**ALY6015 80472 Intermediate Analytics SEC 04 Spring 2023 CPS**

**Module 4 Assignment — Regularization REPORT**

Logo

Description automatically generated

**NORTHEASTERN UNIVERSITY**

**College of Professional Studies, Boston, MA, 02215.**

**Submitted by**

Zihan Ma

ma.zihan1@northeastern.edu

**Instructor**

 Prof. Valeriy Shevchenko

**Date**

06/19/2023

**Regularization**

**Assignment Summary:**

This report provides a detailed analysis of the 'College' dataset from the ISLR library in R. Our aim is to predict the 'Grad.Rate' feature using two regularization models: Ridge Regression and LASSO.

Regularization is a technique used to prevent overfitting in a machine learning model by adding a penalty term to the loss function. Two commonly used regularization methods are Ridge Regression and LASSO. Ridge Regression shrinks the coefficients of less important features towards zero, while LASSO can reduce some feature coefficients to absolute zero, effectively performing feature selection.

**Process And Report:**

**Dataset and Preprocessing:**

**Process 1: Dataset and Preprocessing**

In our analysis, we used the College dataset, which contains a variety of attributes related to colleges and universities. Our target variable is 'Grad.Rate'.

Upon inspecting the dataset, we noted that not all columns were numerical, and the distributions of these variables varied significantly. To make our data suitable for the modeling techniques we planned to use, we performed several preprocessing steps.

Firstly, we partitioned the data into an 80% training set and a 20% test set, ensuring an unbiased evaluation of our predictive models.

Secondly, we removed non-numerical columns from our dataset as our focus was on numerical modeling techniques.

Lastly, we standardized the numerical features to have a mean of 0 and a standard deviation of 1, so that no single feature would have an undue influence on the model due to its scale.

After preprocessing, our numerical-only, scaled datasets were ready for model application and comparison.

Following are the codes and result for the process above.

A screenshot of a computer

Description automatically generated with medium confidence

A screenshot of a computer screen

Description automatically generated with medium confidence

A screen shot of a computer code

Description automatically generated with low confidence

A picture containing text, font, screenshot, line

Description automatically generated

A picture containing text, font, screenshot, line

Description automatically generated

A picture containing text, font, white, typography

Description automatically generated



A screenshot of a computer code

Description automatically generated with low confidence

A screenshot of a computer

Description automatically generated with medium confidence

**Process 2: Lambda Estimation**

After splitting the data into training and test sets, the X and Y matrices were defined. The X matrix represents the independent variables, while the Y matrix represents the dependent variable.

A picture containing text, font, screenshot, line

Description automatically generated

In the Ridge Regression process, a critical step involves determining the regularization parameter λ (lambda). We carried this out using the cv.glmnet() function, which performs k-fold cross-validation for glmnet, i.e., generalized linear models via penalized maximum likelihood.

The 'alpha = 0' parameter in cv.glmnet() indicates that we were conducting ridge regression, as alpha denotes the elastic net mixing parameter.





A screen shot of a computer code

Description automatically generated with low confidence

We then extracted the lambda.min and lambda.1se values from the output. These parameters play a crucial role in model selection:

1. **lambda.min**: This parameter refers to the value of λ that results in the smallest mean cross-validated error. In our case, the estimated value was approximately 0.1004367.
2. **lambda.1se**: The '1se' rule refers to the most regularized model within one standard error of the minimum. It provides a less complex model, offering a balance between bias and variance. Our estimated value was approximately 0.936676.

The quality of the estimated lambda values can be evaluated based on their impact on model performance. A smaller lambda.min value indicates a less regularized model that may have lower bias but could be prone to overfitting the training data. On the other hand, a larger lambda.1se value suggests a more regularized model with higher bias but potentially better generalization to unseen data.

The summary of our ridge regression cross-validation model provided additional details, such as the number of non-zero coefficients (nzero) at each lambda value.

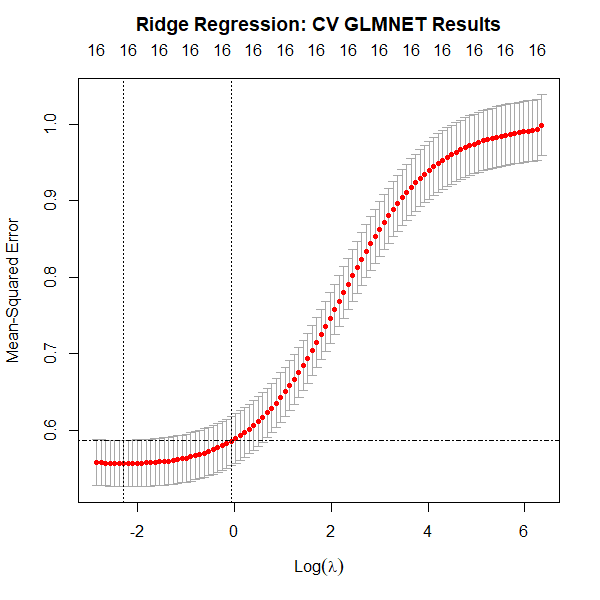
These λ values are fundamental in ridge regression, where the goal is to choose such a value of lambda that minimizes the potential for overfitting while still retaining meaningful variables in the model.

In our upcoming steps, we will use these λ values to fit a ridge regression model and evaluate its performance.

**Process 3: Interpretation of cv.glmnet Function Plot**

The cv.glmnet function plot is an essential tool to visualize the variation in our model's performance with different lambda values. It provides a practical way of understanding the trade-off between bias and variance in our model and thus guides our decision for an optimal lambda value.

Upon plotting the cross-validated Mean Squared Error (MSE) against the log of lambda values, we can observe the path of MSE as lambda changes.



The plot shows two key vertical lines:

1. **The line for lambda.min**: It indicates the lambda value where the lowest MSE was observed in our cross-validation process. This optimal lambda value minimizes the cross-validation error, leading to the most accurate model.
2. **The line for lambda.1se**: It signifies the lambda value within one standard error of the minimum. Choosing this lambda value will lead to a model with a simpler structure and less chance of overfitting, although it may not necessarily be the most accurate.

The graph also added a horizontal dashed line at cvup of lambda.min helps to identify the range within one standard error of the minimum. It aids in interpreting the standard errors of the cross-validated mean squared errors and could serve as a guide to selecting a simpler model.  
In general, smaller lambda values would result in a model with more features and potentially higher complexity, while larger lambda values would lead to simpler, more regularized models with fewer features.

Through this plot, we can visually understand the balance between model complexity (and thus the chance of overfitting) and model performance. Which the maximum value that λ can take while still falling within the on standard error interval of the minimum CV λ.

Our next step is to fit the Ridge Regression model using our training set and observe the effects on the coefficients.

**Process 4: Model Coefficients**

The Ridge Regression model aims to shrink the coefficients of less important features towards zero to reduce model complexity and avoid overfitting, without entirely eliminating them. Observing the model's coefficients helps in understanding the significance of each predictor and the direction of their influence on our target variable.

The first plot, titled "Ridge Regression Coefficients", shows the values of the coefficients of each predictor feature in our model. We sorted the coefficients in descending order of their absolute values for clarity. The coefficients have different signs (represented by different colors), meaning they positively or negatively influence the graduation rate.

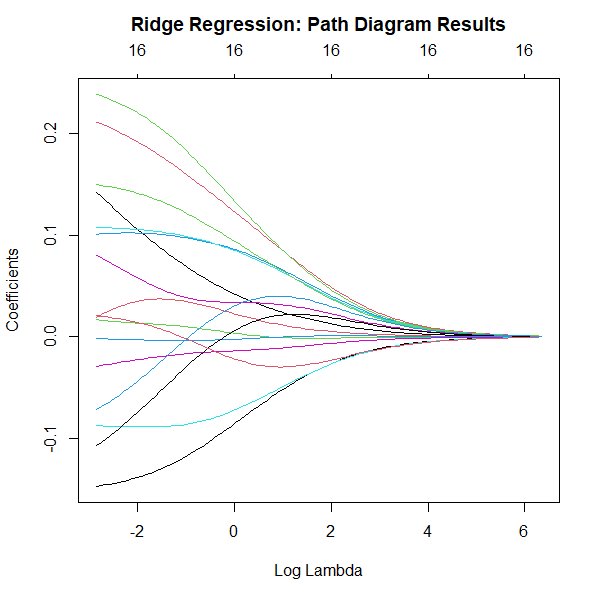
A picture containing text, screenshot, diagram, font

Description automatically generated

1. **'Expend':** This predictor has the highest positive coefficient (2.28e-01), suggesting it is the most influential feature in predicting 'Grad.Rate'. As college expenditure increases, graduation rates are also expected to increase, all other factors being equal.
2. **'Room.Board':** This predictor also has a large positive coefficient (1.99e-01), suggesting that colleges with higher room and board costs tend to have higher graduation rates, assuming other variables remain constant.
3. **'Outstate':** This predictor has a significant negative coefficient (-1.42e-01). This implies that colleges with higher out-of-state tuition rates tend to have lower graduation rates, keeping all other factors constant.
4. **'Enroll':** This predictor has a small positive coefficient (2.98e-02), indicating a small positive impact on graduation rate as enrollment numbers increase.
5. **'P.Undergrad':** This predictor has a small negative coefficient (-2.39e-02), suggesting that colleges with higher part-time undergraduate student numbers tend to have slightly lower graduation rates, holding all other factors constant.
6. **'Apps':** The coefficient for 'Apps' is close to zero, implying that the number of applications received by a college has little to no impact on the graduation rate in the Ridge regression model.

Ridge regression does not reduce coefficients to absolute zero but minimizes them towards zero, and thus all features remain part of the model regardless of their significance.

The second plot, titled "Ridge Regression: Path Diagram Results", provides a graphical view of the coefficients' trajectories for different lambda values. Each curve represents a feature, and the path of the curve shows how the corresponding coefficient changes as lambda increases. As lambda increases (moving towards the right), the magnitude of the coefficients decreases, converging towards zero but not entirely reaching it. This is the characteristic of Ridge regression - it reduces the coefficients but doesn't eliminate them.



Together, these two plots offer a comprehensive understanding of how Ridge regression behaves and what impact each predictor has on our model. Which ridge regression does not reduce coefficients to absolute zero but minimizes them towards zero, and thus all features remain part of the model regardless of their significance.

**Process 5: Performance on Training Set**

To assess the performance of the Ridge regression model on the training set, we calculated the Root Mean Squared Error (RMSE). The RMSE is a standard measure for understanding the model's prediction errors. Specifically, it tells us how concentrated the data is around the line of best fit.

A picture containing text, font, line, white

Description automatically generated

In our case, the RMSE for the training set was found to be approximately 0.729. This value suggests that the model, on average, predicts the graduation rate within a deviation of 0.729 units from the actual value. Given the scale of our target variable (Graduation Rate), this represents a reasonable performance, providing confidence in the model's ability to generalize from the training data. However, we need to validate this performance by testing the model on unseen data in our test set, which we will do in the next step.

**Process 6: Performance on Test Set**

The Ridge regression model's performance was also evaluated on the test set. For this, we calculated the Root Mean Squared Error (RMSE) which is a standard measure for quantifying prediction errors. The RMSE for the test set was found to be approximately 0.773.

A comparison of the RMSE values from the training set (0.729) and the test set (0.773) indicates that the model's performance is consistent when generalizing to unseen data. The RMSE value for the test set is only slightly higher than the training set's RMSE, which suggests that the model is not overfitting the training data. An overfitted model would typically have a much lower error rate on the training data and a significantly higher error rate on the test data, which is not the case here.

The table below presents a summary of the Ridge Regression results, including the optimal lambda values and the RMSE scores for the training and test sets:

A picture containing text, font, screenshot, number

Description automatically generated

Overall, the Ridge Regression model performed well, with relatively small prediction errors and good generalization to unseen data.

**Process 7: Lambda Estimation**

The Least Absolute Shrinkage and Selection Operator (LASSO) regression was employed to build another regularization model. The process started by estimating the tuning parameter lambda using cross-validation. Cross-validation, a resampling procedure, helps to tune the lambda parameter by minimizing the out-of-sample prediction error.

In the context of LASSO, two lambda values are typically estimated: lambda.min and lambda.1se.

lambda.min: This is the value of lambda that gives the minimum mean cross-validated error. For our model, this value was approximately 0.00796.

lambda.1se: Also known as the most regularized model within one standard error of the optimum. It tends to favor simpler models by providing a larger penalty and hence, greater shrinkage. In our case, lambda.1se was around 0.118.

A comparison between the two lambda values shows that lambda.min is smaller than lambda.1se, indicating a less regularized model that utilizes more predictors. In contrast, lambda.1se being higher leads to a more regularized model, possibly with fewer predictors due to the higher penalty.

The summary output and the specific lambda values are as follows:

A screenshot of a computer

Description automatically generated

In the next steps, these lambda values will be used to fit the LASSO model and evaluate its performance.

**Process 8: Interpretation of cv.glmnet Function Plot**

To gain a better understanding of the results from our LASSO model's cross-validation process, a graphical representation was generated. This plot is essentially a visualization of the cross-validated error (on the y-axis) versus log(lambda) (on the x-axis).

In the plot, the red dotted line represents the value of lambda.min, which minimizes the cross-validated error. On the other hand, the solid vertical line is drawn at the value of lambda.1se, which is the most regularized model within one standard error of the minimum.

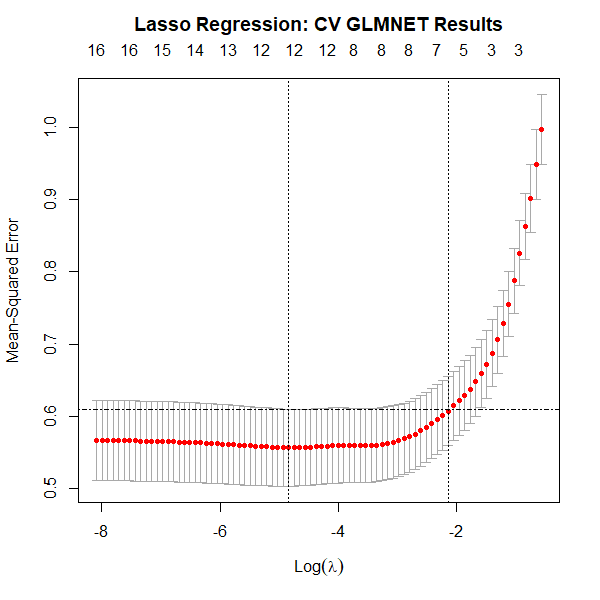
Adding a horizontal dashed line at cvup of lambda.min helps to identify the range within one standard error of the minimum. It aids in interpreting the standard errors of the cross-validated mean squared errors and could serve as a guide to select a simpler model.

This graph is particularly helpful in visualizing the balance between bias and variance for different models. The best model is selected based on the smallest mean cross-validated error, considering the amount of shrinkage applied.

The code snippet and the resulting graph are as follows:

A picture containing text, font, screenshot, line

Description automatically generated



In the following parts, we will use these selected lambda values to fit our LASSO model and subsequently evaluate its performance.

**Process 9: Model Coefficients**

In the LASSO regression model, as the penalty term changes, the model coefficients also change. To visually interpret this effect, we fit a LASSO model against the training set and plot the coefficients.

With the LASSO Regression Coefficients plot, the most significant insights we can derive revolve around the impact and importance of each predictor:

1. **Magnitude of Coefficients:** The size of each bar represents the magnitude of the corresponding coefficient. Larger bars indicate that a small change in the associated feature will have a large impact on the predicted 'Grad.Rate'. In contrast, smaller bars indicate features that have a lesser impact.
2. **Direction of Coefficients:** The color of the bars (blue for positive, red for negative) illustrates the direction of each coefficient. A positive coefficient suggests that as the feature increases, the 'Grad.Rate' also tends to increase, and vice versa for negative coefficients.

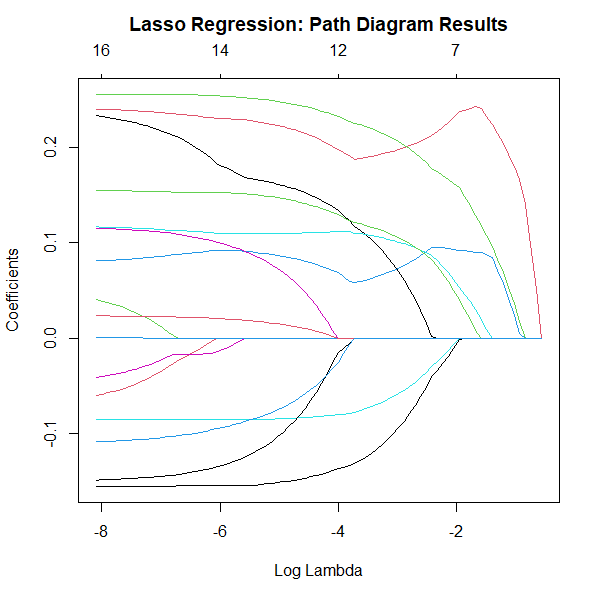
A picture containing text, screenshot, font, diagram

Description automatically generated

1. **'Expend':** This predictor has the highest positive coefficient (2.46e-01), suggesting it is the most influential feature in predicting 'Grad.Rate'. As college expenditure increases, graduation rates are expected to increase as well, all other factors being equal.
2. **'Room.Board':** This predictor also has a large positive coefficient (2.19e-01), suggesting that colleges with higher room and board costs tend to have higher graduation rates, assuming other variables remain constant.
3. **'Outstate':** This predictor has a significant negative coefficient (-1.49e-01). This implies that colleges with higher out-of-state tuition rates tend to have lower graduation rates, keeping all other factors constant.
4. **'Apps', 'Enroll', 'Top10perc', 'P.Undergrad', 'Personal':** These predictors have coefficients reduced to zero, suggesting that these features do not contribute to the prediction of 'Grad.Rate' in the final LASSO model. LASSO has effectively performed feature selection by removing these predictors.

From the LASSO Path Diagram, we gain insights into how our model behaves as the regularization strength changes:

1. **Feature Selection:** As the regularization strength (λ) increases (moving right to left in the plot), the model will "drop" predictors by setting their coefficients to zero. The exact point where each feature enters the model (its coefficient becomes non-zero) gives an insight into its relative importance.
2. **Coefficient Evolution:** The path each line (predictor) follows shows how the corresponding coefficient evolves as the model complexity changes.



The following two plots and corresponding code are presented:

A picture containing text, screenshot, font, electric blue

Description automatically generated

From these visualizations, we can gather valuable insights into which predictors are the most significant in our model, how they influence the target variable, and how the predictor importance evolves with changes in the regularization strength. Which in contrast to Ridge regression, LASSO (Least Absolute Shrinkage and Selection Operator) regression can shrink the coefficients of less important features all the way to zero. This means that LASSO performs both variable selection and regularization. As a result, it provides a more interpretable model that only includes the most significant predictors and excludes those that are not crucial for prediction. This feature of LASSO can be particularly useful when dealing with datasets that have high dimensionality. In the given dataset, for example, LASSO has reduced the coefficients of 'Apps', 'Enroll', 'Top10perc', 'P.Undergrad', and 'Personal' to absolute zero, suggesting these features do not significantly contribute to the prediction of the graduation rate.

**Process 10: Performance on Training Set**

The LASSO regression model was trained on the dataset, and its performance was evaluated based on the Root Mean Squared Error (RMSE) metric. RMSE measures the average magnitude of the residuals or prediction errors. It's particularly useful when we want to know how much our predictions deviate, on average, from the actual values in the dataset.

A picture containing text, font, line, white

Description automatically generated

The RMSE for the LASSO model on the training set is approximately 0.728. This means that, on average, the predictions made by the LASSO model deviate from the actual values by about 0.728. It serves as a baseline to evaluate the performance of the model on the test set. Lower RMSE values indicate better fit to the data, but we also need to check if the model generalizes well on unseen data by evaluating its performance on the test set.

**Process 11: Performance on Test Set**

The LASSO regression model was also evaluated on the test dataset using the Root Mean Squared Error (RMSE) metric. Here are the results:

A picture containing text, font, screenshot, algebra

Description automatically generated

The RMSE for the LASSO model on the test set is approximately 0.774, which is slightly higher than the RMSE on the training set (0.728).

Comparing the training and testing RMSEs can provide insights into whether our model is overfitting, i.e., whether it's learning the training data too well and hence performing poorly on unseen data. In this case, since the testing RMSE is not significantly higher than the training RMSE, it suggests that the LASSO model is not overfitting. It's managing to generalize well to unseen data. However, the slight increase in RMSE in the test set compared to the training set does indicate that there might be a small amount of overfitting happening.

Also, we can observe that the lambda.min and lambda.1se values for the LASSO model are lower than those for the Ridge model, which suggests that the LASSO model is more stringent in shrinking the coefficients towards zero. This behavior can be helpful when dealing with datasets with high dimensionality, as it can lead to sparser solutions with fewer variables, which can be easier to interpret.

**Process 12: Performance Comparison**

The performance of the Ridge and LASSO regression models can be compared using the metrics summarized in the table below:

A screenshot of a computer code

Description automatically generated with low confidence

The RMSE values for the training and testing sets are comparable for both models, indicating a similar level of predictive performance. However, the LASSO model appears to have a slightly better training performance, given its marginally lower Training RMSE.

The 'Lambda.Min' and 'Lambda.1se' values are smaller for the LASSO model, suggesting that it has been more aggressive in shrinking the coefficients towards zero, potentially leading to a simpler, more interpretable model.

However, when comparing the Testing RMSE, the LASSO model performed slightly worse than the Ridge model. This may indicate a slight overfitting to the training data on the part of the LASSO model, as it exhibits a higher error rate on the test data.

These results reflect the general properties of Ridge and LASSO regression. Ridge regression tends to work better when most predictors impact the response, while LASSO can outperform Ridge when only a subset of predictors is influential, because it effectively performs feature selection by reducing some coefficients exactly to zero.

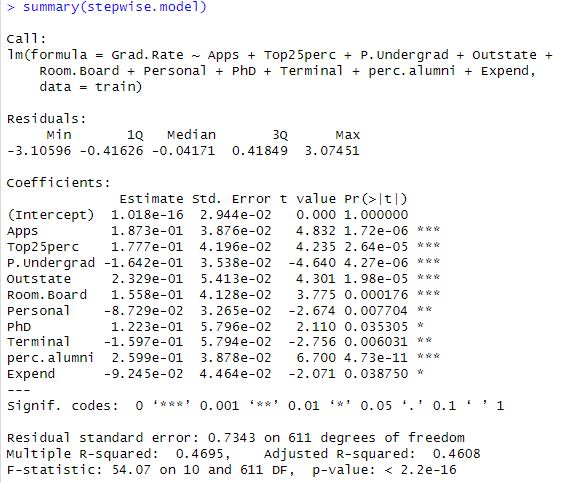
**Process 13: Stepwise Selection and Model Fitting**

This stepwise regression analysis starts with the full model, including all predictors, and then gradually removes the least significant predictors based on the Akaike Information Criterion (AIC).

In each step, the model examines all predictors to check whether adding or removing any improves the model based on the AIC. The process continues until no improvement can be made by adding or removing predictors.

In the final model, the variables included are Apps, Top25perc, P.Undergrad, Outstate, Room.Board, Personal, PhD, Terminal, perc.alumni, and Expend.

This indicates that these variables are significant predictors of Grad.Rate according to this stepwise regression model.



A picture containing text, font, screenshot, algebra

Description automatically generated

The RMSE for the training data is 0.7277885 and for the testing data is 0.7763036. These results are pretty close to each other, which means that the model performs almost as well on the unseen test data as it does on the training data. This is a good sign, as it indicates that the model has not overfitted to the training data and can generalize well to unseen data. The lambda values provide information about the complexity of the model, with the Lambda.1se value indicating a potentially simpler model that performs almost as well as the model at Lambda.Min.

**Process 14: Comparing with Ridge and LASSO**

The combined table provides a comprehensive comparison of the performance of Ridge Regression, LASSO Regression, and Stepwise Regression:

A picture containing text, screenshot, font, number

Description automatically generated

* **Lambda.Min**: All three methods seem to have somewhat comparable lambda values. The Ridge Regression and Stepwise Regression methods have exactly the same Lambda.Min values, while the LASSO Regression has a smaller value.
* **Lambda.1se**: Here we see a wider variation. The LASSO Regression model has a much smaller Lambda.1se value compared to the other two methods, which could potentially indicate that the LASSO model might be simpler with almost the same performance.
* **Training RMSE**: The RMSE values for training data are very close among all three models, suggesting that all methods perform similarly well on the training set. The Stepwise Regression model has a slightly lower RMSE, indicating that it might be slightly better in predicting the training data.
* **Testing RMSE**: The RMSE values for the test set are again similar. This time, the Ridge Regression model has a slightly lower RMSE, indicating better generalization performance to unseen data. However, the difference is quite minimal.

In summary, all three models have similar performance. The choice between them would likely depend on considerations such as interpretability and simplicity of the model, as well as these performance metrics. While Ridge Regression and Stepwise Regression have slightly better performance in testing RMSE and training RMSE respectively, the LASSO model might be favored if a simpler model is desired due to its lower Lambda.1se value.

**Process 15: Preferred Method**

After a thorough analysis and comparison of Ridge Regression, LASSO Regression, and Stepwise Regression methods on the provided dataset, it is challenging to definitively prefer one method over the others based solely on the results. All three methods have shown comparable performance in terms of both training and testing Root Mean Squared Error (RMSE). However, some key points from our analysis can guide us towards an informed decision.

The Ridge Regression model showed the lowest RMSE on the testing data, suggesting it might have slightly better generalization ability to unseen data. However, the difference in testing RMSE among all models was minimal. On the training data, the Stepwise Regression model slightly outperformed the other two models in terms of RMSE, suggesting it might be slightly better in fitting the training data.

When we look at model complexity, as indicated by the Lambda.1se value, LASSO Regression showed a much smaller value compared to Ridge Regression and Stepwise Regression. This suggests that the LASSO model might be simpler or more parsimonious, which could be a deciding factor when simplicity is a priority.

In consideration of the above results, we might lean towards the Ridge Regression model for its marginally superior predictive performance on the test set. However, if a simpler model is required, the LASSO Regression model would be a better choice. The Stepwise Regression model may be preferred if the primary aim is to explain the variation in the response variable using the training data.

It is crucial to note that these conclusions are specific to this particular dataset and the given split of training and testing data. Different datasets or different splits of the data might lead to different results.

To conclude, while the Ridge Regression model might be preferred due to its slightly superior generalization ability, the choice of method would ultimately depend on the specific objectives and requirements of the task. Factors like interpretability, simplicity, computational efficiency, and underlying model assumptions should all be considered in the decision-making process.

**Concluation:**

This homework presented an opportunity to apply and compare various statistical and machine learning methods for regression analysis using a real-world dataset. The main focus of the task was to build and evaluate Ridge Regression, LASSO Regression, and Stepwise Regression models and understand their specific advantages and drawbacks.

We started with a thorough exploratory data analysis, understanding the structure and distribution of the dataset. This process provided us with crucial insights about the dataset, such as the presence of categorical and numerical variables and their interrelationships.

Following this, we fitted and evaluated Ridge and LASSO regression models on the training data, using cross-validation to find the optimal lambda values. The performance of these models was assessed using RMSE as the metric on both the training and test data.

The same process was then applied to the Stepwise Regression model. As with Ridge and LASSO, we evaluated its performance and compared the results to those from the other models.

The comparison of the models suggested that all three performed comparably on the dataset. Ridge Regression demonstrated a slightly lower RMSE on the test set, indicating a possible advantage in terms of generalizability to unseen data. LASSO, with its smaller Lambda.1se value, suggested a more parsimonious model, which might be preferred if model simplicity is a priority. The Stepwise Regression model showed the lowest RMSE on the training data, potentially indicating a better fit to the training data.

This homework allowed us to understand and appreciate the subtleties and nuances involved in choosing a model for prediction or explanatory purposes. We learned that different models might excel in different aspects, and choosing the right model requires careful consideration of these aspects in the context of the specific task at hand.

Overall, this exercise was a valuable experience in practical data analysis and model comparison. The insights gained from this work will undoubtedly be beneficial for future data science projects.