DSCI 445 Project Paper

Paige Galvan, Neha Deshpande, & Witlie Leslie

2024-11-25

Motivation

The goal of our project is to predict mortality from heart failure using behavioral risk factor data. Heart failure is a disease that affects millions of people yearly. Although modern medicine has improved, it can be hard to determine causes of heart failure due to how many variables can affect it. The Heart Failure Clinical Records Dataset provides a collection of medical indicators such as age, ejection fraction, serum creatinine, and co-existing conditions like diabetes and high blood pressure. By analyzing this data, researchers can uncover patterns that contribute to better understanding the progression of heart failure.

The main motivation for our group to study this dataset is to dive a little bit deeper into which factors affect heart failure. Knowing that heart failure is a leading cause of death around the world, finding meaningful patterns can inform public health strategies, such as targeted lifestyle modifications or health care campaigns. The main objective is to transform this raw data into meaningful conclusions on heart disease.

Methodology

Exploratory Analysis Before applying machine learning models, we began by performing an exploratory analysis of the data. This included assessing the linearity and normality of the predictors, identifying any outliers, and exploring potential correlations among the variables. We visualized distributions using histograms and box plots to understand the spread of each feature, and scatter plots to check the relationships between the predictor variables and the target variable (mortality). This helped us determine whether the data required transformations before applying machine learning techniques.

Logistic Regression with Regularization Logistic regression is a go-to method for binary classification, and we explored three versions to analyze predictive performance. First, we fit a basic logistic regression model without regularization as a baseline. While simple, it doesn't handle collinearity or irrelevant predictors. Next, we applied Ridge regression regularization, which penalizes large coefficients to stabilize the model, though it doesn't eliminate predictors, making it less interpretable than Lasso. Finally, we used Lasso regularization, which not only penalizes coefficients, but also performs feature selection by shrinking some to zero, improving interpretability. Comparing their predictive power helps determine which approach balances accuracy and simplicity best.

Because the predictor variables are of varying ranges and units, we began by scaling all continuous features to prevent our regularization techniques from over-penalizing variables with larger ranges. Next, we split our data into a training set (containing 80% of the data) and a test set (containing 20%) so that we could assess the performance of our logistic regression models using cross validation.

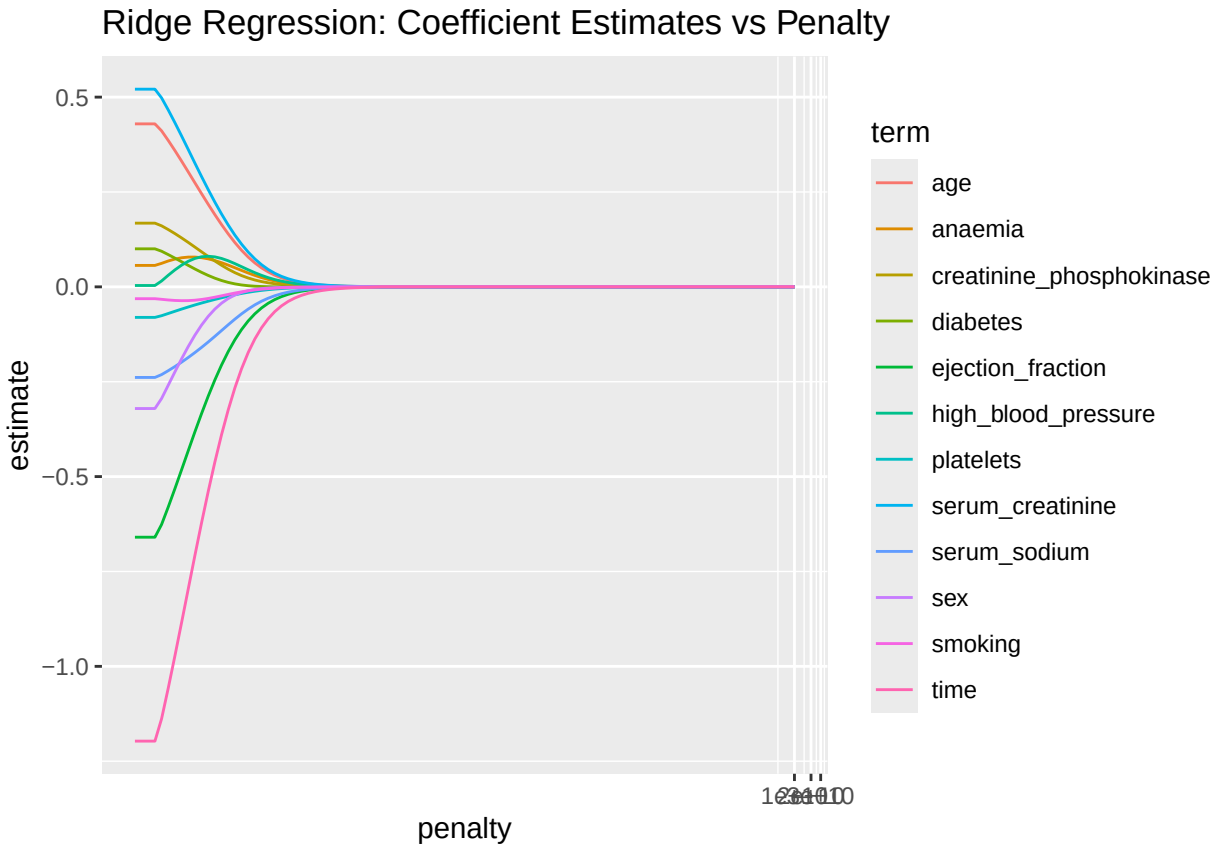
Once this setup was complete, we created our first logistic regression model with no regularization. The metrics we used to assess the performance of these models are roc_auc, which is the area under the receiver-operating characteristic (ROC) curve that represents the probability that the model will correctly rank a randomly selected positive example higher than a negative one, as well as accuracy, which is the proportion of correct

predictions out of all total predictions. Below are the roc_auc and accuracy values of the first logistic regression model with no regularization:

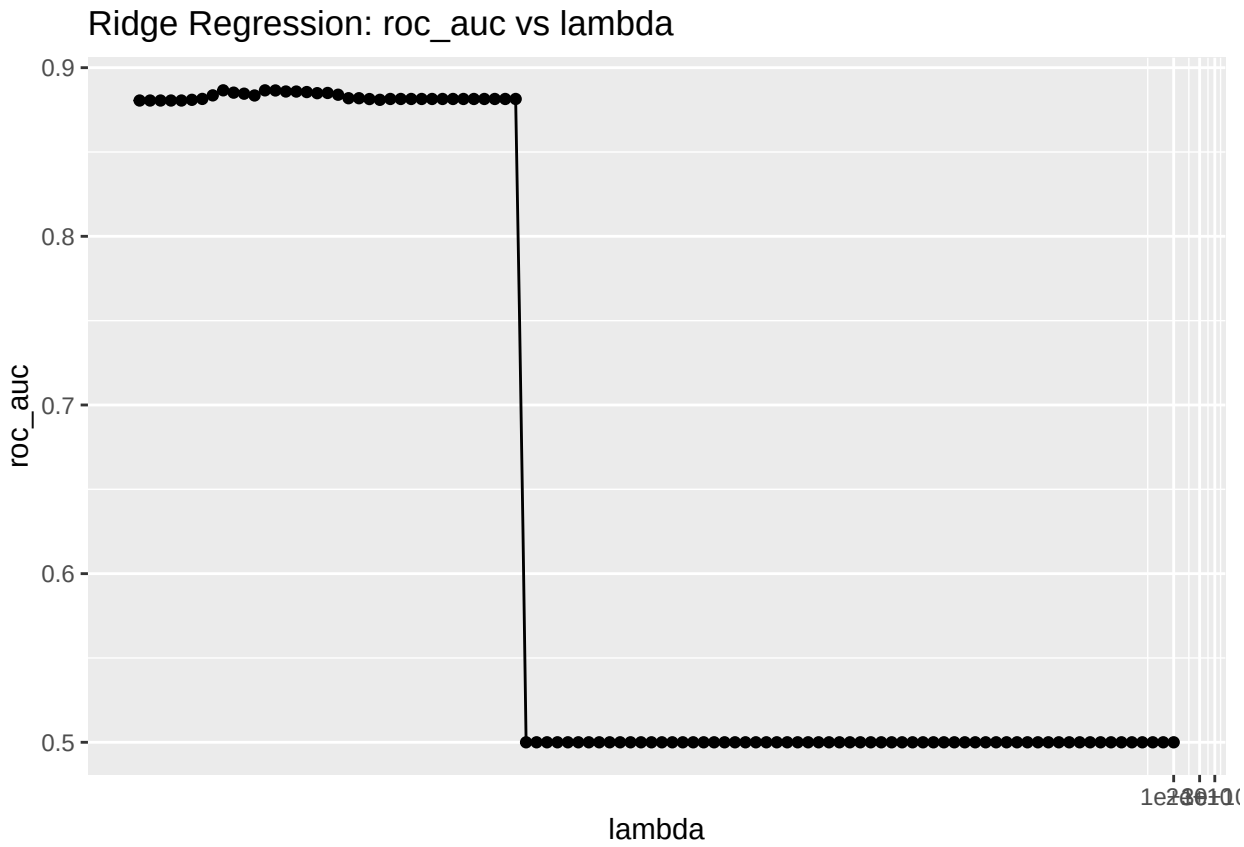
```
## # A tibble: 2 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 roc_auc  binary         0.145
## 2 accuracy binary         0.833
```

The logistic regression model with no regularization gave an roc_auc value of 0.1325 and an accuracy value of 0.8333. This accuracy value is fairly high, but the roc_auc value is very low, which indicates poor performance.

Next, we created a logistic regression model including ridge regression regularization. We determined the optimal lambda penalty value using cross validation.

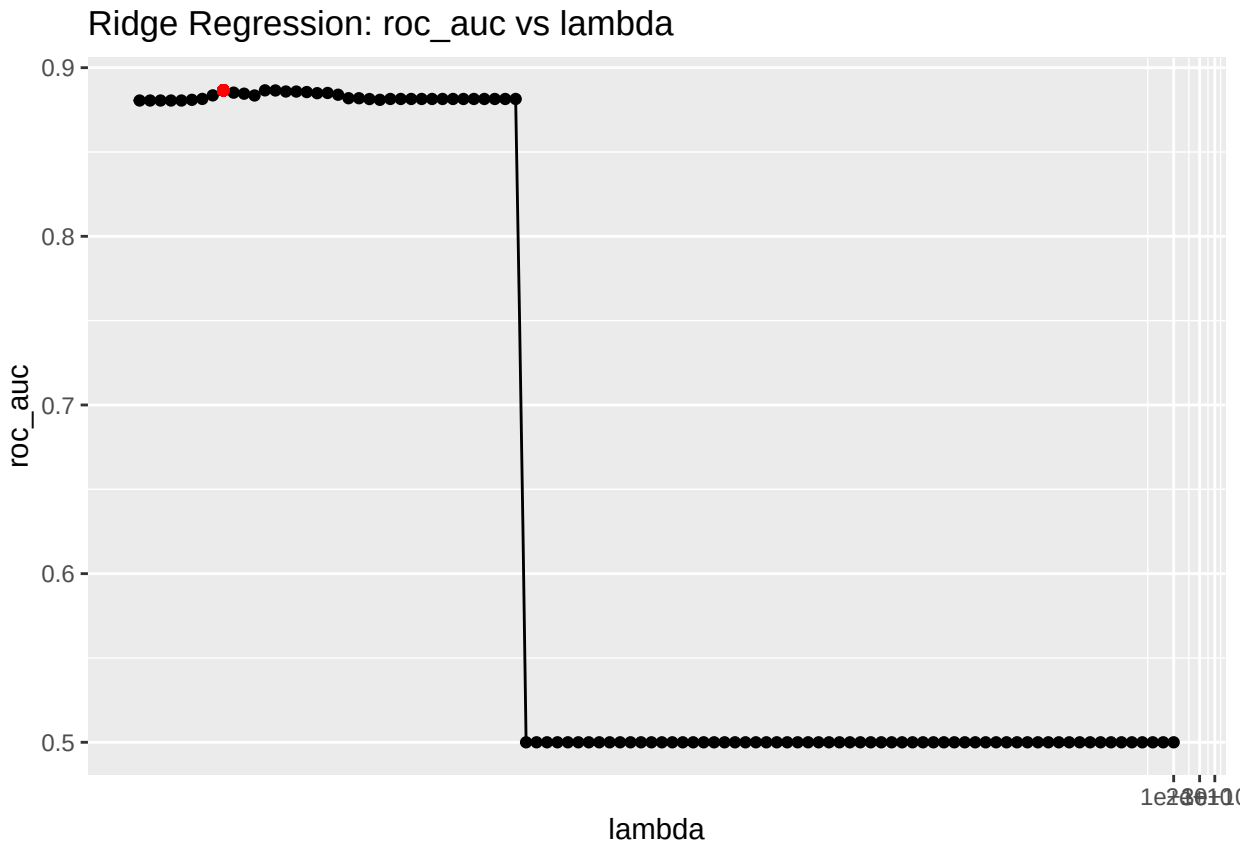


This graph depicts the estimated value of each coefficient corresponding to each feature for each lambda value tested. As the penalty lambda value increases, the coefficient estimates diminish towards zero without being removed entirely.



This graph depicts the roc_auc value for each lambda tested. A higher roc_auc value indicates better performance, and we can see there is a significant and sudden drop in roc_auc value after approximately the first third of lambda values. After completing cross validation, we can see which lambda value demonstrates the best performance:

```
## # A tibble: 1 x 7
##   lambda .metric .estimator  mean     n std_err .config
##    <dbl> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
## 1 0.0933 roc_auc binary     0.887    10  0.0194 Preprocessor1_Model009
```

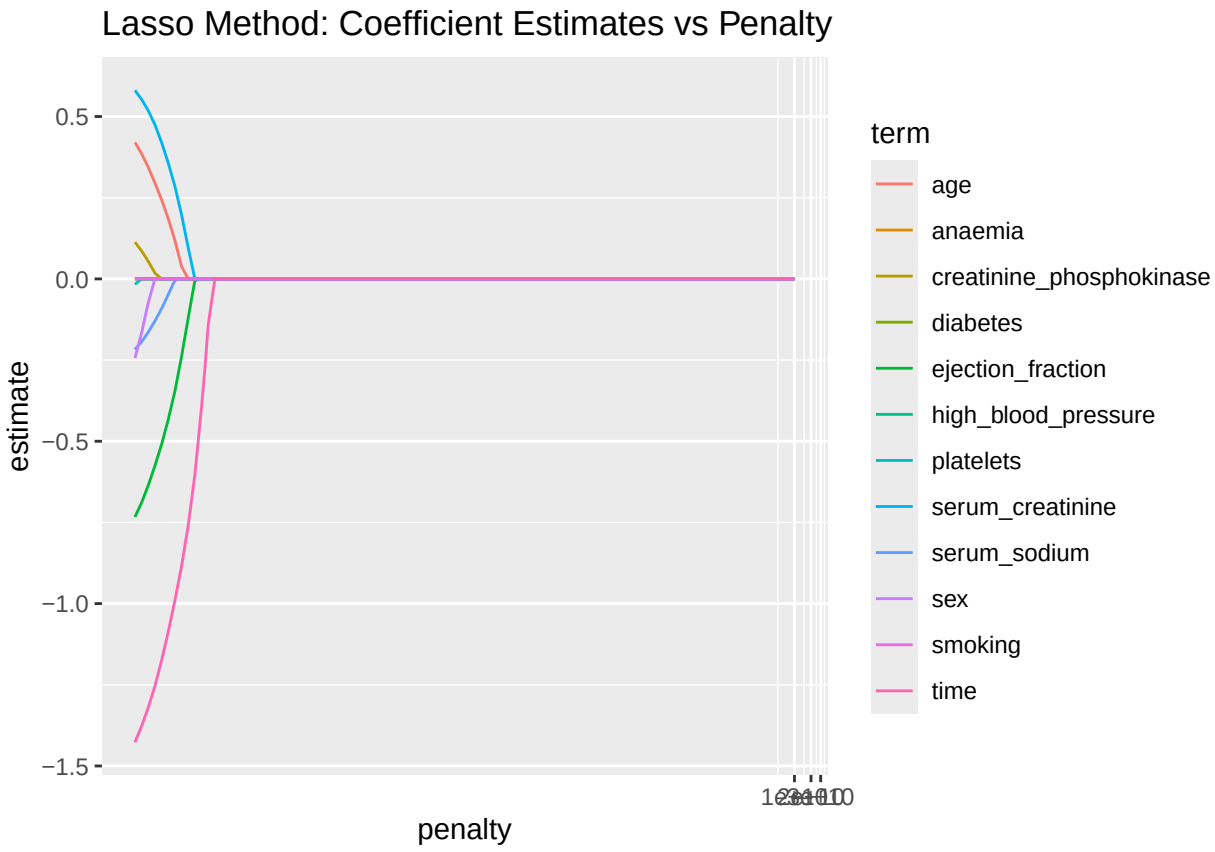


Once we have chosen the optimal penalty, we can fit the ridge regression model with this lambda value and assess its performance with roc_auc and accuracy metrics:

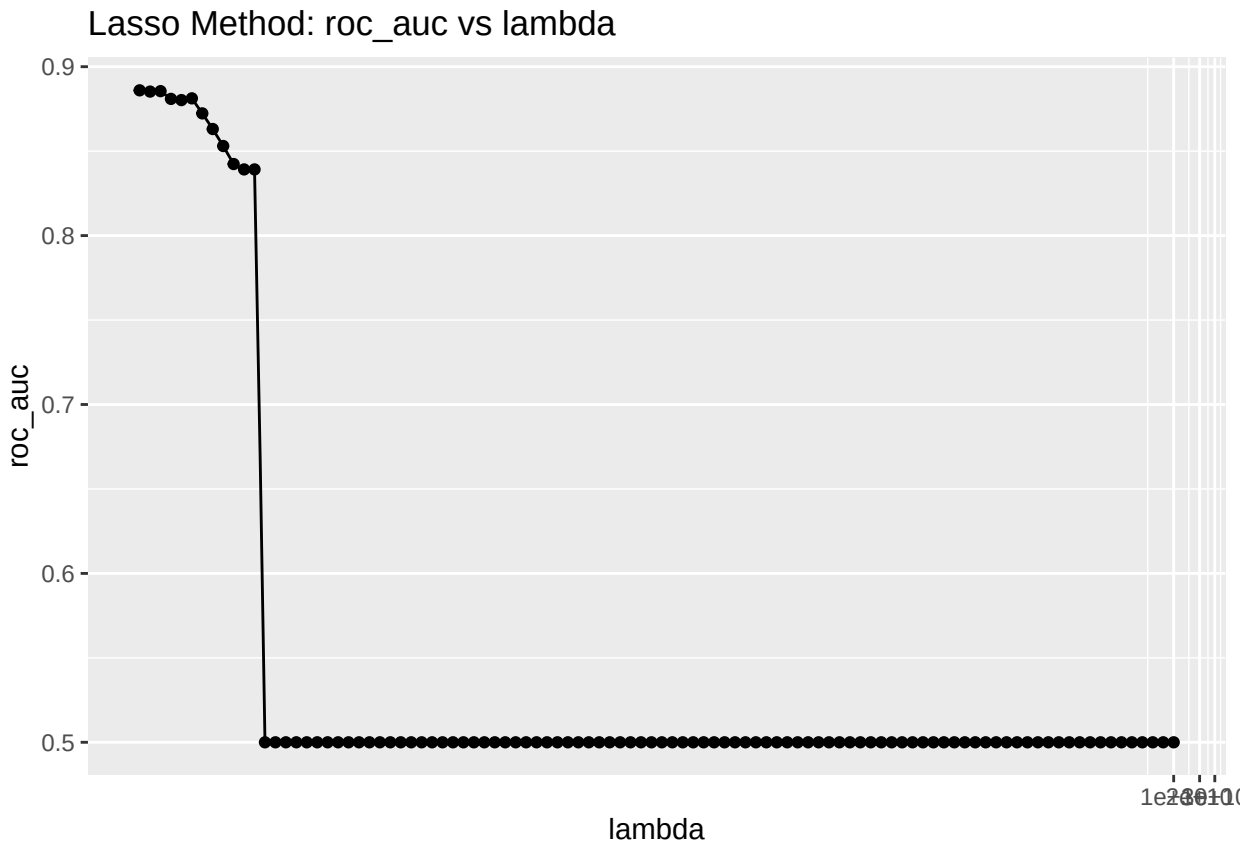
```
## # A tibble: 2 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 roc_auc  binary         0.144
## 2 accuracy binary         0.8
```

The logistic regression model with ridge regression regularization gave an roc_auc value of 0.1374 and an accuracy value of 0.8333. These values are nearly identical to those of the logistic model with no regularization.

For our final logistic regression model, we will implement lasso, once again using cross validation to determine the optimal lambda penalty value.

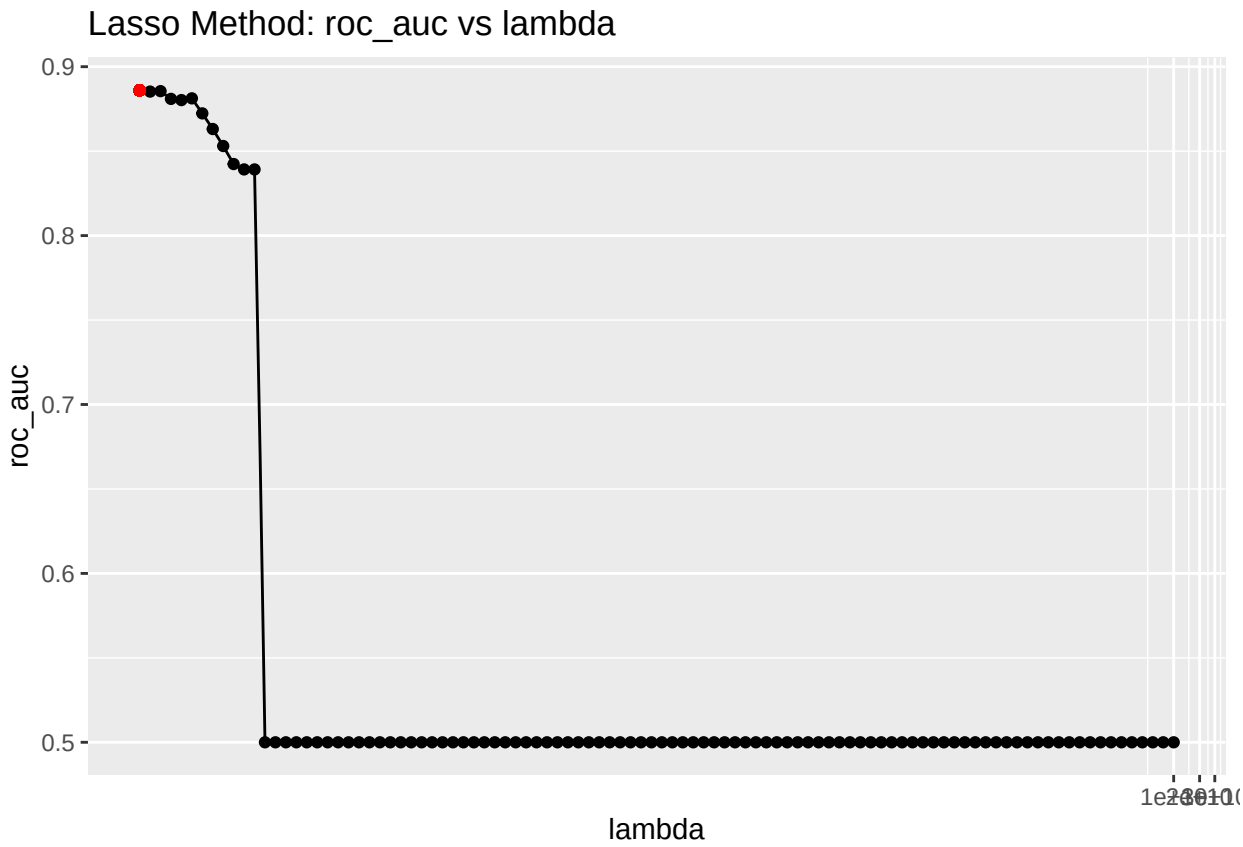


Here we can see the coefficient estimates reduce towards zero as lambda increases. The reduction in coefficient estimates is notably steeper for lasso than with ridge regression. Unlike ridge regression, lasso performs feature selection by driving some coefficients to equal zero, thus eliminating them.



We can see in the graph above a significant and sudden drop in roc_auc value after the first dozen lambda values. After completing cross validation, we can see which lambda value demonstrates the best performance:

```
## # A tibble: 1 x 7
##   lambda .metric .estimator  mean     n std_err .config
##    <dbl> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
## 1   0.01 roc_auc binary     0.886    10  0.0211 Preprocessor1_Model001
```



Again, once we have chosen the optimal penalty, we can fit the lasso model with this lambda value and assess its performance with roc_auc and accuracy metrics:

```
## # A tibble: 2 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 roc_auc  binary         0.134
## 2 accuracy binary         0.833
```

The logistic regression model with ridge regression regularization gave an roc_auc value of 0.1027 and an accuracy value of 0.850. These values are again nearly identical to those of the logistic model with no regularization and the logisitic model with ridge regression. The lasso model provided the highest accuracy of the three models, but also an even lower roc_auc value.

Support Vector Machine Support Vector Machines (SVMs) are primarily binary classifiers. When dealing with more than two classes, SVMs can handle multi-class classification by applying techniques like "one-vs-one" or "one-vs-all," where multiple binary classifications are combined. One of the key advantages of SVMs is their ability to perform non-linear classification, which increases their flexibility and allows them to handle complex decision boundaries. This handels linear and non-linear decision boundaries. Using a linear kernel is good for approximately linear relationships, which our data is. It does not assume any specific distribution of predictors since none of our predictors are normal.

We began with our SVM model looking to the Support Vectors and the parameters of interest. The support vector are important to understanding the model's decisions. They are the informative points and make it the most critical for classification. They demonstrate the most ambiguous points of data.

From this code we conclude that there are only 13 predictive variables we should consider. We also considered

the accuracy of how the SVM model this allows us to understand its predictive capabilities. We also conclude that this model will have an accuracy of 0.817 this inidcates the model.

Random Forest Random Forest is a powerful machine learning algorithm used for both classification and regression. It works by building multiple decision trees and aggregating their predictions to improve accuracy and reduce overfitting. Key advantages include its ability to handle complex, non-linear relationships, manage missing data, and automatically capture feature interactions. Random Forest is also robust to overfitting, particularly compared to individual decision trees, and provides a built-in estimate of model performance through out-of-bag error. Additionally, it offers valuable insights into feature importance, helping to identify which variables most influence the outcome. Overall, Random Forest is particularly effective for high-dimensional datasets, imbalanced classes, and when model interpretability is secondary to prediction accuracy.

Results

None of the three logistic regression models performed well. The model without regularization performed very similarly to those which implemented ridge regression and lasso, indicating that regularization is unnecessary for this data. Roc_auc values of 0.10 - 0.13 indicate that the model is performing worse than chance (0.5). A low roc_auc value indicates poor discrimination ability. Having a high accuracy value with a low roc_auc value might suggest that accuracy is being misleadingly driven up by the model assigning predictions to the majority class (in our case, death event = 0).

References

Davide Chicco, Giuseppe Jurman: Machine learning can predict survival of patients with heart failure from serum creatinine and ejection fraction alone. BMC Medical Informatics and Decision Making 20, 16 (2020). https://bmcmedinformdecismak.biomedcentral.com/articles/10.1186/s12911-020-1023-5