Heart Disease Analysis

Final Project

Jaimee Hope Cox, Robert Anthony Young, Runcong Wu, Tongchen Yi

Group 3

Professor Camila de Souza

SS 3860B: Generalized Linear Models / SS 9155B: Statistical Modelling II

Table of Contents

1. Introduction

Cardiovascular diseases (CVDs) are the leading cause of global mortality and significantly reduce the quality of life; CVDs cause nearly 18 million deaths worldwide each year, and the number keeps increasing (Mensah et al., 2019). CVDs mainly exhibit in the form of heart failure. Heart failure is a condition where the heart cannot pump enough blood to the body. Understanding the risk factors, such as hypertension, diabetes, hyperlipidemia, can be beneficial to prevent heart failure (Chicco & Jurman, 2020). Many hospitals fail to manage and identify high-risk patients; therefore, using a data-driven approach is much preferred to allocate resources to these high-risk patients. Our objective is to develop a highly accurate and interpretable statistical model that can classify high-risk patients in an early stage and identify risk factors associated with heart failure patients.

2. The Data

The data set used in this work is titled *Heart Failure Prediction: 12 clinical features for predicting death events*. It is retrieved from Kaggle.com at the following link: Kaggle Data Set. The data set was originally collected by Chicco and Jurman (2020) from 299 patients at the Faisalabad Institute of Cardiology and the Allied Hospital in Faisalabad in 2015. These patients were classified as stages three and four of heart failure as they all had left ventricular systolic dysfunction and had previous heart failures.

The dataset has 299 observations (rows) across 13 variables (columns), including 12 predictors that can be used to predict heart failure as they give a summary of the patients' clinical, body and lifestyle information, plus one binary response variable - *Death Event*. There are 6 categorical variables: *Death Event*, *Anemia*, *Diabetes*, *High Blood Pressure*, *Sex*, and *Smoking*; as well as 7 continuous: *Time*, *Age*, *Creatinine Phosphokinase*, *Ejection Fraction*, *Platelets*, *Serum Creatinine* and *Serum Sodium*. A description of the variables is as follows:

Feature	Explanation	Measurement
Age	Age of the patient	Years
Anaemia	Decrease of red blood cells or hemoglobin	Boolean (0: No, 1: Yes)
High blood pressure	If the patient has hypertension	Boolean (0: No, 1: Yes)
Creatinine phosphokinase	Level of creatinine phosphokinase in the blood	mcg/L
Diabetes	If the patient has diabetes	Boolean (0: No, 1: Yes)
Ejection fraction	Percentage of blood leaving the heart at each contraction	Percentage
Sex	Female or male	Binary (0: Female, 1: Male)
Platelets	Platelets in the blood	kiloplatelets/mL
Serum creatinine	Level of creatinine in the blood	mg/dL
Serum sodium	Level of sodium in the blood	mEg/L
Smoking	If the patient smokes	Boolean (0: No, 1: Yes)
Time	Follow up period	Days
Death event	If the patient died during the follow up period	Boolean (0: Alive, 1: Death)

Table 1. List of variable descriptions in the dataset.

All data preprocessing and analyses were performed under R version 4.0.3. Various preprocessing steps are carried out to prepare the data to be analyzed. First, we check data completeness. There is no missing value in the data set. Second, we decide to drop the *Time* predictor because it is highly biased by the physicians' best guess on who needs to be checked up on sooner. Therefore, we have 11 predictors in the data set. Next, we define the baseline for all 6 categorical variables and change the data type to R factor type. The target variable is binary, with 0 indicating the patient is alive and 1 indicating the patient is dead.

3. Methods

3.1 Exploratory Data Analysis

The exploratory data analysis includes several univariate, bivariate, and multivariate visualizations. First, we consider the categorical predictors in the data set. For each relevant variable, we create a bar graph to see the count of the observations across the various levels of the categorical predictor. We also use histograms to assess the distribution of the data for all continuous variables. A log-transform is applied to heavily skewed variables to evaluate if any beneficial effect on the data distribution is achieved. Second, boxplots are created to view the distribution of continuous variables with respect to the target variable; this would highlight any large differences in the distribution of data based on the response. Then, side-by-side bar graphs are created to view the distribution of categorical variables with respect to the target variable; this would capture any large differences for each category based on the response. A scatter plot is created to visualize the relationship between two continuous variables. To better understand the relationship between predictors and the response variable, we use the binary response variable to colour code the scatter plot.

3.2 Main Data Analysis

We split 70% of our data into the training dataset to fit models and the rest 30% of the data into the testing dataset to evaluate model performance. Our training dataset has 209 observations; our testing dataset has 90 observations.

For our objective, we chose 7 models: Logistic Regression, LDA, QDA, Classification Tree, Bagging, Random Forest, and Boosting. We chose Logistic Regression because it deals with binary output variables. LDA and QDA were good candidates because most of our continuous predictors followed approximately normal distributions. Tree based methods were chosen because they are more flexible. Bagging reduces the variance in decision trees; however, trees are highly correlated with each other. Random Forest reduced the variance in decision trees and correlations between trees at the same time. Boosting was chosen because it reduces bias gradually.

Logistic Regression (LR):

- Firstly, we fit a model using all 11 features.
- Secondly, we perform feature selection by running the Backwards Selection algorithm with Akaike Information Criterion (AIC) as our criterion.
- Thirdly, we consider adding a quadratic term of the most significant feature, *Ejection Fraction*, in the AIC selected feature space.
- Then, we decide to drop the least significant term, *High Blood Pressure*, from the quadratic model.
- The best LR model is selected using the Likelihood Ratio Test (deviance-based test).
- Hosmer-Lemeshow Test is used to determine the model's Goodness of Fit. The model assumptions are evaluated by the deviance residual plot and half-normal plot.
- Interpretation of coefficients in terms of odds.

Linear Discriminant Analysis (LDA): LDA is used to classify observations as Alive or Dead using linear combinations of the predictors. We used the features found by the best LR model. The LDA model requires the predictors to follow approximately normal distributions, this was taken care of in preprocessing as the non-normal continuous variable, *Serum Creatinine* was transformed into an approximately normal variable using the log transformation.

Quadratic Discriminant Analysis (QDA): QDA creates non-linear decision boundaries to classify observations as Alive or Dead. We used the same features in the LDA model. *Serum Creatinine* was log-transformed to better fit the model assumption of using approximately normal predictors.

Tree-based Method: The random seed was set to 108 to ensure reproducible results for all tree models. All tree-based models are trained using the original dataset without any transformation since they are not sensitive to the range of variables.

Classification Tree: A classification tree is composed of branches and nodes, which form a structural mapping of binary decisions. In practice, the decision process starts at the root node and follows the branches until a leaf node is reached, which leads to decisions about the class of the target variable. We applied this method by building both unpruned tree and pruned tree with the best tree size determined using cross-validation.

Bagging: Bagging is a bootstrap-based approach that aims to reduce the variance in decision trees. It corrects the problem of overfitting by re-sampling the training sets with replacement. We decided to build 200 trees. Each split considers all 11 features.

Random Forest: Random forest is developed from the bagging tree and it further reduces the correlation of each tree by splitting based on a random sample of features. During this process, we can examine the importance of each feature in prediction power and remove the insignificant attributes to avoid overfitting. The model trained 200 trees with 4 variables randomly sampled as candidates at each split.

Boosting: Boosting can gradually reduce the error by fitting many weak trees sequentially. We decided to use a binary classification task because the response variable follows the Bernoulli distribution. The model is trained on 200 trees with a learning rate (shrinkage) of 0.1. We convert the response to a character type because the “gbm” R package needs a character type response instead of a factor type response.

Evaluation Metrics

For each of the models we used above, we performed the Receiver Operating Characteristic (ROC) analysis using the test dataset to calculate the area under the curve (AUC), sensitivity, specificity, accuracy, and decision threshold. The best model will be selected by the highest AUC. Then we will analyze the best model by constructing a confusion matrix using the best decision threshold as well as interpret the result.

4. Results

4.1 Exploratory Data Analysis

We begin by assessing the frequency of the different levels for each categorical predictor used in this work as well as the frequency of the target variables. The majority class (patient is alive) of the response variable, *Death Event*, has twice the observations than the minority class (patient is dead). This indicates our dataset has class imbalance issues with respect to the response variable.

Next, we assessed the distribution of each continuous predictor by using the histogram. The distribution of the *Creatinine Phosphokinase* and *Serum Creatinine* is heavily right skewed with large peaks near 0 and 1 respectively. It can be seen in Figure 1 that the natural logarithm transformation has a significant impact on both of these variables, making the resultant distribution approximately normal. Consequently, we will use the natural logarithm of *Serum Creatinine* as a predictor in the LDA and QDA models.

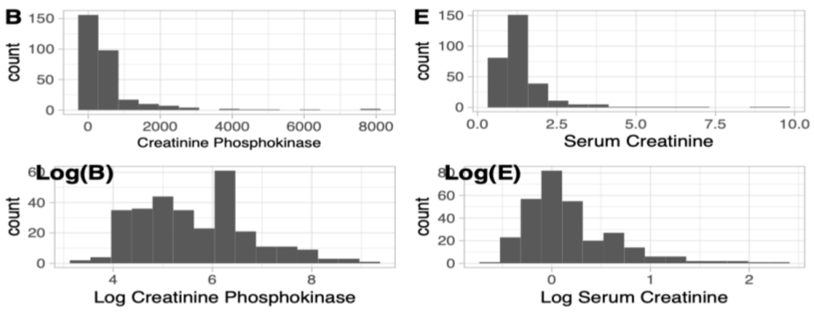


Figure 1. Effect of log transformation on *Creatinine Phosphokinase* and *Serum Creatinine*.

Then, we assessed the relationship between each continuous predictor and the response variable using boxplots shown in Figure 2. There are noticeable differences in the distribution of *Age*, *log(Serum Creatinine)*, *Ejection Fraction*, *Serum Sodium* based on the patient status (alive or dead), which indicate these 4 predictors may have a significant impact on the probability of patient death. Elder patients with higher *log(Serum Creatinine)* have a greater chance of heart failure, while patients with higher *Ejection Fraction* and *Serum Sodium* level have a smaller chance of heart failure.

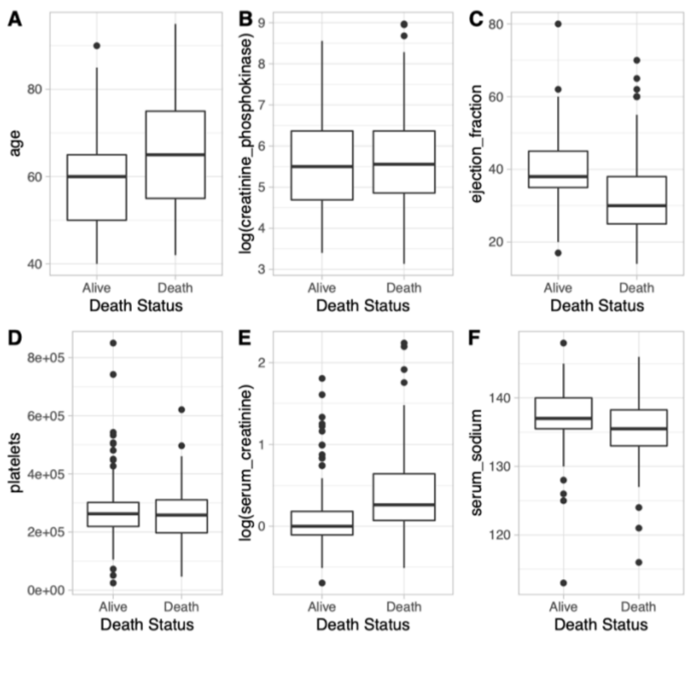


Figure 2. Side-by-side boxplots describe the relationship between each continuous predictor and the response variable.

Side-by-side bar graphs are created to visualize the distribution of *Sex*, *Diabetes* with respect to the response variable. As we can see from Figure 3, both male and female patients have similar proportions of heart failure (A); patients with and without diabetes have similar proportions of heart failure (B). It is interesting to note that female patients with diabetes have a higher chance of heart failure, whereas diabetes has minimal effect on male patients (C).

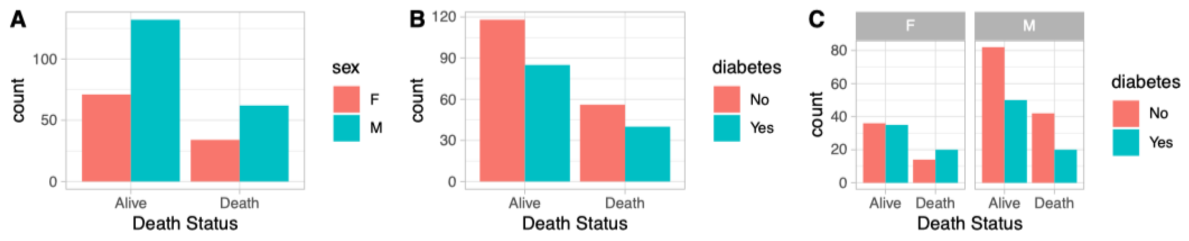


Figure 3. Side-by-side bar plots describe the relationship between *Sex*, *Diabetes* and the response variable.

Finally, we use a scatter plot to visualize the relationship between *Age* and *Ejection Fraction*. The binary response is used to colour code the scatter plot shown in Figure 4. Although there is no clear trend, it seems that a lower *Ejection Fraction* is associated with a greater concentration of deaths for all age levels.

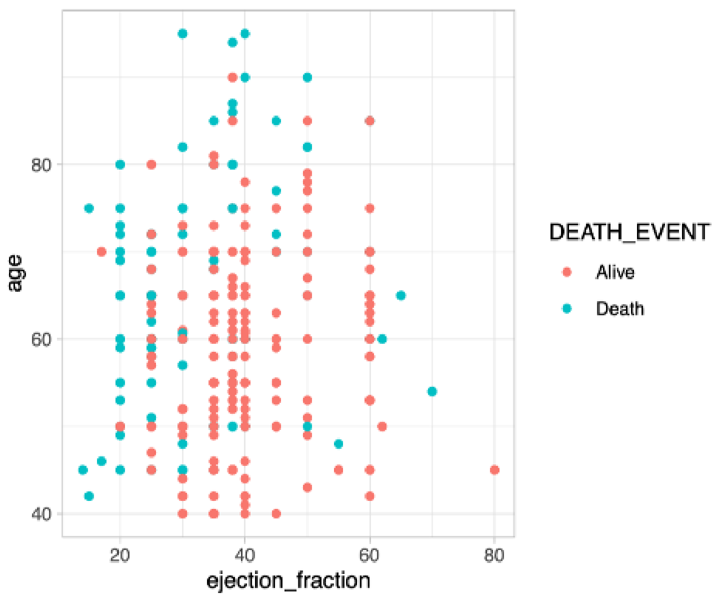


Figure 4. Scatter plot describes the relationship between *Age*, *Ejection Fraction* and the response variable.

4.2 Main Data Analysis

Logistic Regression Results: Our final model is,

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_{int}x_{int} + \beta_{age}x_{age} + \beta_{ejec,1}x_{ejec,1} + \beta_{ejec,2}x_{ejec,2}^2 + \beta_{serum}x_{serum}$$

Here are the steps on how we derive this model.

After fitting the Logistic Regression model to all 11 features, we run the Hosmer-Lemeshow (HL) Test and find evidence that our model fits the data well. We consider 11 features to be too many for any parametric model and so we reduce the dimensions by the AIC stepwise algorithm and obtain a model that uses features; Age, Ejection Fraction, High Blood Pressure (Yes), and Serum Creatinine.

We then conduct the HL Goodness of Fit test for the model that the AIC algorithm produced, and we are given evidence that this model fits that data well, with a p-value of 0.4644. Next, we run a series of Likelihood Ratio Tests to arrive at our best performing Logistic Regression. Our first LRT justifies adding a quadratic term to our most significant predictor, ejection fraction. Then given this new larger model, we find that high blood pressure is no longer needed, and we perform an LRT test that gives proof that a smaller model is adequate.

The results of the first test are adding Ejection Fraction^2 to the AIC model, our LRT test comes back with a 0.01795 p-value and thus we admit the quadratic term into our model. Next, we find that our quadratic model no longer finds High Blood Pressure to be significant, thus we perform an LRT test and receive a p-value of 0.06648. This shows that the model is sufficient without its presence.

Thus, we drop high blood pressure and have arrived at our final model which includes age, ejection fraction, ejection fraction^2, and serum creatinine. The results are given below.

Features	Coefficient Value	P-Value
Intercept	-4.8579	$1.59 * 10^{-6}$
Age	0.0472	0.0021
Ejection Fraction	-8.2447	0.0015
Ejection Fraction^2	6.8103	0.0113
Serum Creatinine	0.7901	0.0016

Table 2. Model summary table for the best Logistic Regression Model.

The best Logistic model seems to be a good fit for our data by passing the Hosmer-Lemeshow Goodness of fit test with a p-value of 0.5023. But it is important that we consider the residual diagnostics to either find any glaring abnormalities or potential outliers. We consider the two methods of diagnostics on our best model; the estimated linear predictor binning and half-normal plot, both are given below, left and right respectively.

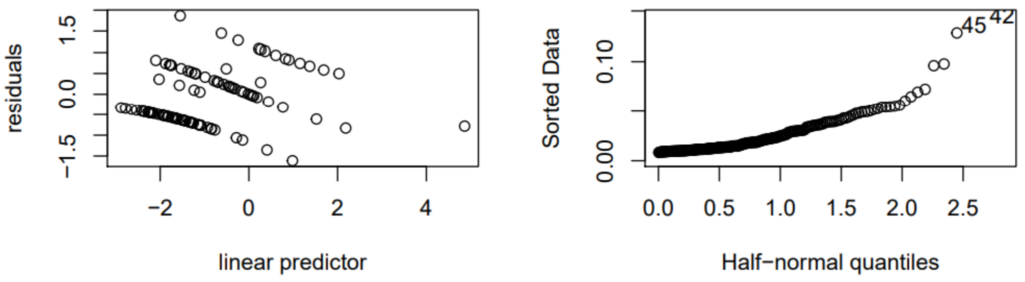


Figure 5. (a) Left: Output of binned residuals against estimated linear predictors. (b) Right: Output from half-normal plot. Both are from the best logistic model found after the LRT tests.

Unfortunately, there is one glaring abnormality. The residuals vs linear predictor plot is supposed to display two lines, one for each classification of the binary response. The third line means that the Logistic model is having trouble truly classifying a good portion of the data. While this is not the end of our model, we should now take its results with this in the back of our minds. On the other hand, the half-normal plot shows a nearly linear relationship except for a few potential outliers. We could opt to remove them from our training dataset but after running this scenario, we find that the model performs worse and does not add any value.

The AUC is adequate at 0.8332391 for a classification method and we see that the Sensitivity is high. Our goal is to ensure a high AUC while also obtaining a high Sensitivity. The Sensitivity is of more importance because our True Positive Rate reflects how well the model predicts the Deaths of patients suffering from Heart Failure. There is not much need for a high prediction accuracy of patients who will not die. Given this, our Logistic model is not a good fit for our goals.

Linear Discriminant Analysis Results: *Age*, *Ejection Fraction*, *Ejection Fraction^2*, and *log(Serum Creatinine)* were used to fit the LDA model. The results come back with an AUC of 0.812886 which is competitive with our LR model and suggests that the data might have a linear decision boundary.

Quadratic Discriminant Analysis Results: *Age*, *Ejection Fraction*, *Ejection Fraction^2*, and *log(Serum Creatinine)* were used to fit the QDA model. The test AUC is 0.7908423.

Classification tree: Starting from building an unpruned tree with 21 terminal nodes, we then find the optimum tree size using cross-validation. In Figure 6(a), it appears that a tree of size 9 has only 24% misclassifications, which is the lowest among all the considered trees. The pruned tree with 9 terminal nodes is shown in Figure 6(b). Only 6 features *Serum Creatinine*, *Ejection Fraction*, *Age*, *Sex*, *Creatinine Phosphokinase* and *platelets* are used in the model. The most important feature is *Serum Creatinine*, which appears in the root node. The classification tree is generated with a test AUC of 0.8115.

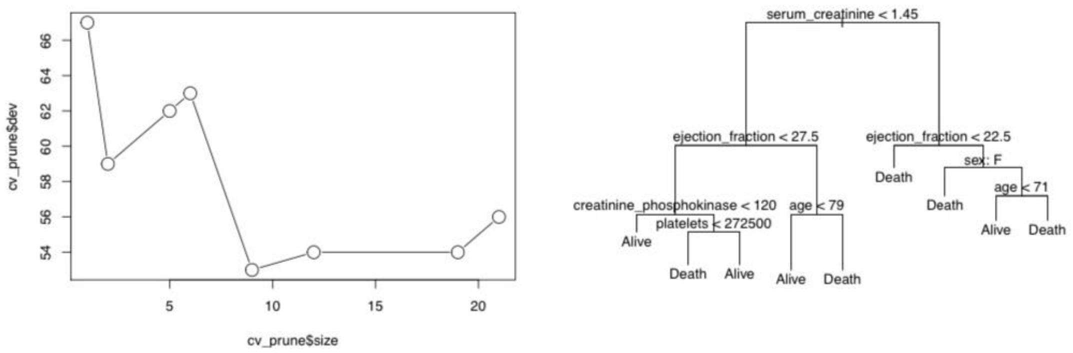


Figure 6. (a) Left: Cross-validation result to find the optimal tree size using the smallest misclassification rate. (b) Right: Prune tree using the best tree size.

Bagging: The cross-validation error determined using the training set is 28.71% for the bagging tree. The classification error rate for the Death class is 53.7%, while only 16.9% of the Alive class are misclassified. The test AUC for the bagging tree model is 0.8174.

Random Forest: For Random Forest, the training model has a cross-validation error of 28.23%. The classification error rate for the Death class is 53.7% which is the same as the bagging tree, while the misclassification rate for the Alive class slightly decreases to 16.2%. The test AUC for the Random Forest model is 0.8383.

Boosting: The boosting model does not perform as well as the other tree models, the AUC score is only 0.797. For this model, *Serum Creatinine* and *Ejection Fraction* are determined as the most important variables, which is very similar to the result of the classification tree.

	Threshold	**Sensitivity**	**Specificity**	**Accuracy**	**AUC**
Logistic Regression	0.2563	0.8621	0.7705	0.8000	0.8332
LDA	0.2385	0.7931	0.7869	0.7889	0.8129
QDA	0.3359	0.7241	0.8525	0.8111	0.7908
Tree	0.1371	0.7931	0.8033	0.8000	0.8115
Bagging	0.1725	0.9310	0.6393	0.7333	0.8174
Random Forest	0.2375	0.8966	0.6721	0.7444	0.8383
Boosting	0.3114	0.7586	0.8361	0.8111	0.7971

Table 3. ROC Analysis using test dataset on 7 classification models.

If we compare the results obtained from all models, the logistic regression and random forest outperform other models with an AUC score greater than 0.83. Since our objective is to study the impact of those features on heart disease mortality, we also need to pay attention to sensitivity which determines the correct classification rate in the death group. It is obvious that the bagging tree has an outstanding sensitivity of 0.931, which means the model could successfully identify 93% of death. The logistic regression and Random Forest also have excellent performance with a sensitivity of 0.8621 and 0.8966 respectively. However, the specificity of bagging and random forest models is relatively lower, while the logistic regression model can still ensure good specificity while obtaining high sensitivity. Thus, the logistic regression model is selected as our final model, and we can interpret the coefficients using Table 4.

Features	$\boldsymbol{e^{\beta} - 1}$
Age	0.0483
Serum Creatinine	1.2037
Ejection Fraction & Ejection Fraction^2 (by 1%)	-0.7617

Table 4. Interpreting the coefficients in terms of odds for the best model (logistic regression) we selected.

An increase in odds means an increase in the probability of success, where we have defined success to be the event of death. We see that an increase in *Age* by one unit increases the odds of success (death) by 4.83%; an increase in *Serum Creatinine* by one unit increases the odds of success (death) by 120.37%. For *Ejection Fraction*, we see that an increase in one unit will decrease the odds of death by 76.17%.

5. Conclusion

We develop a statistical model to predict patients' mortality risk caused by heart failure by giving patients' clinical, body and lifestyle information. The model is expected to be applicable to classify patients into different severity levels and allocate resources to high-risk patients. Building an early warning system that has the ability to identify high-risk patients in advance can be beneficial to patient management. Besides, using a data-driven approach is much preferred and faster than assigning patients manually in the hospitals.

There are 7 statistical models used in our analysis, including 4 traditional machine learning models (LR, LDA, QDA, Classification Tree) and 3 black-boxed methods (Bagging, RF, Boosting). Since our goal is to find a model that balances the model inference and prediction ability, it should be highly interpretable and highly accurate. According to the results of our 7 models, LR is selected as our final model. It is worth mentioning that adding the quadratic term of *Ejection Fraction* in the LR model sacrifices the interpretation ability for better prediction results.

The LR model can easily achieve above 86% recall/sensitivity, but the precision (64%) is not as good as the recall. There's always a trade-off between precision and recall in real cases. Precision measures the proportion of high-risk patients among all the patients that are classified as high-risk patients, while recall measures the proportion of actual high-risk patients that are correctly identified. Low precision indicates that around 35% of the predicted high-risk patients by our model are actual low-risk patients. Increasing the decision threshold can significantly improve the precision but lower the recall which means the model may fail to capture many potential high-risk patients. Another way to improve precision is to add more positive data to the training dataset. As only about 30% of our data is tagged as positive, adding more positive data would help with balancing the dataset and improving the evaluation metrics.

We would like to discuss the limitations of our study from two aspects. One problem is our dataset is relatively small. This can lead to inaccurate predictions. Including more data will improve the performance of our model. The second limitation is our data is heavily imbalanced. We may use under-sampling and over-sampling methods in future studies to achieve better model performance.

6. Student Contributions

Student	Task
Jaimee Cox	Data Set Description, Exploratory Data Analysis, LDA
Rob Young	Introduction, Logistic Regression, and QDA
Runcong Wu	R Coding, Conclusion
Tongchen Yi	Classification Tree, Bagging, Random Forest, Boosting, Models Comparison

Table 5. Student Contributions

7. References

Chicco, D., & Jurman, G. (2020). Machine learning can predict survival of patients with heart failure from serum creatinine and ejection fraction alone. *BMC Medical Informatics and Decision Making*, *20*(1), 1. https://doi.org/10.1186/s12911-020-1023-5

Mensah, G. A., Roth, G. A., & Fuster, V. (2019). The Global Burden of Cardiovascular Diseases and Risk Factors. *Journal of the American College of Cardiology*, *74*(20), 2529–2532. https://doi.org/10.1016/j.jacc.2019.10.009

8. Appendix for R Code

Table of Contents

Binary Response: Death_Event (0 – Alive, 1 – Death)

1. Preprocessing

```r
# load the csv data
heart_data <- read.csv("heart_failure_clinical_records_dataset.csv")

# Check if there is any missing value (number of missing values = 0)
sum(is.na(heart_data))
```

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

```r
# Check nrows and ncols
dim(heart_data)
```

```
## [1] 299  13
```

1.1 Drop time

```r
heart_data <- subset(heart_data, select = -time)
dim(heart_data)
```

```
## [1] 299  12
```

1.2 Define Baseline for Categorical Variables -> Factor Type

```r
heart_data$anaemia <- factor(heart_data$anaemia, levels=c("0","1"), labels=c("No","Yes"))
heart_data$diabetes <- factor(heart_data$diabetes, levels=c("0","1"), labels=c("No","Yes"))
heart_data$high_blood_pressure <- factor(heart_data$high_blood_pressure,
                                         levels=c("0","1"), labels=c("No","Yes"))
```

```r
heart_data$sex <- factor(heart_data$sex, levels=c("0","1"), labels=c("F","M"))
heart_data$smoking <- factor(heart_data$smoking, levels=c("0","1"), labels=c("No","Yes"))
heart_data$DEATH_EVENT <- factor(heart_data$DEATH_EVENT, levels=c("0","1"),
                                 labels=c("Alive","Death"))

# Check the data type to see the changes
str(heart_data)
```

```
## 'data.frame':    299 obs. of  12 variables:
##  $ age                     : num  75 55 65 50 65 90 75 60 65 80 ...
##  $ anaemia                 : Factor w/ 2 levels "No","Yes": 1 1 1 2 2 2 2 2 1 2 ...
##  $ creatinine_phosphokinase: int  582 7861 146 111 160 47 246 315 157 123 ...
##  $ diabetes                : Factor w/ 2 levels "No","Yes": 1 1 1 1 2 1 1 2 1 1 ...
##  $ ejection_fraction       : int  20 38 20 20 20 40 15 60 65 35 ...
##  $ high_blood_pressure     : Factor w/ 2 levels "No","Yes": 2 1 1 1 1 2 1 1 1 2 ...
##  $ platelets               : num  265000 263358 162000 210000 327000 ...
##  $ serum_creatinine        : num  1.9 1.1 1.3 1.9 2.7 2.1 1.2 1.1 1.5 9.4 ...
##  $ serum_sodium            : int  130 136 129 137 116 132 137 131 138 133 ...
##  $ sex                     : Factor w/ 2 levels "F","M": 2 2 2 2 1 2 2 2 1 2 ...
##  $ smoking                 : Factor w/ 2 levels "No","Yes": 1 1 2 1 1 2 1 2 1 2 ...
##  $ DEATH_EVENT             : Factor w/ 2 levels "Alive","Death": 2 2 2 2 2 2 2 2 2 2 ...
```

2. Exploratory Data Analysis

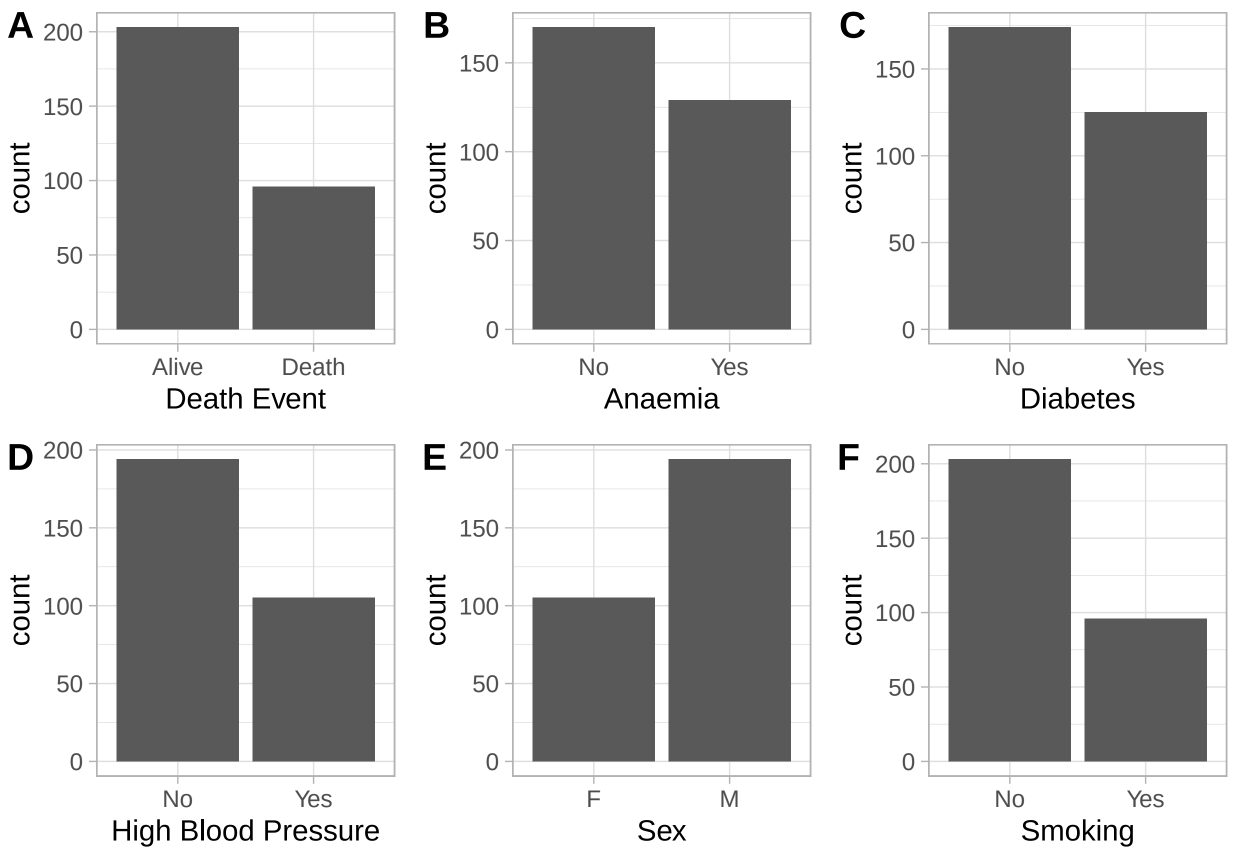
```r
# Generate summary statistics
summary(heart_data)
```

```
##       age        anaemia    creatinine_phosphokinase diabetes   ejection_fraction
##  Min.   :40.00   No :170   Min.   :  23.0            No :174   Min.   :14.00    
##  1st Qu.:51.00   Yes:129   1st Qu.: 116.5            Yes:125   1st Qu.:30.00    
##  Median :60.00             Median : 250.0                      Median :38.00    
##  Mean   :60.83             Mean   : 581.8                      Mean   :38.08    
##  3rd Qu.:70.00             3rd Qu.: 582.0                      3rd Qu.:45.00    
##  Max.   :95.00             Max.   :7861.0                      Max.   :80.00    
##  high_blood_pressure   platelets      serum_creatinine  serum_sodium   sex    
##  No :194             Min.   : 25100   Min.   :0.500    Min.   :113.0   F:105  
##  Yes:105             1st Qu.:212500   1st Qu.:0.900    1st Qu.:134.0   M:194  
##                      Median :262000   Median :1.100    Median :137.0          
##                      Mean   :263358   Mean   :1.394    Mean   :136.6          
##                      3rd Qu.:303500   3rd Qu.:1.400    3rd Qu.:140.0          
##                      Max.   :850000   Max.   :9.400    Max.   :148.0          
##  smoking   DEATH_EVENT
##  No :203   Alive:203  
##  Yes: 96   Death: 96  
##                       
##                       
##                       
## 
```

2.1 Categorical Variable Distributions

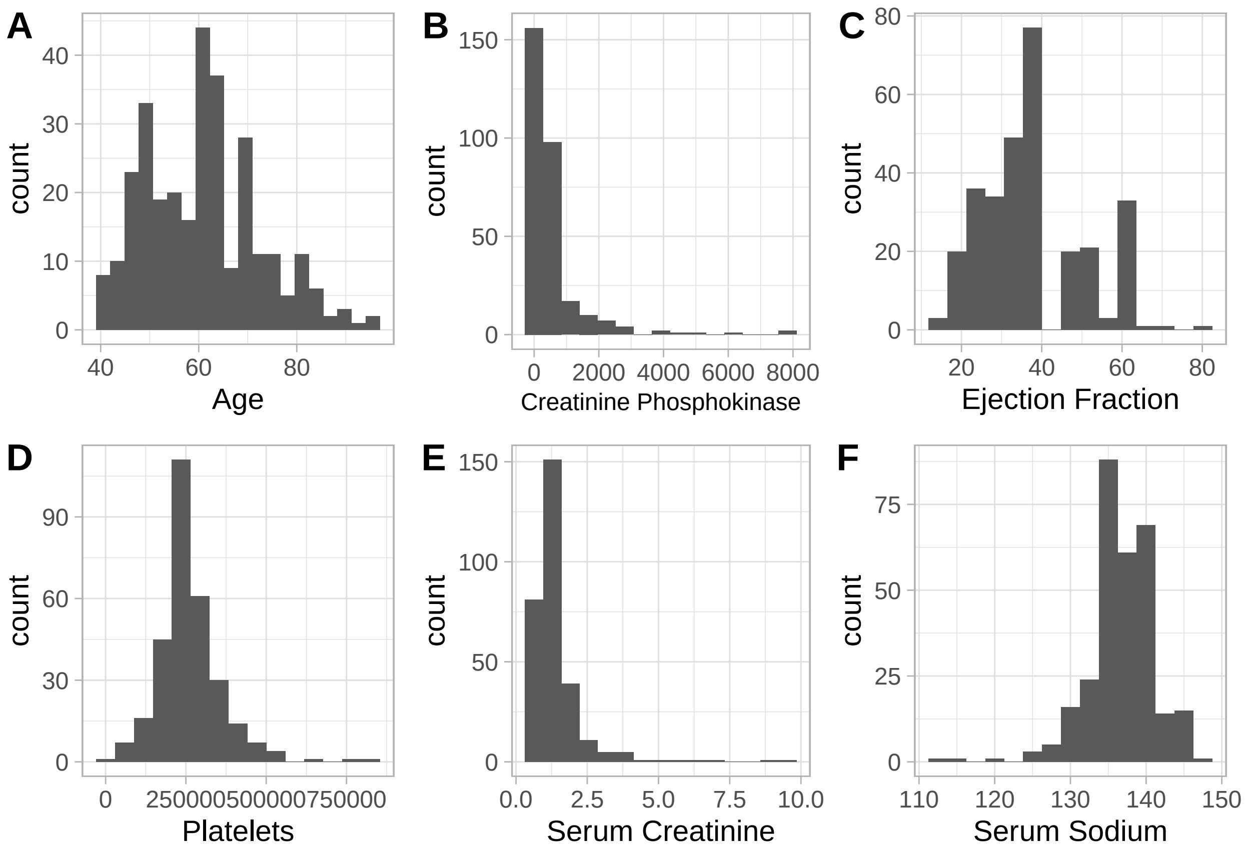
```r
suppressMessages(library(tidyverse))

# response
response <- heart_data %>%
  ggplot(aes(x = DEATH_EVENT)) + geom_bar() + labs(x = "Death Event") + theme_light()

# anaemia
anaemia <- heart_data %>%
  ggplot(aes(x = anaemia)) + geom_bar() + labs(x = "Anaemia") + theme_light()

# diabetes
diabetes <- heart_data %>%
  ggplot(aes(x = diabetes)) + geom_bar() + labs(x = "Diabetes") + theme_light()

# high_blood_pressure
high_blood_pressure <- heart_data %>%
  ggplot(aes(x = high_blood_pressure)) + geom_bar() + labs(x = "High Blood Pressure") +
  theme_light()

# sex
sex <- heart_data %>%
  ggplot(aes(x = sex)) + geom_bar() + labs(x = "Sex") + theme_light()
```

```r
# smoking
smoking <- heart_data %>%
  ggplot(aes(x = smoking)) + geom_bar() + labs(x = "Smoking") + theme_light()

# library allows clear labels and save plot as pdf
suppressMessages(library(cowplot))
(p_cat <- plot_grid(response, anaemia, diabetes, high_blood_pressure, sex, smoking,
                    labels = "AUTO"))
```



2.2 Continuous Variable Distributions

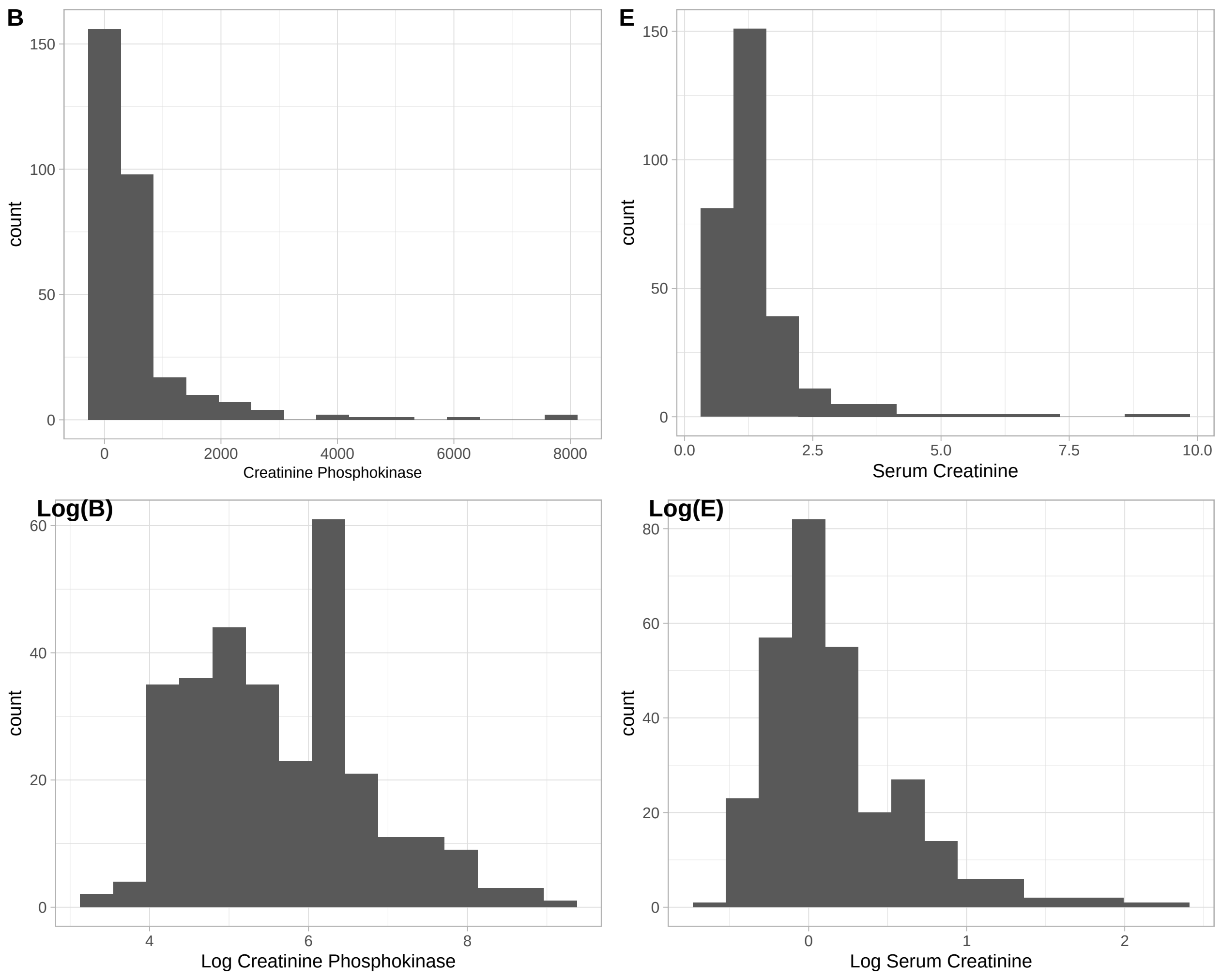
```r
# age
age <- heart_data %>%
  ggplot(aes(x = age)) + geom_histogram(bins=20) + labs(x = "Age") + theme_light()

# creatinine_phosphokinase
creatinine_phosphokinase <- heart_data %>%
  ggplot(aes(x = creatinine_phosphokinase)) + geom_histogram(bins=15) +
  labs(x = "Creatinine Phosphokinase") + theme_light() +
  theme(axis.title.x = element_text(size = 9))

# ejection_fraction
ejection_fraction <- heart_data %>%
  ggplot(aes(x = ejection_fraction)) + geom_histogram(bins=15) +
  labs(x = "Ejection Fraction") + theme_light()

# platelets
platelets <- heart_data %>%
  ggplot(aes(x = platelets)) + geom_histogram(bins=15) + labs(x = "Platelets") +
  theme_light()

# serum_creatinine
serum_creatinine <- heart_data %>%
  ggplot(aes(x = serum_creatinine)) + geom_histogram(bins=15) +
  labs(x = "Serum Creatinine") + theme_light()

# serum_sodium
serum_sodium <- heart_data %>%
  ggplot(aes(x = serum_sodium)) + geom_histogram(bins=15) + labs(x = "Serum Sodium") +
  theme_light()

# plot
(p_con <- plot_grid(age, creatinine_phosphokinase, ejection_fraction, platelets,
                    serum_creatinine, serum_sodium, labels = "AUTO"))
```



- B, E are heavily right-skewed, may consider using log transformation

Log Transform Continuous Variable

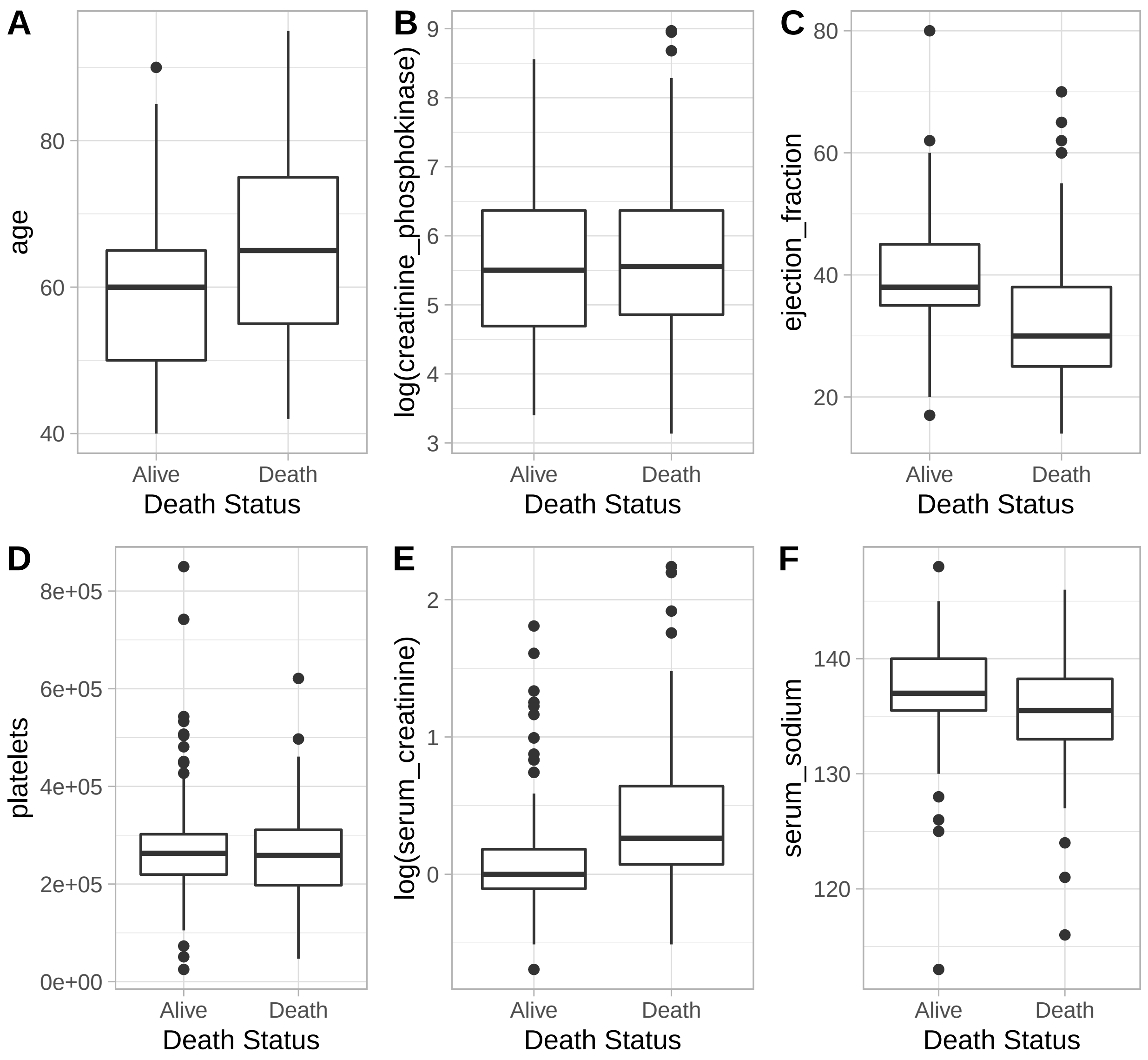
```r
# visualize the result
# creatinine_phosphokinase
log_cp <- heart_data %>%
  ggplot(aes(x = log(creatinine_phosphokinase))) + geom_histogram(bins=15) +
  labs(x = "Log Creatinine Phosphokinase") + theme_light()

# serum_creatinine
log_sc <- heart_data %>%
  ggplot(aes(x = log(serum_creatinine))) + geom_histogram(bins=15) +
  labs(x = "Log Serum Creatinine") + theme_light()

# plot
(p_log <- plot_grid(creatinine_phosphokinase, serum_creatinine,
                    log_cp, log_sc, ncol=2,
                    labels = c("B", "E", "Log(B)", "Log(E)")))
```



- Log-transform decreases skewness in the distributions.

2.3 2D plot

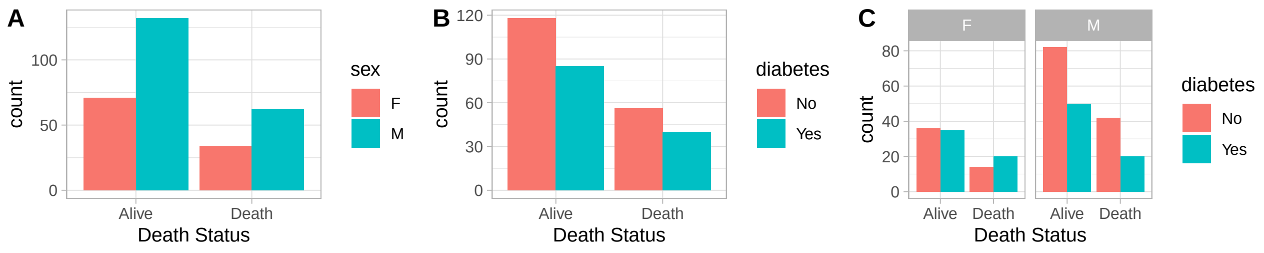
```r
# Age and Response
Box1 <- heart_data %>%
  ggplot(aes(x=DEATH_EVENT, y=age)) + geom_boxplot() + labs(x = "Death Status") +
  theme_light()

# log(creatinine_phosphokinase) and Response
Box2 <- heart_data %>%
  ggplot(aes(x=DEATH_EVENT, y=log(creatinine_phosphokinase))) + geom_boxplot() +
  labs(x = "Death Status") + theme_light()

# Ejection fraction and Response
Box3 <- heart_data %>%
  ggplot(aes(x=DEATH_EVENT, y=ejection_fraction)) + geom_boxplot() +
  labs(x = "Death Status") + theme_light()

# platelets and Response
Box4 <- heart_data %>%
  ggplot(aes(x=DEATH_EVENT, y=platelets)) + geom_boxplot() +
  labs(x = "Death Status") + theme_light()

# log(serum_creatinine) and Response
Box5 <- heart_data %>%
  ggplot(aes(x=DEATH_EVENT, y=log(serum_creatinine))) + geom_boxplot() +
  labs(x = "Death Status") + theme_light()

# serum_sodium and Response
Box6 <- heart_data %>%
  ggplot(aes(x=DEATH_EVENT, y=serum_sodium)) + geom_boxplot() +
  labs(x = "Death Status") + theme_light()

plot_grid(Box1, Box2, Box3, Box4, Box5, Box6, ncol=3, labels = "AUTO")
```



- The median age of deaths is roughly 5 years older than survivors (65 vs 60 yrs)
- It seems lower ejection fraction seems to be associated with greater chance of heart failure according to our sample data.

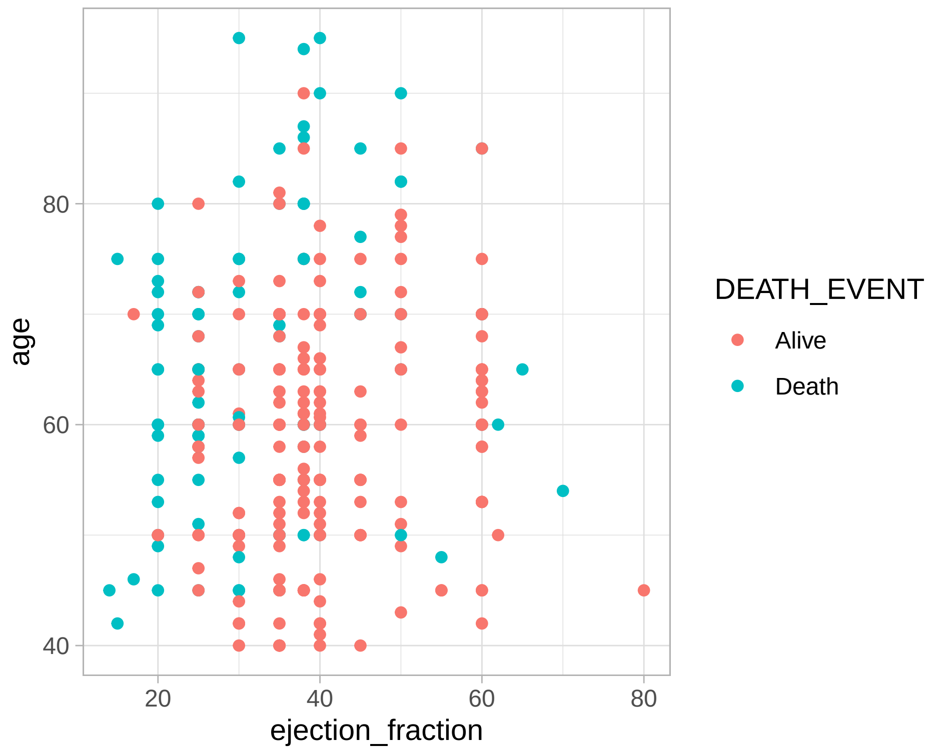
```
# Sex and Response
Bar1 <- heart_data %>%
  ggplot(aes(x=DEATH_EVENT, fill=sex)) + geom_bar(position="dodge") +
  labs(x = "Death Status") + theme_light()

# Diabetes and Response
Bar2 <- heart_data %>%
  ggplot(aes(x=DEATH_EVENT, fill=diabetes)) + geom_bar(position="dodge") +
  labs(x = "Death Status") + theme_light()

# Diabetes, Sex and Response
Bar3 <- heart_data %>%
  ggplot(aes(x=DEATH_EVENT, fill=diabetes)) + geom_bar(position="dodge") +
```

```r
  labs(x = "Death Status") + theme_light() + facet_wrap(~sex)

plot_grid(Bar1, Bar2, Bar3, ncol = 3, labels = "AUTO")
```



- Roughly the same proportion of males and females die of heart failure.
- Roughly the same proportion of heart failure for people with and without diabetes.
- For males, the death ratio is similar bewteen people with and without diabetes.
- For females, the death ratio is higher in people with diabetes.

```r
# Ejection fraction and Age for different Response
heart_data %>%
  ggplot(aes(x=ejection_fraction, y=age, color=DEATH_EVENT)) + geom_point() +
  theme_light()
```



- No clear trends here, though it seems there is a greater concentration of deaths at lower ejection fraction levels at all ages.

3. Data Modelling

3.1 Data Spliting

```
# Let's first split the data into training and test data (70/30)
set.seed(414)
n = nrow(heart_data)
idx_tr <- sample(n, round(0.7*n), replace=FALSE)

# Define training and test data
train = heart_data[idx_tr,]
test = heart_data[-idx_tr,]

dim(train)
```

```
## [1] 209  12
```

```
dim(test)
```

```
## [1] 90 12
```

3.2 Model1: Logistic Regression

Using all the 11 features

```
model1_LR <- glm(DEATH_EVENT ~ ., family=binomial, data=train)

# GOF (Hosmer-Lemeshow) -> p-value > 0.05, no lack of fit
suppressMessages(library(ResourceSelection))
hoslem.test(model1_LR$y,fitted(model1_LR),g=10)
```

```
##
##  Hosmer and Lemeshow goodness of fit (GOF) test
##
## data:  model1_LR$y, fitted(model1_LR)
## X-squared = 15.385, df = 8, p-value = 0.05207
```

Feature selection using AIC

```
# Feature selection using AIC: left with 4 features
model1_LR_small <- step(model1_LR, trace=0)
library(faraway)
```

```
##
## Attaching package: 'faraway'

## The following object is masked _by_ '.GlobalEnv':
##
##     diabetes
```

```
sumary(model1_LR_small)
```

```
##                         Estimate Std. Error z value  Pr(>|z|)
## (Intercept)            -2.458993   1.079707 -2.2775 0.0227586
## age                     0.042042   0.015002  2.8023 0.0050735
## ejection_fraction      -0.065281   0.017665 -3.6956 0.0002194
## high_blood_pressureYes  0.704623   0.349857  2.0140 0.0440059
## serum_creatinine        0.842412   0.246316  3.4200 0.0006261
##
## n = 209 p = 5
## Deviance = 212.38153 Null Deviance = 262.21202 (Difference = 49.83049)
```

```
# GOF: p-value > 0.05, no lack of fit
hoslem.test(model1_LR_small$y,fitted(model1_LR_small),g=10)
```

```
##
##  Hosmer and Lemeshow goodness of fit (GOF) test
##
## data:  model1_LR_small$y, fitted(model1_LR_small)
## X-squared = 7.6892, df = 8, p-value = 0.4644
```

- **ejection_fraction** is the most significant predictor.

Add quadratic term for ejection_fraction

```
model1_LR_small_quadra <- glm(DEATH_EVENT ~ age + poly(ejection_fraction, 2)
                              + high_blood_pressure + serum_creatinine,
                              family=binomial, data=train)
sumary(model1_LR_small_quadra)
```

```
##                              Estimate Std. Error z value  Pr(>|z|)
## (Intercept)                 -5.091122   1.047635 -4.8596 1.176e-06
## age                          0.045397   0.015598  2.9103 0.0036105
## poly(ejection_fraction, 2)1 -8.668854   2.630956 -3.2949 0.0009844
## poly(ejection_fraction, 2)2  6.522273   2.726830  2.3919 0.0167620
## high_blood_pressureYes       0.653746   0.357416  1.8291 0.0673863
## serum_creatinine             0.853430   0.257579  3.3133 0.0009221
##
## n = 209 p = 6
## Deviance = 206.78081 Null Deviance = 262.21202 (Difference = 55.43121)
```

```
# GOF (Hosmer-Lemeshow) -> p-value > 0.05, no lack of fit
hoslem.test(model1_LR_small_quadra$y,fitted(model1_LR_small_quadra),g=10)
```

```
##
##  Hosmer and Lemeshow goodness of fit (GOF) test
##
## data:  model1_LR_small_quadra$y, fitted(model1_LR_small_quadra)
## X-squared = 9.3783, df = 8, p-value = 0.3114
```

Test the significance of quadratic term

$$H_0: \text{Smaller model selected by AIC is adequate} \quad vs. \quad H_a: \text{Larger model with quadratic term is adequate}$$

```r
# Compare with backward selection model
anova(model1_LR_small, model1_LR_small_quadra, test = "Chi")
```

```
## Analysis of Deviance Table
##
## Model 1: DEATH_EVENT ~ age + ejection_fraction + high_blood_pressure +
##     serum_creatinine
## Model 2: DEATH_EVENT ~ age + poly(ejection_fraction, 2) + high_blood_pressure +
##     serum_creatinine
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       204     212.38
## 2       203     206.78  1   5.6007  0.01795 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- LRT: Deviance = 5.6007, follow χ^2_1
- The p-value is 0.01795, which is smaller than 0.05, we have evidence to reject the null hypothesis. Thus, we prefer the larger model with quadratic term.

Drop high_blood_pressure since it is not significant

```r
model1_LR_small_quadra_drop <- glm(DEATH_EVENT ~ age + poly(ejection_fraction, 2)
                                   + serum_creatinine, family=binomial, data=train)
sumary(model1_LR_small_quadra_drop)
```

```
##                               Estimate Std. Error z value  Pr(>|z|)
## (Intercept)                  -4.857947   1.012128 -4.7997 1.589e-06
## age                           0.047169   0.015300  3.0829  0.002050
## poly(ejection_fraction, 2)1 -8.244672   2.590283 -3.1829  0.001458
## poly(ejection_fraction, 2)2  6.810303   2.689282  2.5324  0.011329
## serum_creatinine              0.790146   0.249656  3.1649  0.001551
##
## n = 209 p = 5
## Deviance = 210.14863 Null Deviance = 262.21202 (Difference = 52.06339)
```

```r
# GOF (Hosmer-Lemeshow) -> p-value > 0.05, no lack of fit
hoslem.test(model1_LR_small_quadra_drop$y,fitted(model1_LR_small_quadra_drop),g=10)
```

```
##
##  Hosmer and Lemeshow goodness of fit (GOF) test
##
## data:  model1_LR_small_quadra_drop$y, fitted(model1_LR_small_quadra_drop)
## X-squared = 7.3221, df = 8, p-value = 0.5023
```

Compare model with and without drop high_blood_pressure

$$H_0 : \text{Smaller model without high blood pressure is adequate}$$
$$H_a : \text{larger model with high blood pressure is adequate}$$

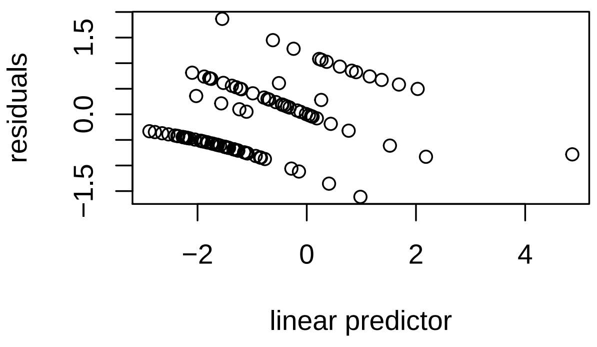
```r
# Compare with backward selection model
anova(model1_LR_small_quadra_drop, model1_LR_small_quadra, test = "Chi")
```

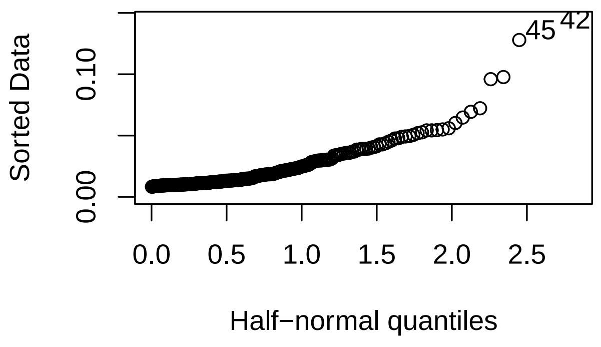
```
## Analysis of Deviance Table
##
## Model 1: DEATH_EVENT ~ age + poly(ejection_fraction, 2) + serum_creatinine
## Model 2: DEATH_EVENT ~ age + poly(ejection_fraction, 2) + high_blood_pressure +
##     serum_creatinine
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       204     210.15
## 2       203     206.78  1   3.3678  0.06648 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- LRT: Deviance = 3.3678, follow χ^2_1
- The p-value is 0.06648, which is larger than 0.05, we fail to reject the null hypothesis. Thus, we prefer the smaller model without high_blood_pressure.

Model Diagnostics

```r
# Deviance Residuals plot
par(mfrow = c(1, 2))
train_assumption <- mutate(train, residuals=residuals(model1_LR_small_quadra_drop),
                           linpred=predict(model1_LR_small_quadra_drop))
gdf <- group_by(train_assumption,
                cut(linpred, breaks=c(min(linpred),
                                      unique(quantile(linpred,(1:100)/101)),max(linpred)),
                    include.lowest = TRUE))
diagdf <- summarise(gdf, residuals=mean(residuals), linpred=mean(linpred), .groups = 'drop')

plot(residuals ~ linpred, diagdf, xlab="linear predictor",cex.lab=1)

# half-normal plot
suppressMessages(library(faraway))
halfnorm(hatvalues(model1_LR_small_quadra_drop))
```





3.3 Model2: LDA

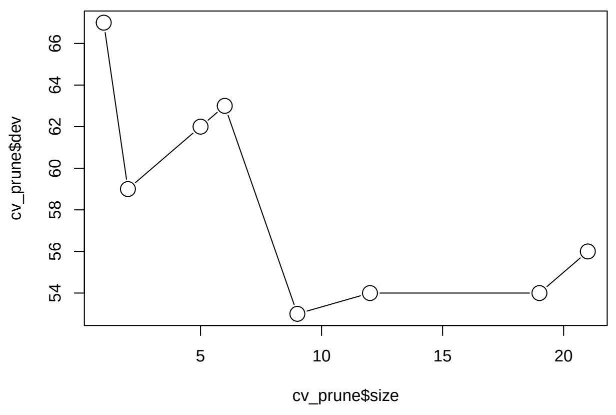
```r
suppressMessages(library(MASS))
model2_LDA = lda(DEATH_EVENT ~ age + poly(ejection_fraction, 2) +
                   log(serum_creatinine), data=train)
```

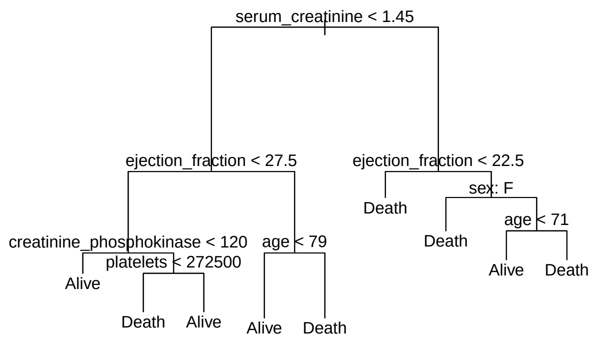
3.4 Model3: QDA

```r
model3_QDA = qda(DEATH_EVENT ~ age + poly(ejection_fraction, 2) +
                   log(serum_creatinine), data=train)
```

3.5 Model4: Classification Tree

```r
suppressMessages(library(tree))
set.seed(108)
model4_tree <- tree(DEATH_EVENT ~ ., data=train)

# pruned tree -> 9 terminal nodes
par(mfrow = c(1, 2))
cv_prune <- cv.tree(model4_tree, FUN=prune.misclass)
plot(cv_prune$size, cv_prune$dev, type='b', cex=2)
model4_tree_prune <- prune.misclass(model4_tree, best=9)
plot(model4_tree_prune); text(model4_tree_prune, pretty=0)
```





3.6 Model5: Bagging

```r
suppressMessages(library(randomForest))
set.seed(108)
model5_bagging <- randomForest(DEATH_EVENT ~ ., data=train, ntree=200,
                        mtry=11, importance=TRUE)
model5_bagging
```

```
##
## Call:
##  randomForest(formula = DEATH_EVENT ~ ., data = train, ntree = 200,      mtry = 11, importance = TRU
##                Type of random forest: classification
##                      Number of trees: 200
## No. of variables tried at each split: 11
##
##         OOB estimate of  error rate: 28.71%
## Confusion matrix:
##       Alive Death class.error
## Alive   118    24   0.1690141
## Death    36    31   0.5373134
```

3.7 Model6: Random Forest

```
set.seed(108)
model6_rf <- randomForest(DEATH_EVENT ~ ., data=train, ntree=200,
                        mtry=4, importance=TRUE)
model6_rf
```

```
##
## Call:
##  randomForest(formula = DEATH_EVENT ~ ., data = train, ntree = 200,      mtry = 4, importance = TRUE
##                Type of random forest: classification
##                      Number of trees: 200
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 28.23%
## Confusion matrix:
##       Alive Death class.error
## Alive   119    23   0.1619718
## Death    36    31   0.5373134
```

3.8 Model7: Boosting

```
suppressMessages(library(gbm))
set.seed(108)

# gbm need character type response instead of factor
train_boost <- train
train_boost$DEATH_EVENT <- as.character(train_boost$DEATH_EVENT)
train_boost <- train_boost %>%
  mutate(DEATH_EVENT = case_when(DEATH_EVENT=="Alive" ~ "0",
                                DEATH_EVENT=="Death" ~ "1"))
test_boost <- test
test_boost$DEATH_EVENT <- as.character(test_boost$DEATH_EVENT)
test_boost <- test_boost %>%
  mutate(DEATH_EVENT = case_when(DEATH_EVENT=="Alive" ~ "0",
                                DEATH_EVENT=="Death" ~ "1"))
```

```
# build the model
model7_boosting <- gbm(DEATH_EVENT ~ ., data=train_boost, distribution="bernoulli",
                n.trees=200, shrinkage=0.1)
model7_boosting
```

```
## gbm(formula = DEATH_EVENT ~ ., distribution = "bernoulli", data = train_boost,
##     n.trees = 200, shrinkage = 0.1)
## A gradient boosted model with bernoulli loss function.
## 200 iterations were performed.
## There were 11 predictors of which 11 had non-zero influence.
```

4. Model Selection using ROC Analysis

```r
suppressMessages(library(pROC))

# Create matrix to store the evaluation metrics for each model
eva_metrics = matrix(0, nrow=7, ncol=5)

# phat
phat1 <- predict(model1_LR_small_quadra_drop, newdata=test, type="response")
phat2 <- predict(model2_LDA, newdata=test)$posterior[,2]
phat3 <- predict(model3_QDA, newdata=test)$posterior[,2]
phat4 <- predict(model4_tree_prune, newdata=test)[,2]
phat5 <- predict(model5_bagging, newdata=test, type="prob")[,2]
phat6 <- predict(model6_rf, newdata=test, type="prob")[,2]
phat7 <- predict(model7_boosting, newdata=test_boost, type="response")

# create roc object
roc_obj1 <- roc(response=test$DEATH_EVENT, predictor=phat1)
roc_obj2 <- roc(response=test$DEATH_EVENT, predictor=phat2)
roc_obj3 <- roc(response=test$DEATH_EVENT, predictor=phat3)
roc_obj4 <- roc(response=test$DEATH_EVENT, predictor=phat4)
roc_obj5 <- roc(response=test$DEATH_EVENT, predictor=phat5)
roc_obj6 <- roc(response=test$DEATH_EVENT, predictor=phat6)
roc_obj7 <- roc(response=test$DEATH_EVENT, predictor=phat7)

# calculate AUC
AUC1 <- auc(roc_obj1)
AUC2 <- auc(roc_obj2)
AUC3 <- auc(roc_obj3)
AUC4 <- auc(roc_obj4)
AUC5 <- auc(roc_obj5)
AUC6 <- auc(roc_obj6)
AUC7 <- auc(roc_obj7)

# show the performance matric
roc_1 <- c(coords(roc_obj1, "b", ret=c("threshold","se","sp","accuracy"),
                         best.method="youden", transpose=TRUE), AUC1)
eva_metrics[1,] <- t(roc_1)

roc_2 <- c(coords(roc_obj2, "b", ret=c("threshold","se","sp","accuracy"),
                         best.method="youden", transpose=TRUE), AUC2)
eva_metrics[2,] <- t(roc_2)

roc_3 <- c(coords(roc_obj3, "b", ret=c("threshold","se","sp","accuracy"),
                         best.method="youden", transpose=TRUE), AUC3)
eva_metrics[3,] <- t(roc_3)

roc_4 <- c(coords(roc_obj4, "b", ret=c("threshold","se","sp","accuracy"),
                         best.method="youden", transpose=TRUE), AUC4)
eva_metrics[4,] <- t(roc_4)

roc_5 <- c(coords(roc_obj5, "b", ret=c("threshold","se","sp","accuracy"),
                         best.method="youden", transpose=TRUE), AUC5)
```

```r
eva_metrics[5,] <- t(roc_5)

roc_6 <- c(coords(roc_obj6, "b", ret=c("threshold","se","sp","accuracy"),
                         best.method="youden", transpose=TRUE), AUC6)
eva_metrics[6,] <- t(roc_6)

roc_7 <- c(coords(roc_obj7, "b", ret=c("threshold","se","sp","accuracy"),
                         best.method="youden", transpose=TRUE), AUC7)
eva_metrics[7,] <- t(roc_7)

# Create metrics df
metrics <- as.data.frame(eva_metrics)
colnames(metrics) = c("Threshold","Sensitivity","Specificity","Accuracy","AUC")
rownames(metrics) = c("Logistic Regression","LDA","QDA","Tree","Bagging","RF","Boosting")
metrics
```

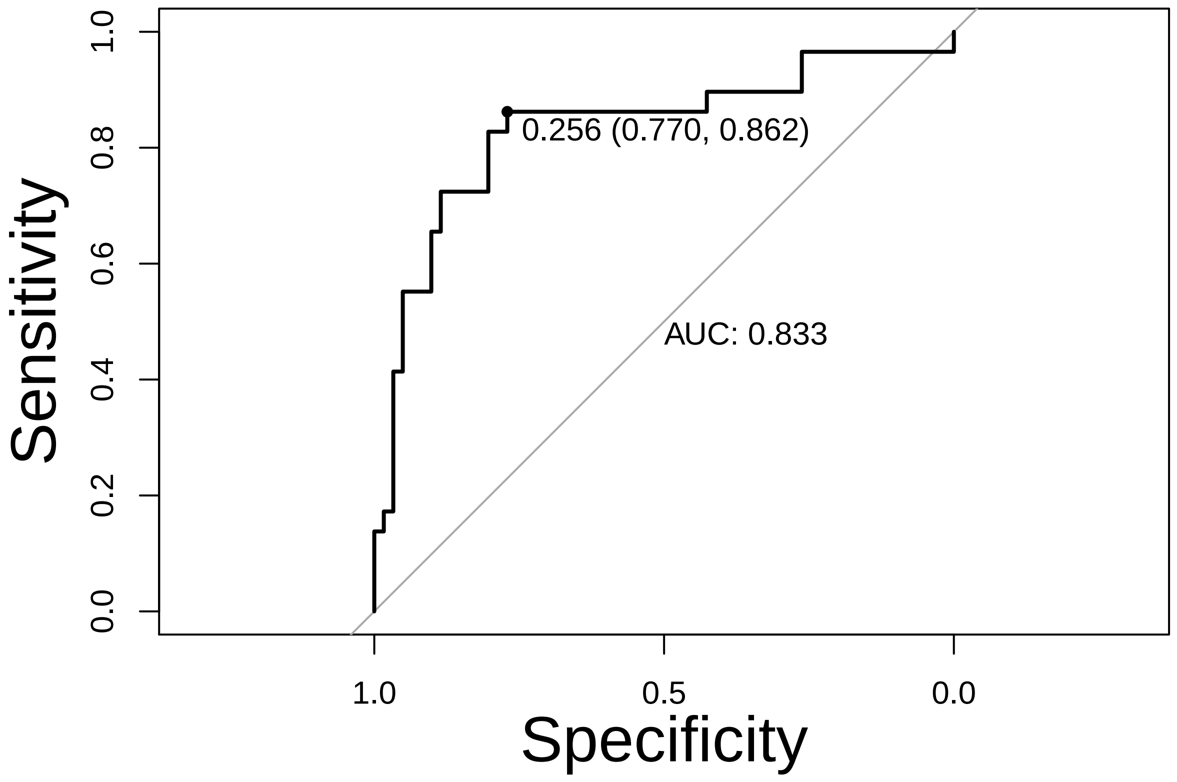
```
##                     Threshold Sensitivity Specificity  Accuracy       AUC
## Logistic Regression 0.2563136   0.8620690   0.7704918 0.8000000 0.8332391
## LDA                 0.2384947   0.7931034   0.7868852 0.7888889 0.8128886
## QDA                 0.3358923   0.7241379   0.8524590 0.8111111 0.7908423
## Tree                0.1370523   0.7931034   0.8032787 0.8000000 0.8114754
## Bagging             0.1725000   0.9310345   0.6393443 0.7333333 0.8174110
## RF                  0.2375000   0.8965517   0.6721311 0.7444444 0.8383267
## Boosting            0.3113842   0.7586207   0.8360656 0.8111111 0.7970605
```

- The best model based on the highest test AUC is Random Forest, the AUC = 0.8383267.
- But Logistic Regression performs (0.8332391) very close to Random Forest, we decide to select Logistic Regression as our final model.

5. Analyze the Best Performing Model - Logistic Regression

5.1 Logistic Regression ROC Analysis

```
# produce ROC Curve
plot(roc_obj1,legacy.axes=FALSE,print.auc=TRUE,print.thres=TRUE,cex.lab=2)
```



5.2 Logistic Regression Confusion Matrix using Best Threshold

```
# Obtain Y_hat values for the data observation (cutoff=0.2563136)
proba_hat <- predict(model1_LR_small_quadra_drop, newdata=test, type="response")

n = nrow(test); y_hat = rep(0,n)
cutoff = 0.2563136; idx = which(proba_hat > cutoff)
y_hat[idx] = 1

# confusion matrix at cutoff=0.2563136
(conf_mat = table(predicted = y_hat, actual = test$DEATH_EVENT))
```

```
##          actual
## predicted Alive Death
##         0    47     4
##         1    14    25
```

```r
# sensitivity/recall
conf_mat[2, 2] / sum(conf_mat[, 2])
```

```
## [1] 0.862069
```

```r
# precision/positive predictive value
conf_mat[2, 2] / sum(conf_mat[2, ])
```

```
## [1] 0.6410256
```

```r
# specificity
conf_mat[1, 1] / sum(conf_mat[, 1])
```

```
## [1] 0.7704918
```

	Summary Metrics for Logistic Regression
Sensitivity/Recall	0.8620690
Precision/Positive Predictive Value	0.6410256
Specificity	0.7704918
Accuracy	0.8000000
AUC	0.8332391

5.3 Logistic Regression Coefficient Interpretation

```r
sumary(model1_LR_small_quadra_drop)
```

```
##                              Estimate Std. Error z value  Pr(>|z|)
## (Intercept)                 -4.857947   1.012128 -4.7997 1.589e-06
## age                          0.047169   0.015300  3.0829  0.002050
## poly(ejection_fraction, 2)1 -8.244672   2.590283 -3.1829  0.001458
## poly(ejection_fraction, 2)2  6.810303   2.689282  2.5324  0.011329
## serum_creatinine             0.790146   0.249656  3.1649  0.001551
##
## n = 209 p = 5
## Deviance = 210.14863 Null Deviance = 262.21202 (Difference = 52.06339)
```

```r
exp(coefficients(model1_LR_small_quadra_drop))
```

```
##                 (Intercept)                         age
##                7.766413e-03                1.048299e+00
## poly(ejection_fraction, 2)1 poly(ejection_fraction, 2)2
##                2.626543e-04                9.071458e+02
##            serum_creatinine
##                2.203718e+00
```