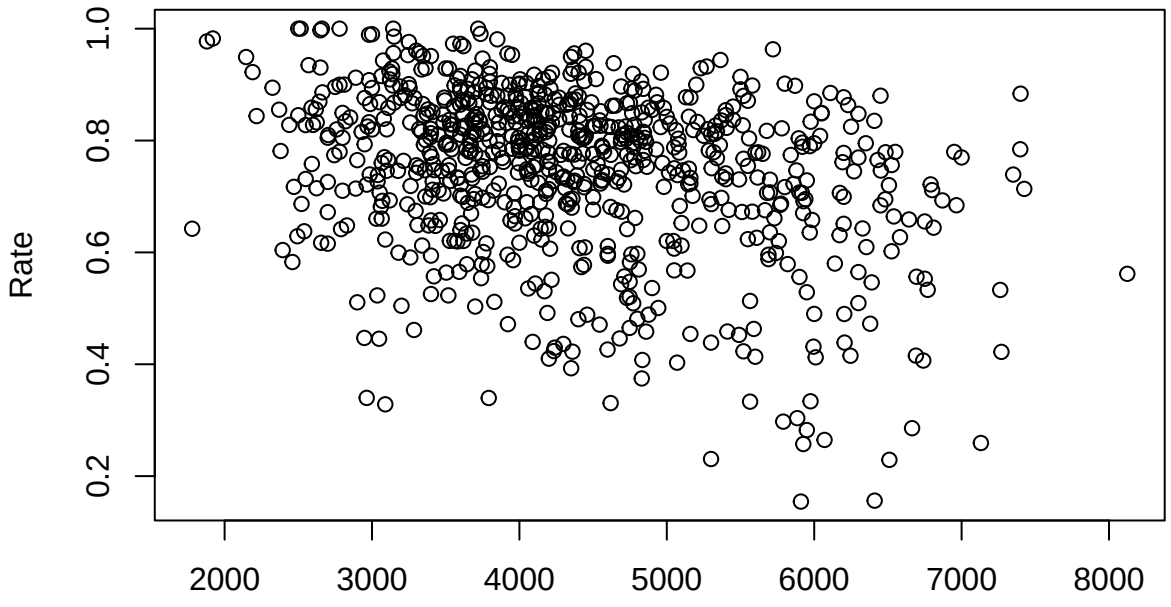

Rate
0.2
0.4
0.6
0.8
1.0
0
20
40
60
80
Top10perc

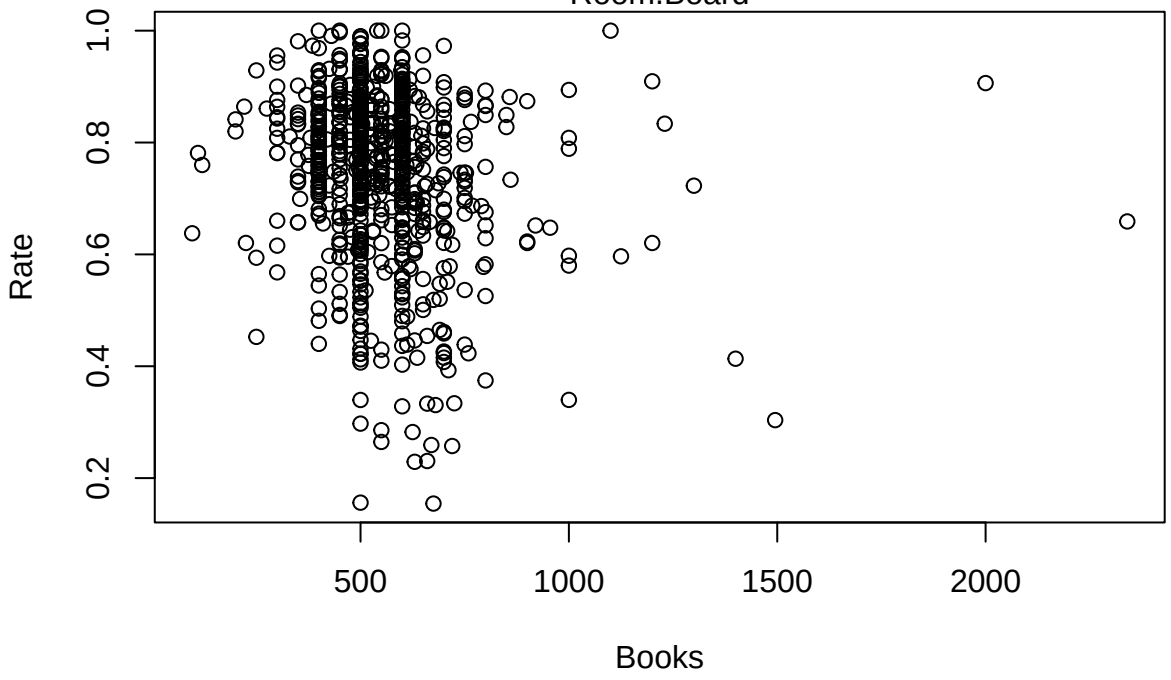

Rate
0.2
0.4
0.6
0.8
1.0
20
40
60
80
100
Top25perc

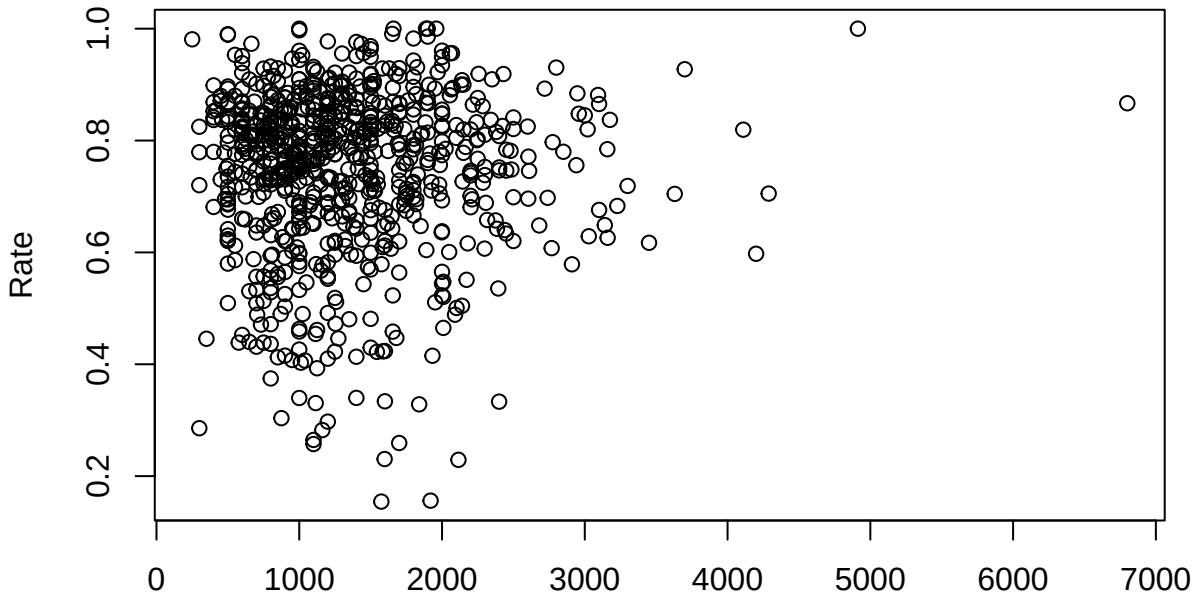

Rate
0.2
0.4
0.6
0.8
1.0
0
5000
10000
15000
20000
25000
30000
F.Undergrad

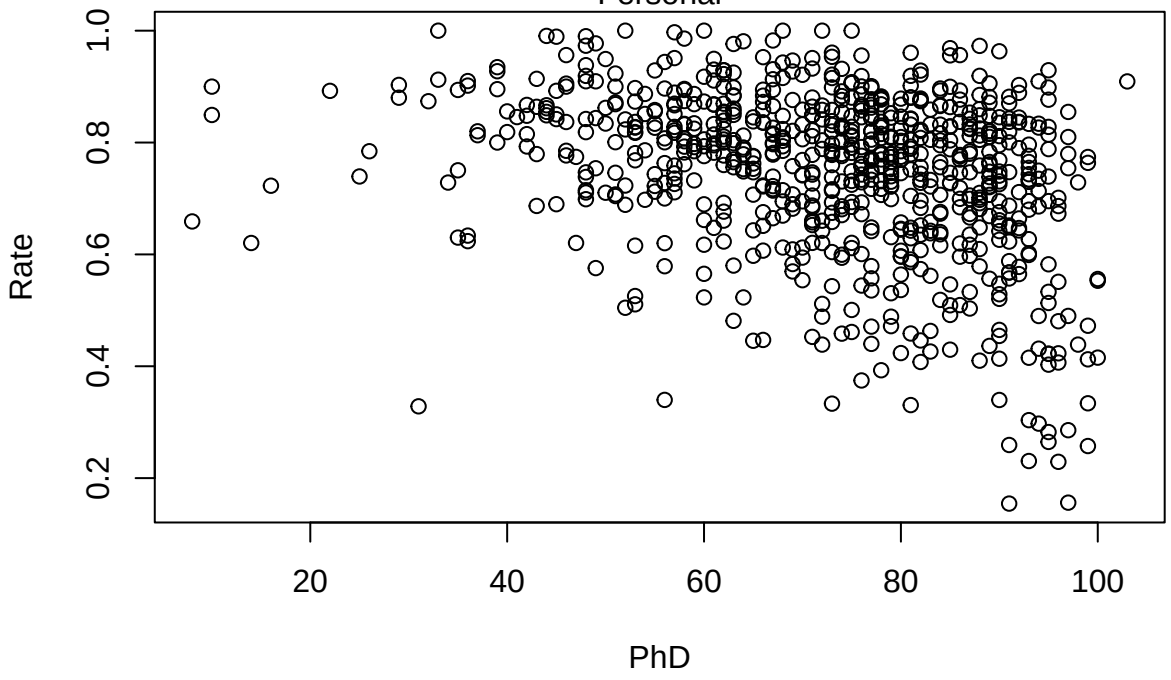

Rate
0.2
0.4
0.6
0.8
1.0
0
5000
10000
15000
20000
P.Undergrad

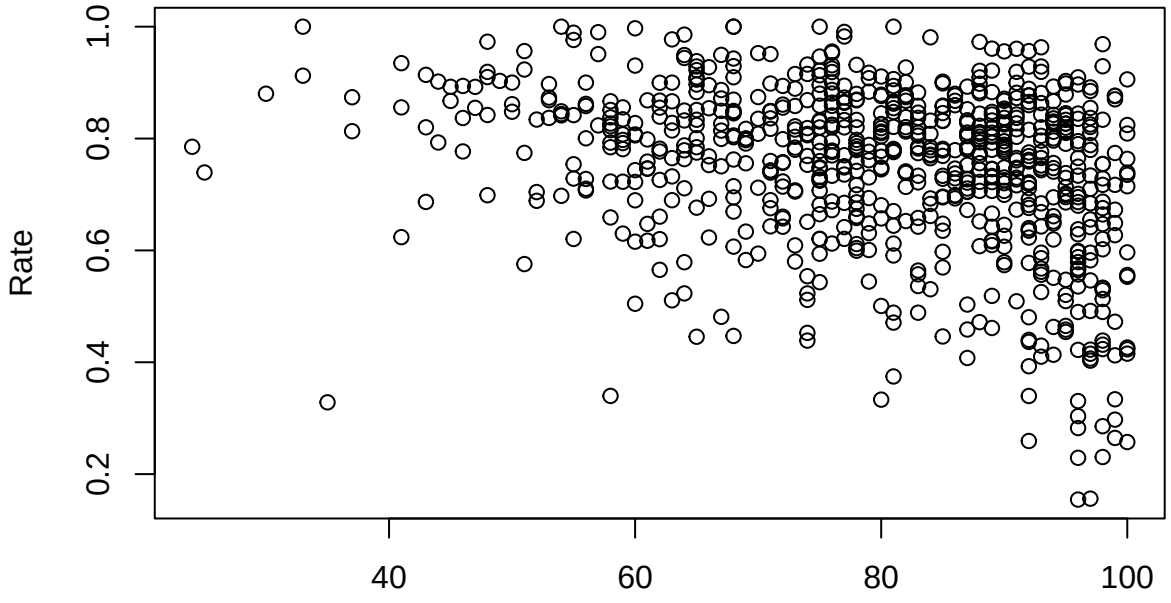

Rate
0.2
0.4
0.6
0.8
1.0
5000
10000
15000
20000
Outstate

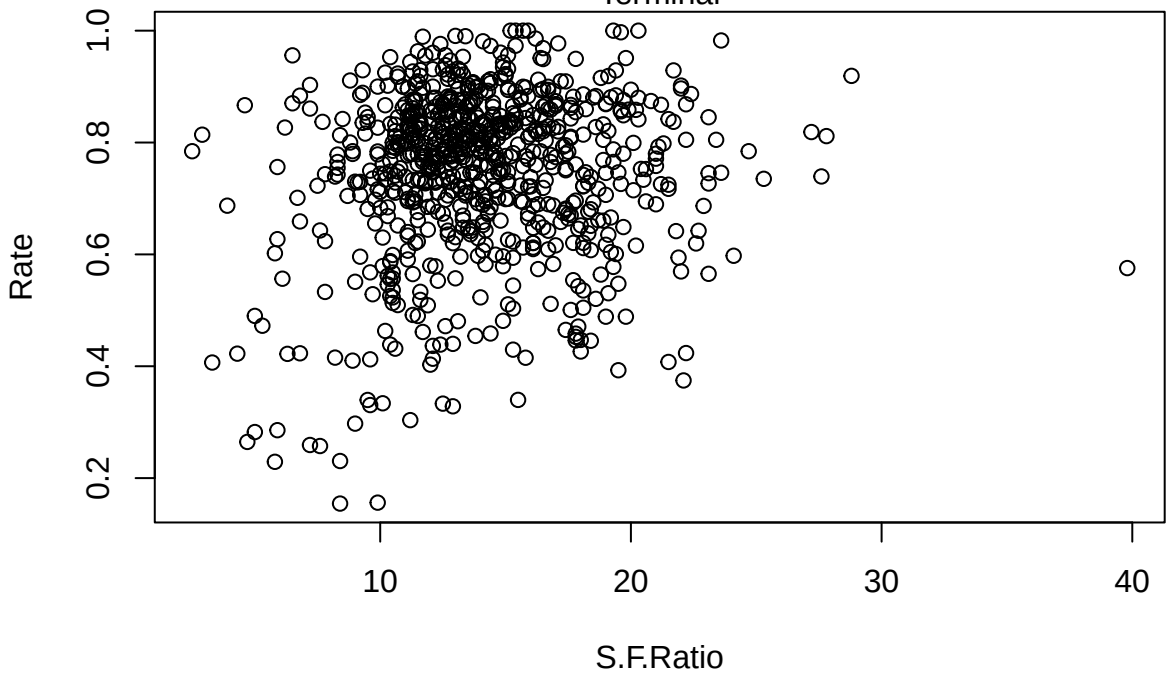

Rate
0.2
0.4
0.6
0.8
1.0
2000
3000
4000
5000
6000
7000
8000
Room.Board

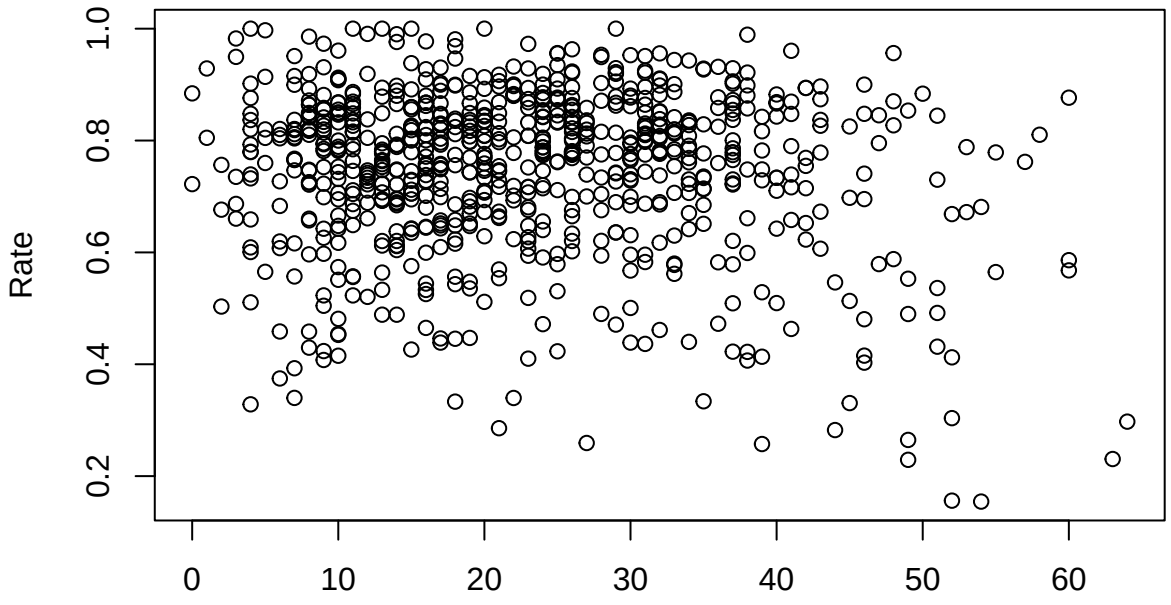

Rate
0.2
0.4
0.6
0.8
1.0
500
1000
1500
2000
Books

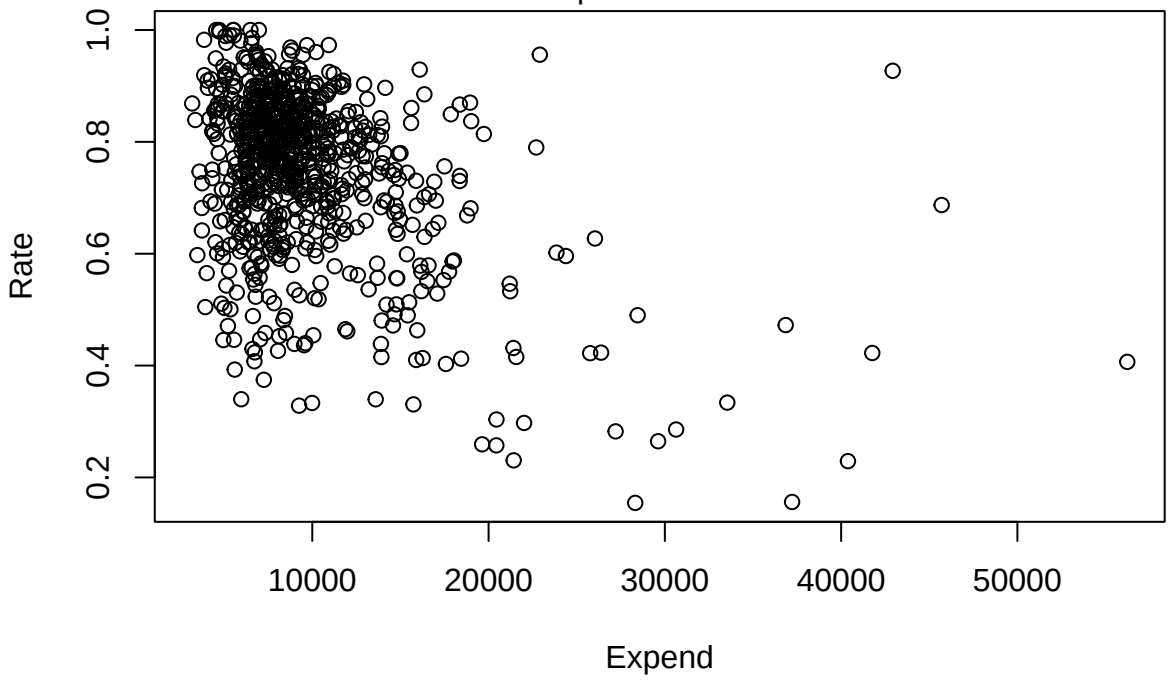

Rate
0.2
0.4
0.6
0.8
1.0
0
1000
2000
3000
4000
5000
6000
7000
Personal

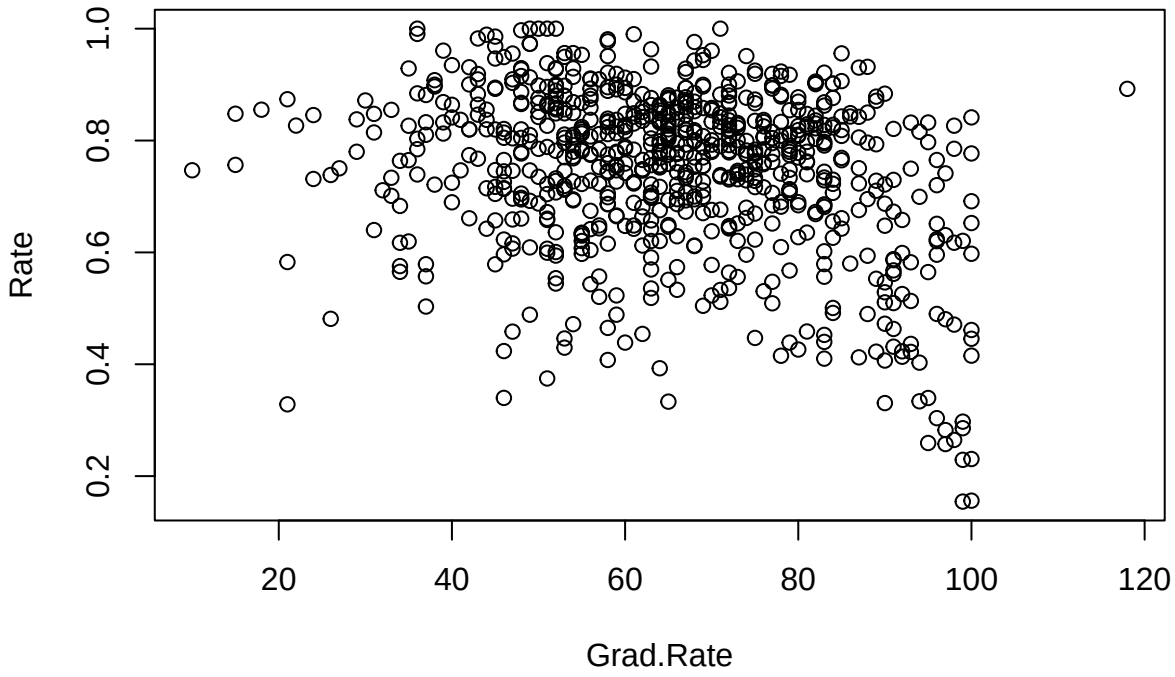

Rate
0.2
0.4
0.6
0.8
1.0
20
40
60
80
100
PhD


Rate
0.2
0.4
0.6
0.8
1.0
40
60
80
100
Terminal


Rate
0.2
0.4
0.6
0.8
1.0
10
20
30
40
S.F.Ratio


Rate
0.2
0.4
0.6
0.8
1.0
0
10
20
30
40
50
60
perc.alumni


Rate
0.2
0.4
0.6
0.8
1.0
10000
20000
30000
40000
50000
Expend



2b

(b) Fit a linear model using least squares on the training set, and report the training and testing error obtained.

```
lm_model <- lm(Rate ~ ., data=train)
summary(lm_model)
```

```
##
## Call:
## lm(formula = Rate ~ ., data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.54015 -0.07271  0.01067  0.08611  0.32045
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.017e+00  5.547e-02  18.328  < 2e-16 ***
## PrivateYes   6.430e-02  1.913e-02   3.361 0.000834 ***
## Enroll       5.499e-05  2.383e-05   2.307 0.021416 *
## Top10perc   -3.411e-03  7.861e-04  -4.339 1.72e-05 ***
## Top25perc    1.845e-04  6.488e-04   0.284 0.776184
## F.Undergrad -6.529e-06  4.815e-06  -1.356 0.175728
## P.Undergrad -1.155e-05  4.219e-06  -2.738 0.006397 **
## Outstate     4.223e-06  2.596e-06   1.627 0.104343
## Room.Board  -1.934e-05  7.012e-06  -2.759 0.006004 **
## Books       -1.178e-04  3.283e-05  -3.589 0.000363 ***
## Personal     1.087e-05  8.654e-06   1.257 0.209488
## PhD          1.820e-04  6.374e-04   0.286 0.775339
## Terminal     4.305e-04  7.035e-04   0.612 0.540826
## S.F.Ratio   -4.232e-03  1.713e-03  -2.471 0.013791 *
## perc.alumni  6.123e-04  5.622e-04   1.089 0.276605
```

```
## Expend      -7.653e-06  1.677e-06  -4.565 6.24e-06 ***
## Grad.Rate   -1.204e-03  4.118e-04  -2.925 0.003597 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1222 on 527 degrees of freedom
## Multiple R-squared:  0.352,  Adjusted R-squared:  0.3323
## F-statistic: 17.89 on 16 and 527 DF,  p-value: < 2.2e-16
```

```
# Root mean squared error of training data. (0.120322)
rmse<-function(x,y) sqrt(mean((x-y)^2))
rmse(fitted(lm_model), train$Rate)
```

```
## [1] 0.120322
```

```
# Root mean squared error of testing data. (0.1138004)
rmse(predict(lm_model,test), test$Rate)
```

```
## [1] 0.1138004
```

2c

(c) Use AIC, BIC, and Adjusted R2 to select a potentially smaller model , from the set of all possible predictors used in the previous question. Report which model each method chose, and the training and testing errors for their chosen model(s).

```
# Do variable selection with leaps.
library(leaps)

b=regsubsets(Rate ~., data = train)
n=dim(train)[1]
msize = 1:16
rs = summary(b)

# Models tested
rs$which
```

```
##   (Intercept) PrivateYes Enroll Top10perc Top25perc F.Undergrad P.Undergrad
## 1        TRUE      FALSE  FALSE      TRUE     FALSE       FALSE       FALSE
## 2        TRUE      FALSE  FALSE      TRUE     FALSE       FALSE        TRUE
## 3        TRUE       TRUE  FALSE      TRUE     FALSE       FALSE       FALSE
## 4        TRUE       TRUE  FALSE      TRUE     FALSE       FALSE       FALSE
## 5        TRUE       TRUE  FALSE      TRUE     FALSE       FALSE       FALSE
## 6        TRUE       TRUE  FALSE      TRUE     FALSE       FALSE       FALSE
## 7        TRUE       TRUE   TRUE      TRUE     FALSE       FALSE       FALSE
## 8        TRUE       TRUE   TRUE      TRUE     FALSE       FALSE        TRUE
##   Outstate Room.Board Books Personal   PhD Terminal S.F.Ratio perc.alumni
## 1    FALSE      FALSE FALSE    FALSE FALSE    FALSE     FALSE       FALSE
## 2    FALSE      FALSE FALSE    FALSE FALSE    FALSE     FALSE       FALSE
## 3    FALSE       TRUE FALSE    FALSE FALSE    FALSE     FALSE       FALSE
## 4    FALSE      FALSE  TRUE    FALSE FALSE    FALSE     FALSE       FALSE
## 5    FALSE       TRUE  TRUE    FALSE FALSE    FALSE     FALSE       FALSE
## 6    FALSE       TRUE  TRUE    FALSE FALSE    FALSE      TRUE       FALSE
## 7    FALSE       TRUE  TRUE    FALSE FALSE    FALSE      TRUE       FALSE
## 8    FALSE      FALSE  TRUE    FALSE FALSE    FALSE      TRUE       FALSE
##   Expend Grad.Rate
## 1  FALSE     FALSE
```

```
## 2  FALSE     FALSE
## 3  FALSE     FALSE
## 4   TRUE     FALSE
## 5   TRUE     FALSE
## 6   TRUE     FALSE
## 7   TRUE     FALSE
## 8   TRUE      TRUE
```

```
# adjusted R^2 values 0.2270797 0.2558832 0.2783246 0.2962523 0.3042864 0.3093118 0.3194597 0.3251439
rs$adjr2
```

```
## [1] 0.2245215 0.2484184 0.2734503 0.2918925 0.3012310 0.3085319 0.3136080
## [8] 0.3219863
```

```
which.max(rs$adjr2)
```

```
## [1] 8
```

```
# BIC values for different models: -128.5280 -143.8937 -155.2599 -163.6539 -164.6113 -163.2682 -166.035
rs$bic
```

```
## [1] -126.7305 -138.4636 -151.5980 -160.2941 -162.2274 -162.6542 -161.3775
## [8] -162.7756
```

```
which.min(rs$bic)
```

```
## [1] 8
```

```
# Aic
Aic = n*log(rs$rss/n) + 2*msize;
which.min(Aic)
```

```
## [1] 8
```

```
names(rs$which[8,])[which(rs$which[8,]==T)]
```

```
## [1] "(Intercept)" "PrivateYes"  "Enroll"      "Top10perc"   "P.Undergrad"
## [6] "Books"       "S.F.Ratio"   "Expend"      "Grad.Rate"
```

So by all three criteria, we choose the 8th model.

The 8th model includes the following predictors: PrivateYes, Enroll, Top10perc, P.Undergrad, Books, S.F.Ratio. Expend, Grad.Rate

Here are the training and test errors of model 8. The errors are slightly larger than our full model with all predictors.

```
model_8th <- lm(Rate ~ Private + Enroll + Top10perc + P.Undergrad
                +  Books + S.F.Ratio + Expend + Grad.Rate, data=train)
summary(model_8th)
```

```
##
## Call:
## lm(formula = Rate ~ Private + Enroll + Top10perc + P.Undergrad +
##     Books + S.F.Ratio + Expend + Grad.Rate, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.55424 -0.07651  0.01187  0.08543  0.34397
##
## Coefficients:
```

```
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.052e+00  4.431e-02  23.733  < 2e-16 ***
## PrivateYes   7.050e-02  1.688e-02   4.177 3.45e-05 ***
## Enroll       2.613e-05  7.778e-06   3.359 0.000837 ***
## Top10perc   -2.893e-03  4.421e-04  -6.545 1.40e-10 ***
## P.Undergrad -1.275e-05  4.006e-06  -3.184 0.001539 **
## Books       -1.325e-04  3.164e-05  -4.188 3.29e-05 ***
## S.F.Ratio   -4.918e-03  1.697e-03  -2.898 0.003911 **
## Expend      -7.254e-06  1.495e-06  -4.850 1.62e-06 ***
## Grad.Rate   -1.105e-03  3.791e-04  -2.915 0.003705 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1232 on 535 degrees of freedom
## Multiple R-squared:  0.332,  Adjusted R-squared:  0.322
## F-statistic: 33.23 on 8 and 535 DF,  p-value: < 2.2e-16
```

```
# Train error for model 8 (0.1221655)
rmse(fitted(model_8th), train$Rate)
```

```
## [1] 0.1221655
```

```
# Test error for model 8 (0.1164683)
rmse(predict(model_8th,test), test$Rate)
```

```
## [1] 0.1164683
```

2d Ridge

(d) Fit a ridge regression model on the training set, with λ chosen by 10-fold cross-validation. Report the training, cross-validated, and testing errors.

```
set.seed(425)
# Use 10 fold CV to find lambda.
library(glmnet)
```

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

```
## Loaded glmnet 4.1-2
```

```
cv.ridge <- cv.glmnet(data.matrix(train[,1:16]), train[,17], alpha=0)
bestlam = cv.ridge$lambda.min
bestlam
```

```
## [1] 0.007104901
```

```
# Fit a model with newly found best lambda.
rgmod <- glmnet(train[,1:16], train[,17], alpha=0, lambda = bestlam)
ridge_pred = predict(rgmod, s = bestlam, newx = data.matrix(test[,1:16])) # Use best lambda to predict
rmse(ridge_pred, test$Rate) # Calculate test MSE
```

```
## [1] 0.1133502
```

```
# Calculate the train MSE
ridge_pred1 = predict(rgmod, s = bestlam, newx = data.matrix(train[,1:16]))
rmse(ridge_pred1, train$Rate)
```

```
## [1] 0.1206716
```

```
# Smallest mean cross validated error
cv.ridge$cvm[which.min(cv.ridge$cvm)]
```

```
## [1] 0.01538668
```

2e Lasso

(e) Fit a lasso model on the training set, with λ chosen by 10-fold cross-validation. Report which variables are included in the model, and the training, cross- validated, and testing errors.

```
cv.las <- cv.glmnet(data.matrix(train[,1:16]), train[,17], alpha=1)
bestlam.las = cv.las$lambda.min
bestlam.las
```

```
## [1] 0.001079882
```

```
# Fit a model with newly found best lambda.
las.mod <- glmnet(train[,1:16], train[,17], alpha=1, lambda = bestlam.las)
las.pred = predict(las.mod, s = bestlam.las, newx = data.matrix(test[,1:16])) # Use best lambda to pred
rmse(las.pred, test$Rate) # Calculate test MSE
```

```
## [1] 0.1134402
```

```
# Calculate the train MSE
las.pred1 = predict(las.mod, s = bestlam.las, newx = data.matrix(train[,1:16]))
rmse(las.pred1, train$Rate)
```

```
## [1] 0.1206919
```

```
# Smallest mean cross validated error
cv.las$cvm[which.min(cv.las$cvm)]
```

```
## [1] 0.01567434
```

2f PCR

(f) Fit a PCR 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

```
# Check correlation between variables
cor(data.matrix(train[,1:16]))
```

```
##                  Private       Enroll   Top10perc   Top25perc  F.Undergrad
## Private      1.00000000 -0.577967328  0.15325799  0.08088110 -0.621383166
## Enroll      -0.57796733  1.000000000  0.19046890  0.23464493  0.968844866
## Top10perc    0.15325799  0.190468904  1.00000000  0.89542622  0.149840633
## Top25perc    0.08088110  0.234644926  0.89542622  1.00000000  0.215072653
## F.Undergrad -0.62138317  0.968844866  0.14984063  0.21507265  1.000000000
## P.Undergrad -0.48028970  0.486209741 -0.11909442 -0.06576051  0.552841537
## Outstate     0.55522223 -0.187331444  0.55247002  0.47757132 -0.242028779
## Room.Board   0.33787601 -0.079963867  0.38669681  0.35846715 -0.100088482
## Books       -0.01727646  0.101842523  0.13690158  0.12672873  0.106113636
## Personal    -0.26753888  0.253702182 -0.08939677 -0.08502618  0.297250608
## PhD         -0.14449249  0.337217823  0.55451554  0.57263730  0.322874759
## Terminal    -0.12087101  0.312622884  0.51204846  0.54768652  0.304715525
## S.F.Ratio   -0.45017444  0.226229990 -0.37727547 -0.28847723  0.267702954
## perc.alumni  0.42031342 -0.169932256  0.45532374  0.41395463 -0.217099138
## Expend       0.25185835  0.046500671  0.66607533  0.52956506  0.004104707
## Grad.Rate    0.32946388 -0.003996233  0.51480716  0.50302764 -0.068072679
```

```
##             P.Undergrad    Outstate  Room.Board       Books     Personal
## Private     -0.48028970  0.55522223  0.33787601 -0.01727646 -0.267538882
## Enroll       0.48620974 -0.18733144 -0.07996387  0.10184252  0.253702182
## Top10perc   -0.11909442  0.55247002  0.38669681  0.13690158 -0.089396774
## Top25perc   -0.06576051  0.47757132  0.35846715  0.12672873 -0.085026185
## F.Undergrad  0.55284154 -0.24202878 -0.10008848  0.10611364  0.297250608
## P.Undergrad  1.00000000 -0.29299106 -0.13240202  0.06771817  0.314900159
## Outstate    -0.29299106  1.00000000  0.65818397  0.02994607 -0.287768282
## Room.Board  -0.13240202  0.65818397  1.00000000  0.14986864 -0.197403614
## Books        0.06771817  0.02994607  0.14986864  1.00000000  0.163822293
## Personal     0.31490016 -0.28776828 -0.19740361  0.16382229  1.000000000
## PhD          0.13847590  0.39769391  0.36700450  0.01813714 -0.006682589
## Terminal     0.13190519  0.41918936  0.40667032  0.10506802 -0.028461925
## S.F.Ratio    0.24738767 -0.54868266 -0.34776064 -0.02535027  0.110525946
## perc.alumni -0.27279746  0.56830741  0.27610160 -0.03457483 -0.257888520
## Expend      -0.10011229  0.66613880  0.49911348  0.11868377 -0.090327214
## Grad.Rate   -0.26564694  0.57345152  0.42060428 -0.01504416 -0.274221443
##                      PhD    Terminal   S.F.Ratio perc.alumni       Expend
## Private     -0.144492494 -0.12087101 -0.45017444  0.42031342  0.251858352
## Enroll       0.337217823  0.31262288  0.22622999 -0.16993226  0.046500671
## Top10perc    0.554515535  0.51204846 -0.37727547  0.45532374  0.666075330
## Top25perc    0.572637302  0.54768652 -0.28847723  0.41395463  0.529565055
## F.Undergrad  0.322874759  0.30471552  0.26770295 -0.21709914  0.004104707
## P.Undergrad  0.138475903  0.13190519  0.24738767 -0.27279746 -0.100112290
## Outstate     0.397693914  0.41918936 -0.54868266  0.56830741  0.666138802
## Room.Board   0.367004504  0.40667032 -0.34776064  0.27610160  0.499113480
## Books        0.018137138  0.10506802 -0.02535027 -0.03457483  0.118683774
## Personal    -0.006682589 -0.02846193  0.11052595 -0.25788852 -0.090327214
## PhD          1.000000000  0.84742013 -0.13254876  0.26636874  0.454216892
## Terminal     0.847420131  1.00000000 -0.17208965  0.28451544  0.450321394
## S.F.Ratio   -0.132548758 -0.17208965  1.00000000 -0.40847014 -0.573734129
## perc.alumni  0.266368739  0.28451544 -0.40847014  1.00000000  0.432079702
## Expend       0.454216892  0.45032139 -0.57373413  0.43207970  1.000000000
## Grad.Rate    0.323966450  0.30313221 -0.33346185  0.47718931  0.404485975
##                Grad.Rate
## Private      0.329463876
## Enroll      -0.003996233
## Top10perc    0.514807164
## Top25perc    0.503027635
## F.Undergrad -0.068072679
## P.Undergrad -0.265646939
## Outstate     0.573451519
## Room.Board   0.420604279
## Books       -0.015044161
## Personal    -0.274221443
## PhD          0.323966450
## Terminal     0.303132212
## S.F.Ratio   -0.333461854
## perc.alumni  0.477189307
## Expend       0.404485975
## Grad.Rate    1.000000000
```

```
# Some mild correlations found. (outstate and expend)
```

```r
#Use function prcomp to extract the Principal Components
pc <- prcomp(x=data.matrix(train[,1:16]), scale=TRUE)
summary
```

```
## standardGeneric for "summary" defined from package "base"
##
## function (object, ...)
## standardGeneric("summary")
## <environment: 0x7f836b335960>
## Methods may be defined for arguments: object
## Use  showMethods(summary)  for currently available ones.
```

```r
# So the first 6 components explain 80% of the variance in the data and the first
# 9 components explain 90% of the variance in the data. (Our best model with variable
# selection uses 8 predictors).
# Looks like PCA isn't particularly helpful for this dataset because many components
# are required to explain at least 90% variance in the data. Let's true to fit a model
# with the first 6 components.
```

```r
modpcr<-lm(train[,17] ~ pc$x[,1:6]) #Using the first 6 components.
summary(modpcr)
```

```
##
## Call:
## lm(formula = train[, 17] ~ pc$x[, 1:6])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.56077 -0.08035  0.02168  0.09597  0.29183
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     0.7440565  0.0056198 132.400  < 2e-16 ***
## pc$x[, 1:6]PC1  0.0234952  0.0024098   9.750  < 2e-16 ***
## pc$x[, 1:6]PC2  0.0213676  0.0029489   7.246  1.5e-12 ***
## pc$x[, 1:6]PC3 -0.0159910  0.0051813  -3.086 0.002131 **
## pc$x[, 1:6]PC4 -0.0075103  0.0058772  -1.278 0.201851
## pc$x[, 1:6]PC5 -0.0205558  0.0060180  -3.416 0.000684 ***
## pc$x[, 1:6]PC6 -0.0006416  0.0064958  -0.099 0.921352
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1311 on 537 degrees of freedom
## Multiple R-squared:  0.2409, Adjusted R-squared:  0.2324
## F-statistic:  28.4 on 6 and 537 DF,  p-value: < 2.2e-16
```

```r
# Root mean squared error of testing data. (0.1138004)
rmse(predict(modpcr,test), test$Rate)
```

```
## Warning: 'newdata' had 233 rows but variables found have 544 rows
```

```
## Warning in x - y: longer object length is not a multiple of shorter object
## length
```

```
## [1] 0.1612622
```

```
# The fit isn't great. R^2 is low and two predictors are not stat. significant.
# ALso the RMSE for test data is quite high compared to other models.
```

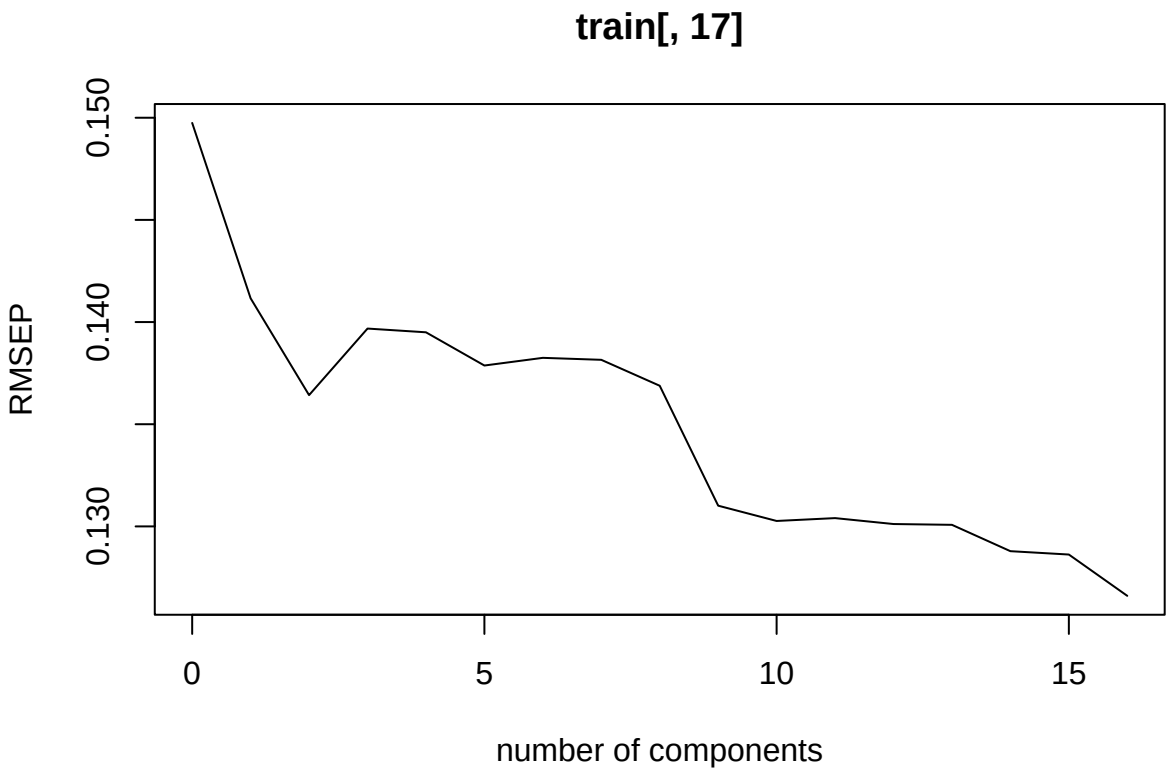
Let's try with CV.

```
library(pls)
```

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

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

```
set.seed(425)
modpcrcv<-pcr(train[,17] ~.,data=train,validation="CV",ncomp=16)
pcrCV<-RMSEP(modpcrcv,estimate="CV")
plot(pcrCV)
```



The best number of components to minimized the CV error is the max number of available predictors.

```
pcpred<-predict(modpcrcv,test,ncomp=16)
rmse(pcpred,test$Rate) # 0.1127521 This is the test RMSE.
```

```
## [1] 0.1127521
```

2g Comments

(g) Comment on the results obtained. How accurately can we predict the acceptance rate overall? Which approach would you recommend for this data set and why?

The full model RMSE is 0.120322 for training data 0.1138004 for test data.

Using AIC, BIC, Adjusted R^2 I found that variable selection algorithm recommends us to use 8 predictors. The revised model has the RMSE 0.1221655 for training data 0.1164683 for testing data

Use Ridge regression with 10 fold CV, I found that the RMSE is 0.1206716 for training data 0.1133502 for testing data (some small improvement on the test RSME)

Use Lasso regression with 10 fold CV, I found that the RMSE is 0.1206919 for training data 0.1134402 for testing data

Use PCR with 10 fold CV, I found that the RMSE is 0.1127521 for testing data

So overall, for new data, we predict the Rate variable with 88% accuracy approx. I would recommend Ridge regression for this dataset because it shows some small improvement on the test RSME compared to other methods. PCR also has improvement, but the number of components is not reduced compared to the full model so it is not clear our significant the result is. PCR with 6 components has fairly high RMSE on test data (0.16 approx. so we'd need to use more components).