Module 4 Assignment — Regularization

Sourabh D. Khot (ID 002754952)

College of Professional Studies, Northeastern University

ALY 6015: Intermediate Analytics CRN 81176

Professor Behzad Ahmadi

May 9, 2022

Table of Contents

Introduction

Concepts of overfitting and regularization in generalized linear models will be understood and implemented in this assignment. Specifically, the L1 (LASSO) and L2 (Ridge) types of regularization will be explored and implemented hands-on in the R language.

The College dataset will be used with the aim of predicting the percentage of students who graduate based on information available about the college, its students, faculty, and alumni. After partitioning the dataset for training and testing purposes, different linear regression models will be built based on Ridge regularization, LASSO regulation, and stepwise feature selection.

The models will be tested and compared against each other based on their complexity and accuracy. Finally, one among all the models will be proposed as the preferred model for the prediction.

Analysis

Data Preparation

1. Split the data into a train and test set.

The College dataset is imported from the ISLR library, which contains details of 777 colleges, their students, faculty, and alumni. It has 777 records and 18 variables, excluding the college name, which is the row label. 'Grad.Rate' is the target variable, which means the percentage of students who graduate. 17 variables are numerical, and 1 is categorical; a description of each variable is given in the appendix.

To partition the data into a train and test set, I have used the createDataPartition() from the caret package, which preserves the distribution of the target variable. Figure 1 shows the distribution of the Grad.Rate (target variable) before and after the partition.

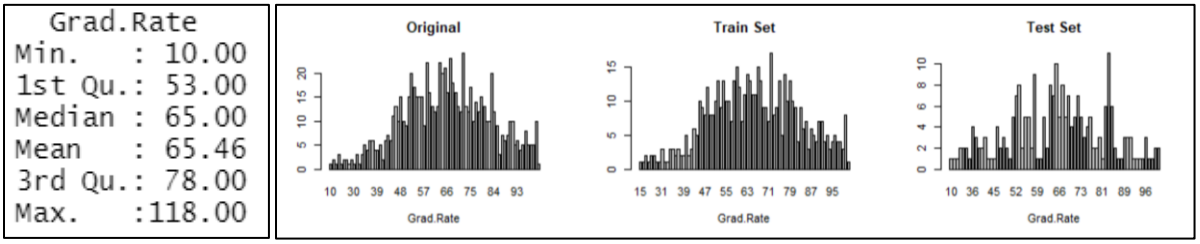

Figure 1. Distribution of Original, Train, Test Sets

Ridge Regression

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

In regularization, λ varies from zero to infinity. At zero, the model is the same as the least square regression, and at infinity, all coefficients tend to zero making the target value constant at all inputs. As λ increases, bias increases, and variance decreases.

```
> cv.ridge <- cv.glmnet(train_x, train_y, nfolds=10, alpha=0)
> cv.ridge$lambda.min
[1] 3.219821
> cv.ridge$lambda.1se
[1] 24.92991
```

Figure 2. λ Values from Ridge Regression Cross-Validation

After performing cross-validation for ridge regularization on this dataset, we obtain λ values as given in Figure 2. Model at λ_{min} of 3.22 will have a minimum error but will be complex, while the one at λ_{1se} 24.93 will be within one standard error of the minimum. The difference in the two λ values is enormous. Next, we will plot and interpret the result.

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

Figure 3 indicates the plot of mean-squared error vs. log(λ) with the number of features at the top. The left vertical dotted line corresponds to λ_{min} while the right is λ_{1se}. The variables remain at 17 since ridge regression does not eliminate variables. Model at λ_{1se} is considered the most parsimonious model, i.e., the simplest model that performs as good as the best model, which can also be visually interpreted from the plot. Hence, we will use λ_{1se} to fit the model.

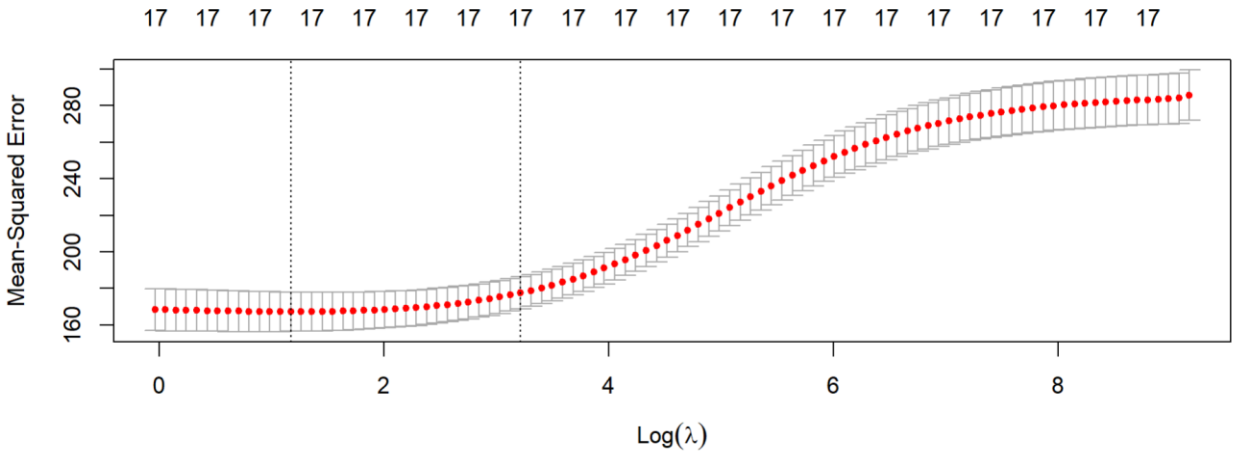

Figure 3: Plot from Ridge Regression Cross-Validation

4. **Fit a Ridge regression model against the training set and report on the coefficients. Is there anything interesting?**

The Ridge regression model is fitted using λ_{1se} = 24.93. The coefficients obtained for the 18 variables are given in Figure 4. Apart from the intercept, only 7 variables (Private, Top10perc, Top25perc, PhD, Terminal, S.F.Ratio, perc.alumni) have coefficients within the second exponent, while the 11 other variables have a much small coefficient, which may not impact the outcome variable.

```
> coef(model.ridge)
18 x 1 sparse Matrix of class "dgCMatrix"
                       s0
(Intercept)  4.255599e+01
PrivateYes   2.704273e+00
Apps         1.595341e-04
Accept       1.427138e-04
Enroll       3.467184e-05
Top10perc    6.742775e-02
Top25perc    6.234887e-02
F.Undergrad -4.819312e-05
P.Undergrad -8.567249e-04
Outstate     4.270602e-04
Room.Board   1.023315e-03
Books       -1.057564e-03
Personal    -1.394080e-03
PhD          4.938993e-02
Terminal     3.199244e-02
S.F.Ratio   -6.888482e-02
perc.alumni  1.469234e-01
Expend       9.951427e-05
```

Figure 4. Ridge Regression Coefficients

5. **Determine the performance of the fit model against the training set by calculating the root mean square error (RMSE).**

RMSE of the λ_{1se} ridge regression model on the train set is 13.21, which is less considerable than the range of Grad.Rate is from 10 to 118.

```
> preds.train.ridge <- predict(model.ridge, newx = train_x)
> rmse(train_y, preds.train.ridge)
[1] 13.20685
```

Figure 5. Ridge Regression Train Set RMSE

6. **Determine the performance of the fit model against the test set by calculating the root mean square error (RMSE). Is your model overfit?**

A model is considered overfitted to the train set when it has a higher error on the test or any new set, compared to the train set. Comparing Figures 5 and 6, the RMSE on test set is 13.66, which is slightly greater than train set error of 13.21, with a difference of 0.45. Hence the model has a negligible overfit.

```
> preds.test.ridge <- predict(model.ridge, newx = test_x)
> rmse(test_y, preds.test.ridge)
[1] 13.6618
```

Figure 6. Ridge Regression Test Set RMSE

LASSO

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

While the interpretation of λ is similar, the calculation of λ in Ridge and LASSO regressions is different as the latter can eliminate variables. Model at λ_{min} of 0.31 will have minimum error but most features, while the one at λ_{1se} 1.49 will have errors within one standard error of the minimum; however, have fewer features. The difference in the two λ values is considerable and will be explored further using a plot.

```
> cv.lasso <- cv.glmnet(train_x, train_y, nfolds=10, alpha=1)
> cv.lasso$lambda.min
[1] 0.3073475
> cv.lasso$lambda.1se
[1] 1.494509
```

Figure 7. λ Values from LASSO Regression Cross-Validation

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

Compared to Ridge, LASSO regression provides simpler, easier to interpret models with accurate predictions. Also, LASSO can eliminate the number of features, as observed in Figure 8, where the number of features indicated at the top reduces as λ increases. λ_{min} (left dotted line) has 13 variables, while λ_{1se} (right dotted line) has just 8 variables and an error within one standard deviation. We will build the model using λ_{1se}, which is the simplest yet accurate model.

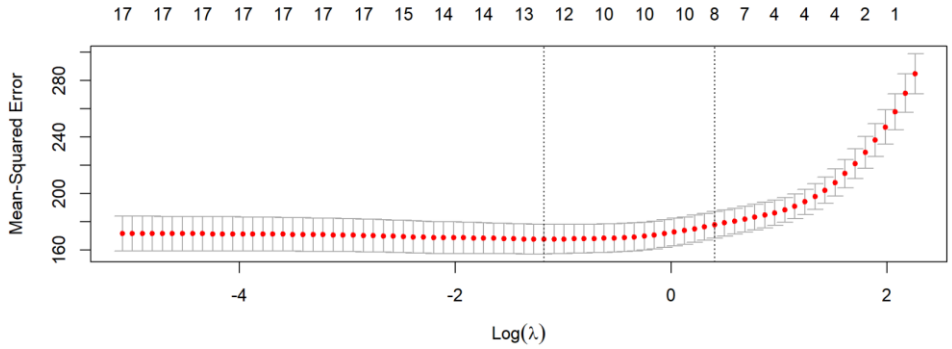

Figure 8: Plot from LASSO Regression Cross-Validation

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

After fitting the LASSO regression model at λ_{1se}, the model coefficients are shown in Figure 9. A total of 9 coefficients are reduced to zero, corresponding to Private, Accept, Enroll, F.Undergrad, Books, PhD, Terminal, S.F.Ratio, and Expend, meaning these features may not significantly impact the output variable and can be excluded from the model. Excluding intercept, 8 features are selected by this LASSO regression model.

```
> coef(model.lasso)
18 x 1 sparse Matrix of class "dgCMatrix"
                      s0
(Intercept)  4.077175e+01
PrivateYes   .
Apps         1.387681e-05
Accept       .
Enroll       .
Top10perc    3.814295e-02
Top25perc    9.040762e-02
F.Undergrad  .
P.Undergrad -5.050928e-04
Outstate     1.148922e-03
Room.Board   6.173271e-04
Books        .
Personal    -5.506170e-04
PhD          .
Terminal     .
S.F.Ratio    .
perc.alumni  2.330188e-01
Expend       .
```

Figure 9. LASSO Regression Coefficients

10. Determine the performance of the fit model against the training set by calculating the root mean square error (RMSE).

LASSO model on train set has RMSE of 13.15, which is small.

```
> preds.train.lasso <- predict(model.lasso, newx = train_x)
> rmse(train_y, preds.train.lasso)
[1] 13.14987
```

Figure 10. LASSO Regression Train Set RMSE

11. Determine the performance of the fit model against the test set by calculating the root mean square error (RMSE). Is your model overfit?

On the test set, the RMSE is 13.49, just slightly above the train set RMSE of 13.15, with a difference of 0.34. Hence the model is not significantly an overfit and can be considered a good model for predicting Grad.Rate.

```
> preds.test.lasso <- predict(model.lasso, newx = test_x)
> rmse(test_y, preds.test.lasso)
[1] 13.49172
```

Figure 11. LASSO Regression Test Set RMSE

Comparison

12. Which model performed better and why? Is that what you expected?

Figure 12 summarizes the parameters of both models. RMSEs of LASSO regression is slightly lower than Ridge regression on both the train and test set. Also, the difference in RMSE is slightly higher in Ridge, meaning the Ridge model is slightly overfit compared to LASSO. On top of that, LASSO has about half the number of features compared to Ridge.

Type of Regularization	No. of Variables	$RMSE_{train}$	$RMSE_{test}$	$RMSE_{diff}$
Ridge Regression	17	13.21	13.66	0.45
LASSO Regression	8	13.15	13.49	0.34

Figure 12. Comparison of Ridge and LASSO Models

Thus, the LASSO regression model performs better than the ridge model because of low error and half the number of features. It aligns with the expectations since LASSO models are simpler, easier to interpret, provide feature selection and predict more accurately.

13. 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?

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

Coefficients:
(Intercept)     Outstate  perc.alumni    Top25perc  P.Undergrad         Apps
 33.8475034    0.0008741    0.2964994    0.1211390   -0.0019855    0.0008423
   Personal   PrivateYes          PhD       Expend   Room.Board     Terminal
 -0.0019078    4.6850156    0.1855169   -0.0003030    0.0014764   -0.1173741
```

Figure 13. Stepwise Both Direction Feature Selection

A both-direction stepwise feature selection is performed using step() function, with output given in Figure 13, shortlisting 11 features (Outstate, perc.alumni, Top25perc, P.Undergrad, Apps, Personal, Private, PhD, Expend, Room.Board, Terminal). Using these

features, a linear regression model is fitted, whose performance is tested on a train and test set, and compared in Figure 14 among the previous ridge and LASSO models.

Type of Regularization	No. of Variables	$RMSE_{train}$	$RMSE_{test}$	$RMSE_{diff}$
Ridge Regression	17	13.21	13.66	0.45
LASSO Regression	8	13.15	13.49	0.34
Stepwise Selected Model	11	12.58	20.54	7.96

Figure 14. Comparison of Ridge, LASSO, Stepwise Models

The stepwise feature selected model has a lower RMSE (12.58) for the train set, but a high RMSE (20.54) for the test set, indicating overfitting to the train set and having low bias high variance. With 11 features, it is more complex than LASSO but simpler than the Ridge model; however, it terms of overfitting, it performs worse than both Ridge and LASSO models.

I would prefer the LASSO Regression Model, which has the lowest RMSE among all the models indicating high accuracy, nearly equal RMSE for train and test set indicating no overfitting, and the least number of features at 8, indicating a simple, easy to interpret the model.

Interpretation & Conclusion

It was observed that the linear regression model (built with features selected from the both-direction stepwise method) had high variance and exhibited the phenomenon of overfitting. Hence, regularization was necessary and beneficial as it reduced the overfit and balanced the tradeoff between bias and variance.

Among regularization techniques, both Ridge and LASSO reduced overfitting and performed well. However, LASSO was slightly more accurate. More importantly, LASSO had about half the number of features as Ridge, making it the simplest among all the models.

The LASSO model is preferred as the best model, being a simple model with no overfit and predicting with low errors.

References

(n.d.). Retrieved from StackOverFlow.com: https://stackoverflow.com/questions/55576482/how-to-fix-error-in-storage-modey-double-invalid-to-change-the-storage

An Introduction to glmnet: stanford.edu. (n.d.). Retrieved from https://glmnet.stanford.edu/articles/glmnet.html

Bluman, A. G. (2018). *Elementary Statistics.* New York: McGraw Hill Education.

Canvas Module 4 Assignment. (n.d.). Retrieved from https://northeastern.instructure.com/courses/110053/assignments/1345435

Linear, Lasso, and Ridge Regression with R: pluralsight.com. (n.d.). Retrieved from https://www.pluralsight.com/guides/linear-lasso-and-ridge-regression-with-r

R Converting from string to double: StackOverFlow.com. (n.d.). Retrieved from https://stackoverflow.com/questions/26734913/r-converting-from-string-to-double

Appendix

Description of Dataset Variables

- Private: A factor with levels No and Yes indicating private or public university
- Apps: Number of applications received
- Accept: Number of applications accepted
- Enroll: Number of new students enrolled
- Top10perc: Pct. new students from top 10% of H.S. class
- Top25perc: Pct. new students from top 25% of H.S. class
- F.Undergrad: Number of fulltime undergraduates
- P.Undergrad: Number of parttime undergraduates
- Outstate: Out-of-state tuition
- Room.Board: Room and board costs
- Books: Estimated book costs
- Personal: Estimated personal spending
- PhD: Pct. of faculty with Ph.D.’s
- Terminal: Pct. of faculty with terminal degree
- S.F.Ratio: Student/faculty ratio
- perc.alumni: Pct. alumni who donate
- Expend: Instructional expenditure per student
- Grad.Rate: Graduation rate

R Code

```
# DATA PREP ####

# Importing libraries and data

library(glmnet)   # for GLM, Ridge, LASSO
library(Metrics)  # helper functions for model measure metrics
library(ISLR)     # for dataset
library(caret)    # for data partition

df <- College

# 1. Splitting data ####

set.seed(952)
trainIndex <- createDataPartition(df$Grad.Rate, p = 0.7, list = FALSE, times = 1)
train <- df[trainIndex,]
test <- df[-trainIndex,]

# Plotting class rations of original and new sets
par(mfrow=c(1,3))
barplot(table(df$Grad.Rate), main="Original", xlab="Grad.Rate")
barplot(table(train$Grad.Rate), main="Train Set", xlab="Grad.Rate")
barplot(table(test$Grad.Rate), main="Test Set", xlab="Grad.Rate")
dev.off()

# Creating matrices for glmnet function
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


# RIDGE REGRESSION ####

# 2. Lambda Values #### alpha=0 for Ridge

set.seed(952)
cv.ridge <- cv.glmnet(train_x, train_y, nfolds=10, alpha=0)
cv.ridge$lambda.min
cv.ridge$lambda.1se

# 3. Lambda Plot ####

plot(cv.ridge)
```

```
# 4. Model Fitting ####

( model.ridge <- glmnet(train_x, train_y, alpha=0, lambda=cv.ridge$lambda.1se) )
coef(model.ridge)

# 5. Perf on Train Set ####

preds.train.ridge <- predict(model.ridge, newx = train_x)
rmse(train_y, preds.train.ridge)

# 6. Perf on Test Set ####

preds.test.ridge <- predict(model.ridge, newx = test_x)
rmse(test_y, preds.test.ridge)


# LASSO REGRESSION ####

# 7. Lamba Values ####

set.seed(952)
cv.lasso <- cv.glmnet(train_x, train_y, nfolds=10, alpha=1)
cv.lasso$lambda.min
cv.lasso$lambda.1se

# 8. Lambda Plot ####

plot(cv.lasso)

# 9. Model Fitting ####

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

# 10. Perf on Train Set ####

preds.train.lasso <- predict(model.lasso, newx = train_x)
rmse(train_y, preds.train.lasso)

# 11. Perf on Test Set ####

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

```
# COMPARISON ####

# 12. Better Model ####

# Explained in Report

# 13. Stepwise Model ####

set.seed(952)
step(lm(Grad.Rate ~ 1, data = train), direction = 'both', scope = ~
Private+Apps+Accept+Enroll+Top10perc+Top25perc+F.Undergrad+P.Undergrad+Outstate+Ro
om.Board+Books+Personal+PhD+Terminal+S.F.Ratio+perc.alumni+Expend)
model.stepwise <- lm(formula = Grad.Rate ~
Outstate+perc.alumni+Top25perc+P.Undergrad+Apps+Personal+Private+PhD+Expend+Room.
Board+Terminal, data = train)
summary(model.stepwise)

# Performance on Train & Test
preds.train.stepwise <- predict(model.stepwise, newx = train_x)
rmse(train_y, preds.train.stepwise)
preds.test.stepwise <- predict(model.stepwise, newx = test_x)
rmse(test_y, preds.test.stepwise)
```

<End of Report>