# Regularization: Graduate Rate of college

# Ridge and LASSO regression

Heejae Roh
Northeastern University
ALY 6015: Intermediate Analytics
Paromita Guha
February 5, 2023

## INTRODUCTION

Regularization prevents overfitting of the model. I will learn about the Ridge (L2) LASSO (L1) method which are used to prevent overfitting. Both methods use a method called 'Lambda' that gives a penalty to the coefficient. I will learn how to execute Ridge and LASSO and also compare these models with stepwise model.

## ABSTRACTION

### Data itself

Dataset is about statistics for US colleges from the 1995 issue of US news and World Report. Dependent variable is Graduate Rate which is numeric variable. There are 16 other numeric variables which contain information about admissions, estimated cost of college life, student/faculty ratio of each college.

### EDA

I found the outlier which Grad.Rate is bigger than 100 and deleted the row. Scatter plot of Grad.Rate shows that Outstate, Top10perc, perc.alumni is highly related to Grad.Rate. I learned for linear regression or generalization, eliminating outlier is important, because I compare the result before and after eliminating the outlier, there was significant difference in lambda and model.

### PREREQUISITE for MODELING

I made a model except only one row which Grad.Rate is higher than 100. After that I split the dataset to train and test whchi contains 543 and 233 observations.

### Ridge Regularization

Ridge use penalty equal to the square of the magnitude of coefficients. It means that the all coefficients may approach zero, but they will never actually equal zero. In other words, Ridge cannot eliminate variables from the model

### LASSO Regularization

On the other hand, LASSO use penalty equal to the absolute value of the magnitude of coefficients. It allows to actually reduce a coefficient to zero. In this project, 9 variables are eliminated with one standard error of the minimum lambda.

### Ridge vs LASSO vs Stepwise

After making each model, I check the Root Mean Square Error (RMSE) and compare each model. I also check the multicollinearity with correlation matrix and VIF values. I thought that LASSO perfoms better than Ridge, because the RMSE is lower, although it contains fewer variables.

As same as comparing Ridge vs LASSO, I compare the RMSE of LASSO and Stepwise. The Stepwise model's RMSE was lower than LASSO. However there is multicollinearity problem in both models, so I fixed it manually.

## CONCLUSION

I learn about Ridge and LASSO Regularization. Also, about the stepwise method. I realize that I have to check the dataset and model respectively, because I can't simply say which model is always good for specific dataset.

## PART 0. INTRODUCTION

Generalization prevents overfitting of the model. Overfitting means using too many variables or having multicollinearity. In this module, I will learn about the Ridge (L2) LASSO (L1) method which are used to prevent overfitting. Both methods use a method called 'Lambda' that gives a penalty to the coefficient. This allows the coefficients to shrink to 0 as Lambda gets greater to prevent overfitting. In this process, we will use the minimum Lambda or one standard error of the minimum Lambda. I will learn how to execute Ridge and LASSO and its advantages and disadvantages while making a model of predicting Graduate Rate of college dataset.

## PART 1. ANALYSIS

**Data Analysis by data explanation**

1. Dataset is about statistics for US Colleges from the the 1995 issue of US News and World Report (R-DATA, n.d.)
2. Data has 777 observations and 18 variables. Our target variable is Grad.Rate which means graduation rate of specific university which is numeric variable. There are 17 numeric variables (including Grad.Rate) information about admissions, estimated cost of college life, student/faculty ratio, and etc.
3. In this analysis, I will find out the model which can predict graduation rate with other numeric variables

## PART 1-1. SUMMARY of EDA and DATA CLEANING

**Headtail of Data 1**

| | Private | Apps | Accept | Enroll | Top10perc | Top25perc | F.undergrad |
|---|---|---|---|---|---|---|---|
| Abilene Christian University | Yes | 1660 | 1232 | 721 | 23 | 52 | 2885 |
| Adelphi University | Yes | 2186 | 1924 | 512 | 16 | 29 | 2683 |
| Adrian College | Yes | 1428 | 1097 | 336 | 22 | 50 | 1036 |
| … | | | | | … | | |
| Xavier University of Louisiana | Yes | 2097 | 1915 | 695 | 34 | 61 | 2793 |
| Yale University | Yes | 10705 | 2453 | 1317 | 95 | 99 | 5217 |
| York College of Pennsylvania | Yes | 2989 | 1855 | 691 | 28 | 63 | 2988 |

**Headtail of Data 2**

| | P.undergrad | Outstate | Room.Board | Books | Personal | Phd |
|---|---|---|---|---|---|---|
| Abilene Christian University | 537 | 7440 | 3300 | 450 | 2200 | 70 |
| Adelphi University | 1227 | 12280 | 6450 | 750 | 1500 | 29 |
| Adrian College | 99 | 11250 | 3750 | 400 | 1165 | 53 |
| … | | | | | | |
| Xavier University of Louisiana | 166 | 6900 | 4200 | 617 | 781 | 67 |
| Yale University | 83 | 19840 | 6510 | 630 | 2115 | 96 |
| York College of Pennsylvania | 1726 | 4990 | 3560 | 500 | 1250 | 75 |

**Headtail of Data 3**

| | Terminal | S.F.Ratio | perc.alumni | Expend | Grad.Rate |
|---|---|---|---|---|---|
| Abilene Christian University | 78 | 18.1 | 12 | 7041 | 60 |
| Adelphi University | 30 | 12.2 | 16 | 10527 | 56 |
| Adrian College | 66 | 12.9 | 30 | 8735 | 54 |
| … | | | | | |
| Xavier University of Louisiana | 75 | 14.4 | 20 | 8323 | 49 |
| Yale University | 96 | 5.8 | 49 | 40386 | 99 |
| York College of Pennsylvania | 75 | 18.1 | 28 | 4509 | 99 |

**Detailed explanation about variables**

1. Top10/25perc: Percentage of new students from top 10/25% of H.S. class
2. Outstate: out of state tuition, it could be indicate estimated or average of tuition
3. Room.Board is room and board costs, Books is estimated book costs
4. Terminal is pct. Of faculty with terminal degree (PhDs or doctorates) and S.F.Ratio: Student/faculty ratio are about education quality
5. Perc.alumni: Pct. Alumni who donate and Expend: Instructional expenditure per student, these are about financial conditions and support
6. Grad.Rate: Graduation rate, I think it could explain the loyalty about school, because if you don't like or trust your school curriculum, it's very likely that you won't graduate (R-DATA, n.d.)

**Summary of Data 1**

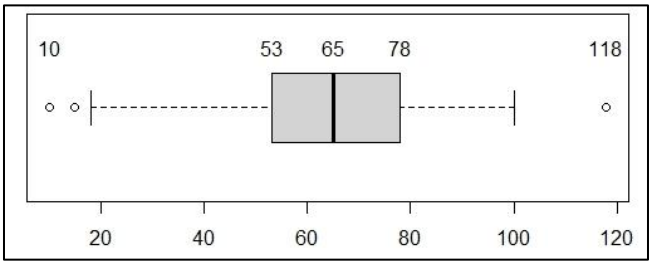
| Private | | | Apps | Accept | Enroll | Top10perc | Top25perc | F.undergrad | P.undergrad | Outstate |
|---|---|---|---|---|---|---|---|---|---|---|
| No | 212 | Min. | 81 | 72 | 35 | 1.00 | 9.0 | 139 | 1.0 | 2340 |
| Yes | 565 | 1st Qu. | 776 | 604 | 242 | 15.00 | 41.0 | 992 | 95.0 | 7320 |
| | | median | 1558 | 1110 | 434 | 23.00 | 54.0 | 1707 | 353.0 | 9990 |
| | | mean | 3002 | 2019 | 780 | 27.56 | 55.8 | 3700 | 855.3 | 10441 |
| | | 3rd Qu. | 3624 | 2424 | 902 | 35.00 | 69.0 | 4005 | 967.0 | 12925 |
| | | Max. | 48094 | 26330 | 6392 | 96.00 | 100.0 | 31643 | 21836.0 | 21700 |

**Summary of Data 2**

| | Room.Board | Books | Personal | PhD | Terminal | S.F.Ratio | perc.alumni | Expend | Grad.Rate |
|---|---|---|---|---|---|---|---|---|---|
| Min. | 1780 | 96.0 | 250 | 8.00 | 24.0 | 2.50 | 0.00 | 3186 | 10.00 |
| 1st Qu. | 3597 | 470.0 | 850 | 62.00 | 71.0 | 11.50 | 13.00 | 6751 | 53.00 |
| median | 4200 | 500.0 | 1200 | 75.00 | 82.0 | 13.60 | 21.00 | 8377 | 65.00 |
| mean | 4358 | 549.4 | 1341 | 72.66 | 79.7 | 14.09 | 22.74 | 9660 | 65.46 |
| 3rd Qu. | 5050 | 600.0 | 1700 | 85.00 | 92.0 | 16.50 | 31.00 | 10830 | 78.00 |
| Max. | 8124 | 2340.0 | 6800 | 103.00 | 100.0 | 39.80 | 64.00 | 56233 | 118.00 |

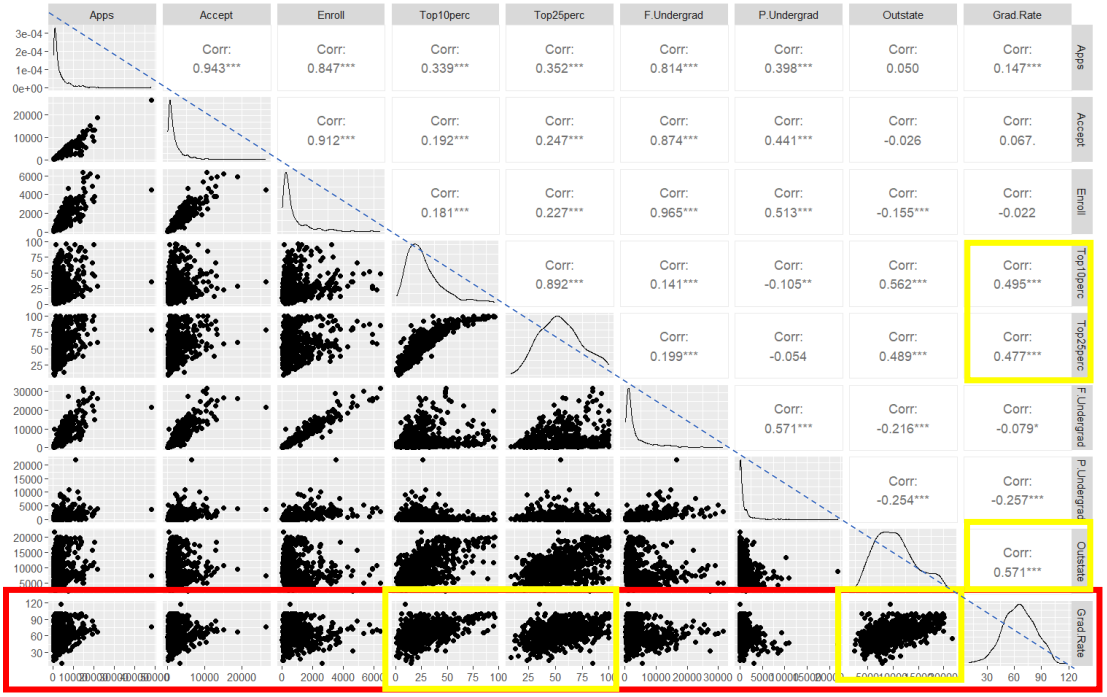
**Data Cleaning**

1. Checking summary(dataset) and duplicated
2. There is no <NA> value in this dataset
3. Checking boxplot of Grad.Rate and found 'Cazenovia College' which value is 118
4. Delete 'Cazenovia College', because rate

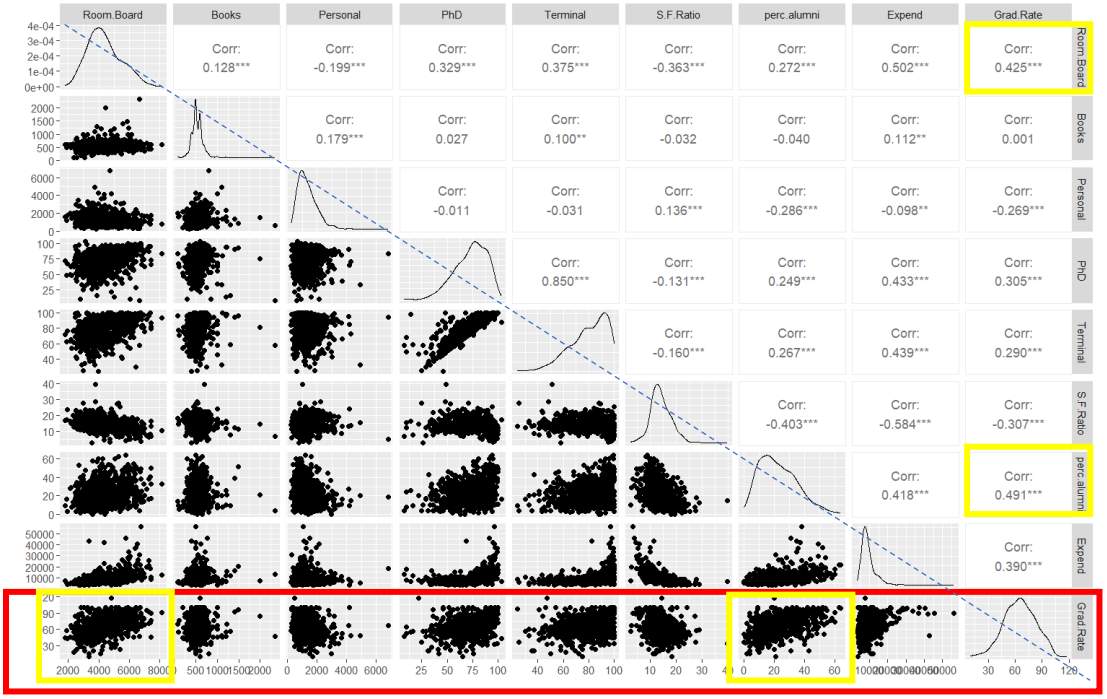


cannot over 100, and check other variables with over 100 in Grad.Rate

## Scatter, Correlation, and Frequency graph 1 with Grad.Rate




## Scatter, Correlation, and Frequency graph 1 with Grad.Rate



**Explanation:**

1. Highest Correlation with Grad.Rate is Outstate (0.571), Top10perc (0.495), perc.alumni (0.491), Top25perc (0.477), and Room.Board (0.425)
2. Red box contains all scatter plot of Grad.Rate and other variables. Yellow box is highly correlated factors. The relationship between the highly correlated factor and Grad.Rate is shown in the region symmetrical around the diagonal blue line

**Trends and interesting stuffs about dataset**

1. Outstate & Room.Board is related to cost according to detailed explanation about variables. And perc.alumni is about alumni loyalty and budgeting
2. Top10perc and Top25perc are about the grades of admitted students. The two factors are likely to be highly correlated with each other, so I will check that later
3. In addition to variables mentioned above, other variables also show quite a correlation.

## PART 1-2. PREREQUISITE for MODELING (Split the data)

**Preparing dataset**

1. Creating partition with the p = 0.70, train dataset has 543 observations and test dataset has 233 observations
2. Making train_x and test_x which does not have without Grad.Rate. Making train_y and train_y which only have Grad.Rate for using gv..glmnet function

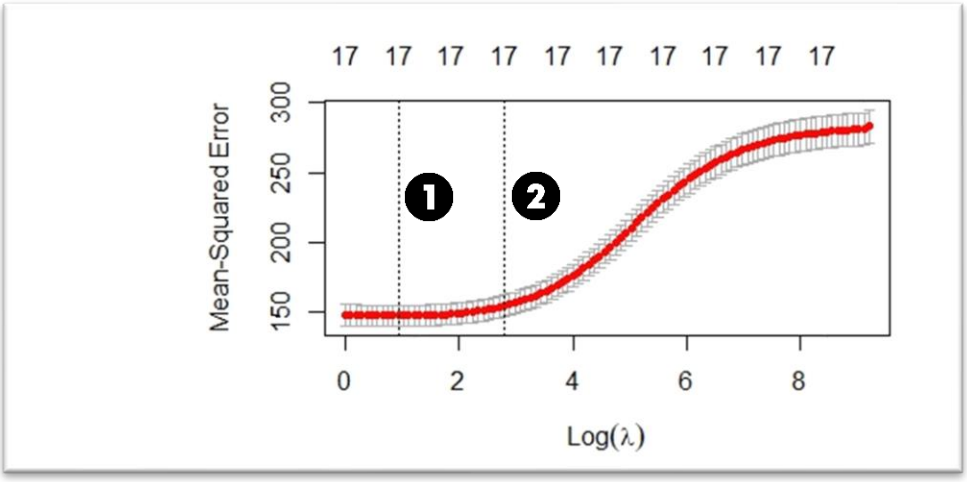
**Why Split the dataset**

1. Dividing train and test dataset allows cross-validation which refers to a set of methods for measuring the performance of a given predictive model on new test data sets (kassambara, 2018)
2. The main idea of splitting the dataset into a validation set is to prevent our model from overfitting i.e., the model becomes really good at classifying the samples in the training set but cannot generalize and make accurate classifications on the data it has not seen before (Pragati, 2023)
3. Dividing the train and test set allows cross-validation and prevent from overfitting. Cross-Validation is checking our model with a new test dataset to see accuracy of our model’s prediction

## PART 1-3. Ridge Regularization Finding the best Lambda

**Understanding:**

1. Ridge Regression has penalty equal to the square of the magnitude of coefficients. It means that all coefficients are shrunk by the same factor, in other words, the coefficients may approach zero, but they will never actually equal zero. This mean Ridge regularization cannot eliminate variables from the model (Goulding, 2022)
2. In this case, nfolds=10 which means randomly divide a dataset into k groups (which is folds) of roughly equal size and choose one of the groups and keep it. After that fit the model with remaining k-1 groups and calculate the test MSE with kept one group. Repeat this process k times, using a different set each time as the holdout set. Add all the MSE in each test and divide by k (ZACH, 2020)




| The Best Lambda | | | |
|---|---|---|---|
| log(λ$min) ① | 1.025761 | λ$min | 2.789219 |
| log(λ$1se) ② | 3.258571 | λ$1se | 26.01235 |

**Interpretation:**

1. X-axis is Log(lambda) and Y-axis is Mean-Squared Error (MSE). X-axis is not the lambda itself which has to be converted from log(lambda) to lambda
2. On the top, there are many '17's which means how many variables used while extracting log (lambda). In this case, we use Ridge which cannot eliminate variables, so everything is same as 17
3. The gray lines in each red dot is confident interval, and red dot represents the loss metric made through the cross-validation.
4. Left dot line (1) represents the minimum value of Lambda, and right dot line (2) is Lambda.1se which is maximum value within one standard error of the minimum. In other words, Lambda.1se is one standard error away from the minimum value of Lambda

## PART 1-4. Fit a model with the best Lambda in Ridge

**Model with Minimum of Lambda**

glmnet(x = train_x, y = train_y, alpha = 0, lambda = cv.ridge$lambda.min)

| | Df | %Dev | Lambda |
|---|---|---|---|
| 1 | 17 | 50.19 | 2.789 |

coef(model with min Lambda)

| 18 x 1 sparse Matrix of class "dgCMatrix" | s0 | λ$min | 1.9486<br>s0 |
|---|---|---|---|
| (Intercept) | 3.088712e+01 | Outstate | 6.318032e-04 |
| PrivateYes | 5.096437e+00 | Room.Board | 2.176416e-03 |
| Apps | 4.412696e-04 | Books | -1.237375e-03 |
| Accept | 2.049314e-04 | Personal | -1.850562e-03 |
| Enroll | 8.239522e-04 | phD | 2.940830e-02 |
| Top10perc | 8.554154e-02 | Terminal | 1.326194e-02 |
| Top25perc | 1.235753e-01 | S.F.Ratio | 6.397139e-02 |
| F.Undergrad | -9.038432e-05 | perc.alumni | 2.372240e-01 |
| P.Undergrad | -1.049755e-03 | Expend | -1.739864e-04 |

**Interpretation:**

1. This is coefficient of model with minimum Lambda. There are 17 variables which contain every variable of data set without deleting the variables
2. Minimum of Lambda by cv.glmnet function with train sets was 1.9486, These coefficient is calculated using Minimum of Lambda

**Model with one standard error of the minimum Lambda**

glmnet(x = train_x, y = train_y, alpha = 0, lambda = cv.ridge$lambda.1se)

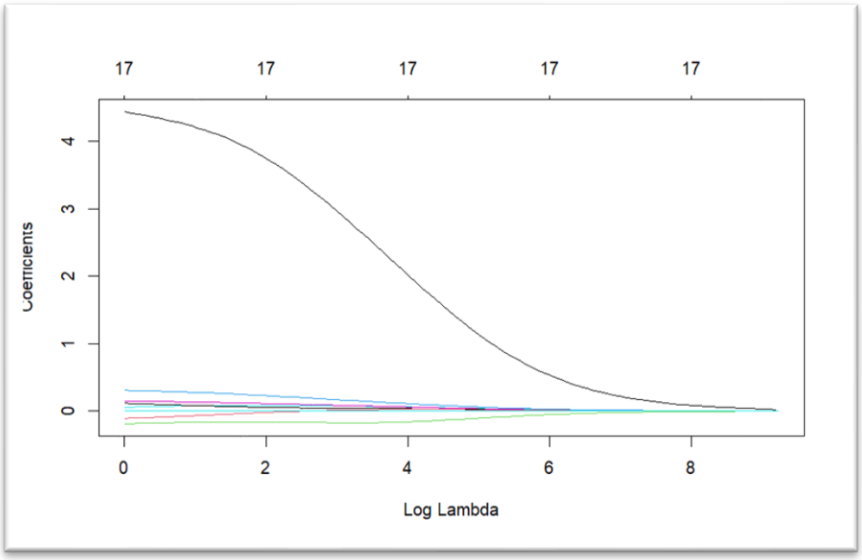
| | Df | %Dev | Lambda |
|---|---|---|---|
| 1 | 17 | 44.28 | 26.01 |

coef(model with 1se Lambda)

| 18 x 1 sparse Matrix of class "dgCMatrix" | s0 | λ$min | 28.93609<br>s0 |
|---|---|---|---|
| (Intercept) | 4.097982e+01 | Outstate | 4.116596e-04 |
| PrivateYes | 2.838922e+00 | Room.Board | 1.288993e-03 |
| Apps | 1.699354e-04 | Books | -5.793816e-04 |
| Accept | 1.615573e-04 | Personal | -1.366184e-03 |
| Enroll | 1.731171e-04 | phD | 3.868448e-02 |
| Top10perc | 7.559080e-02 | Terminal | 4.094918e-02 |
| Top25perc | 7.498129e-02 | S.F.Ratio | -1.100843e-01 |
| F.Undergrad | -2.800045e-05 | perc.alumni | 1.345650e-01 |
| P.Undergrad | -6.022945e-04 | Expend | 7.725665e-05 |

**Interpretation:**

1. This is coefficient of model with one standard error of the minimum Lambda
2. One standard error of the minimum Lambda by cv.glmnet function with train sets was 26.01235, These coefficient is calculated using One standard error of the minimum Lambda
3. After changing the minimum lambda to 1se.lambda, coefficients of almost every variable get closer to 0




4. As you can see in above graph, the black line represent the coefficient of Private Yes. The coefficient of PrivateYes is getting smaller as Log Lambda (which is X-aixs) getting bigger
5. Even though have to check the size of variables, only at the coefficient of PrivateYes takes quite a while to converge to 0 as Log Lambda increases

**coef(model with ols)**

18 x 1 sparse Matrix of class "dgCMatrix"

| | s0 | | s0 |
|---|---|---|---|
| (Intercept) | 26.6092487126 | Outstate | 0.0008044048 |
| PrivateYes | 6.0243476270 | Room.Board | 0.0025028823 |
| Apps | 0.0013312971 | Books | -0.0013928227 |
| Accept | -0.0014867628 | Personal | -0.0017969009 |
| Enroll | 0.0050149877 | phD | 0.0350196994 |
| Top10perc | -0.0143427758 | Terminal | -0.0071864257 |
| Top25perc | 0.2005783857 | S.F.Ratio | 0.1234007155 |
| F.Undergrad | -0.0005991068 | perc.alumni | 0.2624125308 |
| P.Undergrad | -0.0011104587 | Expend | -0.0003702202 |

**Interpretation:**

1. This is coefficient of model with ordinary least squared model without Lambda
2. Coefficients of this model is not close to 0 comparing with other Lambda models. It is without penalty model

| **Determining the performance of the Ridge model with RMSE** | |
|---|---|
| Root Mean Square error of each model | |
| train.rmse (one standard error model) | 12.55167 |
| test.rmse (one standard error model) | 14.54794 |
| ols.rmse | 14.21157 |

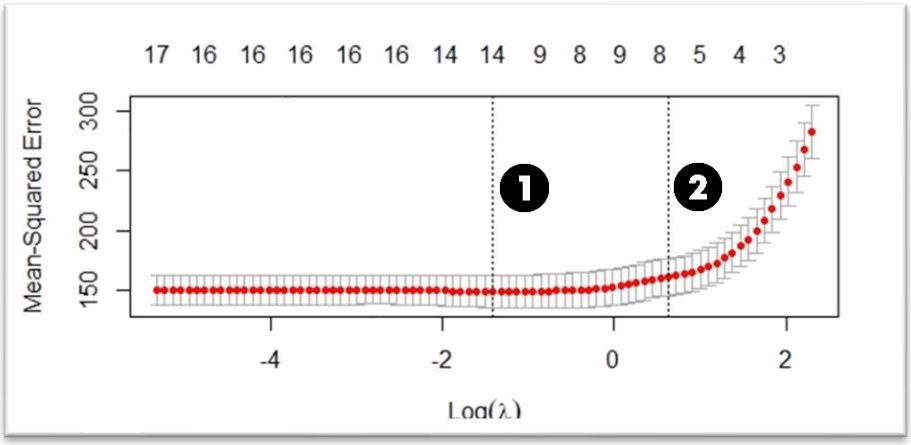
**Interpretation:**

1. Use one standard error model whose lambda is 26.01 and log (Lambda) is 3.26. The RMSE with train set is 12.55167 and the RESE with test set is 14.54794
2. The RMSE with train set is little bit smaller than test set, but the difference is not that big, so we can conclude that we don't have any overfitting going on here
3. The RMSE with ols model is 14.21157, which is not that different from others, therefore with Lambda 1se model, there is no overfitting

## PART 1-5. LASSO Regularization Finding the best Lambda

**Understanding:**

1. LASSO (L1) Regression has penalty equal to the absolute value of the magnitude of coefficients. It allows to actually reduce a coefficient to zero, in other words, we can eliminate variables (Goulding, 2022)
2. Use also nfolds=10 which means randomly divide a dataset into k groups (which is folds) of roughly equal size and choose one of the groups and keep it. To do cross-validation




**The Best Lambda**

| | | | |
|---|---|---|---|
| log(λ$min) ① | -1.416374 | λ$min | 0.242592 |
| log(λ$1se) ② | 0.6303681 | λ$1se | 1.878302 |

**Interpretation:**

1. X-axis is Log(lambda) and Y-axis is Mean-Squared Error(MSE). X-axis is not the lambda itself which has to be converted from log(lambda) to lambda
2. On the top, there are many numbers which means how many variables used while extracting log (lambda).
3. The gray lines in each red dot is confident interval, and red dot represents the loss metric made through the cross-validation.
4. Left dot line (1) represents the minimum value of Lambda, and right dot line (2) is Lambda.1se which is maximum value within one standard error of the minimum. In other words, Lambda.1se is one standard error away from the minimum value of Lambda
5. Left dot line(1) has significance of 14. This model with minimum lambda has 14 non-zero coefficients in the model
6. And right dot line (2) has significance of 8. This model with minimum lambda has 8 non-zero coefficients in the model. This is the simplest model that perform as best model

## PART 1-6. Fit a model with the best Lambda in LASSO

**Model with Minimum of Lambda**

| glmnet(x = train_x, y = train_y, alpha = 1, lambda = cv.lasso$lambda.min) | | | |
|---|---|---|---|
| | Df | %Dev | Lambda |
| 1 | 14 | 50.31 | 0.2426 |

| coef(model with min Lambda) | | | |
|---|---|---|---|
| 18 x 1 sparse Matrix of class "dgCMatrix" | | λ$min | 0.242592 |
| | s0 | | s0 |
| (Intercept) | 3.050567e+01 | Outstate | 6.933263e-04 |
| PrivateYes | 4.745639e+00 | Room.Board | 2.262846e-03 |
| Apps | 6.565080e-04 | Books | -2.545600e-04 |
| Accept | . | Personal | -1.651770e-03 |
| Enroll | 6.183751e-05 | phD | 1.019733e-02 |
| Top10perc | 2.047552e-02 | Terminal | . |
| Top25perc | 1.771778e-01 | S.F.Ratio | 2.813225e-02 |
| F.Undergrad | . | perc.alumni | 2.657025e-01 |
| P.Undergrad | -1.024760e-03 | Expend | -1.819468e-04 |

**Interpretation:**

1. These coefficient is calculated using Minimum of Lambda
2. The Accept, F.Undergrad and Terminal have '.' Which means these are eliminated variables in new model with the minimum Lambda
3. PrivateYes has 4.75 coefficient which is highest, but we have to think about the actual size of the variable itself
4. There are 14 (Df) variables with this minimum Lambda model, the model with min Lambda eliminate 'Accept', 'F.Undergrad' and 'Terminal', because is not significant or no enough regularization to kill one of them (Sergey, 2020)
5. The minimum of Lambda by cv.glmnet function with train sets was 0.242592

**Model with one standard error of the minimum Lambda**

| glmnet(x = train_x, y = train_y, alpha = 1, lambda = cv.lasso$lambda.1se) | | | |
|---|---|---|---|
| | Df | %Dev | Lambda |
| 1 | 8 | 44.41 | 1.878 |

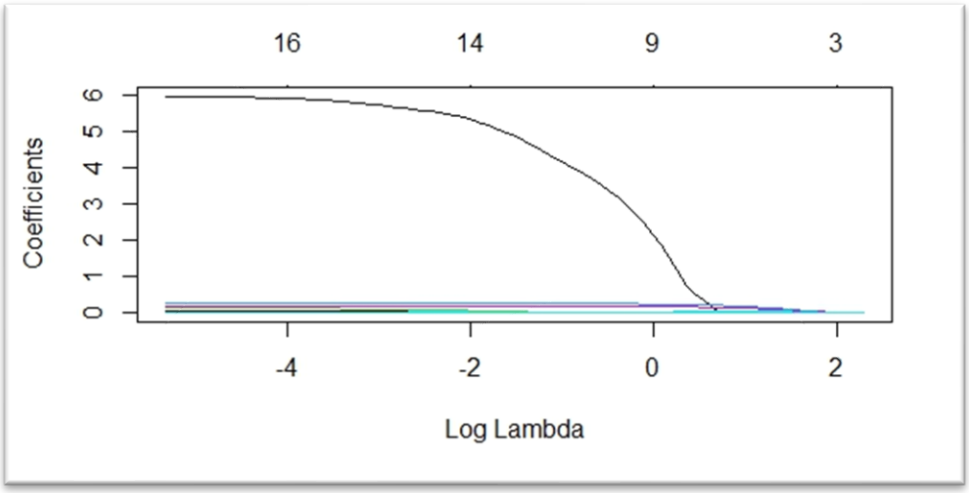
**Interpretation:**

1. Below coefficient of model with one standard error of the minimum Lambda
2. There are 9 (Df) variables with this 1se of the minimum Lambda model
3. Minimum of Lambda by cv.glmnet function with train sets was 1.312 which is greater than minimum lambda

| coef(model with 1se Lambda) | | | |
|---|---|---|---|
| 18 x 1 sparse Matrix of class "dgCMatrix" | | λ$min | 1.878302 |
| | s0 | | s0 |
| (Intercept) | 3.660906e+01 | Outstate | 9.318131e-04 |
| PrivateYes | 1.674888e-01 | Room.Board | 1.439722e-03 |
| Apps | . | Books | . |
| Accept | . | Personal | -1.274488e-04 |
| Enroll | . | phD | . |
| Top10perc | 2.730817e-02 | Terminal | . |
| Top25perc | 1.401610e-01 | S.F.Ratio | . |
| F.Undergrad | . | perc.alumni | 2.042931e-01 |
| P.Undergrad | -5.260135e-05 | Expend | . |

**Interpretation:**

1. These coefficients are calculated using 1se of the minimum Lambda
2. The Accept, F.Undergrad and Terminal have '.' same as the minimum lambda model. Furthermore Apps, Enroll, Books phD, S.F.Ratio and Expend have '.' Which means these are eliminated variables in the model with the 1se of the minimum Lambda. There are more variables which is eliminated comparing with the minimum lambda model, above
3. The other coefficients are shrinking to 0




4. As you can see in above graph, the black line represents the coefficient of Private Yes. The coefficient of PrivateYes is getting smaller as Log Lambda (which is X-aixs) getting bigger
5. On the top, as the log Lambda get greater, the number of variables are getting smaller like 16, 14, 9, and 3
6. There are a few lines which are black, blue and pink takes quite a while to converge to 0 as Log Lambda increases

**coef(model with ols)**

18 x 1 sparse Matrix of class "dgCMatrix"

| | s0 | | s0 |
|---|---|---|---|
| (Intercept) | 26.6092487126 | Outstate | 0.0008044048 |
| PrivateYes | 6.0243476270 | Room.Board | 0.0025028823 |
| Apps | 0.0013312971 | Books | -0.0013928227 |
| Accept | -0.0014867628 | Personal | -0.0017969009 |
| Enroll | 0.0050149877 | phD | 0.0350196994 |
| Top10perc | -0.0143427758 | Terminal | -0.0071864257 |
| Top25perc | 0.2005783857 | S.F.Ratio | 0.1234007155 |
| F.Undergrad | -0.0005991068 | perc.alumni | 0.2624125308 |
| P.Undergrad | -0.0011104587 | Expend | -0.0003702202 |

**Interpretation:**

1. This is coefficient of model with ordinary least squared model without Lambda
2. Coefficients of this model is far from 0 comparing with other Lambda models. It is without penalty model

**Determining the performance of the LASSO and Ridge model with RMSE**

Root Mean Square error of each model

| | LASSO (L1) | Ridge (L2) |
|---|---|---|
| train.rmse (one standard error model) | 12.53699 (-0.01468) | 12.55167 |
| test.rmse (one standard error model) | 14.56093 (0.01299) | 14.54794 |
| ols.rmse | 14.21157 | 14.21157 |

**Interpretation:**

1. Use one standard error model whose lambda is 1.878302 and log (Lambda) is 0.6303681 The RMSE with train set is 12.53699 and the RESE with test set is 14.56093
2. The RMSE with train set is little bit higher than test set, but the difference is not that big, so we can conclude that we don't have any overfitting going on here
3. The RMSE with ols model is 14.21157, which is not that different from others, therefore with Lambda 1se model, there is no overfitting

**Which model perform better:**

1. In my opinion, LASSO (L1) regression performed better than Ridge (L2) regression, because LASSO have smaller variables whose number is 9, but the Root Mean Squared Error is smaller in train set (12.53699) and RMSE in test set is also similar to Ridge's
2. The model is simpler but the performance is almost same, so it's easy to interpret and easy to understand. Also, more likely to avoid multicollinearity and overfitting

## PART 1-7. COMPARISON AND STEPWISE MODELING

### Stepwise Modeling

step(lm(Grad.Rate~., data=C1), direction = 'both')

lm(formula = Grad.Rate ~ Private + Apps + Accept + Top25perc + P.Undergrad + Outstate + Room.Board + Personal + PhD + Terminal + perc.alumni + Expend, data = C1)

Residuals

| Min | 1Q | Median | 3Q | Max |
|---|---|---|---|---|
| -51.449 | -7.380 | -0.367 | 7.378 | 51.207 |

Coefficients:

| LASSO | Stepwise | Estimate | Std. Error | t value | Pr(>\|t\|) |
|---|---|---|---|---|---|
| (Intercept) | (Intercept) | 32.7791997 | 3.3291914 | 9.846 | < 2e-16 *** |
| PrivateYes | PrivateYes | 3.5359337 | 1.6099868 | 2.196 | 0.028374 * |
| | Apps | 0.0014322 | 0.0004092 | 3.500 | 0.000493 *** |
| Top10perc | Accept | -0.0009485 | 0.0006256 | -1.516 | 0.129902 |
| Top25perc | Top25perc | 0.1627464 | 0.0321719 | 5.059 | 5.29e-07 *** |
| P.Undergrad | P.Undergrad | -0.0016014 | 0.0003611 | -4.435 | 1.06e-05 *** |
| Outstate | Outstate | 0.0010593 | 0.0002253 | 4.701 | 3.06e-06 *** |
| Room.Board | Room.Board | 0.0017177 | 0.0005754 | 2.985 | 0.002925 ** |
| Personal | Personal | -0.0016904 | 0.0007403 | -2.283 | 0.022676 * |
| | PhD | 0.1244548 | 0.0550710 | 2.260 | 0.024109 * |
| | Terminal | -0.0923458 | 0.0605889 | -1.524 | 0.127889 |
| perc.alumni | perc.alumni | 0.2820839 | 0.0479949 | 5.877 | 6.23e-09 *** |
| | Expend | -0.0004587 | 0.0001335 | -3.436 | 0.000623 *** |

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 12.58 on 763 degrees of freedom
Multiple R-squared: 0.4664, Adjusted R-squared: 0.458
F-statistic: 55.58 on 12 and 763 DF, p-value: < 2.2e-16

**Interpretation:**

1. The stepwise regression (or stepwise selection) consists of iteratively adding and removing predictors, in the predictive model, in order to find the subset of variables in the data set resulting in the best performing model, that is a model that lowers prediction error (STHDA, n.d.)
2. The stepwise model has more variables than the LASSO model. LASSO model has 8 variables, but Stepwise model uses 12 variables to make fitted model. Since the step-by-step model repeatedly adds and removes variables, the model is created considering various factors

## LASSO (better than Ridge) vs Stepwise Modeling

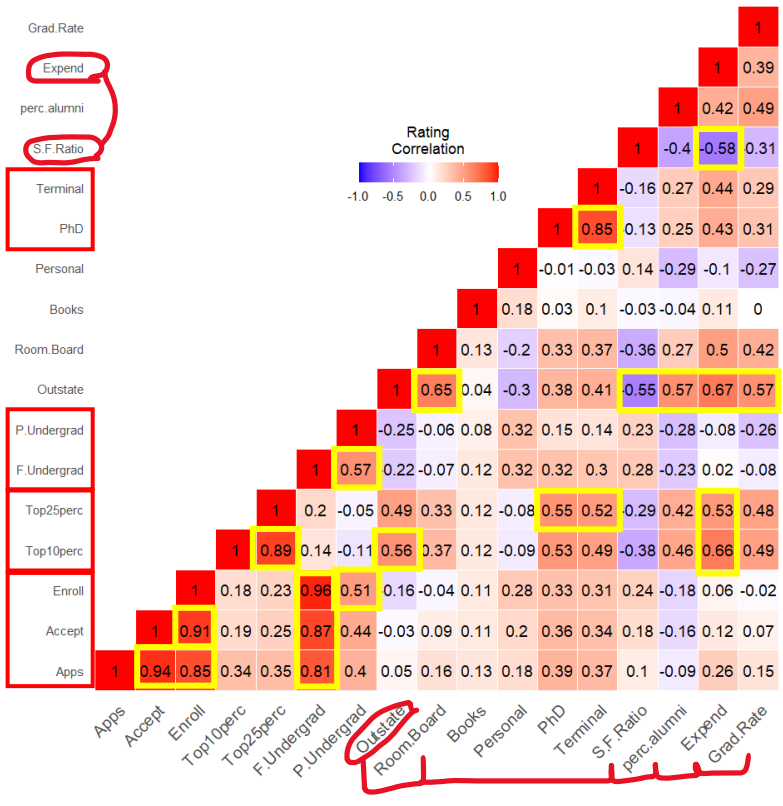
### Determining the performance of the LASSO and Stepwise Modeling

| Root Mean Square error of each model | | |
|---|---|---|
| | LASSO (L1) | Stepwise Modeling |
| train.rmse | 12.53699 | 11.89416 (-0.64283) |
| test.rmse | 14.56093 | 13.72442 (-0.83651) |

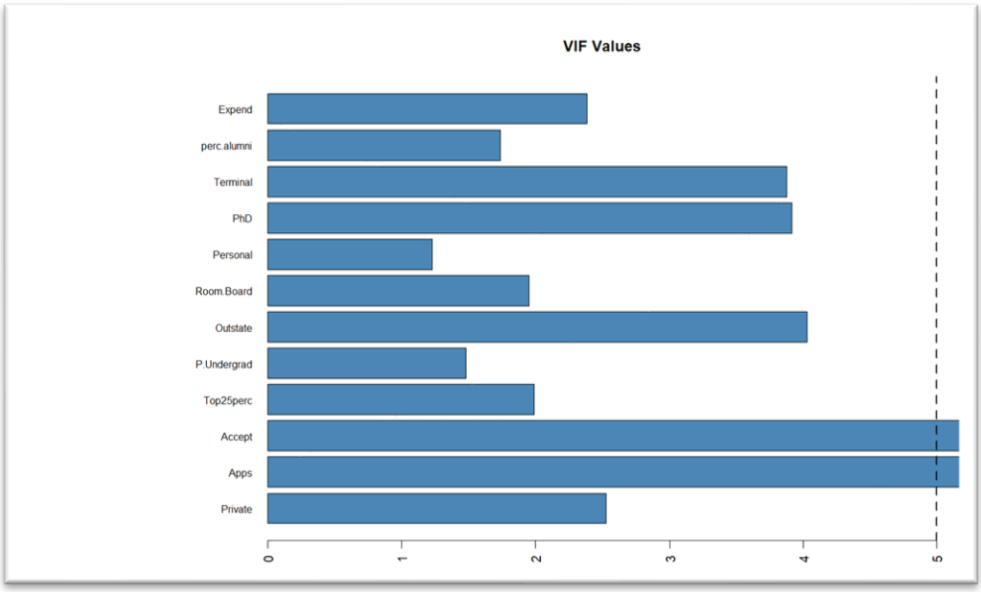
**Interpretation:**

1. In my opinion Stepwise model is better than LASSO which is better than Ridge in this specific case. we can't simply say which model is always good, we need to see the details of model, data, question and outputs
2. RMSE: Root Mean Square Error is the measure of how well a regression line fits the data points. RMSE can also be construed as Standard Deviation in the residuals (vivekchaudhary90, 2022). Lower RMSE indicates better model
3. Looking at the table above, in the RMSE calculation, stepwise modeling showed a lower RMSE in both train and test data
4. LASSO model contains the Top10perc and Top25perc together. In the correlation matrix below, the two variables is one of the highly correlated variables which correlation is 0.89. The fact the model still keeps both may mean two things: either the model deems both important or there no enough regularization to kill one of them (Sergey, 2020)
5. I think the LASSO model deems both important. Anyway they are correlated, so it's better to not use both variables in one model

### Correlation Matrix of numeric variables



**VIF Values of Stepwise model**




**Interpretation:**

1. For a given predictor (p), multicollinearity can assessed by computing a score called the variance inflation factor (or VIF), which measures how much the variance of a regression coefficient is inflated due to multicollinearity in the model (STHDA, n.d.)
2. The VIF values of stepwise model have bigger than 5 values, therefore, I have to manually delete the variables in my model
3. After I compare the AIC of possible models, I decide this model as best model

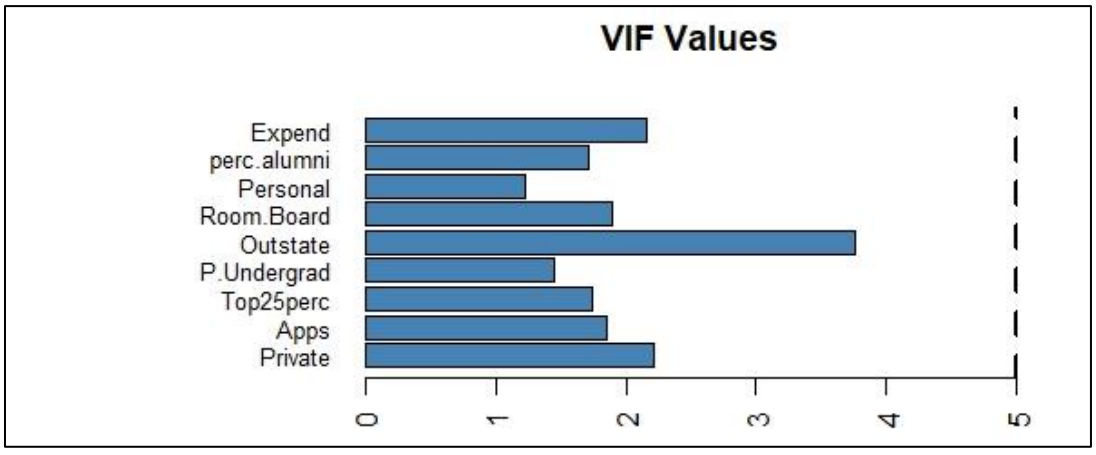
---

**Manually made new model**

lm(formula = Grad.Rate ~ Private + Apps + Top25perc + P.Undergrad + Outstate + Room.Board + Personal + perc.alumni + Expend, data = C1)

---

**VIF Values of New model**



**ANSWER FOR THE QUESTION**

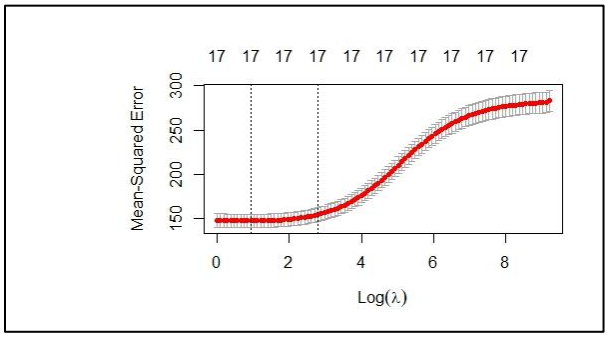
2. (Ridge) Use the cv.glmnet function to estimate the lambda.min and lambda.1se values. Compare and discuss the values.

**The Best Lambda**

| | | | |
|---|---|---|---|
| log(λ$min) ① | 1.025761 | λ$min | 2.789219 |
| log(λ$1se) ② | 3.258571 | λ$1se | 26.01235 |

The minimum Lambda is 2.78219 whose log (Lambda) is 1.025761. Lambda.1se is one standard error of the minimum lambda. The lambda of 1se is 26.01235 whose log (Lambda) is 3.258571. The difference between two seems like huge. The performance of the model with both is not that different.

3. (Ridge) Plot the results from the cv.glmnet (Ridge) function provide an interpretation. What does this plot tell us?




X-axis is Log(lambda) and Y-axis is Mean-Squared Error(MSE). On the top, there are many '17's which means how many variables used while extracting log (lambda). Ridge regression cannot eliminate variables, so everything is same as 17. Left dot line represents the minimum value of Lambda, and right dot line is Lambda.1se which is maximum value within one standard error of the minimum.

4. (Ridge) Fit a Ridge regression model against the training set and report on the coefficients. Is there anything interesting?
   On page 9, there are coefficient tables with minimum lambda and 1se lambda. After changing the minimum lambda to 1se.lambda, coefficients of almost every variable get closer to 0. What's interesting is that moving closer to 0 means that positive numbers get smaller, and negative numbers get bigger.

5. (Ridge) Determine the performance of the fit model against the training set by calculating the root mean square error (RMSE). sqrt(mean((actual - predicted)^2))
6. (Ridge) Determine the performance of the fit model against the test set by calculating the root mean square error (RMSE). Is your model overfit?

**Determining the performance of the Ridge model with RMSE**

| Root Mean Square error of each model | |
|---|---|
| train.rmse (one standard error model) | 12.55167 |
| test.rmse (one standard error model) | 14.54794 |
| ols.rmse | 14.21157 |

The RMSE with train set is smaller than test set, but the difference is not that big,

so we can conclude that we don't have any overfitting going on here. The RMSE with ols model is 14.21157, therefore with Lambda 1se model is better than the ols model. There is no overfitting because the difference between each RMSE is not that huge.

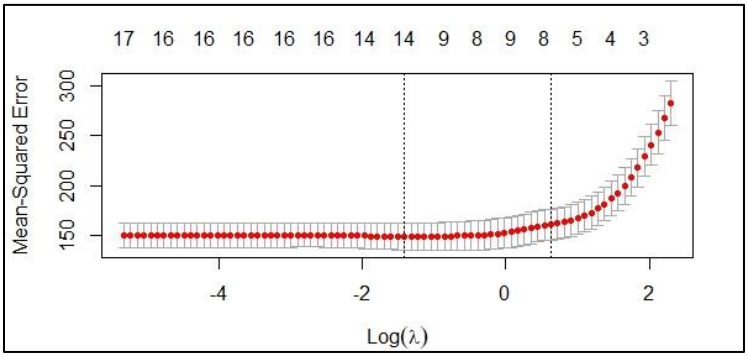
7. (LASSO) Use the cv.glmnet function to estimate the lambda.min and lambda.1se values. Compare and discuss the values.

**The Best Lambda**

| | | | |
|---|---|---|---|
| log(λ$min) ① | -1.416374 | λ$min | 0.242592 |
| log(λ$1se) ② | 0.6303681 | λ$1se | 1.878302 |

The minimum Lambda is 0.242592 whose log (Lambda) is -1.416374. Lambda.1se is one standard error of the minimum lambda. The lambda of 1se is 1.878302 whose log (Lambda) is 0.6303681. The difference between two seems also huge. The performance of the model with both is not that different. It seems that a larger lambda is not needed since we can eliminate the variable compared to the Ridge model.

8. (LASSO) Plot the results from the cv.glmnet function provide an interpretation. What does this plot tell us?




On the top, there are many numbers which means how many variables used while extracting log (lambda). Left dot line has significance of 14. This model with minimum lambda has 14 non-zero coefficients in the model. And right dot line has significance of 8. This model with minimum lambda has 8 non-zero coefficients in the model. This is the simplest model that perform as best model.

9. (LASSO) Fit a LASSO regression model against the training set and report on the coefficients. Do any coefficients reduce to zero? If so, which ones?

On page 13 & 14, there are coefficient tables with minimum lambda and 1se lambda. There are 14 (Df) variables with this minimum Lambda model, the model with min Lambda eliminate 'Accept', 'F.Undergrad', and 'Terminal', because is not significant or no enough regularization to kill one of them (Sergey, 2020). In model with 1se lambda, 'Accept', 'F.Undergrad', and 'Terminal' have '.' Furthermore, Apps, Enroll, Books phD, S.F.Ratio and Expend are eliminated.

10. (LASSO) Determine the performance of the fit model against the training set by calculating the root mean square error (RMSE). sqrt(mean((actual - predicted)^2))
11. (LASSO) Determine the performance of the fit model against the test set by calculating the root mean square error (RMSE). Is your model overfit?

| Root Mean Square error of each model | | |
|---|---|---|
| | LASSO (L1) | Ridge (L2) |
| train.rmse (one standard error model) | 12.53699 (-0.01468) | 12.55167 |
| test.rmse (one standard error model) | 14.56093 (0.01299) | 14.54794 |
| ols.rmse | 14.21157 | 14.21157 |

As same as Ridge model, the RMSE with train set is little bit higher than test set, but the difference is not that big, so we can conclude that we don't have any overfitting going on here. The RMSE with ols model is 14.21157, therefore with Lambda 1se model with train set is better than the ols model. There is no overfitting because the difference between each RMSE is not that huge.

12. (LASSO) Which model performed better and why? Is that what you expected?
LASSO (L1) regression performed better than Ridge (L2) regression, because LASSO have smaller variables whose number is 8, but the Root Mean Squared Error is smaller in train set (12.53699) and RMSE in test set is also similar to Ridge'. The model is simpler, but the performance is almost same, so it's easy to interpret and easy to understand. Also, more likely to avoid multicollinearity and overfitting.

13. (LASSO) Refer to the Intermediate Analytics Feature Selection R.pdf document for how to perform stepwise selection and then fit a model. Did this model perform better or as well as Ridge regression or LASSO? Which method do you prefer and why?

| Root Mean Square error of each model | | |
|---|---|---|
| | LASSO (L1) | Stepwise Modeling |
| train.rmse | 12.53699 | 11.89416 (-0.64283) |
| test.rmse | 14.56093 | 13.72442 (-0.83651) |

I think in this case, stepwise model is better than LASSO which is better than Ridge. In the RMSE calculation, stepwise modeling showed a lower RMSE in both train and test data. LASSO model contains the Top10perc and Top25perc together. The two variables are highly correlated variables which correlation is 0.89 in correlation matrix. The LASSO model deems both variable as important. Anyway they are correlated, so it's better to not use both variables in one model. But we can't simply say which model is always good, and we need to choose the model carefully in relation to the given data and what the business question is.

In this case the stepwise model have the VIF values which are bigger than 5 values, therefore, I have to manually delete the variables in my model. My final

model is the model eliminate accept which is highly correlated to Apps and also delete the PhD and Terminal refer to LASSO model.

## CONCLUSION

In this project, I learned how to run and interpret LASSO and Ridge Regularization. Lasso regularization adds a penalty which is equal to the absolute value of the magnitude of coefficient. This eliminates the variables while making coefficients closer to zero. It helps finding best model which can avoid high levels of multicollinearity. Ridge Regularization add penalty which is equal to the square of the magnitude of coefficients. The biggest difference of two is Ridge cannot eliminate the variables while lambda bigger, because the coefficient will shrink to zero but will never equal zero. I analyzed cv.glmnet () and checked how several variables were used in regularization using lambda.

After comparing the stepwise method and regularization and looking at various cases on the google, I found that there is no one method that is always right. Depending on the data and the problem looking for, I need to decide which method to choose and also there is need to manually add or delete variables. In other words, I have to judge which method is more correct in specific situation. In this case, I used RMSE. However, rethought about what elements the model explains. I learned that I have to think about which factors should be considered. For the same performance, you should choose a simpler model.

# REFERENCE


Goulding, Thomas. 2023. Lesson 4-1 — regularization. Retrieved from https://northeastern.instructure.com/courses/131212/pages/lesson-4-1-regularization?module_item_id=8227336

ZACH. (2020, November 11). Ridge regression in R (step-by-step). STATOLOGY. Retrieved from https://www.statology.org/ridge-regression-in-r/

ZACH. (2020, November 13). Lasso regression in R (step-by-step). STATOLOGY. Retrieved from https://www.statology.org/lasso-regression-in-r/

ZACH. (2020, November 4). An easy guide to K-Fold cross-validation. STATOLOGY. Retrieved from https://www.statology.org/k-fold-cross-validation/

STHDA. (n.d.). Cross-validation essentials in R. Retrieved from http://www.sthda.com/english/articles/38-regression-model-validation/157-cross-validation-essentials-in-r/

Pragati Baheti. (2023, February 3). Train test validation split: how to & best practices [2023]. V7. Retrieved from https://www.v7labs.com/blog/train-validation-test-set

Deepika Singh. (2019, May 17). Linear, Lasso, and Ridge regression with scikit-learn. PLURALSIGHT. Retrieved from https://www.pluralsight.com/guides/linear-lasso-ridge-regression-scikit-learn

Karolis Koncevicius. (2021, December 17). Ridge trace plot - interpretation. Cross Validated. Retrieved from https://stats.stackexchange.com/questions/557428/ridge-trace-plot-interpretation

Sergey Bushmanov. (2020, February 9). Lasso regression won't remove 2 features which are highly correlated. stackoverflow. Retrieved from https://stackoverflow.com/questions/60124199/lasso-regression-wont-remove-2-features-which-are-highly-correlated

STHDA. (n.d.). Stepwise regression essentials in R. Retrieved from http://www.sthda.com/english/articles/37-model-selection-essentials-in-r/154-stepwise-regression-essentials-in-r/

Nizamuddin Siddiqui. (2020, October 17). How to calculate root mean square error for linear model in R. tutorialspoint. Retrieved from https://www.tutorialspoint.com/how-to-calculate-root-mean-square-error-for-linear-model-in-r

vivekchaudhary90. (2022, June 6). ML | mathematical explanation of RMSE and R-squared error. geeksforgeeks. Retrieved from https://www.geeksforgeeks.org/ml-mathematical-explanation-of-rmse-and-r-squared-error/

STHDA. (n.d.). Multicollinearity essentials and VIF in R. Retrieved from http://www.sthda.com/english/articles/39-regression-model-diagnostics/160-multicollinearity-essentials-and-vif-in-r/

STHDA. (n.d.). ggplot2 Quick correlation matrix heatmap - R software and data visualization. Retrieved from http://www.sthda.com/english/wiki/ggplot2-quick-correlation-matrix-heatmap-r-software-and-data-visualization

# R-Codes

```
library(ISLR)
library(psych)
library(dplyr)
library(GGally)
library(glmnet)
library(Metrics)
library(scales)
library(reshape2)
library(car)

C1 <- College
headtail(C1, 5)

#########
# Split data isto train and test sets
#########

# Checking for NA values
complete.cases(C1)
which(!complete.cases(C1))
vis_miss(C1)
sum(is.na(C1))
sum(is.null(C1))

# Checking for Duplication
duplicated(C1)
anyDuplicated(C1)

#####
#EDA
####

# descriptive analysis
summary(C1)
describe(C1)

corr <- select_if(C1, is.numeric)

# correlation of half with Grad.Rate
corr1 <-  corr[ ,c(1:8, 17)]
str(corr1)
ggpairs(corr1)

# correlation of another half with Grad.Rate
```

```
corr2 <-  corr[ ,9:17]
str(corr2)
ggpairs(corr2)

# boxplot
boxplot(C1$Grad.Rate, horizontal=TRUE)
text(x=fivenum(C1$Grad.Rate),labels=fivenum(C1$Grad.Rate),y=1.35)
fivenum(C1$Grad.Rate)

Grad.out <- filter(C1, Grad.Rate > 100)
Grad.out

C1 <- C1[!(row.names(C1) %in% c("Cazenovia College")),]
str(C1)

#########
# Split data isto train and test sets
#########
set.seed(123)
trainIndex <- sample(x=nrow(C1), size= nrow(C1)*0.7)
train <- C1[trainIndex,]
test <- C1[-trainIndex,]

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

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

##############
# Find best values of lamda in Ridge
##############

# Find the best lambda using cross-validation
set.seed(123)
cv.ridge <- cv.glmnet(train_x, train_y, nfolds = 10, alpha=0)
plot(cv.ridge)
summary(cv.ridge)

ridge2 <- glmnet(train_x, train_y, nfolds = 10, alpha=0)
plot(ridge2, xvar = "lambda")

# optimal value of lambda; minimizes the prediction error
# lambda min - minimizes out of sample loss
```

```
# labmda 1se - largest value of lambda within 1 standard error of lambda min
log(cv.ridge$lambda.min)
log(cv.ridge$lambda.1se)

cv.ridge$lambda.min
cv.ridge$lambda.1se

###########
# Fit models based on lambda
###########

# Fit the finla model on the training data using lambda.min
# alpha = 1 for Lasso (L2)
# alpha = 0 for Ridge (L1)
ridge.model.min <- glmnet(train_x, train_y, alpha = 0, lambda = cv.ridge$lambda.min)
ridge.model.min


# Display regression coefficients
coef(ridge.model.min)

# Fit the final model on the training data using lambda.1se
ridge.model.1se <- glmnet(train_x, train_y, alpha =0, lambda = cv.ridge$lambda.1se)
ridge.model.1se

# Display regression coefficients
coef(ridge.model.1se)

# Display coefficients of ols model with no regularization
ols <- lm(Grad.Rate~., data = train)
coef(ols)

# View RMSE of full model
preds.ols <- predict(ols, new = test)
rmse(test$Grad.Rate, preds.ols)

head(preds.ols,10)

###############
#Train set predictions
#############
# Make predictions on the test data using lambda.min
preds.train.ridge <-  predict(ridge.model.1se, newx = train_x)
train.rmse.ridge <-  rmse(train_y, preds.train.ridge)

head(preds.train.ridge,10)
```

```
###############
# Test set predictions
###############
preds.test.ridge <-  predict(ridge.model.1se, newx = test_x)
test.rmse.ridge <-  rmse(test_y, preds.test.ridge)

head(preds.test.ridge ,10)

# Compare rmse values
train.rmse.ridge
test.rmse.ridge

################ Devide line ############
# Find the best lambda using cross-validation
set.seed(123)
cv.lasso <- cv.glmnet(train_x, train_y, nfolds = 10, alpha=1)
plot(cv.lasso)

# optimal value of lambda; minimizes the prediction error
# lambda min - minimizes out of sample loss
# labmda 1se - largest value of lambda within 1 standard error of lambda min
log(cv.lasso$lambda.min)
log(cv.lasso$lambda.1se)

cv.lasso$lambda.min
cv.lasso$lambda.1se

###########
# Fit models based on lambda
###########

# Fit the finla model on the training data using lambda.min
# 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

# Display regression coefficients
coef(model.min)

# Fit the final model on the training data using lambda.1se
model.1se <- glmnet(train_x, train_y, alpha =1, lambda = cv.lasso$lambda.1se)
model.1se

# Display regression coefficients
```

```
coef(model.1se)

# Plot a Lambda graph
lasso2 <- glmnet(train_x, train_y, nfolds = 10, alpha=1)
summary(lasso2)
plot(lasso2, xvar = "lambda")

# Display coefficients of ols model with no regularization
ols <- lm(Grad.Rate~., data = train)
coef(ols)

# View RMSE of full model
preds.ols <- predict(ols, new = test)
rmse(test$Grad.Rate, preds.ols)

###############
#Train set predictions
#############
# Make predictions on the test data using lambda.min
preds.train <-  predict(model.1se, newx = train_x)
train.rmse <-  rmse(train_y, preds.train)

###############
# Test set predictions
###############
preds.test <-  predict(model.1se, newx = test_x)
test.rmse <-  rmse(test_y, preds.test)

# Compare rmse values
train.rmse
test.rmse

# Stepwise selection
step(lm(Grad.Rate~., data=C1), direction = 'both')
model_step <- step(lm(Grad.Rate~., data=C1), direction = 'both')
summary(model_step)

#Correlation heatmap Continuous
cordata3 <- select_if(C1, is.numeric)
aR <- round(cor(cordata3), 2)

cor(cordata3)
aR <- round(cor(cordata3), 2)

get_upper_tri<-function(aR){
  aR[lower.tri(aR)] <- NA
```

```
  return(aR)}

upper_tri <- get_upper_tri(aR)
upper_tri
melted_cormat <- melt(upper_tri, na.rm = TRUE)
melted_cormat

reorder_cormat <- function(aR){
  # Use correlation between variables as distance
  dd <- as.dist((1-cormat)/2)
  hc <- hclust(dd)
  aR <-aR[hc$order, hc$order]}

aR <- reorder_cormat(aR)
aR
upper_tri <- get_upper_tri(aR)

melted_cormat <- melt(upper_tri, na.rm = TRUE)

#NEW
ggheatmap <- ggplot(melted_cormat, aes(Var2, Var1, fill = value))+
  geom_tile(color = "white")+
  scale_fill_gradient2(low = "blue", high = "red", mid = "white",
                       midpoint = 0, limit = c(-1,1), space = "Lab",
                       name="Rating\nCorrelation") +
  theme_minimal()+ # minimal theme
  theme(axis.text.x = element_text(angle = 45, vjust = 1,
                                   size = 12, hjust = 1))+
  coord_fixed()

ggheatmap +
  geom_text(aes(Var2, Var1, label = value), color = "black", size = 4) +
  theme(
    axis.title.x = element_blank(),
    axis.title.y = element_blank(),
    panel.grid.major = element_blank(),
    panel.border = element_blank(),
    panel.background = element_blank(),
    axis.ticks = element_blank(),
    legend.justification = c(1, 0),
    legend.position = c(0.6, 0.7),
    legend.direction = "horizontal")+
  guides(fill = guide_colorbar(barwidth = 7, barheight = 1,
                               title.position = "top", title.hjust = 0.5))

# View RMSE of full model with test
```

```
preds.step <- predict(model_step, new = train)
rmse(train$Grad.Rate, preds.step)

# View RMSE of full model with test
preds.step <- predict(model_step, new = test)
rmse(test$Grad.Rate, preds.step)


# Finding RMSE of step model
sqrt(mean(model_step$residuals^2))

# checking vif
vif_step <- vif(model_step)
vif_step

opar <- par(no.readonly = TRUE)
par(fig=c(0.2, 1, 0, 1))
barplot(vif_step, main = "VIF Values", horiz = TRUE, las=2, cex.names=0.8, xlim=c(0,5), col =
"steelblue")

#add vertical line at 5
abline(v = 5, lwd = 3, lty = 2)

# new model
new.model <-  lm(formula = Grad.Rate ~ Private + Apps + Top25perc + P.Undergrad + Outstate
+ Room.Board + Personal + perc.alumni + Expend, data = C1)
AIC(new.model)
AIC(model_step)

# checking vif
vif_new <- vif(new.model)
vif_new

opar <- par(no.readonly = TRUE)
par(fig=c(0.2, 1, 0, 1))
barplot(vif_new, main = "VIF Values", horiz = TRUE, las=2, cex.names=0.8, xlim=c(0,5), col =
"steelblue")

#add vertical line at 5
abline(v = 5, lwd = 3, lty = 2)

str(train_x)
train_x <- as.data.frame(train_x)
colnames(train_x)[which(names(train_x) == "PrivateYes")] <- "Private"
```