Report on Heart Failure Prediction

Chao Wu

2020/12/30

Contents

Executive summary

In this choose-your-own project, we can choose our own dataset that is not well-known. I picked up the heart failure dataset for this project. The dataset contains 299 observations and 12 features which include body, lifestyle, and clinical information; the outcomes indicate whether the patient survived or not from the heart failure in binary format. The goal of this project is to use at least two different machine learning algorithms to predict the death events by heart failure and at least one of the algorithms are advanced than Logistic Regression.

Before creating models, I applied the following steps:

- Data extraction – I downloaded the dataset from the website, then uploaded to my GitHub repository. So, the dataset is ready to download from the repository.
- Data exploration – By exploring the data, I analysed how risky smoking, diabetes, and high blood pressure are to the death by heart failure. I also analysed whether smoking, diabetes, and high blood pressure would give different genders different impact to heart failure.
- Data wrangling – Because features are in different ranges, I performed standardization to the feature columns. So, features will result in a zero mean and unit variance.
- Data partition – To improve the accuracy, I analysed the data partition ratio which used to split the dataset. The best ratio that gives the highest accuracy on training will be used to split the dataset. The dataset will be split into training and test sets using the best ratio.

I started with Logistic Regression, as this is the basic algorithm to solve classification problem. I also created other models using K-nearest neighbour (KNN), LDA, Decision tree, Random forest, and SVM Linear algorithms. The Ensemble model has been created based on the majority of the votes using all predicted results obtained from previous models. I evaluated each model by computing accuracy on training, accuracy on test, sensitivity (TPR), the false positive rate (FPR), recall, precision, and F1-score. From the evaluation results, I compared each model's performance and identified which model is more accurate when classifying each class.

The most important features given by each model shows which feature(s) having more predictive power. By analysing the correlations between different feature columns and the outcome column, to find out whether there is any connection between the correlation and predictive power in a model. There are some limitations in the models. To improve the speed and accuracy in the future, I looked at dimension reduction, prevalence in the dataset, and other classification algorithms that might be more closely to fit the dataset.

Introduction

This choose-your-own project uses machine learning algorithms to predict death events by heart failure.

For this project, we can choose a dataset that is not well-known. So, I chose the dataset which consists of heart failure clinical records from 299 patients. The dataset was downloaded from Kaggle: https://www.kaggle.com/andrewmvd/heart-failure-clinical-data. It contains 299 observations and 12 features which include body, lifestyle, and clinical information. There are only two outcomes, i.e., survival or death caused by heart failure.

The objectives of this project are to use at least two different algorithms to predict the death events, with at least one being more advanced than Logistic Regression.

As there are two classes to predict, this is a classification problem. To solve this problem, the key steps I took are the following:

1. Extract the dataset;
2. Explore the dataset;
3. Check for missing values and then perform standardization or Z-score normalization in each column;
4. Analyse data partition ratios and then split the dataset into training and test sets based on the best ratio from the analysis;
5. Start with Logistic Regression;
6. Train the model using cross-validation;
7. Use the trained model to predict the results;
8. Evaluate the model by computing accuracy on training, accuracy on test, sensitivity (TPR), the false positive rate (FPR), recall, precision, and F1-score;
9. Repeat steps 5, 6, and 7 for KNN, LDA, Decision tree, Random forest, SVM Linear;
10. Build an ensemble based on the majority of the votes using all predicted results obtained from previous models; Evaluate the ensemble by computing accuracy on test, sensitivity (TPR), the false positive rate (FPR), recall, precision, and F1-score;
11. Compare all models' results and performance;
12. Analyse features to find some insights and see what can be improved in the future.

Library

In this project, I used the following R libraries:

```
library(tidyverse)
library(matrixStats)
library(caret)
library(e1071)
library(rpart)
library(kernlab)
```

Data extraction

The website Kaggle will require login to be able to download the dataset. To avoid the login process, I downloaded the dataset from Kaggle website first; then uploaded it to my GitHub repository. The dataset is ready for download from the repository. As the data file is in CSV format, I can read data directly from the downloaded CSV file in R.

Data exploration

From Table 1, we can see what the dataset looks like.

Table 1: some data rows in the dataset

age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	platelets	serum_creatinine	serum_sodium	sex	smoking	time	DEATH_EVENT
75	0	582	0	20	1	265000	1.9	130	1	0	4	1
55	0	7861	0	38	0	263358	1.1	136	1	0	6	1
65	0	146	0	20	0	162000	1.3	129	1	1	7	1
50	1	111	0	20	0	210000	1.9	137	1	0	7	1
65	1	160	1	20	0	327000	2.7	116	0	0	8	1
90	1	47	0	40	1	204000	2.1	132	1	1	8	1
75	1	246	0	15	0	127000	1.2	137	1	0	10	1
60	1	315	1	60	0	454000	1.1	131	1	1	10	1
65	0	157	0	65	0	263358	1.5	138	0	0	10	1
80	1	123	0	35	1	388000	9.4	133	1	1	10	1

There are 299 samples and 12 features in the dataset. Each data row represents a heart failure clinic record from a patient. Some features represent as binary (0 or 1) or Boolean (0 or 1), e.g., in the 'diabetes' column, 0 means not having diabetes and 1 means having diabetes; in the 'sex' column, 0 means woman and 1 means man; and in the 'DEATH_EVENT' column, 0 means survival and 1 for death. The 'time' column represents follow-up period in days.

Key findings:

- 32.10702% of patients died caused by heart failure. This is the prevalence or death rate of the dataset, which is low.
- We know that smoking is not good for health. Just by looking at the 'smoking' column, the data doesn't approve that there are more patients died who smoke than the patients who don't smoke; it shows that smoking doesn't have strong relationship with death event (Table 2). This means smoking can be considered as a low risk factor. That is, a patient may die not only depending on the amount of cigarette intakes, but also more about the patient's additional body, lifestyle, and clinical information.

Table 2: proportion of death grouped by the patients who smoke or not

smoking	death
0	0.32512
1	0.31250

- By looking at the 'diabetes' column, the data shows that diabetes doesn't have strong relationship with death event (Table 3). Diabetes can be considered as a low risk factor as well.

Table 3: proportion of death grouped by the patients who have diabetes or not

diabetes	death
0	0.32184
1	0.32000

- By looking at the 'high_blood_pressure' column, the data shows that high blood pressure doesn't have strong relationship with death event (Table 4). Like smoking and diabetes, high blood pressure can be considered as a low risk factor as well.

Table 4: proportion of death grouped by the patients who have high blood pressure or not

high_blood_pressure	death
0	0.29381
1	0.37143

- There are more female than male patients died by heart failure who smoke (Table 5). This might indicate that smoking increases more risk to women of having heart failure.

Table 5: proportion of death grouped by gender where patients smoke

sex	death
0	0.75000
1	0.29348

- There are slightly more female than male patients died by heart failure who have diabetes (Table 6). This might indicate that diabetes increases slightly more risk to women of having heart failure.

Table 6: proportion of death grouped by gender where patients have diabetes

sex	death
0	0.36364
1	0.28571

- There are similar number of female and male patients died by heart failure who have high blood pressure (Table 7). This might indicate that high blood pressure has similar risk level for heart failure to different genders.

Table 7: proportion of death grouped by gender where patients have high blood pressure

sex	death
0	0.38636
1	0.36066

- All female patients died by heart failure who smoke and having diabetes (Table 8). This might indicate that it is absolutely risky to women of having heart failure when they have diabetes and also smoke. This might due to smoking and diabetes increase more risk to women of having heart failure (Table 5 and 6).

Table 8: proportion of death grouped by gender where patients smoke and have diabetes

sex	death
0	1.00000
1	0.35714

- There are more female than male patients died by heart failure who smoke and also have high blood pressure (Table 9). The proportion of female patients died is significant smaller than that of Table 8. This might due to less impact from high blood pressure to gender (Table 7).

Table 9: proportion of death grouped by gender where patients smoke and have high blood pressure

sex	death
0	0.66667
1	0.44444

Data wrangling

Check for missing values:

There is 0 missing value in the dataset, which means we don't need to fill in any values.

Standardization/ Z-score normalization:

From Table 1, we can see that features span different ranges. I applied standardization to all the feature columns, those features will result in a zero mean and unit variance. As standardization may speed up the distance-based algorithms, such as K-Means, KNN and so on. In addition, "the range of all features should be normalized so that each feature contributes approximately proportionately to the final distance" ("Feature scaling," 2020).

The formula of standardization can be represented as following:

$$x_{scaled} = (x - \bar{x})/SD$$

Where:

- x is the original feature vector;
- $\bar{x}$ is the mean of that feature;
- SD is the standard deviation of that feature.

Before standardization:

The column 'platelets' has the highest mean 2.63358×10^5; column 'smoking' has the lowest mean 0.32107.

After standardization:

Here are some statistics and exploration from the scaled features:

- The first feature column has standard deviation 1 and mean -1.99604×10^{-17}. Note that the mean value is approximately equal to zero, but not zero, this is an expected behaviour due to 15 decimal digits of precision of floats.
- The average distance between the first "survival" sample and other "survival" samples is 5.07225.
- The average distance between the first "survival" sample and "death" samples is 4.55338.
- These two average distances are very close, this classification task might be tricky for distance-based algorithms. As the samples of different classes might be too close to each other to be differentiated.

Data partition

I will split the whole dataset into training set and test set. The training set will be used to develop the algorithms, as well as optimize the algorithm parameters. The test set will be only used for evaluation.

Analysis of data partition ratios:

To find the best partition ratio to improve classification performance, I analysed different partition ratios. Logistic Regression is a basic algorithm to solve classification problem, so, I used Logistic Regression to create a model for this analysis. To avoid overfitting, I only used training set in this analysis. I look for the ratio that gives the highest accuracy on training.

I took the following steps:

1. Create a sequence of partition ratios for test set - from 0.1 to 0.5, jump by 0.1;
2. Split the dataset into training and test sets using a ratio in the sequence;
3. To make the code run a bit faster, use 10-fold cross-validation with 10% of the observations to train the Logistic Regression model;
4. Obtain the training accuracy;
5. Repeat steps 2, 3, and 4 for each ratio in the sequence;
6. Compare all accuracies and find the ratio that gives the highest accuracy.

From Table 10, we can see that the ratio 0.3 on test set gives the highest accuracy on training. So, I will use this ratio to create training and test sets.

Table 10: display partition ratio on test set and its corresponding accuracy on training

Ratio_on_test_set	Accuracy_on_training
0.1	0.80989
0.2	0.82020
0.3	0.83753
0.4	0.83235
0.5	0.79143

Spliting dataset into training set and test set:

The whole dataset has been splitted into training set and test set using the best ratio obtained from the analysis above. That is, 70% of the data will be used for training and 30% for test.

In the original dataset, 67.89298% of samples are indicated as survival. Training set and test set both have similar proportion of samples that indicated as survial, they are 67.94258% and 67.77778%, respectively.

Methods/Algorithms

I started with Logistic Regression, then used six different algorithms as comparison.

Logistic Regression:

Logistic Regression model uses logistic function and log odds to solve binary classification task. Logistic Regression gives probability of the likelihood of an event to happen between 0 and 1, as well as how much more likely something

will happen. As there are only two outcomes or classes (0 or 1) in the dataset, this is a binary classification problem. So, I picked up Logistic Regression to start with.

I used cross-validation to train and tune the model.

KNN:

Similar cases with the same class labels are near each other. For example, similar "survival" cases have some common body, lifestyle, and clinical information. KNN is selected to create a model to build on the similarity of one case to the others.

I used cross-validation to train and tune the model.

Parameter tuning:

The parameter k is the tuning parameter in this model. I created a sequence of k value for tuning - from 1 to 21, jump by 3. Cross-validation will find the best tuned k value while training the model.

LDA:

LDA can be used to solve binary classification problem. After the standardization, all the columns have the same standard deviation 1. By forcing the assumption that all features share the same correlations, LDA will be similar to Logistic Regression, the boundary will be a line to separate two classes.

I used cross-validation to train and tune the model.

Decision tree:

Decision tree can be used to solve classification problem. It is a flowchart-like structure that builds all possible decision paths in the form of a tree, the leaf nodes contain the possible outcomes. The heart failure dataset contains body, lifestyle, and clinical information, there might be some decision rules that can be used to determined whether a patient will survive or died.

I used cross-validation to train and tune the model.

Parameter tuning:

The parameter *cp* is the tuning parameter in this model. I created a sequence of *cp* value for tuning - from 0 to 0.1, jump by 0.001. Cross-validation will find the best tuned *cp* value while training the model.

Random forest:

The reason of choosing Random forest is similar to Decision tree, as they are all tree-based algorithm. Random forest randomly selects number of features and constructs multiple trees which solves overfitting problem where the Decision tree might have; as well as to improve the prediction performance.

I used cross-validation to train and tune the model.

Parameter tuning:

The parameter *mtry* is the tuning parameter in this model. I created a sequence of *mtry* value for tuning - from 3 to 11, jump by 2. Cross-validation will find the best tuned *mtry* value while training the model.

SVM Linear:

SVM Linear is used to solve bi-classification problem, which could be used to solve this classification problem. It assumes the two classes are linear separable. The boundary will be line or hyper-plane to separate two classes.

I used cross-validation to train and tune the model.

Ensemble:

To improve the accuracy, I built an Ensemble model. The ensemble draws the prediction based on the majority of the votes using all predicted results obtained from previous models. Therefore, there is no training for this model.

Results and Performance

To evaluate how a model does, I computed accuracy on training, accuracy on test, sensitivity (TPR), the false positive rate (FPR), recall, precision, and F1-score.

To understand more about the trained model, I printed important features that have been used in that model. By visualizing the predicted results, I can see what have been predicted correctly and wrongly. In the plots, I picked up the first two features as x-axis and y-axis, respectively.

Note that, the positive class used in all models is 0, the "survival" class. As 0 appears before 1 in the dataset.

Logistic Regression:

The Logistic Regression model does reasonably well (Table 11).

- The accuracy on test is higher than the accuracy on training.
- The FPR is low, while TPR is high. The F1-score shows the precision and recall are well balanced.
- This model has lower performance when classifying the "death" class (by FPR and Figure 1).

Table 11: display evaluation results of the Logistic Regression model

Method	Accuracy_on_training	Accuracy_on_test	FPR	TPR	Recall	Precision	F_1
Logistic regression	0.76223	0.87778	0.2069	0.91803	0.91803	0.90323	0.91057

Figure 1: visualize predicted results from the Logistic Regression model

The top three important features in the Logistic Regression model are 'time', 'ejection_fraction', and 'serum_creatinine', as below.

```
## glm variable importance
##
##                           Overall
## time                      100.000
## ejection_fraction          74.030
## serum_creatinine           57.629
## serum_sodium               44.446
## age                        44.441
## sex                        15.487
## creatinine_phosphokinase   15.026
## high_blood_pressure         9.033
## platelets                   3.524
## diabetes                    2.145
## smoking                     0.818
## anaemia                     0.000
```

KNN:

The KNN model is the worst model so far (Table 12):

- By looking at the accuracy on test, the KNN model is worse than the Logistic Regression model.
- The accuracy on test is higher than the accuracy on training.

- Although the TPR is high and F1-score shows the precision and recall are well balanced. FPR is very high.
- This model fails to classify actual "death" as "death" (by FPR and Figure 2). Because the prevalence is low in the dataset, failing to classify the death samples does not lower the accuracy as much.

Table 12: display evaluation results of the KNN model

Method	Accuracy_on_training	Accuracy_on_test	FPR	TPR	Recall	Precision	F_1
Logistic regression	0.76223	0.87778	0.20690	0.91803	0.91803	0.90323	0.91057
KNN	0.73123	0.74444	0.72414	0.96721	0.96721	0.73750	0.91057

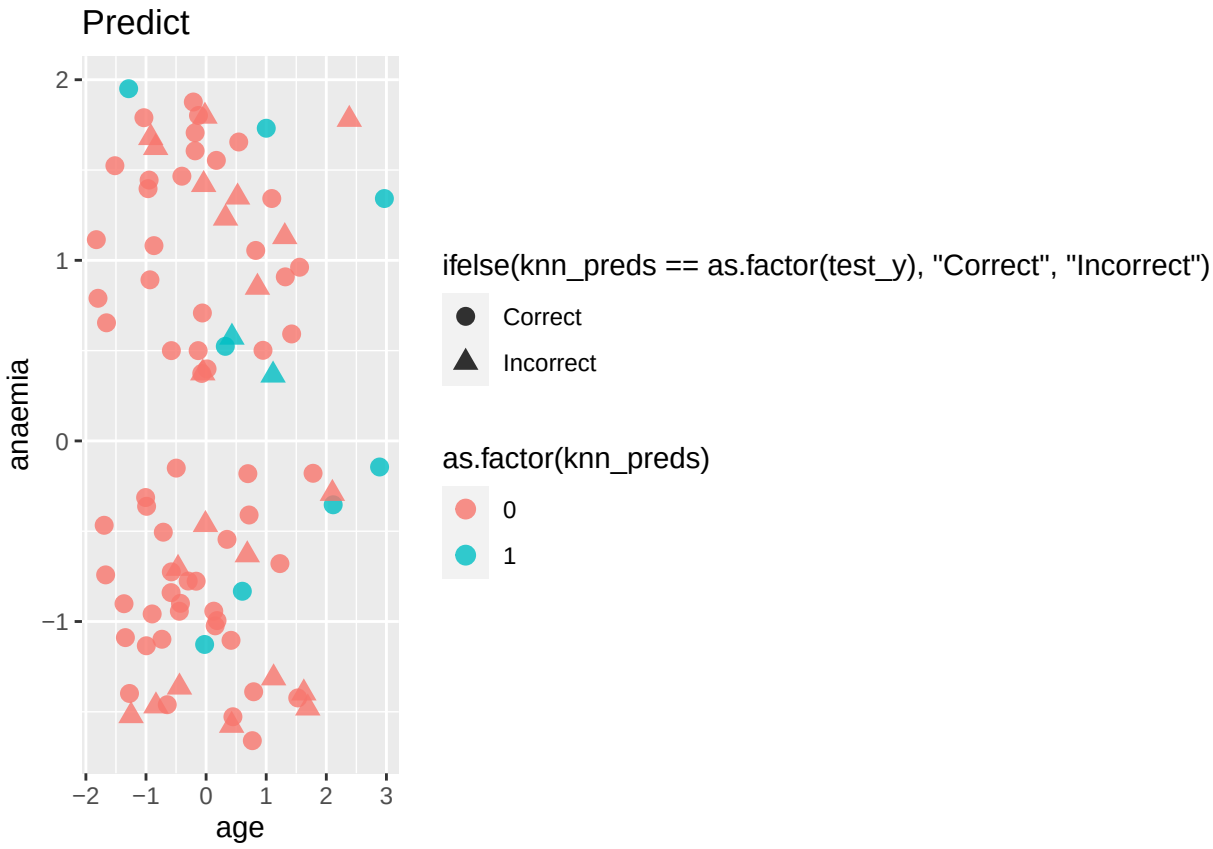

Figure 2: visualize predicted results from the KNN model

The top three important features in the KNN model are 'time', 'serum_creatinine', and 'ejection_fraction', as below.

```
## ROC curve variable importance
##
##                          Importance
## time                        100.000
## serum_creatinine             71.324
## ejection_fraction            64.953
## serum_sodium                 50.128
## age                          37.660
## high_blood_pressure          24.406
## anaemia                      15.389
```

```
## diabetes                      13.459
## sex                            5.722
## platelets                      4.116
## creatinine_phosphokinase       0.649
## smoking                        0.000
```

The optimized parameter k that gives the highest accuracy is 10.

LDA:

The LDA model does reasonably well, the results are the same as of the Logistic Regression model, apart from the accuracy on training (Table 13):

- The accuracy on test is higher than the accuracy on training.
- The FPR is low, while TPR is high. The F1-score shows the precision and recall are well balanced.
- This model has lower performance when classifying the "death" class (by FPR and Figure 3).

Table 13: display evaluation results of the LDA model

Method	Accuracy_on_training	Accuracy_on_test	FPR	TPR	Recall	Precision	F_1
Logistic regression	0.76223	0.87778	0.20690	0.91803	0.91803	0.90323	0.91057
KNN	0.73123	0.74444	0.72414	0.96721	0.96721	0.73750	0.91057
LDA	0.76660	0.87778	0.20690	0.91803	0.91803	0.90323	0.91057

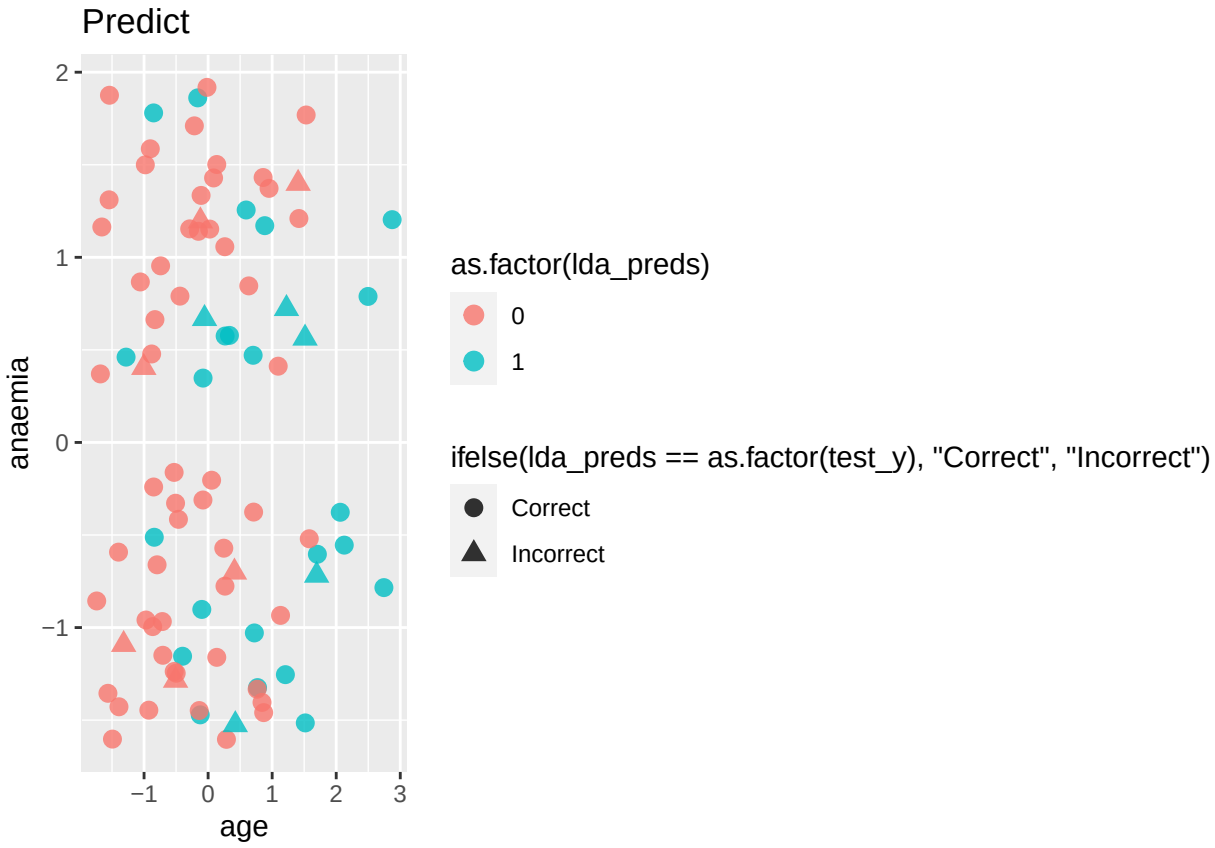

Figure 3: visualize predicted results from the LDA model

The top three important features in the LDA model are 'time', 'serum_creatinine', and 'ejection_fraction', as below.

```
## ROC curve variable importance
##
##                          Importance
## time                        100.000
## serum_creatinine             71.324
## ejection_fraction            64.953
## serum_sodium                 50.128
## age                          37.660
## high_blood_pressure          24.406
## anaemia                      15.389
## diabetes                     13.459
## sex                           5.722
## platelets                     4.116
## creatinine_phosphokinase      0.649
## smoking                       0.000
```

Decision tree:

The Decision tree model is the best model so far (Table 14):

- By looking at the accuracy on test, the Decision tree model is better than the Logistic Regression model.
- The accuracy on test is higher than the accuracy on training.
- The FPR is low, while TPR is high. The F1-score shows the precision and recall are well balanced.
- Comparing to the Logistic Regression model, the FPR and F1-score are the same, while the Decision tree model successfully classifies more "survival" samples (by TPR and Figure 4).

Table 14: display evaluation results of the Decision tree model

Method	Accuracy_on_training	Accuracy_on_test	FPR	TPR	Recall	Precision	F_1
Logistic regression	0.76223	0.87778	0.20690	0.91803	0.91803	0.90323	0.91057
KNN	0.73123	0.74444	0.72414	0.96721	0.96721	0.73750	0.91057
LDA	0.76660	0.87778	0.20690	0.91803	0.91803	0.90323	0.91057
Decision tree	0.79365	0.91111	0.20690	0.96721	0.96721	0.90769	0.91057

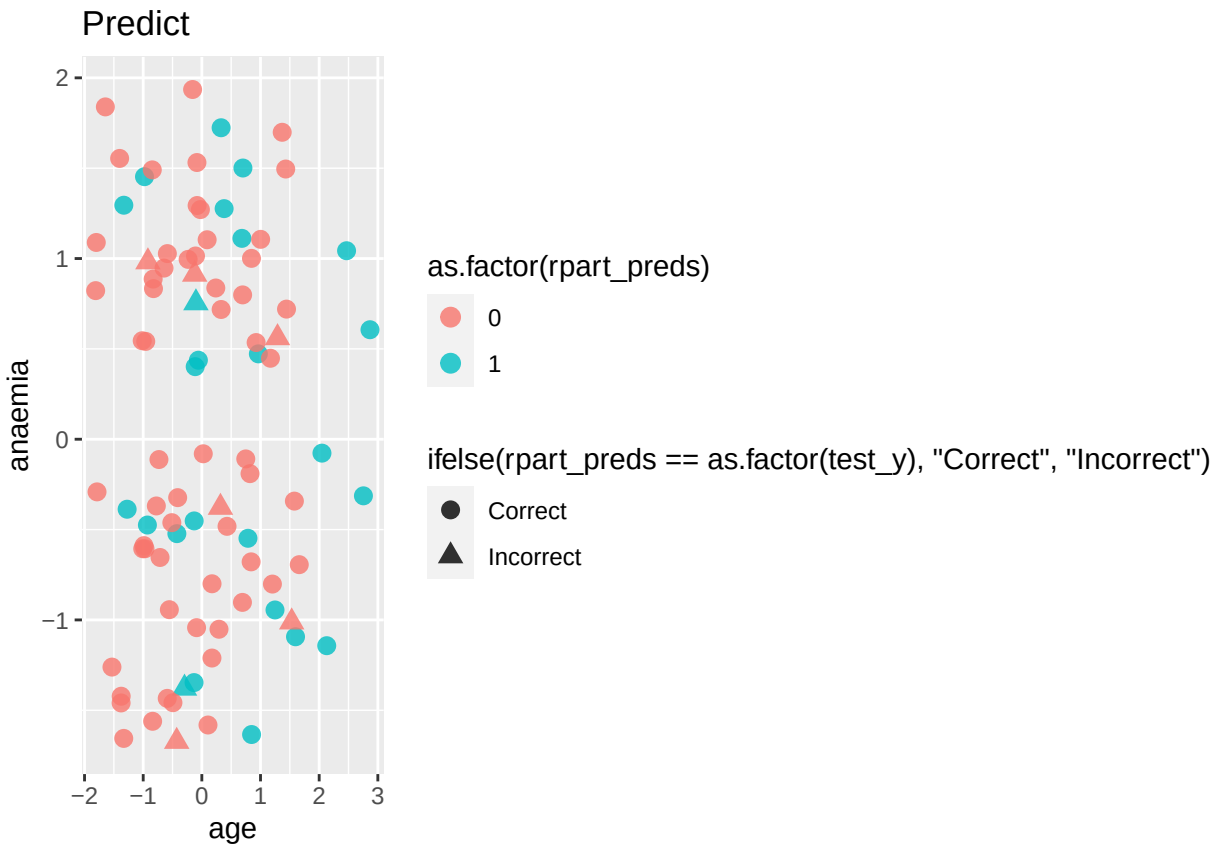

Figure 4: visualize predicted results from the Decision tree model

The Decision tree model only uses one feature, i.e., 'time'. We can interpret the decision rule as "if the time or follow-up period is equal or more than -0.7313 (the value is after the scale), the sample can be marked as 'survival'; otherwise 'death' " (Figure 5).

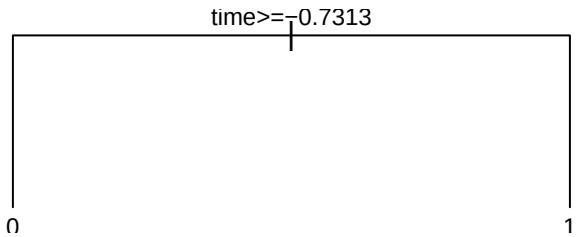

Figure 5: display the shape of the tree

The top three important features in the Decision tree model are 'time', 'ejection_fraction', and 'serum_creatinine', as below.

```
## rpart variable importance
##
##                           Overall
## time                        100.0
## ejection_fraction            52.8
## serum_creatinine             41.3
## age                          26.9
## serum_sodium                 22.9
## platelets                     0.0
## sex                           0.0
## high_blood_pressure           0.0
## diabetes                      0.0
## creatinine_phosphokinase      0.0
## anaemia                       0.0
## smoking                       0.0
```

The optimized parameter *cp* that gives the highest accuracy is 0.061.

Random forest:

The Random forest model does reasonably well (Table 15):

- By looking at the accuracy on test, the Random forest model is better than the Logistic Regression model, but worse than the Decision tree model.
- The accuracy on test is higher than the accuracy on training.
- The FPR is low, while TPR is high. The F1-score shows the precision and recall are well balanced.
- Comparing to the Logistic Regression model, the F1-score is the same, while other results are better than the Logistic Regression model.
- Comparing to the Decision tree model, this Random forest model classifies more "death" samples, but less "survival" samples (by FPR, TPR, and Figure 6), although the overall accuracy is lower than the Decision tree model.

Table 15: display evaluation results of the Random forest model

Method	Accuracy_on_training	Accuracy_on_test	FPR	TPR	Recall	Precision	F_1
Logistic regression	0.76223	0.87778	0.20690	0.91803	0.91803	0.90323	0.91057
KNN	0.73123	0.74444	0.72414	0.96721	0.96721	0.73750	0.91057
LDA	0.76660	0.87778	0.20690	0.91803	0.91803	0.90323	0.91057
Decision tree	0.79365	0.91111	0.20690	0.96721	0.96721	0.90769	0.91057
Random forest	0.81747	0.90000	0.17241	0.93443	0.93443	0.91935	0.91057

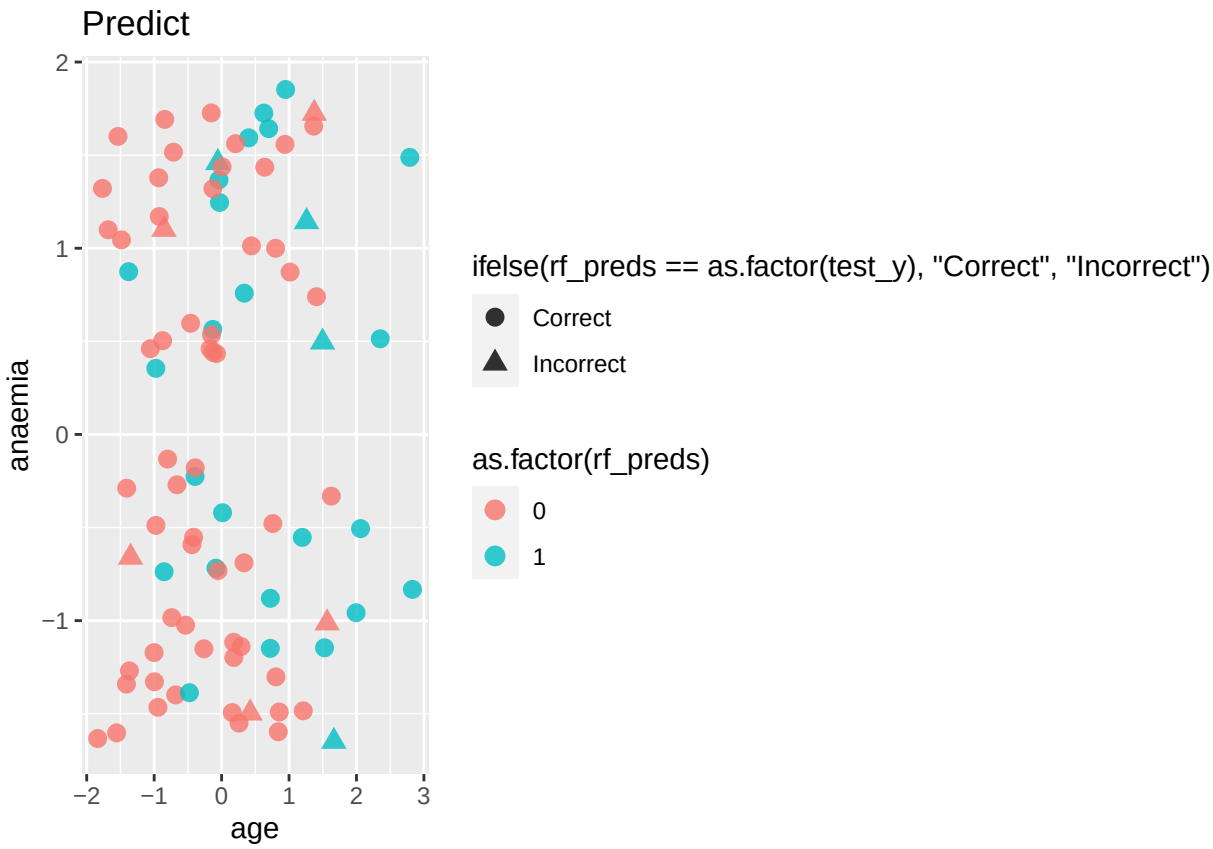

Figure 6: visualize predicted results from the Random forest model

As Random forest randomly selects features, by selecting about 2 to 3 features, the model reaches the highest accuracy (Figure 7).

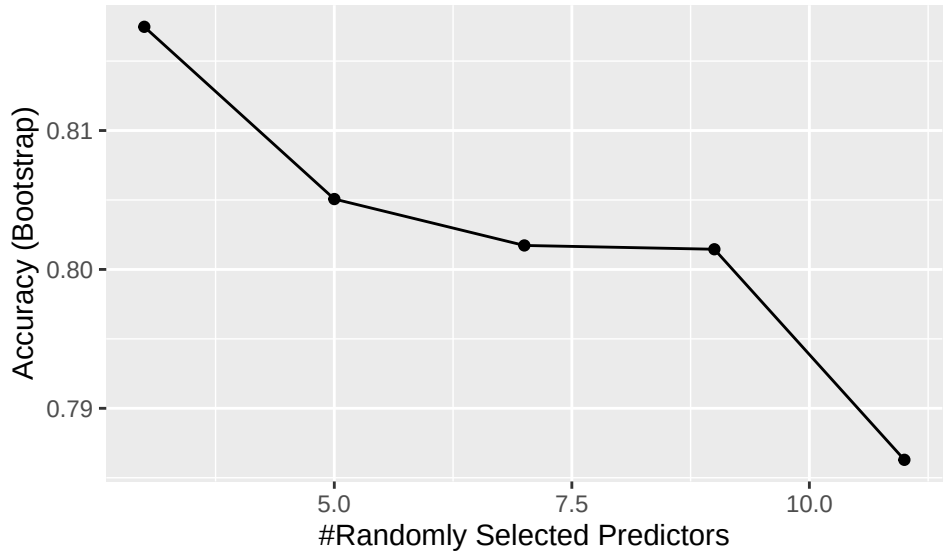

Figure 7: display accuracy against number of features being selected

The top three important features in the Random forest model are 'time', 'ejection_fraction', and 'serum_creatinine', as below.

```
## rf variable importance
##
##                           Importance
## time                          100.00
## ejection_fraction              53.44
## serum_creatinine               49.47
## serum_sodium                   21.96
## platelets                      17.72
## age                            13.13
## high_blood_pressure            12.40
## anaemia                        11.94
## sex                             7.74
## creatinine_phosphokinase        5.13
## diabetes                        3.48
## smoking                         0.00
```

The optimized parameter *mtry* that gives the highest accuracy is 3.

SVM Linear:

The SVM Linear model does reasonably well, the results are the same as of the Logistic Regression model, apart from the accuracy on training (Table 16):

- The accuracy on test is higher than the accuracy on training.
- The FPR is low, while TPR is high. The F1-score shows the precision and recall are well balanced.
- This model has lower performance when classifying the "death" class (by FPR and Figure 8).

Table 16: display evaluation results of the SVM Linear model

Method	Accuracy_on_training	Accuracy_on_test	FPR	TPR	Recall	Precision	F_1
Logistic regression	0.76223	0.87778	0.20690	0.91803	0.91803	0.90323	0.91057
KNN	0.73123	0.74444	0.72414	0.96721	0.96721	0.73750	0.91057
LDA	0.76660	0.87778	0.20690	0.91803	0.91803	0.90323	0.91057
Decision tree	0.79365	0.91111	0.20690	0.96721	0.96721	0.90769	0.91057
Random forest	0.81747	0.90000	0.17241	0.93443	0.93443	0.91935	0.91057
SVM Linear	0.76585	0.87778	0.20690	0.91803	0.91803	0.90323	0.91057

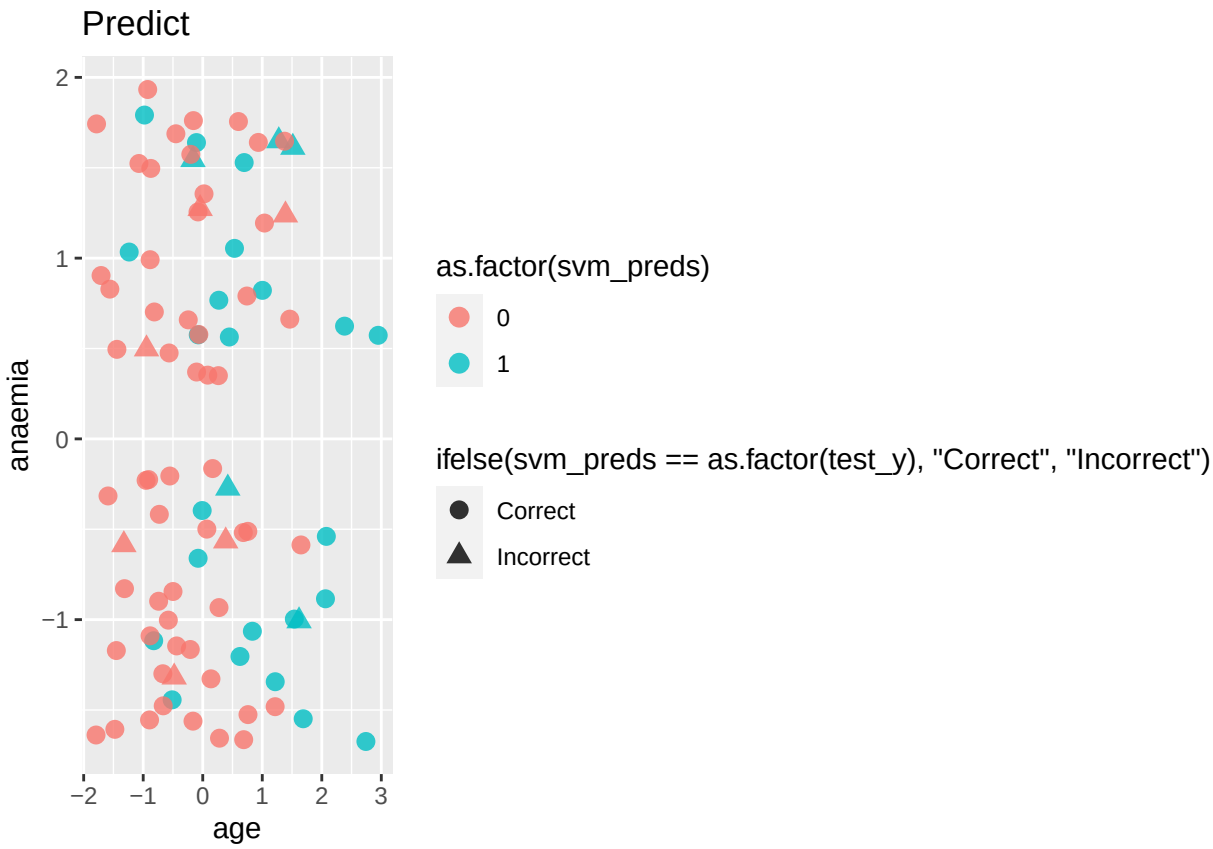

Figure 8: visualize predicted results from the SVM Linear model

The top three important features in the SVM Linear model are 'time', 'serum_creatinine', and 'ejection_fraction', as below.

```
## ROC curve variable importance
##
##                           Importance
## time                         100.000
## serum_creatinine              71.324
## ejection_fraction             64.953
## serum_sodium                  50.128
## age                           37.660
## high_blood_pressure           24.406
## anaemia                       15.389
## diabetes                      13.459
## sex                            5.722
## platelets                      4.116
## creatinine_phosphokinase       0.649
## smoking                        0.000
```

Ensemble:

As the Ensemble combines all the predicted results from previous models, no accuracy will be given on training, as well as no important features.

The Ensemble model is one of the best models, apart from the Decision tree model (Table 17):

- By looking at the accuracy on test, the Ensemble model is better than the Logistic Regression model and the same as of the Decision tree model.
- The accuracy on test is higher than the accuracy on training.
- The FPR is low, while TPR is high. The F1-score shows the precision and recall are well balanced.
- Comparing to the Logistic Regression and Decision tree models, the F1-score is the same, while there are some highs and lows on other results. This Ensemble model classifies more "survival" samples, but less "death" samples (by TPR, FPR, and Figure 9).

Table 17: display evaluation results of the Ensemble

Method	Accuracy_on_training	Accuracy_on_test	FPR	TPR	Recall	Precision	F_1
Logistic regression	0.76223	0.87778	0.20690	0.91803	0.91803	0.90323	0.91057
KNN	0.73123	0.74444	0.72414	0.96721	0.96721	0.73750	0.91057
LDA	0.76660	0.87778	0.20690	0.91803	0.91803	0.90323	0.91057
Decision tree	0.79365	0.91111	0.20690	0.96721	0.96721	0.90769	0.91057
Random forest	0.81747	0.90000	0.17241	0.93443	0.93443	0.91935	0.91057
SVM Linear	0.76585	0.87778	0.20690	0.91803	0.91803	0.90323	0.91057
Ensemble	0.00000	0.91111	0.24138	0.98361	0.98361	0.89552	0.91057

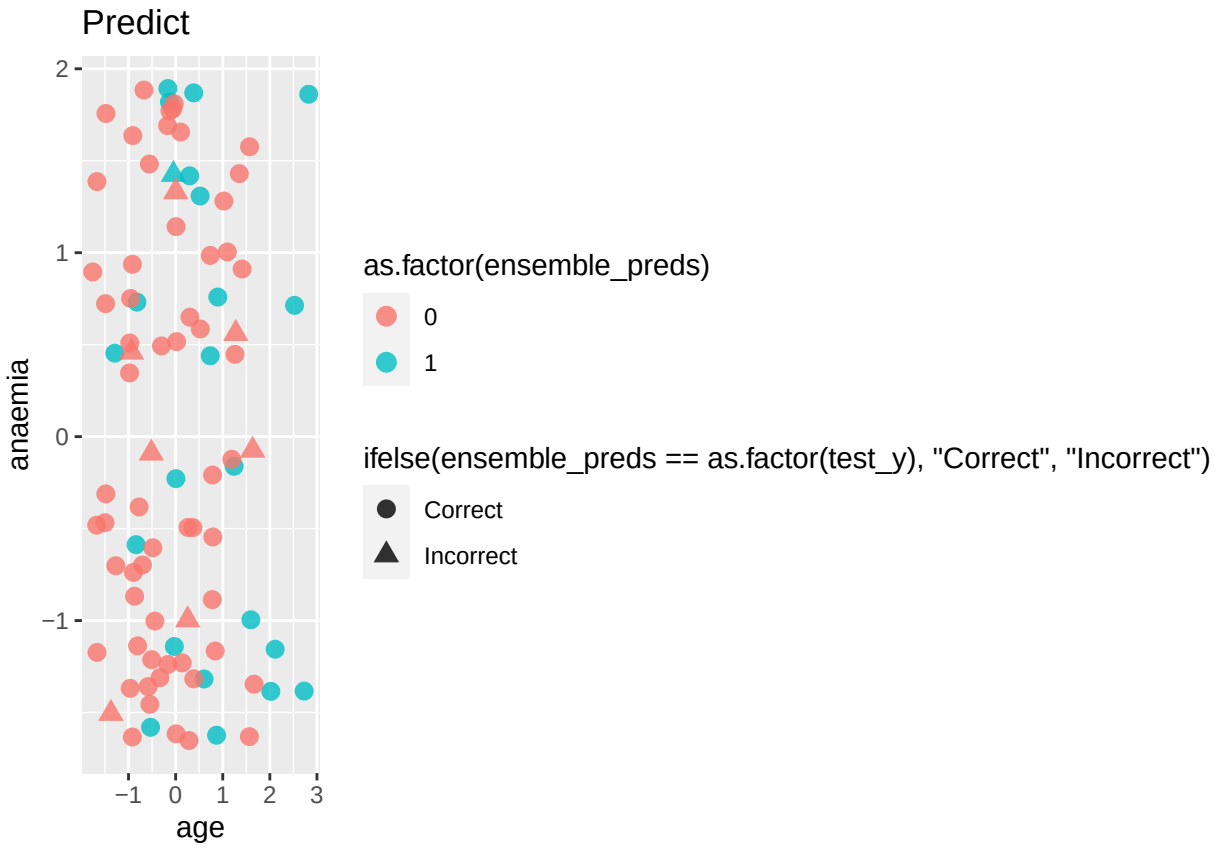

Figure 9: visualize predicted results from the Ensemble

Summary:

The average accuracy on test is 0.87143.

The models that are equal or above the average are Logistic regression, LDA, Decision tree, Random forest, SVM Linear, Ensemble.

Overall, no model is overtrained, as the accuracy on test is higher than the accuracy on training. The Decision tree and Ensemble models give the highest accuracy which is 0.91111. The KNN model gives the lowest accuracy which is 0.74444. The Ensemble model is more accurate when classifying the "survival" class. The Random forest model is more accurate when classifying the "death" class; while the KNN model performs poorly when classifying the "death" class. (Table 17)

Linear algorithms, e.g., Logistic Regression, LDA, and SVM Linear, give the same accuracy on test. Tree-based algorithms are better than linear or distance-based algorithms in this classification task. Because the data points of the two classes are mixed together and some data points of these two classes are very close in distance.

The Decision tree and Random forest models are slow compared to other models. Because Decision tree uses recursive partitioning to create the model and Random forest creates a large number of tree, these make algorithms slow. The Decision tree, Random forest, and Ensemble models have good overall performance. They are more advanced than Logistic Regression model.

Conclusion

Insights:

From the most important features given by each model, we can see that the feature 'time' is given 100% on the top of the most important feature list. It means the 'time' feature has 100% predictive power in the model. Features 'serum_creatinine' and 'ejection_fraction' are always ranked as second or third on the list. These two features also have some predictive power in the model. While 'smoking', 'diabetes', and 'high_blood_pressure' are far more down the list, i.e., they don't have much predictive power in the model.

We can see the reflection from the correlation between the outcome and these features. Feature 'time' has great negative relationship to the outcome; 'ejection_fraction' and 'serum_creatinine' have less negative and positive relationships to the outcome, respectively; while 'smoking', 'diabetes', and 'high_blood_pressure' have much weaker relationships to the outcome. (Table 18)

Table 18: display the correlation between the outcome and the feature

Feature	Correlation
'time'	-0.52696
'ejection_fraction'	-0.26860
'serum_creatinine'	0.29428
'smoking'	-0.01262
'diabetes'	-0.00194
'high_blood_pressure'	0.07935

These findings approve that smoking, diabetes, and high blood pressure can be considered as low risk factors to the death by heart failure; while 'ejection_fraction' and 'serum_creatinine' are high risk factors. Features 'time', 'ejection_fraction', and 'serum_creatinine' have predictive power.

Limitations:

There are some limitations in this project:

- Because the prevalence is low in the dataset and the positive class is 0 ("survival" class), failing to classify the death samples does not lower the accuracy as much.

- I used all features in the model, the prediction process could take longer. For Logistic Regression model, when using cross-validation to train, the time matters, the machine may crash when the process is long.
- As the data points of the two outcomes are mixed together, which is not linearly separable. Some of the algorithms I picked up are linear-based, the resulting models cannot separate two classes just by drawing a line, plane or hyper-plane.

Future work:

I might consider the following in the future work:

- By changing the positive class to the "death" class, we can see how the model classifies the "death" class clearer. As failing to classify the "death" class is more serious, a patient may miss a chance of having a medical treatment.
- To speed up the process, I will consider dimension reduction. The findings tell us that we can obtain the same results by only using the 'time' feature.
- We can see that features 'ejection_fraction' and 'serum_creatinine' also have predictive power. By plotting the data using these two features, we can see two clusters, one for each class (Figure 10). If the outcomes can be predicted only using these two features, the model can directly help medical doctors. "medical doctors aiming at understanding if a patient will survive after heart failure may focus mainly on serum creatinine and ejection fraction." (Chicco, & Jurman, 2020)

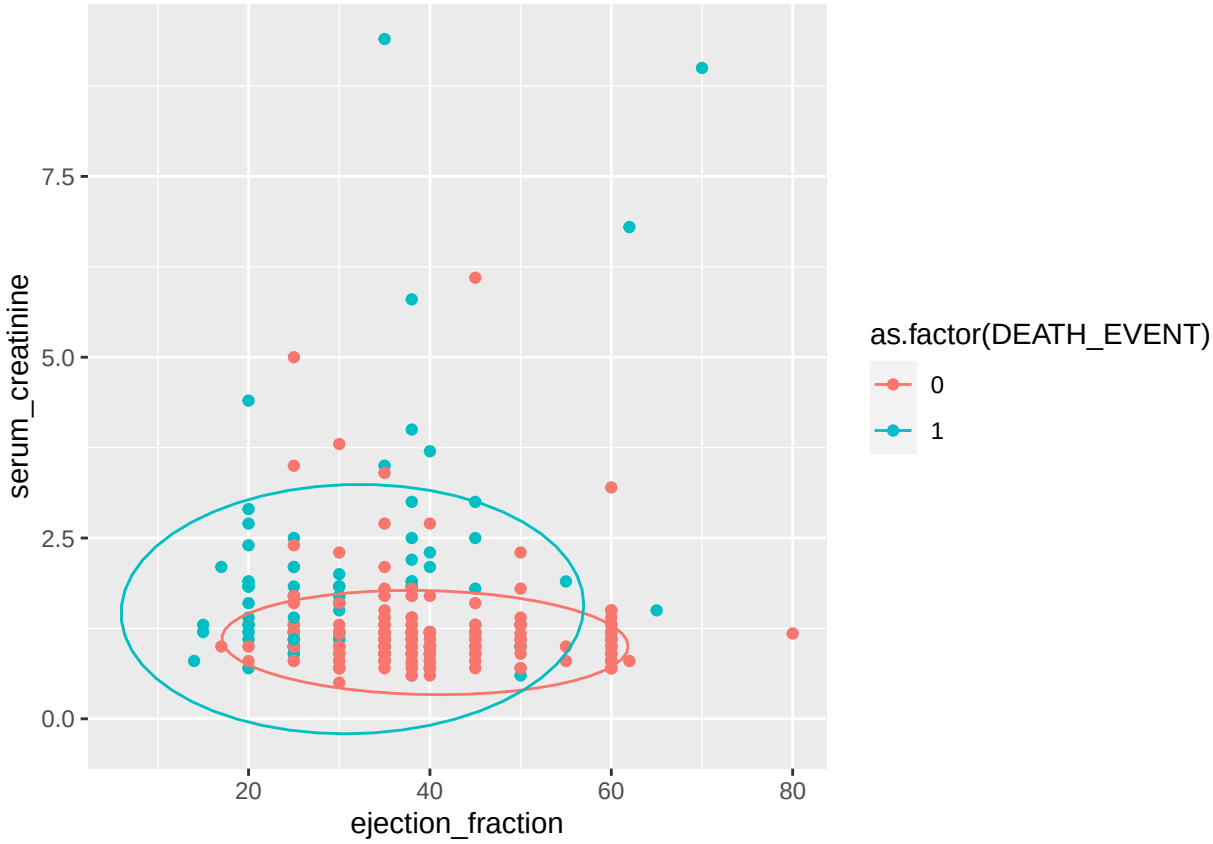

Figure 10: display data by plotting 'serum_creatinine' against 'ejection_fraction'

- The data is not linearly separable. To improve the accuracy, I may try Neural Network and non-linear algorithms, such as SVM with kernels. These algorithms can be used to classify the data that is not linearly separable.

Conclusion:

I chose the heart failure dataset for this choose-your-own project. This project is about predicting the death event by heart failure using machine learning algorithms.

I applied standardization to each feature column, so that these features will result in a zero mean and unit variance. Based on the analysis, data partition ratio 0.3 on test gives the highest accuracy on training. So, I split the dataset into training and test sets, in which training set takes 70% of the dataset and the rest becomes test set. From data exploration, I found out that smoking, diabetes, and high blood pressure are low risk factors to heart failure; ejection fraction and serum creatinine are high risk factors. Furthermore, smoking and diabetes increase more risk to women of having heart failure. As the data is not linear separable, linear algorithms (Logistic Regression, LDA, and SVM Linear) are worse than tree-based algorithms (Decision tree and Random forest). The worst algorithm is KNN that gives the lowest accuracy on test; this might due to the close distances between the data points of two different classes. From the most important features given by each model, the feature with most predictive power is 'time'; 'ejection_fraction' and 'serum_creatinine' also have predictive power. The feature that is more correlated to the outcome will have more predictive power in the model.

To improve the speed in the future, I might consider dimension reduction, since features 'time', 'ejection_fraction', and 'serum_creatinine' have much more predictive power than others. As the prevalence is low in the dataset, I will change the positive class to the "death" class. When selecting algorithms, I will consider Neural Network and non-linear algorithms in addition to the tree-based algorithms.

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*, 16. https://bmcmedinformdecismak.biomedcentral.com/articles/10.1186/s12911-020-1023-5
Feature scaling. (2020, October). Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Feature_scaling
Heart Failure Prediction. (2020, July). Retrieved from Kaggle: https://www.kaggle.com/andrewmvd/heart-failure-clinical-data