GLM and Logistic Regression with LASSO/Ridge Regularization

Tianze Hua

3/4/2022

```
# load multiple packages with two lines
library(pacman)
p_load(ISLR, caret, tidyverse, gridExtra, pROC, psych, knitr,
       broom, gmodels, glmnet, Metrics)

# dataset used is College
attach(College)
md = College
```

Context Description Statistics for a large number of US Colleges from the 1995 issue of US News and World Report.

Format A data frame with 777 observations on the following 18 variables.

Private A factor with levels No and Yes indicating private or public university

Apps Number of applications received

Accept Number of applications accepted

Enroll Number of new students enrolled

Top10perc Pct. new students from top 10% of H.S. class

Top25perc Pct. new students from top 25% of H.S. class

F.Undergrad Number of fulltime undergraduates

P.Undergrad Number of parttime undergraduates

Outstate Out-of-state tuition

Room.Board Room and board costs

Books Estimated book costs

Personal Estimated personal spending

PhD Pct. of faculty with Ph.D.'s

Terminal Pct. of faculty with terminal degree

S.F.Ratio Student/faculty ratio

perc.alumni Pct. alumni who donate

Expend Instructional expenditure per student

Grad.Rate Graduation rate

```
glimpse(md)
```

```
## Rows: 777
## Columns: 18
## $ Private     <fct> Yes, Yes, Yes, Yes, Yes, Yes, Yes, Yes, Yes, Yes, Yes, Yes~
## $ Apps        <dbl> 1660, 2186, 1428, 417, 193, 587, 353, 1899, 1038, 582, 173~
## $ Accept      <dbl> 1232, 1924, 1097, 349, 146, 479, 340, 1720, 839, 498, 1425~
## $ Enroll      <dbl> 721, 512, 336, 137, 55, 158, 103, 489, 227, 172, 472, 484,~
## $ Top10perc   <dbl> 23, 16, 22, 60, 16, 38, 17, 37, 30, 21, 37, 44, 38, 44, 23~
## $ Top25perc   <dbl> 52, 29, 50, 89, 44, 62, 45, 68, 63, 44, 75, 77, 64, 73, 46~
## $ F.Undergrad <dbl> 2885, 2683, 1036, 510, 249, 678, 416, 1594, 973, 799, 1830~
## $ P.Undergrad <dbl> 537, 1227, 99, 63, 869, 41, 230, 32, 306, 78, 110, 44, 638~
## $ Outstate    <dbl> 7440, 12280, 11250, 12960, 7560, 13500, 13290, 13868, 1559~
## $ Room.Board  <dbl> 3300, 6450, 3750, 5450, 4120, 3335, 5720, 4826, 4400, 3380~
## $ Books       <dbl> 450, 750, 400, 450, 800, 500, 500, 450, 300, 660, 500, 400~
## $ Personal    <dbl> 2200, 1500, 1165, 875, 1500, 675, 1500, 850, 500, 1800, 60~
## $ PhD         <dbl> 70, 29, 53, 92, 76, 67, 90, 89, 79, 40, 82, 73, 60, 79, 36~
## $ Terminal    <dbl> 78, 30, 66, 97, 72, 73, 93, 100, 84, 41, 88, 91, 84, 87, 6~
## $ S.F.Ratio   <dbl> 18.1, 12.2, 12.9, 7.7, 11.9, 9.4, 11.5, 13.7, 11.3, 11.5, ~
## $ perc.alumni <dbl> 12, 16, 30, 37, 2, 11, 26, 37, 23, 15, 31, 41, 21, 32, 26,~
## $ Expend      <dbl> 7041, 10527, 8735, 19016, 10922, 9727, 8861, 11487, 11644,~
## $ Grad.Rate   <dbl> 60, 56, 54, 59, 15, 55, 63, 73, 80, 52, 73, 76, 74, 68, 55~
```

```
psych::describe(md, fast=TRUE) %>% slice(2:n()) %>% select(3:8) %>% kable()
```

	mean	sd	min	max	range	se
Apps	3001.63835	3870.201484	81.0	48094.0	48013.0	138.8427049
Accept	2018.80438	2451.113971	72.0	26330.0	26258.0	87.9332239
Enroll	779.97297	929.176190	35.0	6392.0	6357.0	33.3340101
Top10perc	27.55856	17.640364	1.0	96.0	95.0	0.6328445
Top25perc	55.79665	19.804778	9.0	100.0	91.0	0.7104924
F.Undergrad	3699.90734	4850.420531	139.0	31643.0	31504.0	174.0078673
P.Undergrad	855.29858	1522.431887	1.0	21836.0	21835.0	54.6169397
Outstate	10440.66924	4023.016484	2340.0	21700.0	19360.0	144.3249124
Room.Board	4357.52638	1096.696416	1780.0	8124.0	6344.0	39.3437648
Books	549.38095	165.105360	96.0	2340.0	2244.0	5.9231218
Personal	1340.64221	677.071454	250.0	6800.0	6550.0	24.2898031
PhD	72.66023	16.328155	8.0	103.0	95.0	0.5857693
Terminal	79.70270	14.722359	24.0	100.0	76.0	0.5281617
S.F.Ratio	14.08970	3.958349	2.5	39.8	37.3	0.1420050
perc.alumni	22.74389	12.391801	0.0	64.0	64.0	0.4445534
Expend	9660.17117	5221.768440	3186.0	56233.0	53047.0	187.3298993
Grad.Rate	65.46332	17.177710	10.0	118.0	108.0	0.6162469

```
# Frequency table for Private College and Public
table(factor(md$Private))
```

```
##
##  No Yes
## 212 565
```

```
# set Private college to be the baseline
md$Private <- relevel(md$Private, 'No')
```

```r
# descriptive statistics for continuous variables
continous = select_if(md, is.numeric)
summary(continous)
```

```
##       Apps           Accept          Enroll       Top10perc       Top25perc
##  Min.   :   81   Min.   :   72   Min.   :  35   Min.   : 1.00   Min.   :  9.0
##  1st Qu.:  776   1st Qu.:  604   1st Qu.: 242   1st Qu.:15.00   1st Qu.: 41.0
##  Median : 1558   Median : 1110   Median : 434   Median :23.00   Median : 54.0
##  Mean   : 3002   Mean   : 2019   Mean   : 780   Mean   :27.56   Mean   : 55.8
##  3rd Qu.: 3624   3rd Qu.: 2424   3rd Qu.: 902   3rd Qu.:35.00   3rd Qu.: 69.0
##  Max.   :48094   Max.   :26330   Max.   :6392   Max.   :96.00   Max.   :100.0
##   F.Undergrad     P.Undergrad         Outstate       Room.Board
##  Min.   :  139   Min.   :    1.0   Min.   : 2340   Min.   :1780
##  1st Qu.:  992   1st Qu.:   95.0   1st Qu.: 7320   1st Qu.:3597
##  Median : 1707   Median :  353.0   Median : 9990   Median :4200
##  Mean   : 3700   Mean   :  855.3   Mean   :10441   Mean   :4358
##  3rd Qu.: 4005   3rd Qu.:  967.0   3rd Qu.:12925   3rd Qu.:5050
##  Max.   :31643   Max.   :21836.0   Max.   :21700   Max.   :8124
##      Books           Personal         PhD            Terminal
##  Min.   :  96.0   Min.   : 250   Min.   :  8.00   Min.   : 24.0
##  1st Qu.: 470.0   1st Qu.: 850   1st Qu.: 62.00   1st Qu.: 71.0
##  Median : 500.0   Median :1200   Median : 75.00   Median : 82.0
##  Mean   : 549.4   Mean   :1341   Mean   : 72.66   Mean   : 79.7
##  3rd Qu.: 600.0   3rd Qu.:1700   3rd Qu.: 85.00   3rd Qu.: 92.0
##  Max.   :2340.0   Max.   :6800   Max.   :103.00   Max.   :100.0
##    S.F.Ratio      perc.alumni        Expend        Grad.Rate
##  Min.   : 2.50   Min.   : 0.00   Min.   : 3186   Min.   : 10.00
##  1st Qu.:11.50   1st Qu.:13.00   1st Qu.: 6751   1st Qu.: 53.00
##  Median :13.60   Median :21.00   Median : 8377   Median : 65.00
##  Mean   :14.09   Mean   :22.74   Mean   : 9660   Mean   : 65.46
##  3rd Qu.:16.50   3rd Qu.:31.00   3rd Qu.:10830   3rd Qu.: 78.00
##  Max.   :39.80   Max.   :64.00   Max.   :56233   Max.   :118.00
```

```r
set.seed(16)
train_index <- createDataPartition(md$Private, p=0.75, list = FALSE, times = 1)
train_data <- md[train_index,]
test_data <- md[-train_index,]
```

```r
model <- glm(
  Private ~ Personal + PhD + Grad.Rate + S.F.Ratio,
  data = md,
  family = binomial(link = 'logit'))

summary(model)
```

```
##
## Call:
## glm(formula = Private ~ Personal + PhD + Grad.Rate + S.F.Ratio,
##     family = binomial(link = "logit"), data = md)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.8164  -0.3215   0.2789   0.5333   3.8655
##
## Coefficients:
```

```
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  9.7915017  1.0455663   9.365  < 2e-16 ***
## Personal    -0.0007948  0.0001510  -5.263 1.42e-07 ***
## PhD         -0.0745440  0.0093540  -7.969 1.60e-15 ***
## Grad.Rate    0.0540927  0.0076088   7.109 1.17e-12 ***
## S.F.Ratio   -0.3681819  0.0347994 -10.580  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 910.75  on 776  degrees of freedom
## Residual deviance: 553.67  on 772  degrees of freedom
## AIC: 563.67
##
## Number of Fisher Scoring iterations: 6
```

```
# convert coefficient log odds into odds
exp(coef(model)) %>% kable()
```

	x
(Intercept)	1.788114e+04
Personal	9.992055e-01
PhD	9.281667e-01
Grad.Rate	1.055582e+00
S.F.Ratio	6.919913e-01

```
prob_train <- predict(model, newdata = train_data, type = 'response')
predict_result <- as.factor(ifelse(prob_train >= 0.5, 'Yes', 'No'))
```

```
confusionMatrix(predict_result, train_data$Private, positive = 'Yes')
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  108  30
##        Yes  51 394
##
##                Accuracy : 0.8611
##                  95% CI : (0.8303, 0.8881)
##     No Information Rate : 0.7273
##     P-Value [Acc > NIR] : 6.75e-15
##
##                   Kappa : 0.6347
##
##  Mcnemar's Test P-Value : 0.02627
##
##             Sensitivity : 0.9292
##             Specificity : 0.6792
##          Pos Pred Value : 0.8854
##          Neg Pred Value : 0.7826
##              Prevalence : 0.7273
##          Detection Rate : 0.6758
```

```
##    Detection Prevalence : 0.7633
##       Balanced Accuracy : 0.8042
##
##        'Positive' Class : Yes
##
```

```
CrossTable(predict_result, train_data$Private)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## | Chi-square contribution |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  583
##
##
##                | train_data$Private
## predict_result |        No |       Yes | Row Total |
## ---------------|-----------|-----------|-----------|
##             No |       108 |        30 |       138 |
##                |   131.549 |    49.331 |           |
##                |     0.783 |     0.217 |     0.237 |
##                |     0.679 |     0.071 |           |
##                |     0.185 |     0.051 |           |
## ---------------|-----------|-----------|-----------|
##            Yes |        51 |       394 |       445 |
##                |    40.795 |    15.298 |           |
##                |     0.115 |     0.885 |     0.763 |
##                |     0.321 |     0.929 |           |
##                |     0.087 |     0.676 |           |
## ---------------|-----------|-----------|-----------|
##   Column Total |       159 |       424 |       583 |
##                |     0.273 |     0.727 |           |
## ---------------|-----------|-----------|-----------|
##
##
```

```
prob_test <- predict(model, newdata = test_data, type = 'response')
predict_result <- as.factor(ifelse(prob_test >= 0.5, 'Yes', 'No'))

confusionMatrix(predict_result, test_data$Private, positive = 'Yes')
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No   33   9
##        Yes  20 132
```

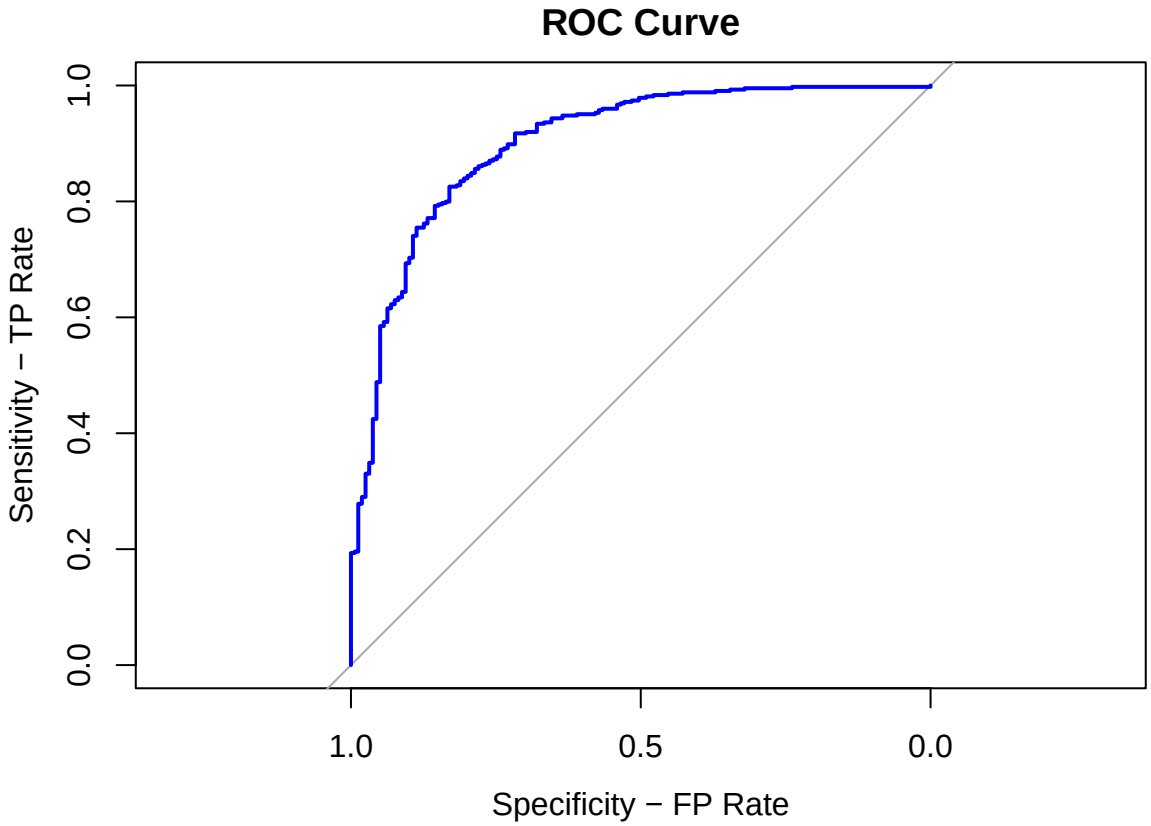
```
##
##                Accuracy : 0.8505
##                  95% CI : (0.7924, 0.8975)
##     No Information Rate : 0.7268
##     P-Value [Acc > NIR] : 3.121e-05
##
##                   Kappa : 0.5975
##
##  Mcnemar's Test P-Value : 0.06332
##
##             Sensitivity : 0.9362
##             Specificity : 0.6226
##          Pos Pred Value : 0.8684
##          Neg Pred Value : 0.7857
##              Prevalence : 0.7268
##          Detection Rate : 0.6804
##    Detection Prevalence : 0.7835
##       Balanced Accuracy : 0.7794
##
##        'Positive' Class : Yes
##
```

```
CrossTable(predict_result, test_data$Private)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## | Chi-square contribution |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  194
##
##
##                | test_data$Private
## predict_result |        No |       Yes | Row Total |
## ---------------|-----------|-----------|-----------|
##             No |        33 |         9 |        42 |
##                |    40.383 |    15.179 |           |
##                |     0.786 |     0.214 |     0.216 |
##                |     0.623 |     0.064 |           |
##                |     0.170 |     0.046 |           |
## ---------------|-----------|-----------|-----------|
##            Yes |        20 |       132 |       152 |
##                |    11.158 |     4.194 |           |
##                |     0.132 |     0.868 |     0.784 |
##                |     0.377 |     0.936 |           |
##                |     0.103 |     0.680 |           |
## ---------------|-----------|-----------|-----------|
##   Column Total |        53 |       141 |       194 |
```

```
##                 |     0.273 |     0.727 |           |
## ----------------|-----------|-----------|-----------|
##
##
```

```
curve <- roc(train_data$Private, prob_train)
plot(curve,
     col = 'Blue', ylab='Sensitivity - TP Rate',
     xlab = 'Specificity - FP Rate', main="ROC Curve")
```




```
pROC::auc(curve)
```

```
## Area under the curve: 0.9005
```

```
auc
```

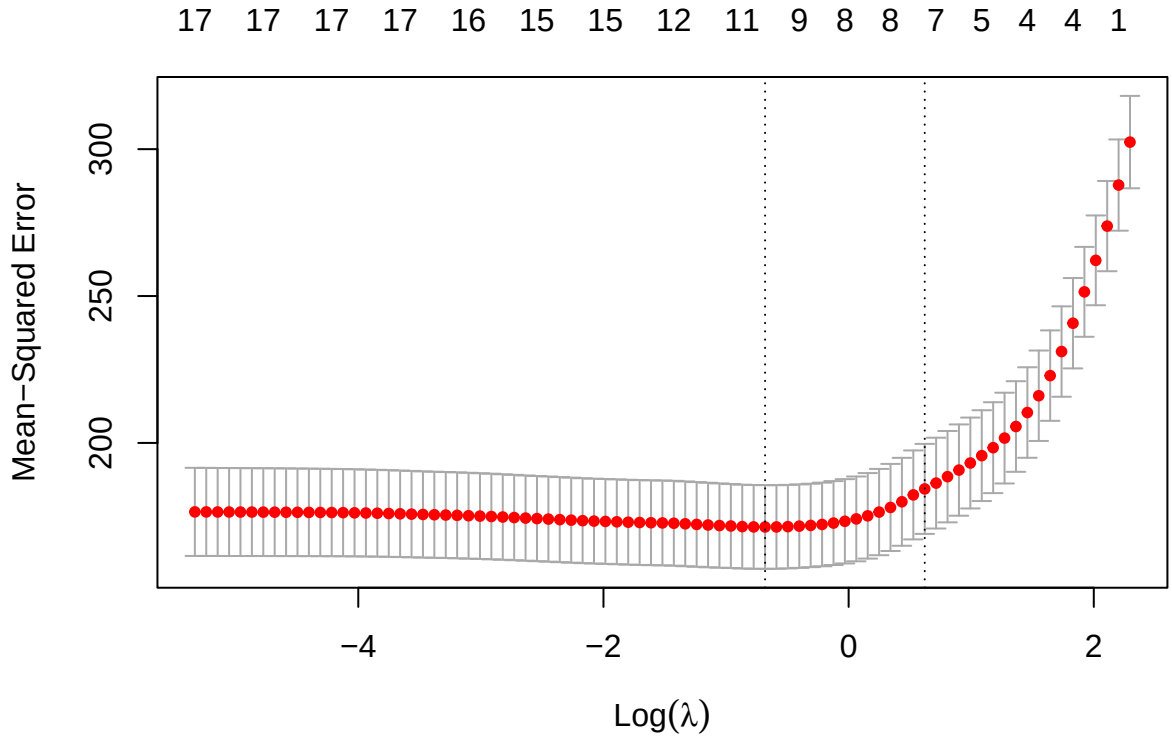
```
## function (actual, predicted)
## {
##     if (length(actual) != length(predicted)) {
##         msg <- "longer object length is not a multiple of shorter object length"
##         warning(msg)
##     }
##     r <- rank(predicted)
##     n_pos <- as.numeric(sum(actual == 1))
##     n_neg <- length(actual) - n_pos
##     return((sum(r[actual == 1]) - n_pos * (n_pos + 1)/2)/(n_pos *
##         n_neg))
## }
## <bytecode: 0x555bbd8ac1b0>
## <environment: namespace:Metrics>
```

```
set.seed(16)
train_index <- createDataPartition(md$Grad.Rate, p=0.8, list = FALSE, times = 1)
train <- md[train_index,]
test <- md[-train_index,]

train_x <- model.matrix(Grad.Rate ~., train)[,-1]
test_x <- model.matrix(Grad.Rate ~., test)[,-1]

train_y <- train$Grad.Rate
test_y <- test$Grad.Rate
```

```
cv.lasso <- cv.glmnet(train_x, train_y, nfolds = 10)
plot(cv.lasso)
```




```
log(cv.lasso$lambda.min)
```

```
## [1] -0.6815882
```

```
log(cv.lasso$lambda.1se)
```

```
## [1] 0.6208842
```

```
cv.lasso$lambda.min
```

```
## [1] 0.505813
```

```
# alpha = 1 for Lasso(L2)
# alpha = 0 for Ridge(L1)
model.min <- glmnet(train_x, train_y, alpha = 1, lambda = cv.lasso$lambda.min)
model.min
```

```
##
## Call:  glmnet(x = train_x, y = train_y, alpha = 1, lambda = cv.lasso$lambda.min)
```

```
##
##   Df  %Dev Lambda
## 1 10 45.72 0.5058
```

```
coef(model.min)
```

```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                         s0
## (Intercept) 36.5715021109
## PrivateYes   1.1226756560
## Apps         0.0006072198
## Accept       .
## Enroll       .
## Top10perc    0.0113548295
## Top25perc    0.1510988240
## F.Undergrad  .
## P.Undergrad -0.0014758616
## Outstate     0.0008664659
## Room.Board   0.0015830035
## Books        .
## Personal    -0.0023355305
## PhD          0.0004115897
## Terminal     .
## S.F.Ratio    .
## perc.alumni  0.2543466553
## Expend       .
```

```
model.1se <- glmnet(train_x, train_y, alpha = 1, lambda = cv.lasso$lambda.1se)
model.1se
```

```
##
## Call:  glmnet(x = train_x, y = train_y, alpha = 1, lambda = cv.lasso$lambda.1se)
##
##   Df %Dev Lambda
## 1  7 40.7  1.861
```

```
coef(model.1se)
```

```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                         s0
## (Intercept) 39.7944017980
## PrivateYes   .
## Apps         .
## Accept       .
## Enroll       .
## Top10perc    0.0415259483
## Top25perc    0.1265319936
## F.Undergrad  .
## P.Undergrad -0.0002931201
## Outstate     0.0010329156
## Room.Board   0.0008025894
## Books        .
## Personal    -0.0008607865
## PhD          .
## Terminal     .
## S.F.Ratio    .
```

```
## perc.alumni  0.1962613507
## Expend       .
```

```
model1.min <- glmnet(train_x, train_y, alpha = 0, lambda = cv.lasso$lambda.min)
model1.min
```

```
##
## Call:  glmnet(x = train_x, y = train_y, alpha = 0, lambda = cv.lasso$lambda.min)
##
##   Df %Dev Lambda
## 1 17 47.3 0.5058
```

```
coef(model1.min)
```

```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                         s0
## (Intercept) 35.0645821500
## PrivateYes   2.7303609139
## Apps         0.0008102179
## Accept       0.0001797282
## Enroll       0.0011386504
## Top10perc    0.0729534048
## Top25perc    0.1282571474
## F.Undergrad -0.0003054878
## P.Undergrad -0.0016993342
## Outstate     0.0008868921
## Room.Board   0.0020275449
## Books       -0.0009130863
## Personal    -0.0025415915
## PhD          0.1076786971
## Terminal    -0.1067852794
## S.F.Ratio    0.0898381512
## perc.alumni  0.2778837945
## Expend      -0.0003212874
```

```
model1.1se <- glmnet(train_x, train_y, alpha = 0, lambda = cv.lasso$lambda.1se)
model1.1se
```

```
##
## Call:  glmnet(x = train_x, y = train_y, alpha = 0, lambda = cv.lasso$lambda.1se)
##
##   Df  %Dev Lambda
## 1 17 46.96  1.861
```

```
coef(model1.1se)
```

```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                         s0
## (Intercept) 35.3741151796
## PrivateYes   2.9538103593
## Apps         0.0005807021
## Accept       0.0004173222
## Enroll       0.0005020891
## Top10perc    0.0873982732
## Top25perc    0.1163862379
## F.Undergrad -0.0001706039
## P.Undergrad -0.0015936925
```

```
## Outstate     0.0007844837
## Room.Board   0.0019363801
## Books       -0.0011059531
## Personal    -0.0025499319
## PhD          0.0807778172
## Terminal    -0.0677521657
## S.F.Ratio    0.0733677926
## perc.alumni  0.2591299704
## Expend      -0.0002170669
```

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

```
##
## Call:
## lm(formula = Grad.Rate ~ ., data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -54.457  -7.348  -0.383   7.013  53.495
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 34.6833008  5.5555005   6.243 8.08e-10 ***
## PrivateYes   2.6622354  1.9337425   1.377 0.169106
## Apps         0.0011790  0.0004850   2.431 0.015357 *
## Accept      -0.0005123  0.0009568  -0.535 0.592572
## Enroll       0.0027282  0.0024545   1.112 0.266787
## Top10perc    0.0422300  0.0808707   0.522 0.601728
## Top25perc    0.1438433  0.0607717   2.367 0.018249 *
## F.Undergrad -0.0005206  0.0004213  -1.236 0.217072
## P.Undergrad -0.0017412  0.0004291  -4.058 5.60e-05 ***
## Outstate     0.0009724  0.0002617   3.715 0.000222 ***
## Room.Board   0.0020516  0.0006634   3.092 0.002076 **
## Books       -0.0006681  0.0035764  -0.187 0.851873
## Personal    -0.0025334  0.0008670  -2.922 0.003609 **
## PhD          0.1277650  0.0617219   2.070 0.038875 *
## Terminal    -0.1289929  0.0678786  -1.900 0.057863 .
## S.F.Ratio    0.0925586  0.1881683   0.492 0.622974
## perc.alumni  0.2824721  0.0566339   4.988 7.99e-07 ***
## Expend      -0.0003938  0.0001747  -2.255 0.024510 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.8 on 605 degrees of freedom
## Multiple R-squared:  0.4741, Adjusted R-squared:  0.4593
## F-statistic: 32.08 on 17 and 605 DF,  p-value: < 2.2e-16
```

```
preds.ols <- predict(ols, new = test)
rmse(test$Grad.Rate, preds.ols)
```

```
## [1] 12.6856
```

```
rmse(train$Grad.Rate, preds.ols)
```

```
## [1] 21.09384
```

```
preds.train <- predict(model.1se, newx = train_x)
preds.train1 <- predict(model1.1se, newx = train_x)
train.rmse <- rmse(train_y, preds.train)
train1.rmse <- rmse(train_y, preds.train1)
```

```
preds.test<- predict(model.1se, newx = test_x)
test.rmse <- rmse(test_y, preds.test)

preds.test1<- predict(model1.1se, newx = test_x)
test1.rmse <- rmse(test_y, preds.test1)

train1.rmse
```

```
## [1] 12.66518
```

```
train.rmse
```

```
## [1] 13.39232
```

```
test.rmse
```

```
## [1] 12.65516
```

```
test1.rmse
```

```
## [1] 12.6153
```