Logo

Description automatically generated

**College of Professional Studies**

**Northeastern University San Jose**

**MPS Analytics**

**Course: ALY6015: Intermediate Analytics**

**Assignment:**

Module 4 Assignment -  Regularization

**Submitted on:**

February 5, 2023

**Submitted to:**  **Submitted by:**

Professor: PAROMITA GUHA NIKSHITA RANGANATHAN

# **INTRODUCTION**

Regularization is a machine learning strategy that avoids overfitting, a problem that frequently arises during model training. Overfitting is the result of a model that has been incredibly complex and overtrained on the training data, which results in a poor generalization of new data.

There are various types of regularization, such as Elastic Net regularization, L1 regularization (also known as Lasso regularization), and L2 regularization (also known as Ridge regularization).

It is a crucial machine learning technique that helps prevent overfitting, improve generalization, reduce variance, and prevent coefficient bloat. By using regularization, machine learning practitioners can build more robust and accurate models capable of making predictions on new data.

**Lasso:**

Lasso is a type of regularization method used in linear and logistic regression models to improve model interpretability. The coefficients' absolute value are the same as the penalty term added by Lasso.The selection of features is carried out using this technique in statistical modeling and machine learning. This means it helps to identify the most important features that contribute to the target variable and eliminates the less important features.

Here is an example of how Lasso can be used in a real-world scenario: Suppose you have a dataset that contains information about various properties and their corresponding sales prices. The dataset includes features such as square footage, number of bedrooms, number of bathrooms, location, and age of the property.

To predict the sales price of a property, you build a regression model that takes into account these features. The sales price, however, may not be significantly impacted by any of these aspects. For example, the age of the property might not be an important factor in determining the sales price.

In this scenario, you can use Lasso Regression to identify the most important features that contribute to the sales price and exclude the rest.

**Ridge:**

In Ridge Regression, a penalty term is added to the sum of the squared residuals, which is the difference between the actual value and the predicted value for each data point. This approach is useful in cases where the number of features is large, and all the features are relevant and have a strong effect on the target variable. Ridge Regression is better when there is multicollinearity between the features, as it can provide a more stable and reliable solution compared to Lasso.

This is due to the fact that Lasso performs feature selection by effectively removing unnecessary features from the model by decreasing their coefficients to zero. As a result, the model has fewer characteristics and is simpler to read and comprehend.

**Purpose:**

The purpose of the assignment is to use the College dataset from the ISLR library to implement Ridge and Lasso regression models using the glmnet() package in R. The objective is to predict Grad.Rate using regularization techniques and understand the relationships among variables through regularization methods.

The dataset includes the following features:

|  |  |  |
| --- | --- | --- |
| **No** | **Feature** | **Dictionary** |
| 1 | Private | whether the college is private or public (binary variable) |
| 2 | Apps | total applications received |
| 3 | Accept | total applications accepted |
| 4 | Enroll | total students enrolled |
| 5 | Top10perc | Top 10% - new students from their high school class (in %) |
| 6 | Top25perc | Top 25% - new students from their high school class (in %) |
| 7 | F.Undergrad | the number of undergraduates enrolled full-time |
| 8 | P.Undergrad | the number of undergraduates enrolled part-time |
| 9 | Outstate | Tuition (out-of-state) |
| 10 | Room.Board | Cost for the room and board |
| 11 | Books | Cost of books (Estimate) |
| 12 | Personal | Personal spending (Estimate) |
| 13 | PhD | faculty with PhD. (in %) |
| 14 | Terminal | faculty with a terminal degree (in %) |
| 15 | S.F.Ratio | student/faculty ratio |
| 16 | perc.alumni | alumni who donate (in %) |
| 17 | Expend | the instructional cost for each student |
| 18 | Grad.Rate | rate of graduating |

*Table 1: Features of the College Data Set with their dictionary*

**ANALYSIS & INTERPRETATION**

* Installing and loading the libraries
* Importing the dataset

In this step, the "College" dataset was imported into the R environment.

* Understanding the College dataset
  + The college dataset includes a variety of information on U.S. colleges and universities. It contains variables such as the college name, private/public status, region, acceptance rate, tuition fees, enrollment, etc. By analyzing this dataset, we can gain insights and make predictions about various aspects of a college education.
  + This dataset contains a total of 777 observations, each with 18 different variables.

**Text, letter

Description automatically generated**

**Figure 1 – glimpse()**

* Descriptive statistics
* The College dataset from ISLR has no missing values, which means that the analysis can be conducted without the need for any imputation techniques.
* It consists of one variable (Private) which is a factor and 17 numerical variables. The categorical variable "Private" indicates whether the university is private or public, while the numerical variables represent various characteristics of the universities such as the number of applications, enrollments, graduation rate, etc. This information can be used to explore patterns within the data.

Text, table

Description automatically generated with medium confidence

**Figure 2 – skim()**

* The mean value of the variable "Grad.Rate" is 65.46, the median value is 65, and the minimum and maximum values are 10 and 118 respectively.
* The mean and median values of Grad.Rate are approximately equal which suggests that the distribution is symmetrical. In other words, the data values are evenly distributed around the mean and median values.
* The graduation rate variable has a skew of -0.11 indicating that the distribution is slightly skewed to the left. The negative kurtosis (-0.22) indicates that there are fewer extreme values (outliers) in the data compared to a normal distribution.

A picture containing text

Description automatically generated

**Figure 3 – describe()**

When we use describeBy to analyze college data by groups (Public and Private), we can observe that:

* There is a total of 777 universities listed out of which 565 of them are private and 212 are public.
* The mean graduate rate for public universities is more than that of private institutions. This can be due to a multitude of factors that can influence the graduation rate of a university, including the type of institution, location, student demographics, academic programs offered, and resources available to support student success.

**Text

Description automatically generated**

**A picture containing table

Description automatically generated**

**Figure 4 – describeBy()**

* Checking for NA values and duplicate rows

There are no missing values/ duplicate rows. Having a dataset without missing values or duplicate rows is advantageous, as it provides a complete and accurate representation of the data.

* Exploratory Data Analysis

**Scatterplots:**

The out-of-state tuition and graduation rate for both public and private universities are positively correlated. This may be because of a variety of factors such as quality of education, research opportunities, the reputation of the universities, demographic, and socioeconomic factors, etc.

**Chart, scatter chart

Description automatically generated**

**Figure 5 – Scatterplot (Graduation rate vs Outstate tuition)**

**Histograms:**

The graduation rate histogram appears to display a nearly normal distribution.

The histogram of private universities has a higher peak than that of public universities, which means that private universities have a higher average graduation rate than public universities.

Chart, histogram

Description automatically generated

**Figure 6 – Histograms**

**Correlation:**

* Grad.Rate has a very moderate positive relationship with Outstate, perc.alumni, Top10perc, and Top25perc (Correlation Coefficient – 0.57,0.49,0.49 and 0.48 respectively)
* Grad.Rate has a very moderate positive relationship with Room.Board(Correlation Coefficient –0.42)

A picture containing scatter chart

Description automatically generated

A picture containing table

Description automatically generated

**Figure 7 – Correlograms**

**Splitting the data into a train and test set**

Dependent variable - ***Grad.Rate***

This is a common technique used in machine learning and data analysis to assess the performance of a model. The purpose of this split is to assess the performance of a model and to avoid overfitting, which occurs when a model is too closely fit to the training data.

The training set is used to train the model and optimize its parameters, while the test set is used to evaluate the performance of the model. The ratio of the split depends on the size of the dataset, but a common ratio is 80/20 or 70/30, where 80% or 70% of the data is used for training and the remaining 20% or 30% is used for testing.

It's important to randomly split the data into train and test sets to ensure that the distribution of the data in both sets is representative of the overall population. This helps to prevent any biases or inconsistencies in the data that could impact the accuracy of the model.

**Ridge Regression**

**cv.glmnet() function**

cv.glmnet is a part of the package "glmnet" which performs cross-validation for a glmnet model. A glmnet model is a type of generalized linear model that includes Lasso regularization to handle problems of overfitting and collinearity in the data.

This function returns an object of class "cv.glmnet", which contains information about the cross-validation results, including the mean square error for each value of Lambda and the selected Lambda value. This information can be used to evaluate the performance of the model and to select the best combination of predictors to include in the final model.

Text

Description automatically generated

**Figure 8 – Ridge cv.glm()**

Text

Description automatically generated with medium confidence

**Figure 9 – lambda.min and lambda.1se - Ridge**

Lambda is used to control the complexity of the model and prevent overfitting by adding a penalty term to the loss function that the model is trying to minimize. The smaller the value of Lambda, the more complex the model can be, while larger values of Lambda result in simpler models.

The minimum value of Lambda: **3.126**

The value of Lambda at one standard error: **29.156**

The minimum logarithmic value of Lambda: **1.13984**

The logarithmic value of Lambda at one standard error: **3.37265**

The results suggest that the minimum value of Lambda **(3.126)** is much smaller than the value of Lambda at one standard error **(29.156)**. This means that the best model is complex, with a relatively low penalty for overfitting. The threshold of **29.156** means that models with a Lambda value larger than this might not fit the data well enough.

**Plot**

A typical cv.glmnet plot includes the following elements:

* The cross-validated mean square error for each value of Lambda.
* The plot is often displayed on a logarithmic scale to show the full range of Lambda values and the behavior of the model as Lambda increases.
* The ideal value of Lambda is the one that balances these two conflicting goals, producing a model with both low bias and low variance.

Chart, histogram

Description automatically generated

**Figure 10 – cv.glmnet Plot**

From the above plot, we can say that:

* There are 17 non-zero coefficients (the model was able to use 17 features to make its predictions).
* The model is most accurate when Lambda is set to a value between 1.13984 and 3.37265.
* The constant non-zero coefficient (17) indicates that the optimal number of features has been reached.

**Ridge regression model against the training set**

I have considered the lambda value at one standard error (i.e., lambda.1se) for this regularization model as it balances the trade-off between underfitting and overfitting. It also provides a good balance between the goodness of fit and the model's ability to generalize to new, unseen data.

This output shows the results of the Ridge Regression model using glmnet, with the regularization parameter lambda 1 standard error. It shows that for the best performance, 17 out of the 18 predictors were used in the model, with a 36.38% reduction in deviance. The best lambda value is 29.16.

A picture containing chart

Description automatically generated

**Figure 11 – Ridge model on training data**

These coefficients are the estimates for the regression coefficients of the ridge regression model. The values show the relationship between each predictor variable and the response variable after being penalized by ridge regularization.

The "(Intercept)" represents the constant term in the regression equation.

The magnitude of the coefficients gives us the importance of each predictor in the model. The positive coefficients indicate a positive relationship between the predictor and the response, while the negative coefficients indicate a negative relationship. The smaller magnitude of coefficients compared to the unpenalized regression model suggests that the ridge regularization has shrunk some of the coefficients towards zero to reduce overfitting.

Table

Description automatically generated with low confidence

**Figure 12 – Coefficient values – Ridge model(lambda.1se)**

**It's interesting to note that most of the coefficients have relatively small absolute values, which suggests that regularization has reduced their magnitude.**

The magnitude of the coefficients can help to identify the most important features in the model. In this case, the features that might be important are **PrivateYes, Top10perc, Top25perc, PhD, Terminal, and perc.alumni**.

**Performance of the fit model against the training set**

RMSE stands for Root Mean Squared Error, it is a commonly used evaluation metric for regression models. It represents the average difference between the actual and predicted values in the units of the response variable. The smaller the RMSE, the better the performance of the model. An RMSE of 0 indicates that the model perfectly predicts the target values, whereas an RMSE with a large value indicates a poor fit.

Text

Description automatically generated with low confidence

**Figure 13 – Ridge model RMSE values**

The RMSE value of the regression model (Train data) is **13.67612**.

In this case, a RMSE of **13.67612** for the regression model's training data can be considered a reasonably good model performance given that the mean value of Grad.Rate (the target variable) is approximately 65.

**Performance of the fit model against the testing set**

The RMSE value of the regression model (Test data) is **12.85973.**

It can be concluded that the RMSE value of **12.85973** on the testing data indicates that the model has reasonable accuracy in predicting the target values.

**Is the Model Overfitting?**

A common metric for evaluating overfitting is the difference between the training and testing error, and if the difference is significant, it may indicate overfitting. In this case, the difference is not significant, so the model is not likely to overfit.

**Lasso Regression**

Table

Description automatically generated with low confidence

**Figure 14 – Lasso cv.glm()**

Text

Description automatically generated with medium confidence

**Figure 15 – lambda.min and lambda.1se - Lasso**

The minimum value of Lambda: **0.1708**

The value of Lambda at one standard error: **1.7478**

The minimum logarithmic value of Lambda: **-1.767464**

The logarithmic value of Lambda at one standard error: **0.5583792**

The minimum value of Lambda, **0.1708**, is the value that provides the best performance for the model according to cross-validation accuracy.

The value of Lambda at one standard error, **1.7478**, is a measure of the uncertainty around the optimal value of Lambda.

The minimum value of Lambda **(0.1708)** is smaller than the value of Lambda at one standard error **(1.7478)**.

This means that the optimal value of Lambda, which provides the best performance for the model, is closer to weaker regularization.

**Plot**

Chart, histogram

Description automatically generated

**Figure 16 – cv.glmnet Plot**

From the above plot, we can say that:

* The model is most accurate when Lambda is set to a value between **-1.7675** and **0.5584**.
* For minimum mean-squared error, a maximum of 13 features can be considered, whereas for mean-squared error at 1 standard error, around 8 features can be considered.

**Lasso regression model against the training set**

The model is fitted using the glmnet function with the "alpha" parameter set to 1, indicating that it is a Lasso regression.

The "Df" column shows the number of variables included in the final model (7), the "%Dev" column shows the percentage of deviance explained by the model (37.17), and the "Lambda" column shows the value of the regularization parameter that was used (1.748).

A picture containing text

Description automatically generated

**Figure 17 – Lasso model on training data**

Below we can see the output of the coefficients of the Lasso Regression model. The output shows that some of the coefficients are set to zero, which means that those features were not selected by the model as significant predictors. The features that have a non-zero coefficient are: **"Top10perc", "Top25perc", "P.Undergrad", "Outstate", "Room.Board", "Personal", and "perc.alumni"**.

The value of the coefficients for these features show their effect on the outcome variable (in this case, graduation rates). The dot (.) symbol represents a coefficient that was set to zero by the Lasso Regression.

Table

Description automatically generated with low confidence

**Figure 18 – Coefficient values – Lasso model(lambda.1se)**

**Performance of the fit model against the training set**

Text

Description automatically generated with medium confidence

**Figure 19 – Lasso model RMSE values**

The RMSE value of the regression model (Train data) is **13.59122**.

In this situation, the RMSE value of **13.59122** means that on average, the model's predictions deviate from the actual values by approximately 13.59122 units.

**Performance of the fit model against the testing set**

The RMSE value of the regression model (Test data) is **12.62484.**

The RMSE value of **12.62484** suggests that the model has a good generalization performance and is capable of making accurate predictions on new, unseen data.

**Is the Model Overfitting?**

The RMSE value for the training data is **13.59122** and the RMSE value for the test data is **12.62484**. Although the training RMSE is higher, the difference between the two is not large, indicating that the model is not overfitting.

**Which model is better?**

The Lasso regularization model (RMSE Value – 12.62) outperformed the Ridge regression model (RMSE Value – 12.85) by selecting a smaller subset of significant features and achieving a lower Root Mean Squared Error (RMSE) value.

This indicates that the Lasso model is a better fit for the data and makes more accurate predictions.

**Feature selection**

Stepwise selection is a method used in regression analysis to select a subset of predictor variables for a model. This method works by adding or removing variables based on a statistical criterion, such as the p-value or the Akaike information criterion (AIC) until the best model is obtained.

**Training data**

First, I applied stepwise selection for the training dataset. We can see the result below.

A p-value less than 0.05 is typically considered significant and suggests that the corresponding predictor is important in explaining the response variable.

Text

Description automatically generated

**Figure 20 – Model created using features from stepwise selection (Train)**

A picture containing text

Description automatically generated

**Figure 21 – RMSE value (Stepwise – Train)**

R2 value **– 0.42**

RMSE value of the model**– 12.96**

**Testing data**

I did the same process again for the test dataset.

A screenshot of a computer

Description automatically generated with low confidence

**Figure 22 – Model created using features from stepwise selection (Test)**

A picture containing text

Description automatically generated

\

**Figure 23 – RMSE value (Stepwise – Test)**

R2 value **– 0.53**

RMSE value of the model**– 11.70**

**Which method is better?**

The Stepwise Selection method was better than the other two methods, Lasso Regularization and Ridge Regression. This is because of its lower Root Mean Square Error (RMSE) value.

**CONCLUSION**

Through this assignment, we were able to understand the concepts of Ridge and Lasso regression and the importance of using regularization techniques in building predictive models.

The key findings are :

* The analysis started by importing the College dataset and conducting exploratory data analysis through descriptive statistics and visualizations to gain insights into the data.
* The data was then divided into a training set and a test set, with a 70-30 ratio, to assess the model's accuracy.
* The Lasso and Ridge methods were applied, and they helped to reduce the variance in the data and prevent overfitting, leading to more accurate predictions of the Grad.Rate
* The accuracy of the models was then further improved by using the Stepwise selection method to assist find the most significant predictor variables.
* RMSE value for the Stepwise method (11.7) was the least among the three so we concluded that this method performed better.

**REFERENCES**

Bluman, A. G. (2018). Elementary Statistics, 10th ed. McGraw Hill.

Kabacoff, R. I. (2011). R in action: Data analysis and graphics with R. Manning Publications Co.

Home - RDocumentation. (n.d.). https://www.rdocumentation.org/

Penalized Logistic Regression Essentials in R: Ridge, Lasso and Elastic Net. (2018, March 11). Articles - STHDA.

http://www.sthda.com/english/articles/36-classification-methods-essentials/149-penalized-logistic-regression-essentials-in-r-ridge-lasso-and-elastic-net/

**APPENDIX: CODE**

#---------------------- Week\_4\_Module\_4 R Script ----------------------#

print("Author : Nikshita Ranganathan")

print("Module 4 Assignment - Regularization")

print("Course Name - ALY6015: Intermediate Analytics")

# Installing and loading the packages

library(ISLR)

library(glmnet)

library(Metrics)

library(corrgram)

library(hrbrthemes)

library(caret)

library(dplyr)

library(skimr)

library(MASS)

library(psych)

# Load the data

attach(College)

College

glimpse(College)

skim(College)

describe(College)

describeBy(College,Private)

# Scatterplot

qplot(x=Outstate,y=Grad.Rate,color=Private,shape=Private)+scale\_shape\_manual(values = c(16, 17))+scale\_color\_brewer(palette="Set1")

# Histogram

ggplot(College,aes(x=Grad.Rate,fill=Private)) +geom\_histogram(color="#e9ecef", alpha=0.5, position = 'identity')+scale\_fill\_manual(values=c("#69b3a2", "#404080")) +ggtitle("Histogram for Graduation Rate")+theme\_ipsum()

# Correlation

corr <- select\_if(College, is.numeric)

cormatrix<-round(cor(corr,method = "pearson"),digits=2)

corrgram(corr, order = TRUE, upper.panel = panel.pie,lower.panel=panel.shade,

main = "Correlogram",)

corrgram(corr, lower.panel=panel.pts, upper.panel=panel.conf,

diag.panel=panel.density)

# Split data into train and test sets

set.seed(123)

datasplit <- createDataPartition(College$Grad.Rate, p = 0.7, list = FALSE)

train <- College[datasplit,]

test <- College[-datasplit,]

# Create a model matrix

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

# Find best values of Lambda

# Find the best lambda using cross-validation

set.seed(123)

cv\_ridge <- cv.glmnet(train\_x, train\_y, alpha = 0, nfolds = 10)

cv\_ridge

# Optimal Value of Lambda; Minimizes the Prediction Error

# Lambda Min - Minimizes out of sample loss

# Lambda 1se - Largest value of Lambda within 1 Standard Error of Lambda Min

log(cv\_ridge$lambda.min)

log(cv\_ridge$lambda.1se)

# Plot

plot(cv\_ridge)

# Fit models based on lambda

# Fit the final model on Training Set using lambda.min

# alpha = 1 for Lasso (L2)

# alpha = 0 for Ridge (L1)

# Ridge Regression

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)

plot(coef(ridge\_model.min))

# Fit the model on training set 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)

plot(coef(ridge\_model.1se))

# Display coefficients of the OLS model with no regularization

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

coef(ols)

# View RMSE of the full model

ridge\_preds.ols <- predict(ols, new = test)

rmse(test$Grad.Rate, ridge\_preds.ols)

# Train set predictions

ridge\_pred.train <- predict(ridge\_model.1se, newx = train\_x)

ridge\_train.rmse <- rmse(train\_y, ridge\_pred.train)

# Test set predictions

ridge\_pred.test <- predict(ridge\_model.1se, newx = test\_x)

ridge\_test.rmse <- rmse(test\_y, ridge\_pred.test)

# Comparing rmse values of train and test set predictions

ridge\_train.rmse

ridge\_test.rmse

# Find best values of Lambda

# Find the best lambda using cross-validation

set.seed(123)

cv\_lasso <- cv.glmnet(train\_x, train\_y, alpha = 1, nfolds = 10)

cv\_lasso

# Optimal Value of Lambda; Minimizes the Prediction Error

# Lambda Min - Minimizes out of sample loss

# Lambda 1se - Largest value of Lambda within 1 Standard Error of Lambda Min

log(cv\_lasso$lambda.min)

log(cv\_lasso$lambda.1se)

# Plot

plot(cv\_lasso)

# Lasso Regression

lasso\_model.min <- glmnet(train\_x,train\_y,alpha=1,lambda = cv\_lasso$lambda.min)

lasso\_model.min

# Display Regression Coefficients

coef(lasso\_model.min)

plot(coef(lasso\_model.min))

# Fit the model on the training set using lambda.1se

lasso\_model.1se <- glmnet(train\_x,train\_y, alpha = 1, lambda = cv\_lasso$lambda.1se)

lasso\_model.1se

# Display Regression Coefficients

coef(lasso\_model.1se)

plot(coef(lasso\_model.1se))

# Display coefficients of the OLS model with no regularization

ols\_train <- lm(Grad.Rate ~ ., data = train)

coef(ols\_train)

# View RMSE of the full model

lasso\_preds.ols <- predict(ols\_train, new = test)

rmse(test$Grad.Rate, lasso\_preds.ols)

# Train set predictions

lasso\_pred.train <- predict(lasso\_model.1se, newx = train\_x)

lasso\_train.rmse <- rmse(train\_y, lasso\_pred.train)

# Test set predictions

lasso\_pred.test <- predict(lasso\_model.1se, newx = test\_x)

lasso\_test.rmse <- rmse(test\_y, lasso\_pred.test)

# Comparing rmse values of train and test set predictions

lasso\_train.rmse

lasso\_test.rmse

# Stepwise Selection (Train set)

model1<-lm(formula = Grad.Rate ~ ., data = train)

summary(model1)

# Stepwise Stepwise Selection

stepAIC(model1, direction = "both")

model.final=step(model1)

# Regression Model with optimal features of stepwise selection (Train data)

model2 <- lm(formula = Grad.Rate ~ Private + Apps + Top25perc + P.Undergrad + Outstate + Room.Board + Personal + perc.alumni + Expend, data = train)

summary1<-summary(model2)

summary1

rmse\_stepwise\_train<-sqrt(mean(summary1$residuals^2))

rmse\_stepwise\_train

# Stepwise Selection (Train set)

model3<-lm(formula = Grad.Rate ~ ., data = test)

summary(model3)

# Stepwise Stepwise Selection

stepAIC(model3, direction = "both")

model.final=step(model3)

# Regression Model with optimal features of stepwise selection (Train data)

model4 <- lm(formula = Grad.Rate ~ Apps + Top25perc + P.Undergrad + Outstate + Room.Board + PhD+ perc.alumni + Terminal, data = test)

summary2<-summary(model4)

summary2

rmse\_stepwise\_test<-sqrt(mean(summary2$residuals^2))

rmse\_stepwise\_test