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 affectheart 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 carecampaigns. The main objective is to transform this raw data into meaningful conclusions on heart disease.

Variables

In our project, we aimed to develop a predictive model to determine the likelihood of death resulting from heart failure using clinical patient data. The binary response variable, "DEATH_EVENT," indicates whether a patient has died (value of 1) or survived (value of 0). The dataset includes 12 risk factor variables, comprising of 5 binary variables—anemia status, diabetes status, high blood pressure status, sex, and smoking status—and 7 numerical variables, including age, creatine phosphokinase level, ejection fraction, platelet concentration, serum creatinine level, serum sodium level, and the length of the follow-up period.

Creatine phosphokinase (CPK) is an enzyme found in the blood that indicates tissue stress, often used to assess muscle or heart injury. Ejection fraction refers to the percentage of blood the heart pumps with each contraction, crucial for evaluating heart function. Platelet concentration is the number of platelets in a unit of blood, important for clotting. Serum creatinine measures creatinine levels in the blood, helping assess kidney function, with high levels indicating possible kidney issues. Serum sodium tests the concentration of sodium in the blood, vital for nerve and muscle function and fluid balance.

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.

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

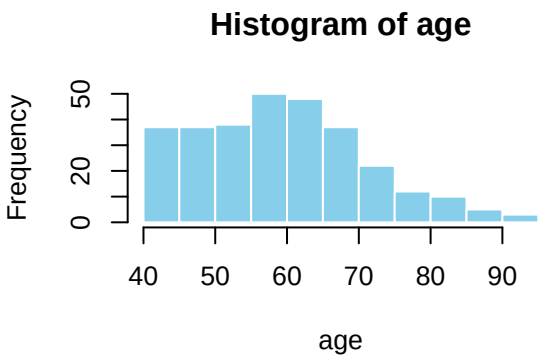
correlations among the variables. We visualized distributions using histograms and box plots to understand the spread of each feature, and scatter plots to check the relationships between the predictor variables and the target variable (mortality). This helped us determine whether the data required transformations before applying machine learning techniques.

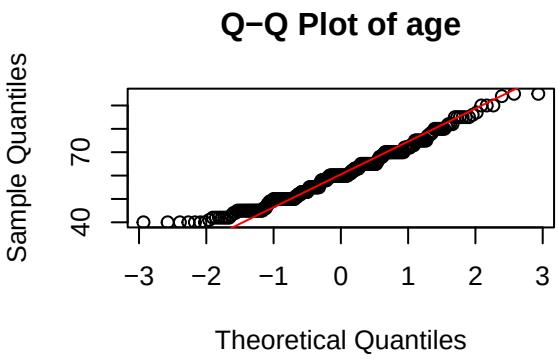
Linearity and Normality

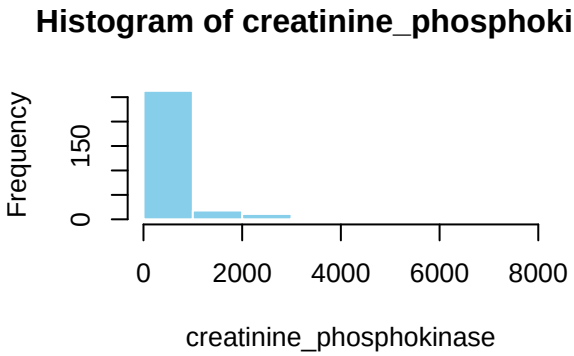
In this analysis, we begin by testing the normality of several continuous variables in the dataset, including age, creatinine phosphokinase, ejection fraction, platelets, serum creatinine, and serum sodium. First, we used histograms and Q-Q plots to visually inspect the distribution of these variables. The histograms for most variables indicated skewness, either to the right or left, suggesting that these variables do not follow a normal distribution. Platelets and serum sodium appeared to be the most normally distributed, but even they showed some deviation from normality. The Q-Q plots confirmed these observations, showing that age, ejection fraction, serum sodium, and platelets were closer to a normal distribution, while other variables exhibited greater deviations.

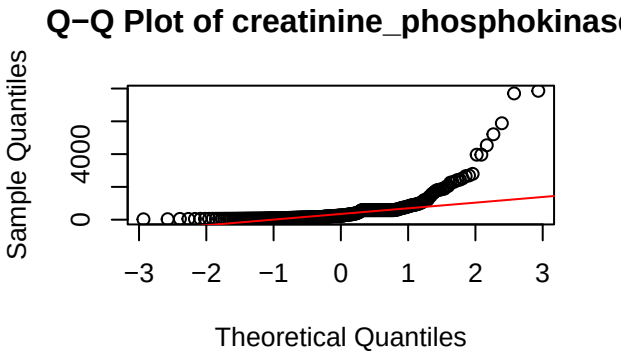
To check the normality of the continuous variables, we performed the Shapiro-Wilk test, and all p-values were below 0.05, indicating that none of the variables followed a normal distribution.

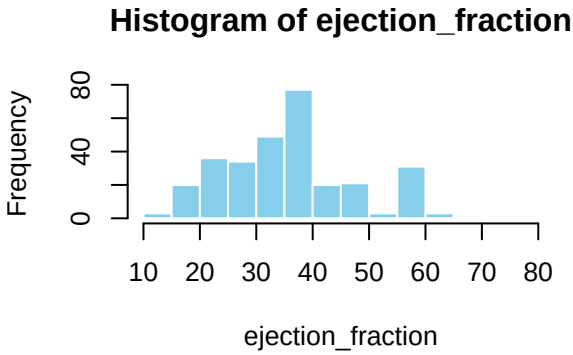
```
data <- read.csv("heart_failure_clinical_records_dataset.csv")
continuous_vars <- c("age", "creatinine_phosphokinase", "ejection_fraction","platelets", "serum_creatini
par(mfrow = c(2, 2)) # Set up a grid for multiple plots
for (var in continuous_vars) {
hist(data[[var]], main = paste("Histogram of", var), xlab = var, col = "skyblue", border = "white")
qqnorm(data[[var]], main = paste("Q-Q Plot of", var))
qqline(data[[var]], col = "red")
}
```

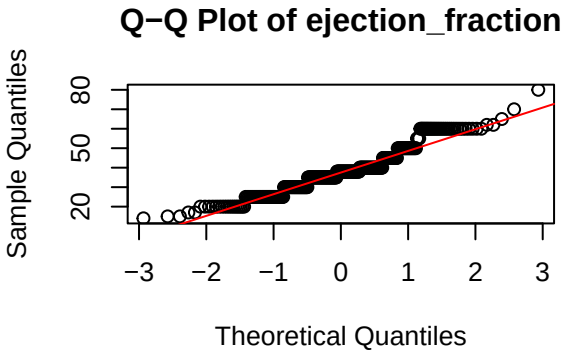

Histogram of age
Frequency
0
20
50
40
50
60
70
80
90
age

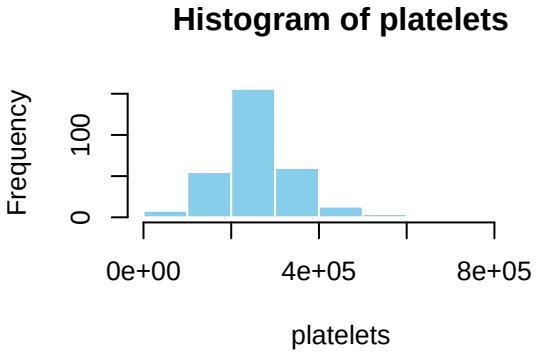

Q−Q Plot of age
Sample Quantiles
40
70
−3
−2
−1
0
1
2
3
Theoretical Quantiles

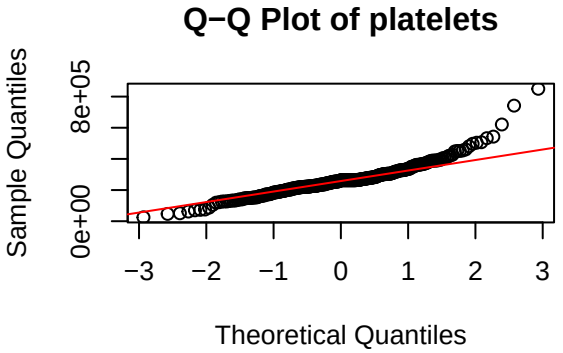

Histogram of creatinine_phosphokinas
Frequency
0
150
0
2000
4000
6000
8000
creatinine_phosphokinase

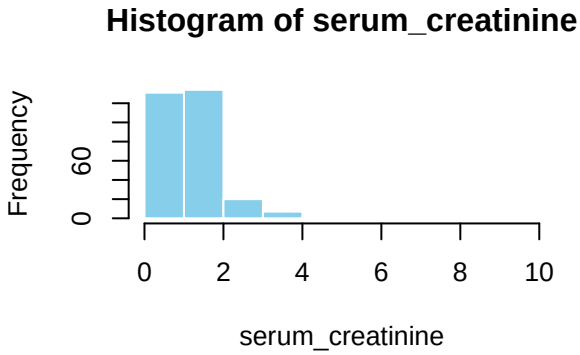

Q−Q Plot of creatinine_phosphokinas
Sample Quantiles
0
4000
−3
−2
−1
0
1
2
3
Theoretical Quantiles

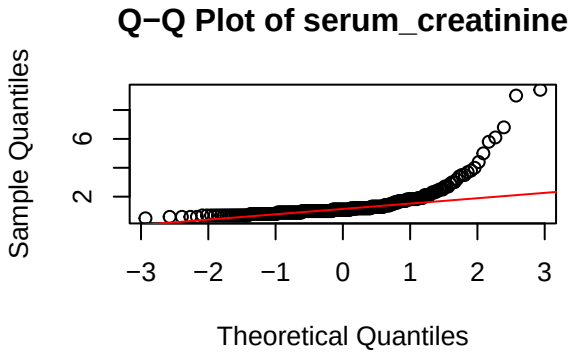

Histogram of ejection_fraction
Frequency
0
40
80
10
20
30
40
50
60
70
80
ejection_fraction

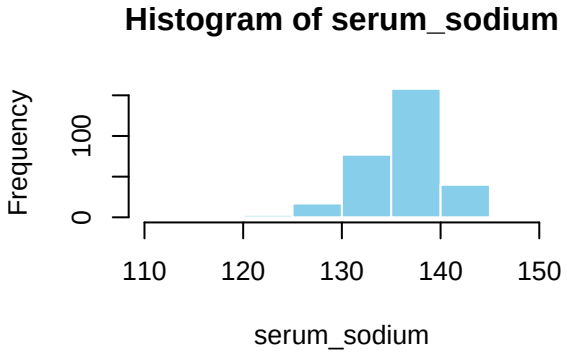

Q−Q Plot of ejection_fraction
Sample Quantiles
20
50
80
−3
−2
−1
0
1
2
3
Theoretical Quantiles

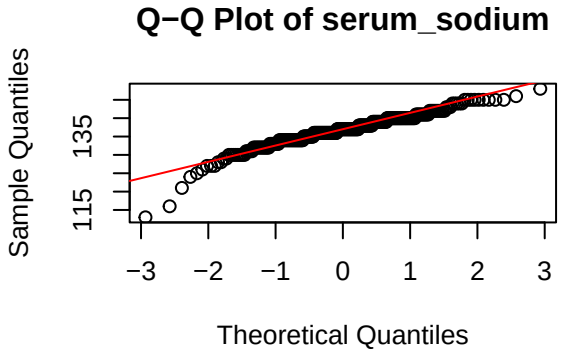

Histogram of platelets
Frequency
0
100
0e+00
4e+05
8e+05
platelets


Q−Q Plot of platelets
Sample Quantiles
0e+00
8e+05
−3
−2
−1
0
1
2
3
Theoretical Quantiles



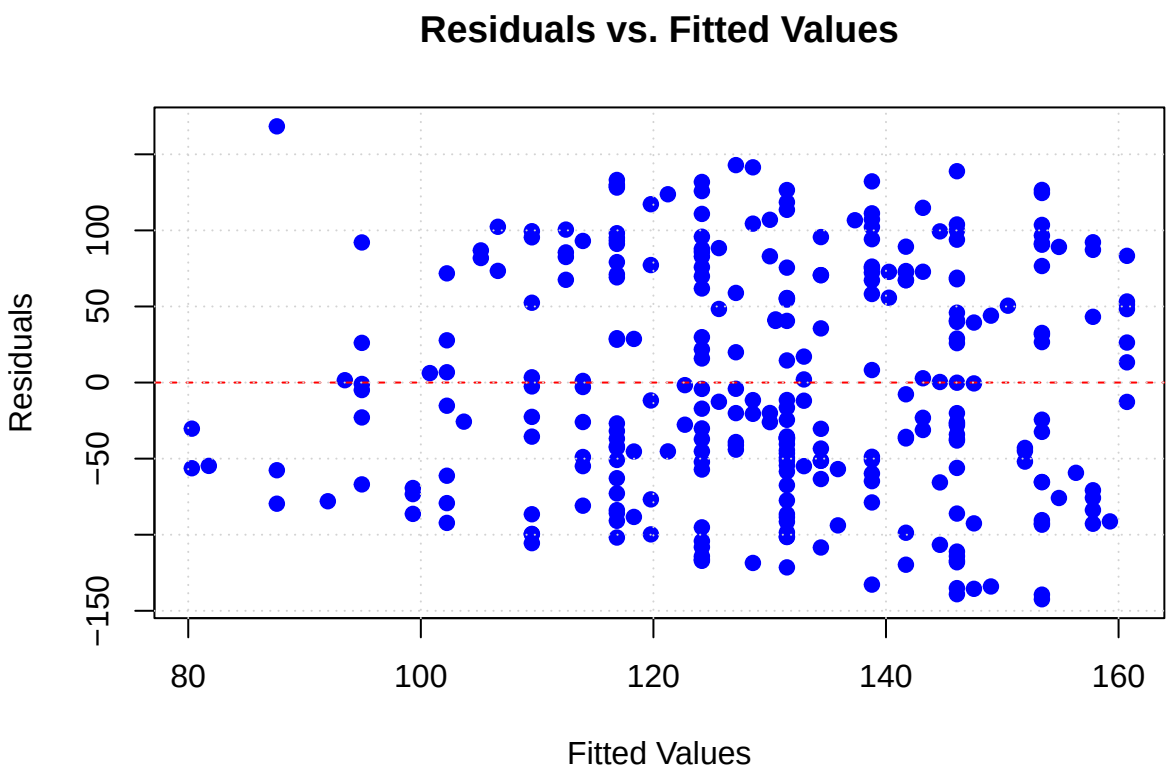






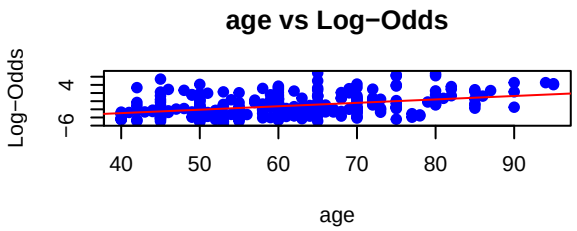
Logistic regression analysis showed that predictor variables, such as age and serum creatinine, were significantly associated with the likelihood of a DEATH_EVENT. Residual diagnostics confirmed a good model fit, with minimal issues in linearity or outliers. Testing quadratic terms for the predictors showed no substantial improvement, indicating that the relationships between the variables and DEATH_EVENT were primarily linear.

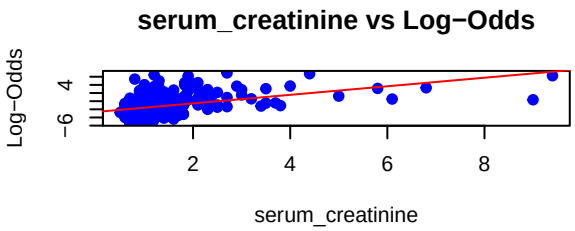
```
data <- read.csv("heart_failure_clinical_records_dataset.csv")
model <- lm(time ~ age, data = data)
fitted_values <- fitted(model)
residuals <- resid(model)
plot(fitted_values, residuals,
main = "Residuals vs. Fitted Values",
xlab = "Fitted Values",
ylab = "Residuals",
pch = 19, col = "blue")
abline(h = 0, col = "red", lty = 2)
grid()
```

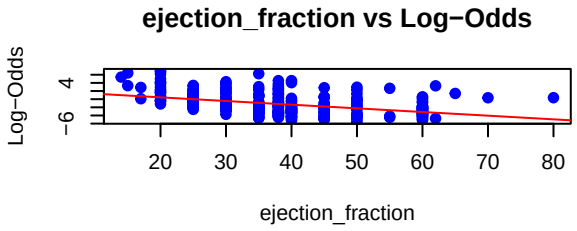


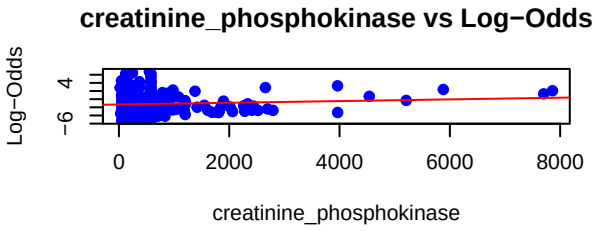
Polynomial Model

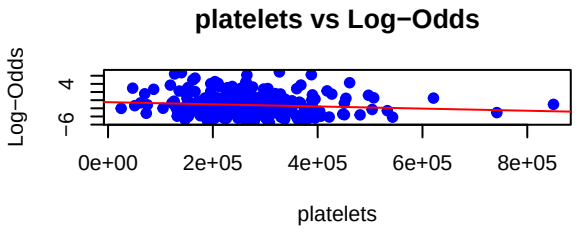
```
poly_model <- glm(DEATH_EVENT ~ poly(age, 2) + poly(serum_creatinine, 2) +
                  poly(ejection_fraction, 2) + poly(creatinine_phosphokinase, 2) +
                  poly(platelets, 2) + poly(serum_sodium, 2) + poly(time, 2) +
                  anaemia + diabetes + high_blood_pressure + sex + smoking,
                  family = binomial, data = data)
data$log_odds <- predict(poly_model, type = "link")  # type = "link" is appropriate for glm

continuous_vars <- c("age", "serum_creatinine", "ejection_fraction",
                     "creatinine_phosphokinase", "platelets", "serum_sodium", "time")

par(mfrow = c(3, 2))

for (var in continuous_vars) {
  plot(data[[var]], data$log_odds,
       main = paste(var, "vs Log-Odds"),
       xlab = var, ylab = "Log-Odds", col = "blue", pch = 19)
  abline(lm(data$log_odds ~ data[[var]]), col = "red")  # Add a red linear trendline
}
```

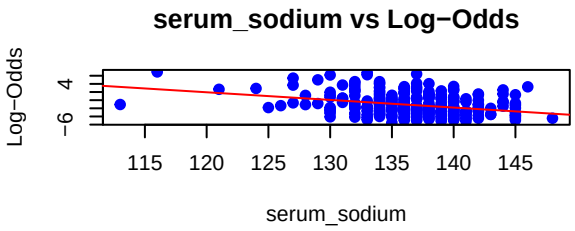








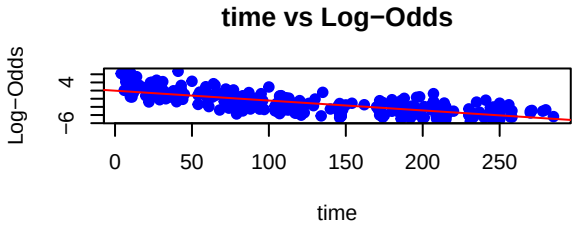





```r
poly_model <- glm(DEATH_EVENT ~ poly(age, 2) + poly(serum_creatinine, 2) +
                  poly(ejection_fraction, 2) + poly(creatinine_phosphokinase, 2) +
                  poly(platelets, 2) + poly(serum_sodium, 2) + poly(time, 2) +
                  anaemia + diabetes + high_blood_pressure + sex + smoking,
                  family = binomial, data = data)

summary(poly_model)
```

```
##
## Call:
## glm(formula = DEATH_EVENT ~ poly(age, 2) + poly(serum_creatinine,
##     2) + poly(ejection_fraction, 2) + poly(creatinine_phosphokinase,
##     2) + poly(platelets, 2) + poly(serum_sodium, 2) + poly(time,
##     2) + anaemia + diabetes + high_blood_pressure + sex + smoking,
##     family = binomial, data = data)
##
## Coefficients:
##                                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)                         -0.93142    0.45420  -2.051  0.04030 *
## poly(age, 2)1                       11.62911    3.63139   3.202  0.00136 **
## poly(age, 2)2                        4.63351    3.95451   1.172  0.24132
## poly(serum_creatinine, 2)1           9.73324    3.22032   3.022  0.00251 **
## poly(serum_creatinine, 2)2          -1.76066    3.48741  -0.505  0.61366
## poly(ejection_fraction, 2)1        -16.22632    3.60869  -4.496 6.91e-06 ***
## poly(ejection_fraction, 2)2          9.93105    3.06586   3.239  0.00120 **
## poly(creatinine_phosphokinase, 2)1   4.54761    3.18791   1.427  0.15372
```

```
## poly(creatinine_phosphokinase, 2)2  -1.92229    3.20433  -0.600  0.54857
## poly(platelets, 2)1                 -2.97233    3.17068  -0.937  0.34853
## poly(platelets, 2)2                  3.26681    2.87279   1.137  0.25547
## poly(serum_sodium, 2)1              -4.57133    3.40748  -1.342  0.17974
## poly(serum_sodium, 2)2               0.02429    3.15791   0.008  0.99386
## poly(time, 2)1                     -25.52735    4.14293  -6.162 7.20e-10 ***
## poly(time, 2)2                      10.33136    3.69314   2.797  0.00515 **
## anaemia                             -0.03995    0.39430  -0.101  0.91929
## diabetes                             0.25905    0.38705   0.669  0.50331
## high_blood_pressure                  0.04917    0.40095   0.123  0.90239
## sex                                 -0.52925    0.44117  -1.200  0.23028
## smoking                             -0.04753    0.44388  -0.107  0.91473
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 375.35  on 298  degrees of freedom
## Residual deviance: 198.74  on 279  degrees of freedom
## AIC: 238.74
##
## Number of Fisher Scoring iterations: 6
```



The logistic regression model includes polynomial terms for several continuous variables, such as age, serum creatinine, and ejection fraction, to account for non-linear relationships between these predictors and the outcome variable (death event). Using polynomial terms allows the model to capture more complex patterns

that a simple linear relationship might miss. For example, the effect of age or serum creatinine on the likelihood of a death event might not be constant, but could change as these variables increase. The polynomial terms help to model this non-linearity, improving the model's ability to fit the data and providing more accurate predictions. This approach ensures that the relationships between predictors and the outcome are represented more flexibly, which is particularly important when dealing with real-world data that may not always follow linear trends

Logistic Regression

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.

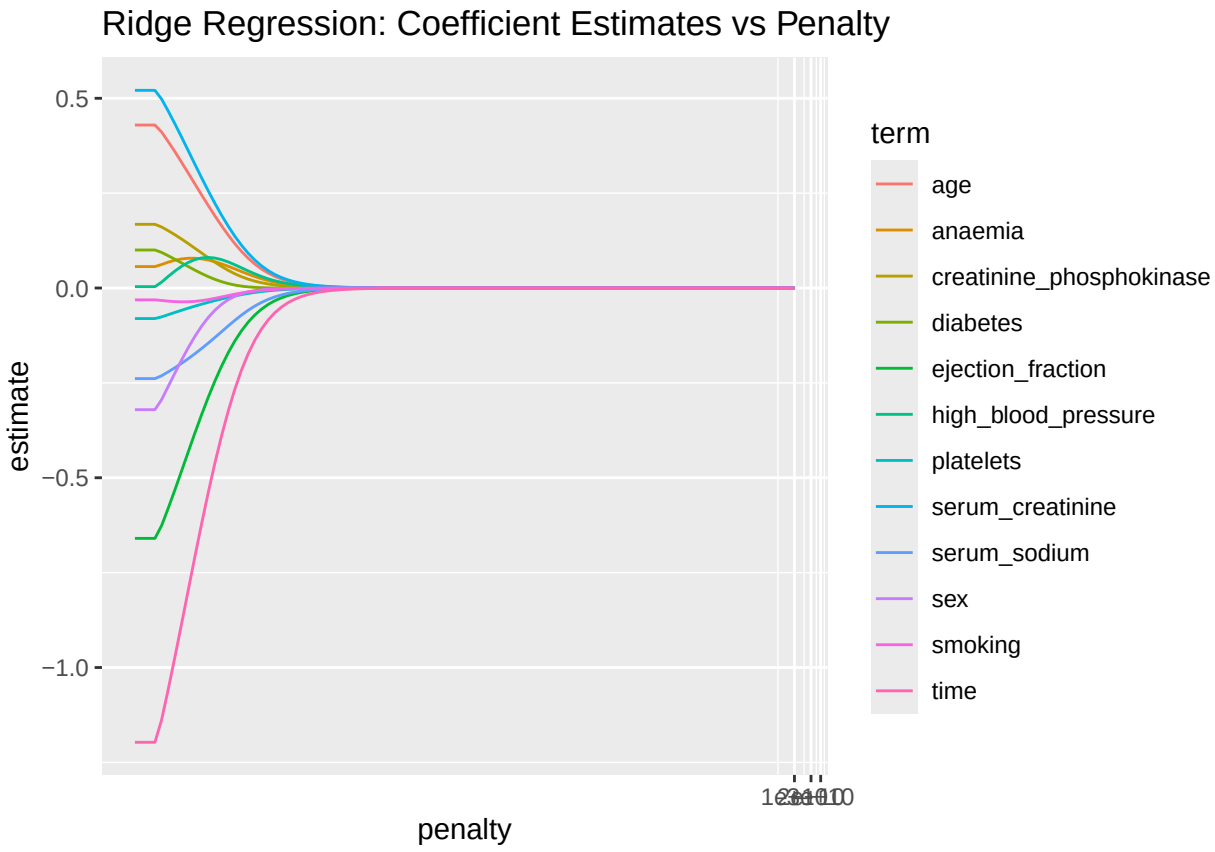
Because the predictor variables are of varying ranges and units, we began by scaling all continuous features to prevent our regularization techniques from over-penalizing variables with larger ranges. Next, we split our data into a training set (containing 80% of the data) and a test set (containing 20%) so that we could assess the performance of our logistic regression models using 10-fold cross validation.

Table 1: Logistic Regression Coefficients and P-values

	Estimate	**Std..Error**	**p.value**
(Intercept)	-1.034	0.487	0.034
age	0.915	0.242	0.000
anaemia	-0.059	0.414	0.886
creatinine_phosphokinase	0.429	0.230	0.062
diabetes	0.286	0.394	0.468
ejection_fraction	-0.897	0.212	0.000
high_blood_pressure	-0.253	0.418	0.544
platelets	-0.207	0.213	0.333
serum_creatinine	0.676	0.213	0.001
serum_sodium	-0.185	0.194	0.342
sex	-0.702	0.472	0.137
smoking	-0.041	0.472	0.931
time	-1.715	0.277	0.000

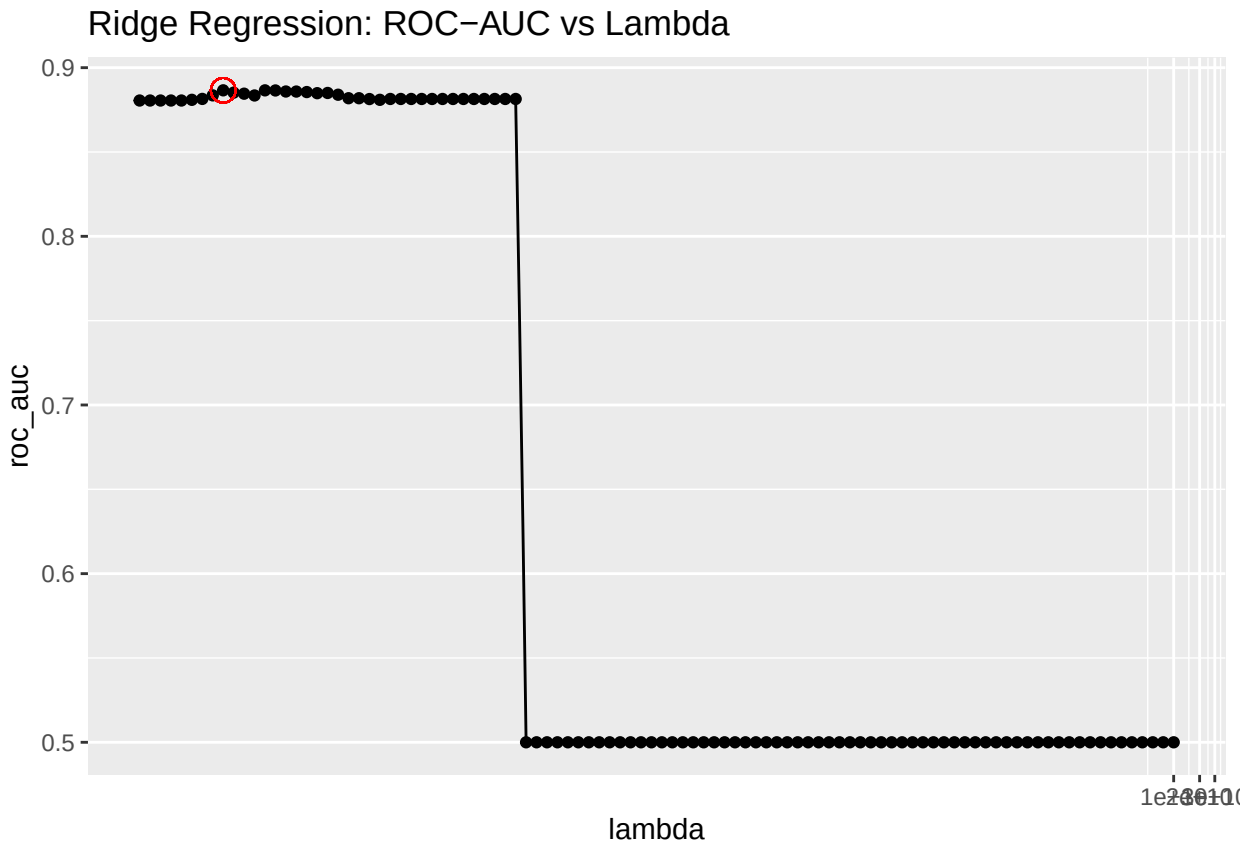
The logistic regression model found that four predictors displayed significance: age, ejection fraction, serum creatine, and time.

Next, we created a logistic regression model with ridge regression regularization. We determined the optimal lambda penalty value using 10-fold cross validation.



The graph above depicts the estimated value of each coefficient for each lambda value tested. As the lambda penalty value increases, the coefficient estimates gradually diminish towards zero without being removed entirely.

The graph below depicts the ROC-AUC value for each lambda tested. 10-fold cross validation determined the best penalty to be lambda = 0.0933 with an ROC-AUC value of 0.887, which is indicated on the graph below with a red circle.



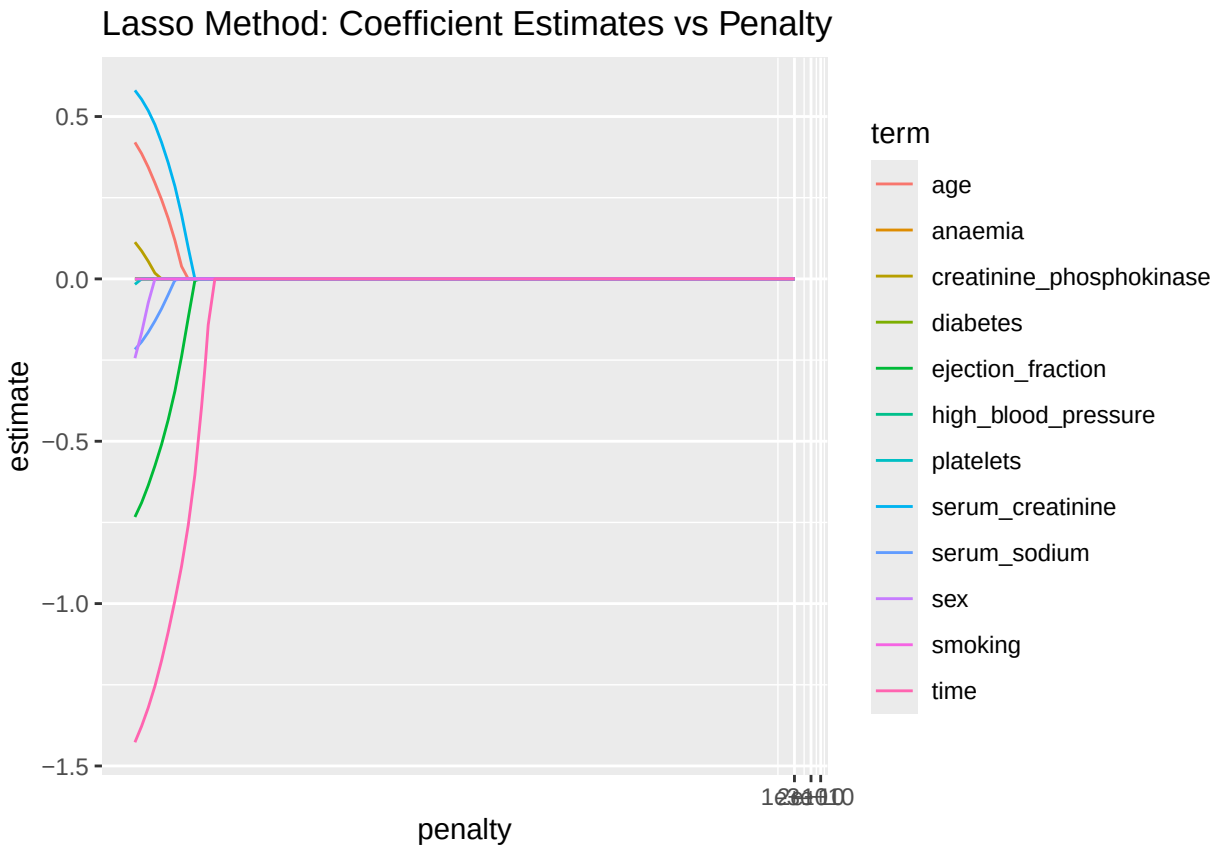
Once we have chosen the optimal penalty, we can fit the ridge regression model with this lambda value and view its estimates.

Table 2: Ridge Regression Coefficients and P-values

	Estimate	**Std..Error**	**p.value**
(Intercept)	-1.034	0.487	0.034
age	0.915	0.242	0.000
anaemia	-0.059	0.414	0.886
creatinine_phosphokinase	0.429	0.230	0.062
diabetes	0.286	0.394	0.468
ejection_fraction	-0.897	0.212	0.000
high_blood_pressure	-0.253	0.418	0.544
platelets	-0.207	0.213	0.333
serum_creatinine	0.676	0.213	0.001
serum_sodium	-0.185	0.194	0.342
sex	-0.702	0.472	0.137
smoking	-0.041	0.472	0.931
time	-1.715	0.277	0.000

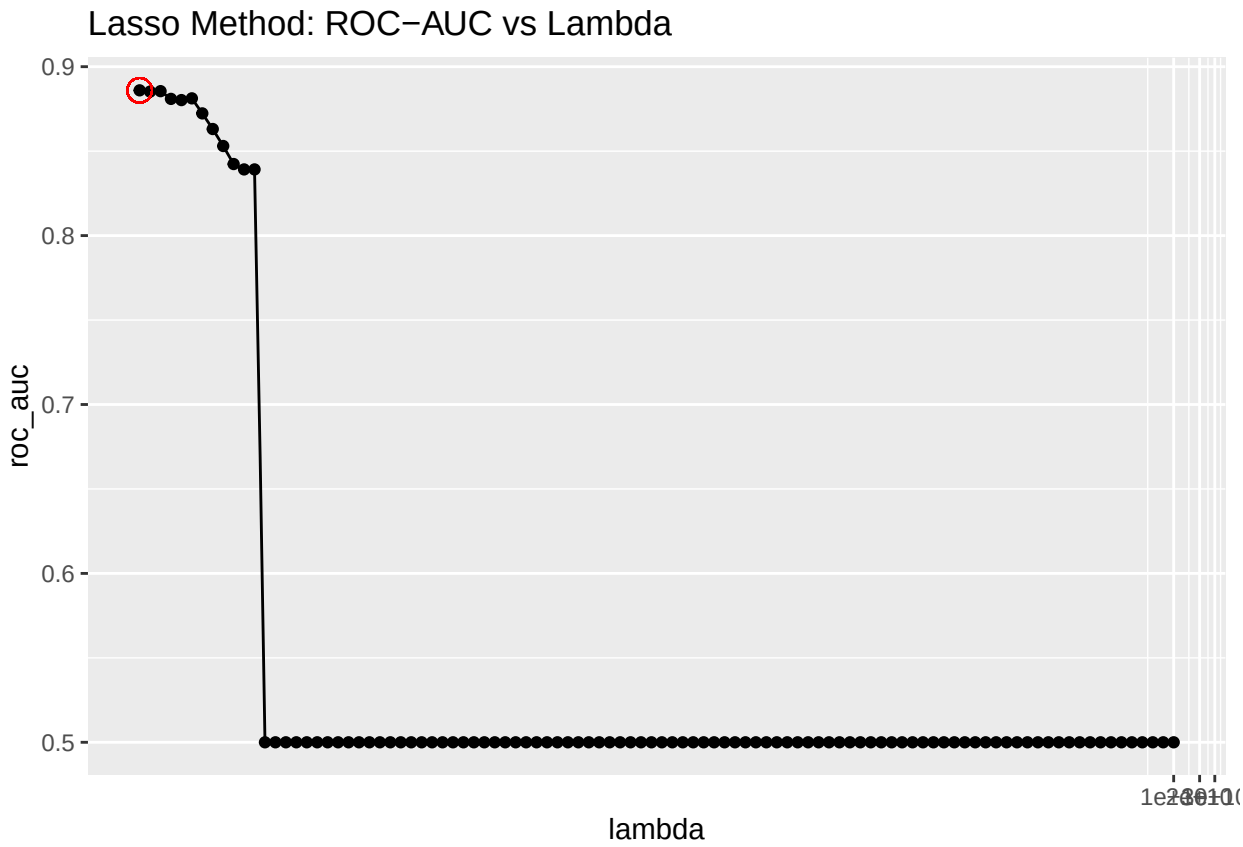
The ridge regression model found the same four predictor variables significant as the logistic regression model without regularization: age, ejection fraction, serum creatine, and time. The coefficient estimates and p-values are overall very similar to those of the first logistic regression model.

For our final logistic regression model, we will implement lasso, once again using 10-fold cross validation to determine the optimal lambda penalty value.



In the graph above, we can see the coefficient estimates diminish 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 from the model.

The graph below depicts the ROC-AUC value for each lambda tested. 10-fold cross validation determined the best penalty to be lambda = 0.01 with an ROC-AUC value of 0.879, which is indicated on the graph below with a red circle.



Now with our chosen penalty, we can fit the lasso model with this lambda value and view its estimates.

Table 3: Lasso Coefficients and P-values

	Estimate	**Std..Error**	**p.value**
(Intercept)	-1.034	0.487	0.034
age	0.915	0.242	0.000
anaemia	-0.059	0.414	0.886
creatinine_phosphokinase	0.429	0.230	0.062
diabetes	0.286	0.394	0.468
ejection_fraction	-0.897	0.212	0.000
high_blood_pressure	-0.253	0.418	0.544
platelets	-0.207	0.213	0.333
serum_creatinine	0.676	0.213	0.001
serum_sodium	-0.185	0.194	0.342
sex	-0.702	0.472	0.137
smoking	-0.041	0.472	0.931
time	-1.715	0.277	0.000

The lasso model again found the same four predictor variables significant as both the logistic regression and ridge regression models: age, ejection fraction, serum creatine, and time. The coefficient estimates and p-values are again very similar to those of the logistic regression and ridge regression models.

Because the ridge regression model and the lasso model resulted in estimates very similar to those of the logistic model without regularization, this indicates that regularization was likely unnecessary for our data set.

Now we will investigate the predictive performance of our three logistic regression models. The metrics we used to assess the performance of these models are ROC-AUC—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.

Logistic Model Metrics:

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

Ridge Model Metrics:

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

Lasso Model Metrics:

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

The 3 logistic models all had very similar performance metrics. This further indicates that regularization was not necessary for this data set. All 3 models had decently high accuracy values (0.83, 0.80, 0.82), but very poor ROC-AUC values (0.145, 0.144, 0.133). Such low ROC-AUC values indicate poor discrimination ability. Because accuracy scores are high and ROC-AUC values are so low, it is likely that accuracy is being erroneously driven up by predictions being assigned to the majority class. The majority class in our data set is DEATH_EVENT = 0, indicating a patient had not deceased during the follow-up period.

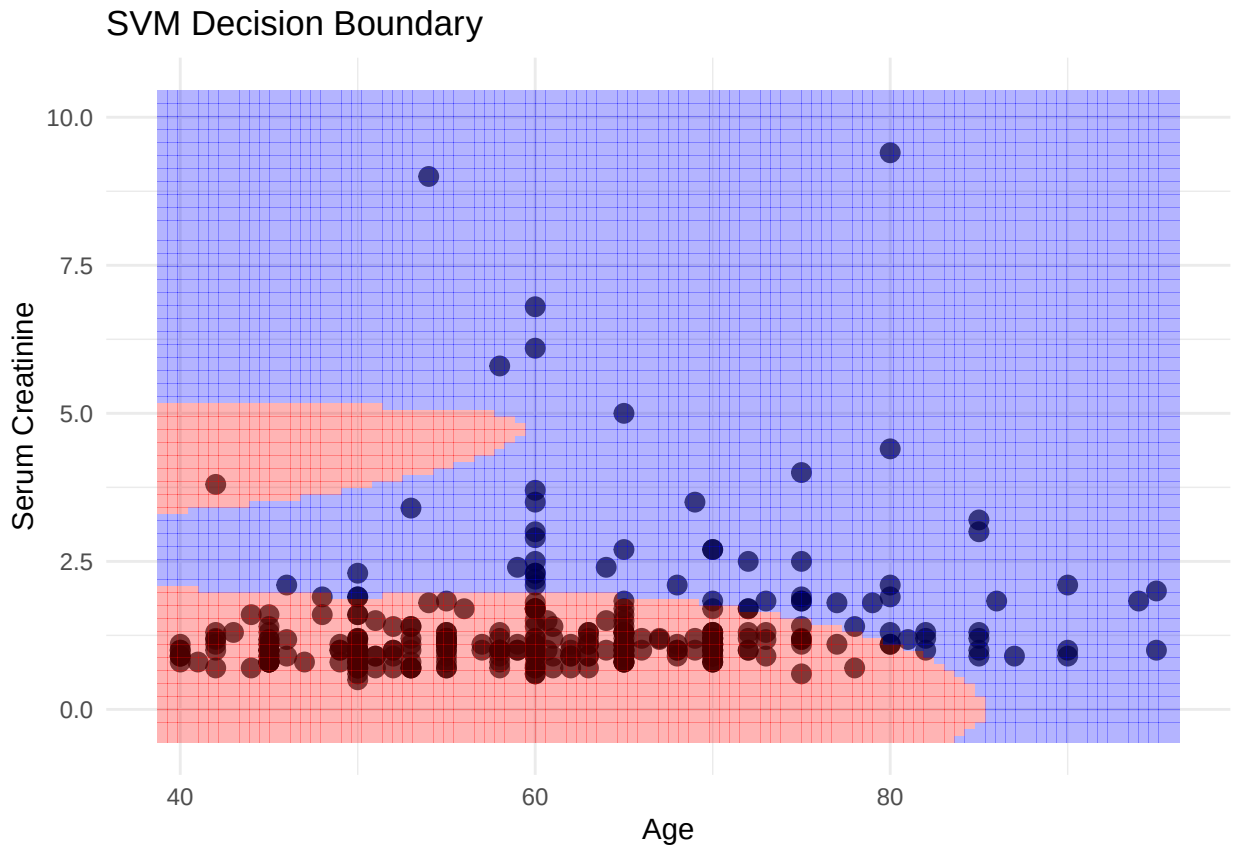
Next we will explore alternative modeling methods that might address the limitations and performance issues of our logistic regression models.

Support Vector Machine

Support Vector Machines (SVMs) are binary classifiers. SVMs can handle multi-class classification including our response variable which is just two classes. 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. We decided to use SVM as our data was only somewhat linear. Earlier we established age, ejection fraction, serum sodium, and platelets were closer to a normal distribution, while other variables had deviations. We also found that all p-values were below 0.05, indicating that none of the variables followed a normal distribution. Using this information we were able to confirm age and serum creatinine were associated with the likelihood of death. Because of this I decided to use age and serum creatinine as our main points of analysis.

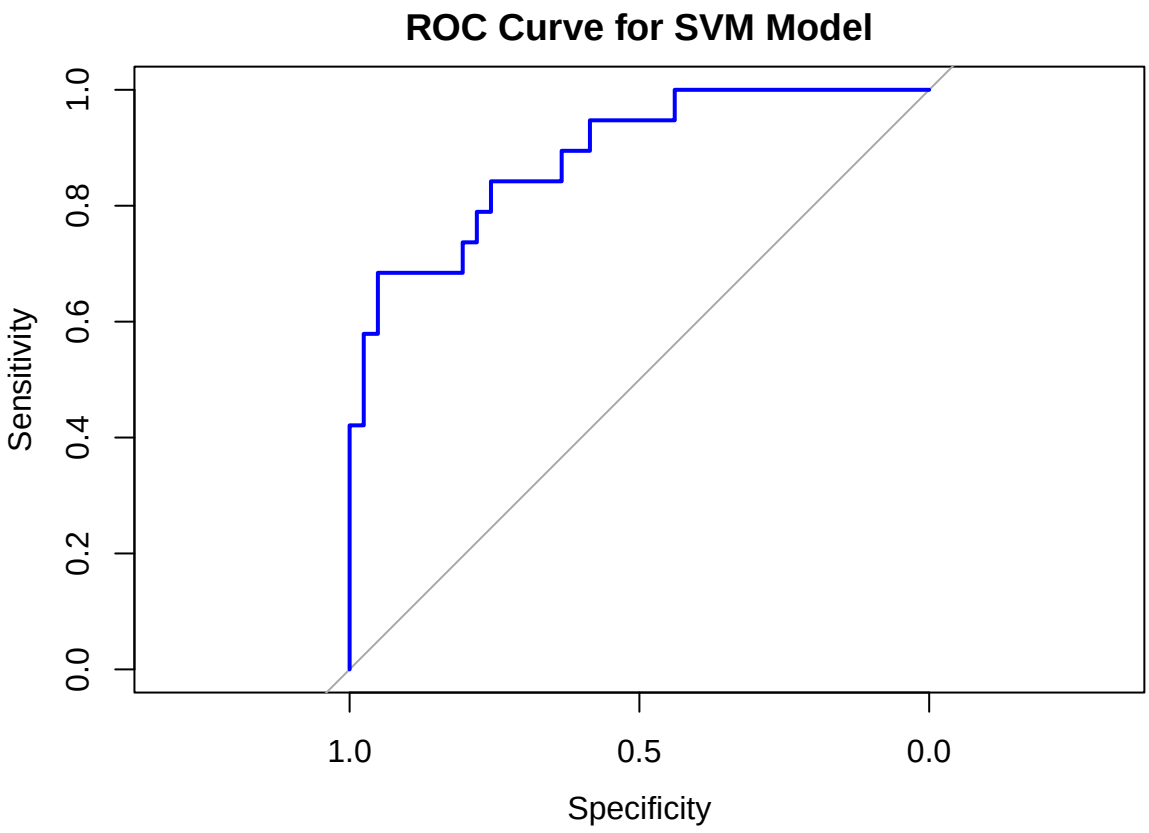
We began with our SVM model looking to the Support Vectors and the parameters of interest. The support vector are important to understanding the model's decisions. They are the informative points and make it the most critical for classification. We started by splitting the data into training set, with 80% of the data, and test set, with 20% of our data. We also tuned the SVM model with `C = 0.5` and `sigma = 0.0562`, in order to produce our best results.

From this code we conclude that there are only 13 predictive variables we should consider. We also looked into the accuracy of the SVM model this allows us to understand its predictive capabilities. We conclude that this model will have an accuracy of 0.817 this indicates the model has a 81.7% of making correct predictions. Using this information we decided to look into a SVM Descision Boundry.



The SVM Boundry analyzes the age vs serum creatinine using the death effect for males vs females. The Red of the grpah represents the Male populations, meanwhile Blue represents the Female population. Overall, serium creatinine and age has little effect on the death in males. Meanwhile age and serum creatinine has a large response on females.

Finally, to confirm and quantity these results we performed a Receiver Operator Characteristic, a ROC curve is a plot used to show the diagnostic ability of binary classifiers. A ROC curve plots the true positive rate (TPR) against the false positive rate (FPR). In the graph below we analyze the Death Effect and the overall true positive rate of this model.



Using this we can see a ROC Curve of 0.86. This graph demonstrated the Death effect from the SVM Model had a sensitivity of 86%, and a specificity of 66%. Indicating a True Positive rate of 86 and a False Positive Rate of 66.

Using all of the information concluded from this model we can see that it overall performed better than any of the logistic regression approaches. However, I wanted to try one more approach so I began Random Forest.

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 good for overfitting, compared to individual decision trees, and provides a built-in estimate of model performance through 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.

We decided to use this model as our data was not perfectly linear and had normality issues. We began by splitting the data into training set, with 80% of the data, and test set, with 20% of our data. We found that we should use an Mtry=6 to preform best.

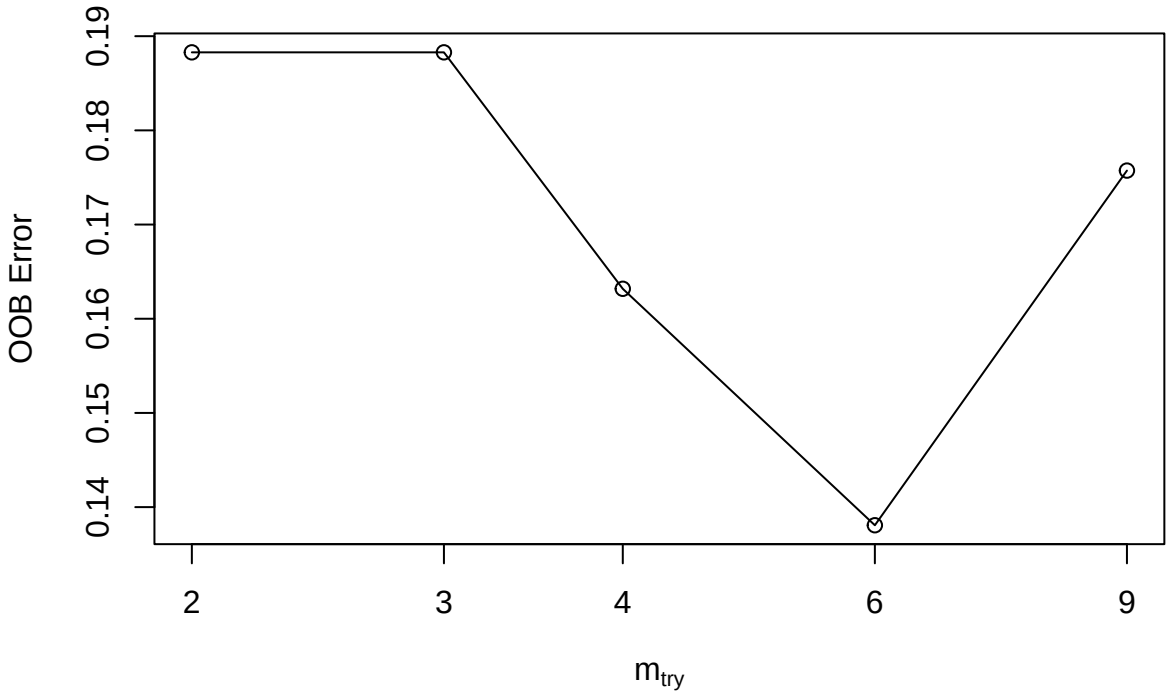
```
## [1] 0
```

```
## [1] 239  13
```

```
## [1] 60 13
```

```
##                                0           1 MeanDecreaseAccuracy
## age                       4.3187246  7.01817289           8.02313874
## anaemia                  -0.5750991 -0.40993099          -0.77722533
## creatinine_phosphokinase -0.8313186  0.30496375          -0.48385634
## diabetes                  0.1540913 -0.87686572          -0.50017880
## ejection_fraction         8.0017788  9.95829603          11.83050036
## high_blood_pressure       0.2668167 -0.07193728           0.06459977
## platelets                -0.1324908 -0.96907947          -0.58833876
## serum_creatinine         13.7955903 14.92363666          19.07434646
## serum_sodium              1.4128593  8.47793838           6.64839609
## sex                       0.5043524 -1.41995691          -0.47369219
## smoking                   2.0409700 -0.29764431           1.46177533
## time                     32.6080324 29.45940873          37.62779320
##                          MeanDecreaseGini
## age                              9.612949
## anaemia                          1.631675
## creatinine_phosphokinase         8.607178
## diabetes                         1.402893
## ejection_fraction               11.341475
## high_blood_pressure              1.632764
## platelets                        8.409864
## serum_creatinine                16.632142
## serum_sodium                     8.938591
## sex                              1.513133
## smoking                          1.180270
## time                            32.726817
```

```
## mtry = 3  OOB error = 18.83%
## Searching left ...
## mtry = 2     OOB error = 18.83%
## 0 0.01
## Searching right ...
## mtry = 4     OOB error = 16.32%
## 0.1333333 0.01
## mtry = 6     OOB error = 13.81%
## 0.1538462 0.01
## mtry = 9     OOB error = 17.57%
## -0.2727273 0.01
```

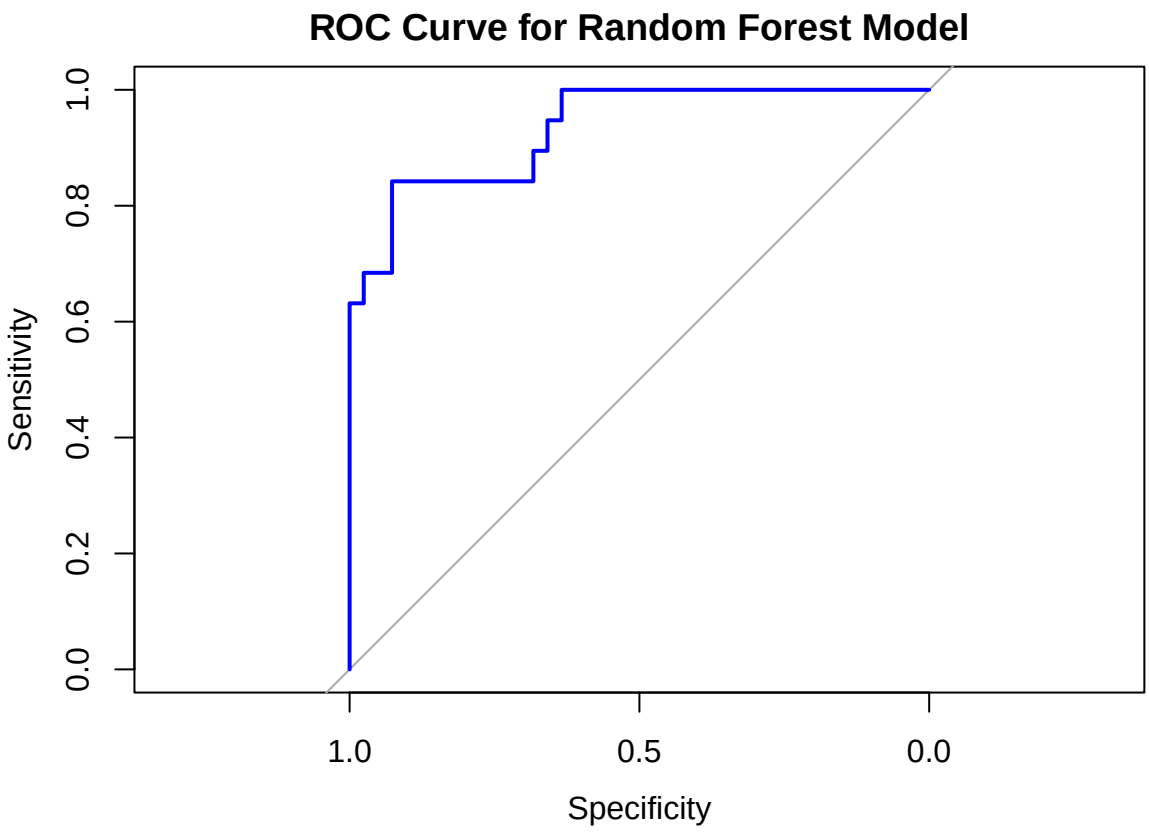



```
##       mtry  OOBError
## 2.OOB    2 0.1882845
## 3.OOB    3 0.1882845
## 4.OOB    4 0.1631799
## 6.OOB    6 0.1380753
## 9.OOB    9 0.1757322
```

```
##          Actual
## Predicted  0  1
##         0 38  3
##         1  3 16
```

```
## Accuracy:  0.9
```

Using the information above we found the dimensions of both the training data and testing data. We also broke down the rf_model accuracy of each variable for the model. We analyzed each Mtry and what the OOB error of each would be. Finally we found the confusion matrix and the accuracy of the overall model. The confusion matrix quantifies the model's predictions using this we saw the majority of true labels correctly matched with the predicted labels, indicating high accuracy. We also found an accuracy score 0.9 from our random forest model. This indicates that our model should offer 90% true positive predictions. Inorder to confirm these results we performed a ROC curve.



With the ROC Curve graph above we found a 0.91 rate. It indicated a 91% sensitivity rate indicating 91% true positive rate. However, this model has a moderate specifity with a 69% rate.

Overall, the Random Forest model performed better, especially in terms of overall accuracy and detecting positive cases.

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).

Performing both SVM and Random Forest we were able to conclude our best model was Random Forest. Our SVM held a high accuracy rating and a high ROC-AUC value indicating it would be a good prediction model for our data. However, once we tested the accuracy and ROC-AUC of Random Forest we can see this highly outranks any other options. This model performs the best overall with the predictions to the majority class 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

Mount Sinai. "Creatine Phosphokinase Test." Mount Sinai Health Library, https://www.mountsinai.org/health-library/tests/creatine-phosphokinase-test.

Cleveland Clinic. "Ejection Fraction." Cleveland Clinic, https://my.clevelandclinic.org/health/articles/16950-ejection-fraction.

RegenLab. "Platelet Concentration: Concentration Factor." RegenLab, 7 Apr. 2022, https://www.regenlab.com/2022/04/07/platelet-concentration-concentration-factor/#:~:text=Platelet%20concentration%20is%20the%20number,exp

Kidney Fund. "Serum Creatinine Test." American Kidney Fund, https://www.kidneyfund.org/all-about-kidneys/tests/serum-creatinine-test.

Healthline. "Sodium Blood Test." Healthline, 16 Nov. 2023, https://www.healthline.com/health/sodium-blood#:~:text=A%20sodium%20blood%20test%20is,for%20nerve%20and%20muscle%20function.

Chicco, Davide, and 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, vol. 20, no. 16, 2020, https://bmcmedinformdecismak.biomedcentral.com/articles/10.1186/s12911-020-1023-5.