STA363SecAProj2-Code Appendix

Rickey Huang

3/16/2021

```
# Loading the data
college <- read.csv("~/Desktop/2021Spring/STA-363/Projects/Project 2/STA363Porject2/collegedata1.csv")
```

```
# Every package required in the project is libraried here
# Package for the BSS
library(leaps)
# Packages for Ridge Regression
# Need to run the next line if the package glmnet is not installed
#install.packages("glmnet")
library(Matrix)
library(glmnet)
```

```
## Loaded glmnet 4.1-1
```

```
# For table output
library(knitr)
# For elastic net technique
# Need to run the next line if the package caret is not installed
#install.packages("caret")
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
# Show the dimension of the data set
dim(college)
```

```
## [1] 777  19
```

```
# Examinate any massing data in the data set
which(is.na(college))
```

```
## integer(0)
```

```
# Remove the column Enroll as it is usually unavailable
college <- subset(college, select = -Enroll)
# To make sure the column Enroll is actually removed
dim(college)
```

```
## [1] 777  18
```

```
# Add a column which represents the acceptance rate
college$Rate <- college$Accept / college$Apps
# Remove the column representing the number of acceptance then
college <- subset(college, select = -Accept)
# Remove the college name column
```

```r
college <- subset(college, select = -College)
# Change variable Private to PrivateYes
college$PrivateYes[college$Private=="Yes"] <- 1
college$PrivateYes[college$Private!="Yes"] <- 0
# Remove the Private column
college <- subset(college, select = -Private)
# To check whether the data is actually modified
head(college)
```

```
##   Apps Top10perc Top25perc F.Undergrad P.Undergrad Outstate Room.Board Books
## 1 1660        23        52        2885         537     7440       3300   450
## 2 2186        16        29        2683        1227    12280       6450   750
## 3 1428        22        50        1036          99    11250       3750   400
## 4  417        60        89         510          63    12960       5450   450
## 5  193        16        44         249         869     7560       4120   800
## 6  587        38        62         678          41    13500       3335   500
##   Personal PhD Terminal S.F.Ratio perc.alumni Expend Grad.Rate      Rate
## 1     2200  70       78      18.1          12   7041        60 0.7421687
## 2     1500  29       30      12.2          16  10527        56 0.8801464
## 3     1165  53       66      12.9          30   8735        54 0.7682073
## 4      875  92       97       7.7          37  19016        59 0.8369305
## 5     1500  76       72      11.9           2  10922        15 0.7564767
## 6      675  67       73       9.4          11   9727        55 0.8160136
##   PrivateYes
## 1          1
## 2          1
## 3          1
## 4          1
## 5          1
## 6          1
```

```r
dim(college)
```

```
## [1] 777  17
```

```r
# Stage 1
# Fit and choose the best models amoung the models with different amount of variables and store
# them in BSSout.
# nvmax is set to be 16, since the full model contains 16 exploratory variables in it.
BSSout <- regsubsets( Apps ~ ., data = college, nvmax = 16)
# Just for checking
#BSSout$rss
#plot(BSSout$rss)
```

```r
# The adjusted R-squared for the models from the stage 1
summary(BSSout)$adjr2
```

```
##  [1] 0.6629606 0.7351032 0.7609193 0.7652828 0.7691150 0.7711374 0.7731134
##  [8] 0.7740025 0.7742142 0.7745535 0.7747108 0.7745266 0.7743210 0.7740949
## [15] 0.7738017 0.7735079
```

```r
# Plot the adjusted R-squareds for models
plot(BSSout, scale = "adjr2")
```

```r
# Fit the LSLR models with the features we selected
LSLR <- lm(Apps ~ PrivateYes + F.Undergrad + P.Undergrad + Outstate + Room.Board + Terminal + perc.alumn
```

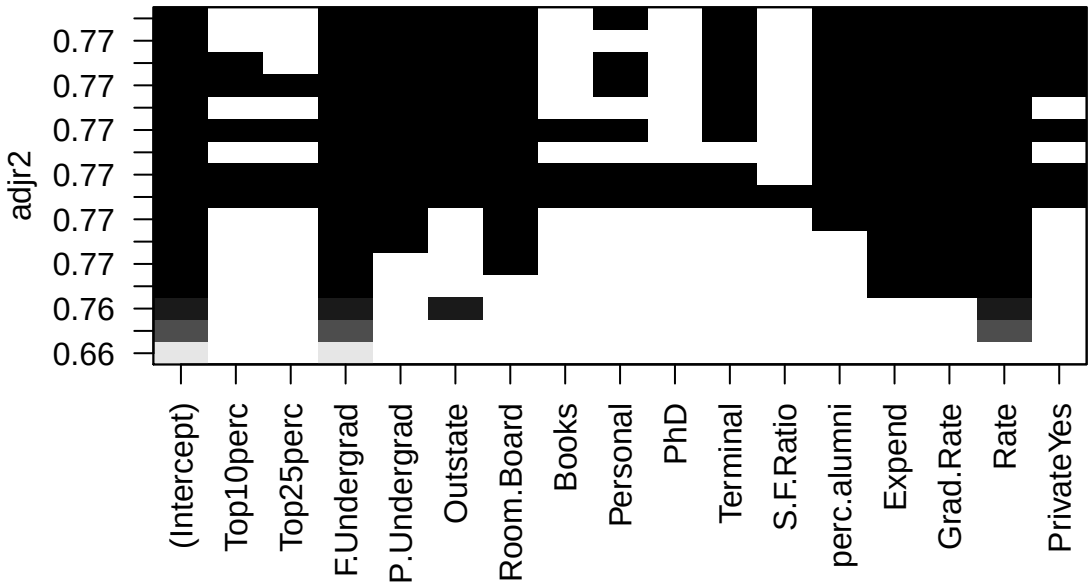

Figure 1: Adjusted R-squareds for models created in the stage 1 of BSS

```
# Output the estimates for the LSLR model
knitr::kable(summary(LSLR)$coefficients, caption = "\\label{tab:LSLRbetas}The estimates for the LSLR mo
```

Table 1: The estimates for the LSLR model

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	1994.5038628	701.8798770	2.841660	0.0046072
PrivateYes	-353.5375145	240.8623931	-1.467799	0.1425694
F.Undergrad	0.6555742	0.0201696	32.503022	0.0000000
P.Undergrad	-0.1597435	0.0554461	-2.881058	0.0040742
Outstate	0.0838324	0.0328084	2.555209	0.0108046
Room.Board	0.2356826	0.0844043	2.792307	0.0053638
Terminal	-9.8869963	5.8877399	-1.679251	0.0935108
perc.alumni	-20.1480419	7.0000823	-2.878258	0.0041101
Expend	0.0689980	0.0187457	3.680732	0.0002489
Grad.Rate	19.0549834	5.0978967	3.737813	0.0001994
Rate	-4812.5857721	525.3775219	-9.160243	0.0000000

```
# Implement the k-fold technique
# Create two null matrices to store the residuals of the LSLR model
residualskfold <- matrix(NA, nrow=777, ncol=1)
# Set up k
# Choose 21, since 21 is a factor of the number of rows 777
k <- 21
#set random seed
set.seed(100)
#create folds
folds <- sample(rep(1:k, 37),777,replace = FALSE)
#for loop
for(i in 1:k){
  #find the rows in fold i
  infold <- which(folds==i)
  #create a kfoldCV training set
  kfoldCVTraining <- college[-infold,]
  #create a kfoldCV test set
  kfoldCVTest <- college[infold,]
  #train the LSLR model
```

```r
cor(college)
```

```
##                    Apps  Top10perc   Top25perc F.Undergrad P.Undergrad
## Apps         1.00000000  0.3388337  0.35163990  0.81449058  0.39826427
## Top10perc    0.33883368  1.0000000  0.89199497  0.14128873 -0.10535628
## Top25perc    0.35163990  0.8919950  1.00000000  0.19944466 -0.05357664
## F.Undergrad  0.81449058  0.1412887  0.19944466  1.00000000  0.57051219
## P.Undergrad  0.39826427 -0.1053563 -0.05357664  0.57051219  1.00000000
## Outstate     0.05015903  0.5623305  0.48939383 -0.21574200 -0.25351232
## Room.Board   0.16493896  0.3714804  0.33148989 -0.06889039 -0.06132551
## Books        0.13255860  0.1188584  0.11552713  0.11554976  0.08119952
## Personal     0.17873085 -0.0933164 -0.08081027  0.31719954  0.31988162
## PhD          0.39069733  0.5318280  0.54586221  0.31833697  0.14911422
## Terminal     0.36949147  0.4911350  0.52474884  0.30001894  0.14190357
## S.F.Ratio    0.09563303 -0.3848745 -0.29462884  0.27970335  0.23253051
## perc.alumni -0.09022589  0.4554853  0.41786429 -0.22946222 -0.28079236
## Expend       0.25959198  0.6609134  0.52744743  0.01865162 -0.08356842
## Grad.Rate    0.14675460  0.4949892  0.47728116 -0.07877313 -0.25700099
## Rate        -0.39255498 -0.4786754 -0.43467244 -0.15565379 -0.09228664
## PrivateYes  -0.43209471  0.1641322  0.09575224 -0.61556054 -0.45208775
##                 Outstate  Room.Board        Books    Personal         PhD
## Apps         0.05015903  0.16493896  0.132558598  0.17873085  0.39069733
## Top10perc    0.56233054  0.37148038  0.118858431 -0.09331640  0.53182802
## Top25perc    0.48939383  0.33148989  0.115527130 -0.08081027  0.54586221
## F.Undergrad -0.21574200 -0.06889039  0.115549761  0.31719954  0.31833697
## P.Undergrad -0.25351232 -0.06132551  0.081199521  0.31988162  0.14911422
## Outstate     1.00000000  0.65425640  0.038854868 -0.29908690  0.38298241
## Room.Board   0.65425640  1.00000000  0.127962970 -0.19942818  0.32920228
## Books        0.03885487  0.12796297  1.000000000  0.17929476  0.02690573
## Personal    -0.29908690 -0.19942818  0.179294764  1.00000000 -0.01093579
## PhD          0.38298241  0.32920228  0.026905731 -0.01093579  1.00000000
## Terminal     0.40798320  0.37453955  0.099954700 -0.03061311  0.84958703
## S.F.Ratio   -0.55482128 -0.36262774 -0.031929274  0.13634483 -0.13053011
## perc.alumni  0.56626242  0.27236345 -0.040207736 -0.28596808  0.24900866
## Expend       0.67277862  0.50173942  0.112409075 -0.09789189  0.43276168
## Grad.Rate    0.57128993  0.42494154  0.001060894 -0.26934396  0.30503785
## Rate        -0.24095073 -0.31030204 -0.174072883  0.01997851 -0.31833394
## PrivateYes   0.55264990  0.34053206 -0.018548975 -0.30448505 -0.15671437
##                 Terminal   S.F.Ratio perc.alumni      Expend    Grad.Rate
## Apps         0.36949147  0.09563303 -0.09022589  0.25959198  0.146754600
## Top10perc    0.49113502 -0.38487451  0.45548526  0.66091341  0.494989235
## Top25perc    0.52474884 -0.29462884  0.41786429  0.52744743  0.477281164
## F.Undergrad  0.30001894  0.27970335 -0.22946222  0.01865162 -0.078773129
## P.Undergrad  0.14190357  0.23253051 -0.28079236 -0.08356842 -0.257000991
## Outstate     0.40798320 -0.55482128  0.56626242  0.67277862  0.571289928
## Room.Board   0.37453955 -0.36262774  0.27236345  0.50173942  0.424941541
## Books        0.09995470 -0.03192927 -0.04020774  0.11240908  0.001060894
## Personal    -0.03061311  0.13634483 -0.28596808 -0.09789189 -0.269343964
## PhD          0.84958703 -0.13053011  0.24900866  0.43276168  0.305037850
## Terminal     1.00000000 -0.16010395  0.26713029  0.43879922  0.289527232
## S.F.Ratio   -0.16010395  1.00000000 -0.40292917 -0.58383204 -0.306710405
## perc.alumni  0.26713029 -0.40292917  1.00000000  0.41771172  0.490897562
## Expend       0.43879922 -0.58383204  0.41771172  1.00000000  0.390342696
## Grad.Rate    0.28952723 -0.30671041  0.49089756  0.39034270  1.000000000
```

```
## Rate        -0.30379999  0.10998188 -0.13210402 -0.40862232 -0.286971505
## PrivateYes  -0.12961994 -0.47220474  0.41477493  0.25846068  0.336162290
##                    Rate  PrivateYes
## Apps        -0.39255498 -0.43209471
## Top10perc   -0.47867539  0.16413220
## Top25perc   -0.43467244  0.09575224
## F.Undergrad -0.15565379 -0.61556054
## P.Undergrad -0.09228664 -0.45208775
## Outstate    -0.24095073  0.55264990
## Room.Board  -0.31030204  0.34053206
## Books       -0.17407288 -0.01854897
## Personal     0.01997851 -0.30448505
## PhD         -0.31833394 -0.15671437
## Terminal    -0.30379999 -0.12961994
## S.F.Ratio    0.10998188 -0.47220474
## perc.alumni -0.13210402  0.41477493
## Expend      -0.40862232  0.25846068
## Grad.Rate   -0.28697150  0.33616229
## Rate         1.00000000  0.08499047
## PrivateYes   0.08499047  1.00000000
```

```
# Create the design matrix, using all variables we have
XD <- model.matrix(Apps ~ ., data = college)
# Doing the cross validation to choose the best tuning parameter
set.seed(100)
# As requested by the client, we want try the tuning parameters from 0 to 1000 by 0.5
cv.ridge <- cv.glmnet(XD[, -1], college$Apps, alpha = 0, nfold = 21, lambda = seq(from = 0, to = 1000,
# Plot the result
plot(cv.ridge)
```

```
# Return the smallest test MSE and the corresponding Lambda
min(cv.ridge$cvm)
```

```
## [1] 3567052
```

```
#cv.ridge$lambda.min
# Compute the RMSE for the full model and ridge model
sqrt(cv.ridge$cvm[1])
```

```
## [1] 2014.712
```

```
sqrt(cv.ridge$cvm[which.min(cv.ridge$cvm)])
```

```
## [1] 1888.664
```

```
# Show the percentage in improvement
(sqrt(cv.ridge$cvm[1]) - sqrt(cv.ridge$cvm[which.min(cv.ridge$cvm)])) / sqrt(cv.ridge$cvm[1])
```

```
## [1] 0.06256349
```

```
# Create the final Ridge model
RidgeModel <- glmnet(XD[, -1], college$Apps, alpha = 0, lambda = cv.ridge$lambda.min)
```

```
# Create a matrix to store the coefficients for the Full and Ridge model
Mat <- cbind(FullModel = coefficients(glmnet(XD[,-1],college$Apps, alpha = 0, lambda = 0)), Shrinkage =
Mat <- as.matrix(Mat)
# Add name to the matrix
colnames(Mat) <- c("Full Model", "Shrinkage")
```

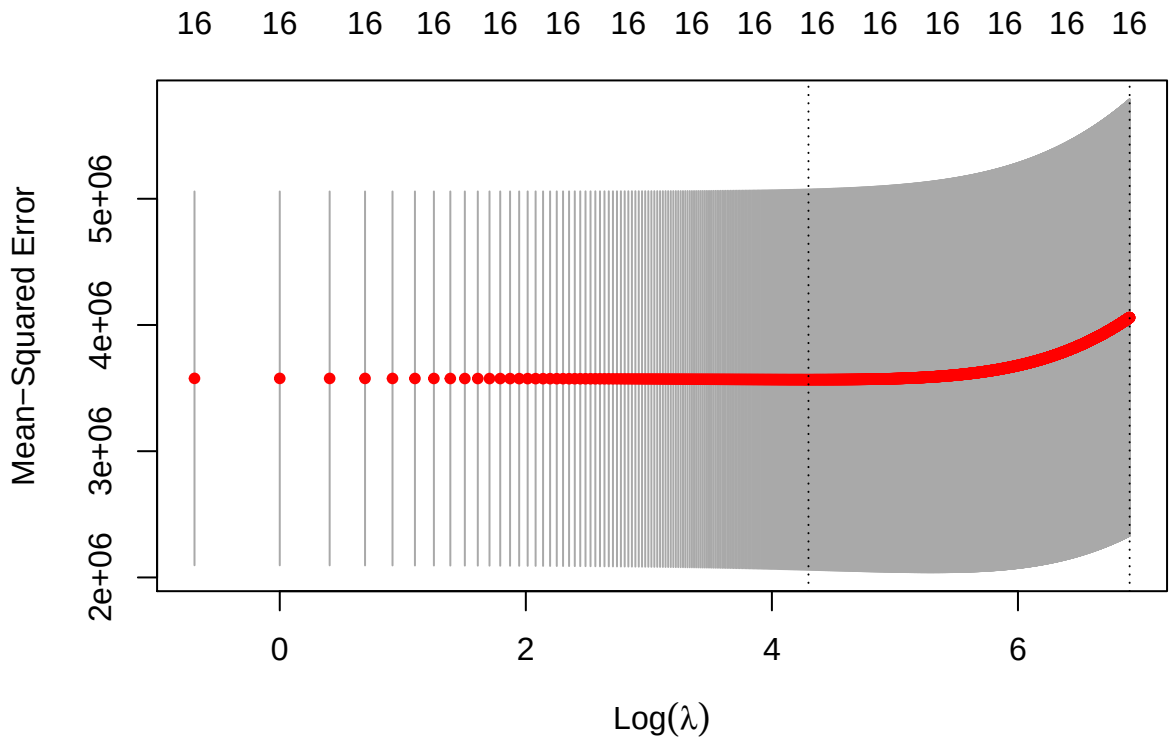

Figure 2: test MSE vs. log(Lambda) for the Ridge model

```
# Store the data into a dataframe
Mat <- data.frame(Mat)
# Output the coefficients for the full model and the Ridge model side by side
knitr::kable(Mat, caption = "\\label{tab:Ridgebetas}Comparing the Coefficients: Full model vs. Ridge mo
```

Table 2: Comparing the Coefficients: Full model vs. Ridge model

	Full.Model	Shrinkage
(Intercept)	2347.9828469	2053.3203010
Top10perc	7.8638193	8.4805942
Top25perc	-4.0746483	-2.3875501
F.Undergrad	0.6573680	0.6285696
P.Undergrad	-0.1459711	-0.1101358
Outstate	0.0784838	0.0729730
Room.Board	0.2395610	0.2425719
Books	-0.2110964	-0.1702510
Personal	-0.1280348	-0.1100649
PhD	-0.9091787	-0.8032072
Terminal	-9.2029383	-7.9782245
S.F.Ratio	2.6902737	6.0486138
perc.alumni	-21.5745533	-21.5946430
Expend	0.0662131	0.0676779
Grad.Rate	18.1436018	18.6297680
Rate	-4740.8089022	-4562.0644359
PrivateYes	-347.5734261	-449.8102435

```
# Create the design matrix, using all variables we have
XD <- model.matrix(Apps ~ ., data = college)
# Doing the cross validation to choose the best tuning parameter
set.seed(100)
# As requested by the client, we want try the tuning parameters from 0 to 1000 by 0.5
cv.lasso <- cv.glmnet(XD[, -1], college$Apps, alpha = 1, nfold = 21, lambda = seq(from = 0, to = 1000,
# Plot the result
```

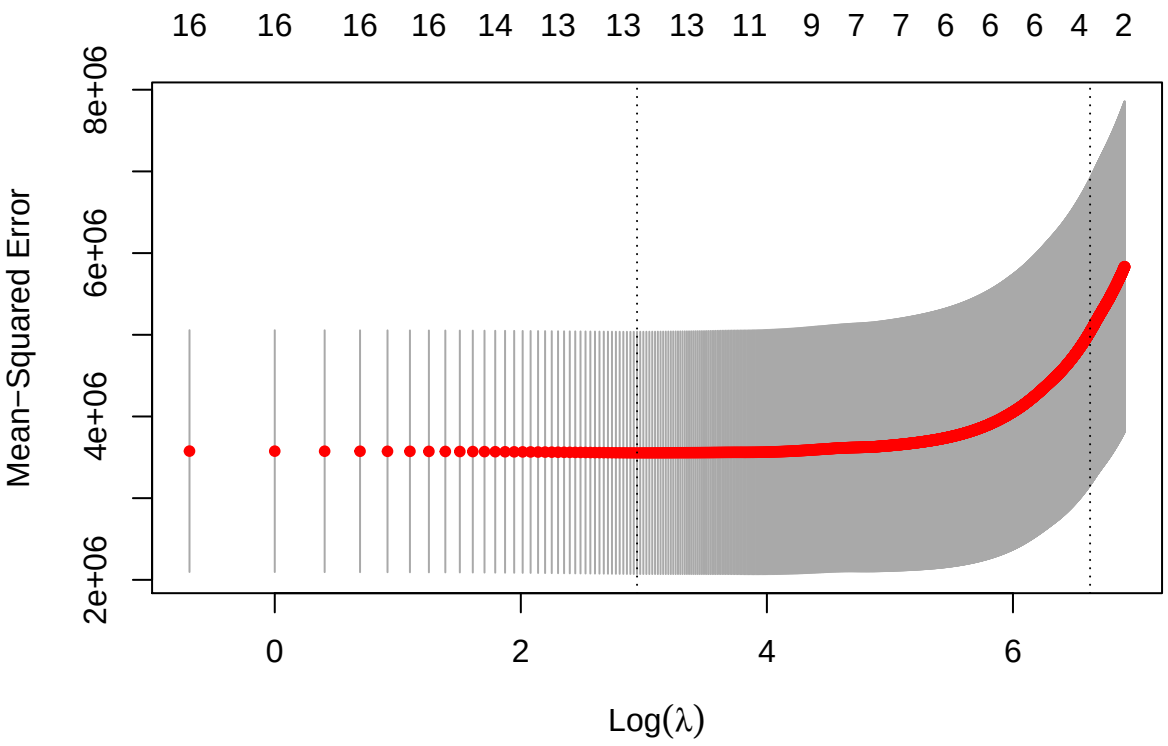

Figure 3: test MSE vs. log(Lambda) for the Lasso model

```
# Create the final Ridge model
LassoModel <- glmnet(XD[, -1], college$Apps, alpha = 1, lambda = cv.lasso$lambda.min)
```

```
# Create a matrix to store the coefficients for the Full model, Ridge model, and Lasso model
Mat <- cbind(FullModel = coefficients(glmnet(XD[,-1],college$Apps, alpha = 0, lambda = 0)), Shrinkage =
Mat <- as.matrix(Mat)
# Add name to the matrix
colnames(Mat) <- c("Full Model", "Shrinkage", "Lasso")
# Store the data into a dataframe
Mat <- data.frame(Mat)
# Output the coefficients for the full model, the Ridge model, and the Lasso model side by side
knitr::kable(Mat, caption = "\\label{tab:Lassobetas}Comparing the Coefficients: Full model vs. Ridge mo
```

Table 3: Comparing the Coefficients: Full model vs. Ridge model vs. Lasso model

	Full.Model	Shrinkage	Lasso
(Intercept)	2347.9828469	2053.3203010	1970.8274189
Top10perc	7.8638193	8.4805942	2.8468785
Top25perc	-4.0746483	-2.3875501	0.0000000
F.Undergrad	0.6573680	0.6285696	0.6494415
P.Undergrad	-0.1459711	-0.1101358	-0.1205914
Outstate	0.0784838	0.0729730	0.0633135
Room.Board	0.2395610	0.2425719	0.2233467
Books	-0.2110964	-0.1702510	-0.0799176
Personal	-0.1280348	-0.1100649	-0.1005813
PhD	-0.9091787	-0.8032072	0.0000000

	Full.Model	Shrinkage	Lasso
Terminal	-9.2029383	-7.9782245	-5.8787418
S.F.Ratio	2.6902737	6.0486138	0.0000000
perc.alumni	-21.5745533	-21.5946430	-18.3912000
Expend	0.0662131	0.0676779	0.0648436
Grad.Rate	18.1436018	18.6297680	17.1839692
Rate	-4740.8089022	-4562.0644359	-4701.8525447
PrivateYes	-347.5734261	-449.8102435	-213.7627808

```
# Trainning the Elastic Net Model
ElnetModel <- train(
  Apps ~ ., data = college,
  method = "glmnet",
  trControl = trainControl(method = "cv", number = 21)
)
# Create a matrix to store the coefficients for the Full model, Ridge model, Lasso model, and Elastic N
Mat <- cbind(FullModel = coefficients(glmnet(XD[,-1],college$Apps, alpha = 0, lambda = 0)), Shrinkage =
Mat <- as.matrix(Mat)
# Add name to the matrix
colnames(Mat) <- c("Full Model", "Shrinkage", "Lasso", "Elastic Net")
# Store the data into a dataframe
Mat <- data.frame(Mat)
# Output the coefficients for the full model, the Ridge model, the Lasso model, the Elastic Net model s
knitr::kable(Mat, caption = "\\label{tab:Elnetbetas}Comparing the Coefficients: Full model vs. Ridge mo
```

Table 4: Comparing the Coefficients: Full model vs. Ridge model vs. Lasso model vs. Elastic Net model

	Full.Model	Shrinkage	Lasso	Elastic.Net
(Intercept)	2347.9828469	2053.3203010	1970.8274189	2012.2948856
Top10perc	7.8638193	8.4805942	2.8468785	5.4355978
Top25perc	-4.0746483	-2.3875501	0.0000000	-0.1753023
F.Undergrad	0.6573680	0.6285696	0.6494415	0.6322090
P.Undergrad	-0.1459711	-0.1101358	-0.1205914	-0.1097671
Outstate	0.0784838	0.0729730	0.0633135	0.0681927
Room.Board	0.2395610	0.2425719	0.2233467	0.2376621
Books	-0.2110964	-0.1702510	-0.0799176	-0.1330824
Personal	-0.1280348	-0.1100649	-0.1005813	-0.1060022
PhD	-0.9091787	-0.8032072	0.0000000	0.0000000
Terminal	-9.2029383	-7.9782245	-5.8787418	-7.8027002
S.F.Ratio	2.6902737	6.0486138	0.0000000	2.1517864
perc.alumni	-21.5745533	-21.5946430	-18.3912000	-20.6252852
Expend	0.0662131	0.0676779	0.0648436	0.0670443
Grad.Rate	18.1436018	18.6297680	17.1839692	18.1880286
Rate	-4740.8089022	-4562.0644359	-4701.8525447	-4584.7702978
PrivateYes	-347.5734261	-449.8102435	-213.7627808	-387.0064176

```
# return the test RMSE
ElnetModel
```

```
## glmnet
##
```

```
## 777 samples
##  16 predictor
##
## No pre-processing
## Resampling: Cross-Validated (21 fold)
## Summary of sample sizes: 739, 741, 740, 740, 740, 740, ...
## Resampling results across tuning parameters:
##
##   alpha  lambda      RMSE      Rsquared   MAE
##   0.10     6.300427  1628.784  0.8313788  1000.6191
##   0.10    63.004271  1624.236  0.8323231   997.4237
##   0.10   630.042706  1684.448  0.8262015  1004.9927
##   0.55     6.300427  1628.888  0.8315239   999.4192
##   0.55    63.004271  1625.509  0.8328906   988.5645
##   0.55   630.042706  1764.469  0.8242229  1012.9412
##   1.00     6.300427  1627.798  0.8318897   997.7343
##   1.00    63.004271  1628.953  0.8326987   982.0657
##   1.00   630.042706  1903.976  0.7999342  1122.9951
##
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were alpha = 0.1 and lambda = 63.00427.
```

Table 5: Models with their test RMSE

models	**test RMSE**
LSLR model	1881.8
Ridge model	1888.64
Lasso model	1885.252
Elastic Net model	1624.236