**Module 4 Assignment — Regularization**

**Sriram Kannepalli**

**Northeastern University: College of Professional Studies**

[**ALY6015.71591.202515**](https://northeastern.instructure.com/courses/196378) **: Intermediate Analytics**

**Professor: Roy Wada**

**November 30th, 2024**

**Introduction**

This report aims to consolidate theoretical understanding and practical application of regularization methods, focusing on Ridge and LASSO regression, to build predictive models. The regularization process helps to mitigate overfitting, improve generalization, and handle multicollinearity in datasets. Additionally, stepwise regression is included as a baseline comparison for feature selection.

The analysis uses the College dataset from the ISLR library to predict the graduation rate (Grad.Rate) using several predictors. Regularization parameters (λ or lambda) for Ridge and LASSO are determined using cross-validation to select the best values for minimizing the Mean Squared Error (MSE). The report also compares the models based on Root Mean Squared Error (RMSE) and provides insights into variable selection, model performance, and overfitting.

**Analysis**

**Data Preparation**

The College dataset contains 18 variables, including one categorical predictor (Private) and several numerical predictors such as Apps, Accept, and Room.Board. The data was split into 70% for training and 30% for testing using the caret package.

**Code Snippet (Refer to Appendix):**  
See **Appendix - Section 1** for the R code used for data preparation.

**1. Ridge Regression**

Ridge regression retains all predictors by shrinking coefficients. The cv.glmnet function was used to identify optimal lambda values:

* **λ\_min (lambda.min):** 3.126
* **λ\_1se (lambda.1se):** 7.228

**Key Outputs:**

* **Training RMSE:** 12.987
* **Testing RMSE:** 12.045

**Interpretation:**  
The cross-validation plot highlights the trade-off between lambda and MSE. As λ increases, the model complexity reduces, lowering overfitting risks but possibly increasing bias. At λ\_min, all predictors are retained but penalized for multicollinearity. At λ\_1se, coefficients are further shrunk for parsimony.

**Visualization:**  
Refer to **Appendix - Figure 1** for the Ridge Cross-Validation Plot.

**2. LASSO Regression**

LASSO regression reduces coefficients to zero, performing variable selection. The cv.glmnet function identified:

* **λ\_min (lambda.min):** 0.171
* **λ\_1se (lambda.1se):** 0.342

**Key Outputs:**

* **Training RMSE:** 12.938
* **Testing RMSE:** 11.989
* **Predictors Reduced to Zero:** 7 predictors were removed, retaining 10 predictors.

**Interpretation:**  
The cross-validation plot illustrates the relationship between lambda and MSE. LASSO selected the most influential predictors, discarding those with negligible impact. This improves model interpretability without compromising prediction accuracy.

**Visualization:**  
Refer to **Appendix - Figure 2** for the LASSO Cross-Validation Plot.

**3. Stepwise Regression**

Stepwise regression, using AIC as the selection criterion, reduced the model to 12 predictors. This method iteratively added and removed variables to optimize performance.

**Key Outputs:**

* **Training RMSE:** 12.960
* **Testing RMSE:** 12.005

**Interpretation:**  
Stepwise regression provides a parsimonious model but lacks the penalization effect of regularization, making it less robust for datasets prone to multicollinearity.

**Model Comparison**

| **Model** | **λ\_min** | **Predictors Retained** | **RMSE (Train)** | **RMSE (Test)** |
| --- | --- | --- | --- | --- |
| Ridge | 3.126 | 17 | 12.987 | 12.045 |
| LASSO | 0.171 | 10 | 12.938 | 11.989 |
| Stepwise | NA | 12 | 12.960 | 12.005 |

**Insights:**

* **Best Performing Model:** LASSO achieved the lowest testing RMSE (11.989), indicating it generalized better than Ridge and Stepwise models.
* **Feature Selection:** Ridge retained all predictors, while LASSO reduced 7 predictors to zero. Stepwise selected 12 predictors.
* **Overfitting:** None of the models exhibited significant overfitting, as RMSE values for training and testing were comparable.

**Conclusion**

1. **Preferred Model:** LASSO regression is preferred due to its balance of interpretability and predictive performance.
2. **Model Selection:** Ridge regression is suitable for datasets with high multicollinearity, while Stepwise is simple but less robust.
3. **Recommendations:** For future analyses, combining LASSO with feature engineering may yield even better results.

**References**

R-bloggers. (2018, February 12). Regularization: Ridge, LASSO, and Elastic Net. Retrieved November 24, 2024, from <https://www.r-bloggers.com/2018/02/regularization-ridge-lasso-and-elastic-net/>

R-bloggers. (2019, July 5). A Practical Guide to Stepwise Regression in R. Retrieved November 24, 2024, from <https://www.r-bloggers.com/2019/07/a-practical-guide-to-stepwise-regression-in-r/>

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An Introduction to Statistical Learning with Applications in R*. Springer. Retrieved November 24, 2024, from <https://www.statlearning.com/>

Friedman, J., Hastie, T., & Tibshirani, R. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer. Retrieved November 26, 2024, from <https://web.stanford.edu/~hastie/ElemStatLearn/>

R Core Team. (2024). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Retrieved November 27, 2024, from <https://www.R-project.org/>

CRAN. (n.d.). *glmnet: Lasso and Elastic-Net Regularized Generalized Linear Models*. Retrieved November 28, 2024, from <https://cran.r-project.org/web/packages/glmnet/>

CRAN. (n.d.). *ISLR: Data for An Introduction to Statistical Learning with Applications in R*. Retrieved November 28, 2024, from <https://cran.r-project.org/web/packages/ISLR/>

**Appendix**

**Section 1: Data Preparation Code**

|  |
| --- |
| set.seed(123)  train\_index <- createDataPartition(College$Grad.Rate, p = 0.7, list = FALSE)  train\_data <- College[train\_index, ]  test\_data <- College[-train\_index, ]  x\_train <- model.matrix(Grad.Rate ~ ., train\_data)[, -1]  y\_train <- train\_data$Grad.Rate  x\_test <- model.matrix(Grad.Rate ~ ., test\_data)[, -1]  y\_test <- test\_data$Grad.Rate |

**Section 2: Ridge Regression Code**

|  |
| --- |
| cv\_ridge <- cv.glmnet(x\_train, y\_train, alpha = 0)  lambda\_min\_ridge <- cv\_ridge$lambda.min  lambda\_1se\_ridge <- cv\_ridge$lambda.1se  ridge\_model <- glmnet(x\_train, y\_train, alpha = 0, lambda = lambda\_min\_ridge)  ridge\_pred\_train <- predict(ridge\_model, newx = x\_train)  ridge\_pred\_test <- predict(ridge\_model, newx = x\_test)  rmse\_train\_ridge <- sqrt(mean((y\_train - ridge\_pred\_train)^2))  rmse\_test\_ridge <- sqrt(mean((y\_test - ridge\_pred\_test)^2)) |

**Figure 1: Ridge Cross-Validation Plot**This plot illustrates the relationship between lambda and mean squared error (MSE) for Ridge regression, highlighting the selected lambda values (lambda.min and lambda.1se).

A graph with a red dotted line

Description automatically generated

**Section 3: LASSO Regression Code**

|  |
| --- |
| cv\_lasso <- cv.glmnet(x\_train, y\_train, alpha = 1)  lambda\_min\_lasso <- cv\_lasso$lambda.min  lambda\_1se\_lasso <- cv\_lasso$lambda.1se  lasso\_model <- glmnet(x\_train, y\_train, alpha = 1, lambda = lambda\_min\_lasso)  lasso\_pred\_train <- predict(lasso\_model, newx = x\_train)  lasso\_pred\_test <- predict(lasso\_model, newx = x\_test)  rmse\_train\_lasso <- sqrt(mean((y\_train - lasso\_pred\_train)^2))  rmse\_test\_lasso <- sqrt(mean((y\_test - lasso\_pred\_test)^2)) |

**Figure 2: LASSO Cross-Validation Plot**This plot illustrates the relationship between lambda and mean squared error (MSE) for LASSO regression, highlighting the selected lambda values (lambda.min and lambda.1se).

A graph with a line

Description automatically generated

**Section 4: Stepwise Regression Code**

|  |
| --- |
| full\_model <- lm(Grad.Rate ~ ., data = train\_data)  step\_model <- step(full\_model, direction = "both")  step\_pred\_train <- predict(step\_model, newdata = train\_data)  step\_pred\_test <- predict(step\_model, newdata = test\_data)  rmse\_train\_step <- sqrt(mean((y\_train - step\_pred\_train)^2))  rmse\_test\_step <- sqrt(mean((y\_test - step\_pred\_test)^2)) |