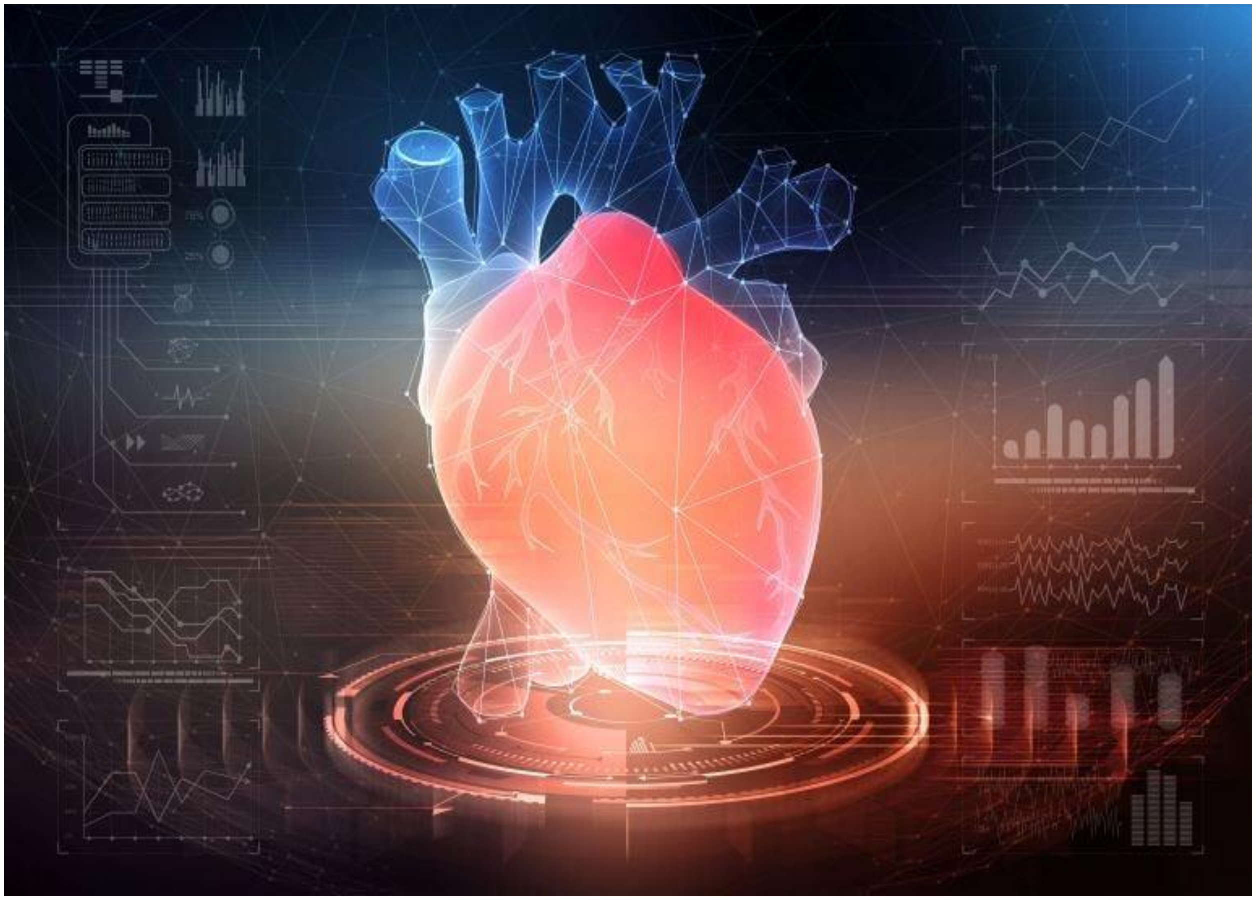
PULSE UNVEILED



SURVIVAL PREDICTION OF HEART FAILURE PATIENTS USING DATA MINING TECHNIQUES

Task 1 : Predict the survival onset of the patients with Heart Failure based on the diagnostics clinical records using Logistic Regression Analysis and R – programming.

CONTENTS

TASK 01 :

PREDICTING THE PROBABILITY OF THE SURVIVAL ONSET OF THE PATIENTS WITH HEART FAILURE BASED ON THE DIAGNOSTICS CLINICAL RECORDS USING LOGISTIC REGRESSION ANALYSIS AND R – PROGRAMMING.

1.1. INTRODUCTION

Heart failure is a complex cardiovascular condition that presents a significant global health challenge. It is crucial to accurately predict patient outcomes to deliver personalized and effective healthcare. Each year, around 17 million people worldwide die from cardiovascular diseases, primarily including heart failure and myocardial infarction (MI). Heart failure occurs when the heart cannot adequately pump blood to meet the body's demands. There is a growing demand for advanced strategies that can enhance prognostic capabilities and tailor treatment plans based on evolving insights into this condition. The Heart Failure Clinical Records dataset, utilized in this study, is a valuable resource for training and evaluating predictive models due to its extensive clinical data covering a wide array of parameters.

The UCI heart failure dataset offers an extensive collection of clinical data that is ideal for using logistic regression, a potent classification algorithm, to forecast survival rates for patients with heart failure. This dataset offers thorough data on a range of patient characteristics, enabling a thorough analysis through the use of logistic regression to identify predictive patterns linked to survival events. Using this dataset and logistic regression, researchers can find important risk factors and create models that help with early diagnosis and individualized treatment plans for heart failure patients. By using data-driven methods, the combination of this particular dataset and logistic regression techniques presents a significant opportunity to advance our knowledge of cardiovascular health and enhance patient care.

These findings have the potential to inform early intervention strategies and personalized care plans for individuals affected by heart failure. This report serves as a roadmap for implementing machine learning techniques to advance more efficient and data-driven approaches to managing cardiac health using Logistic Regression.

1.2. DATA SET

The dataset that we used for this task was obtained from the University of California, Irvine, Machine Learning Repository using the link below.
https://archive.ics.uci.edu/dataset/519/heart+failure+clinical+records

This dataset enables the development of a predictive model for estimating the onset of survival among heart failure patients treated at the Faisalabad Institute of Cardiology and the Allied Health Hospital in Faisalabad (Punjab, Pakistan) between April and December 2015. The dataset includes information on various patient characteristics. All patients included in the dataset had left ventricular systolic dysfunction and a history of prior heart failures, classified as New York Heart Association's (NYHA) classes III and IV based on their cardiac health status and left ventricular systolic dysfunction (Chicco and Jurman, 2020).

Based on the physical symptoms and pertinent test results, doctors may not always be able to determine which factor is the primary cause of the disease in many patients. They will be able to form a more accurate opinion about what they should focus on more in diabetes patients to increase their survival rate if they can find it by identifying patterns and insights in previous data.

1.3. EXPLANATION AND PREPARATION OF THE DATA SET

1.3.1. Explanation of the Data Set

The Heart Failure Clinical Records dataset was utilized for a classification task and comprises medical records from 299 patients diagnosed with heart failure. The patients' ages ranged from 40 to 95 years, with 105 women and 194 men included in the dataset. The dataset consists of 300 rows and 13 columns, representing 13 clinical features (variables), and importantly, there are no missing values present.

The 13 variables within the dataset are as follows :

1. ***age :*** Age of the patient in years
2. ***anaemia :*** Decrease of red blood cells or hemoglobin (Yes = 1, No = 0)
3. ***creatinine_phosphokinase :*** Level of the CPK enzyme in the blood in mcg/L
4. ***diabetes :*** If the patient has diabetes (Yes = 1, No = 0)
5. ***ejection_fraction :*** Percentage of blood leaving the heart at each contraction in %
6. ***high_blood_pressure :*** If the patient has hypertension (Yes = 1, No = 0)
7. ***platelets :*** platelets in the blood in kiloplatelets/ml

8. ***serum_creatinine :*** Level of serum creatinine in the blood in mg/dL
9. ***serum_sodium :*** Level of serum sodium in the blood in mEq/L
10. ***sex :*** Female = 0 , Male = 1
11. ***smoking :*** If the patient smokes or not (Yes = 1, No = 0)
12. ***time :*** Follow-up period in days
13. ***DEATH_EVENT :*** if the patient died during the follow-up period (Yes = 1, No = 0)

Accordingly, the last column 'DEATH_EVENT' is the label (dependent) variable of the dataset

Consequently, the final column labeled 'DEATH_EVENT' serves as the dependent variable in the dataset, while all other columns except for the last one are considered independent variables.

1.3.2. Preparation of the Data Set

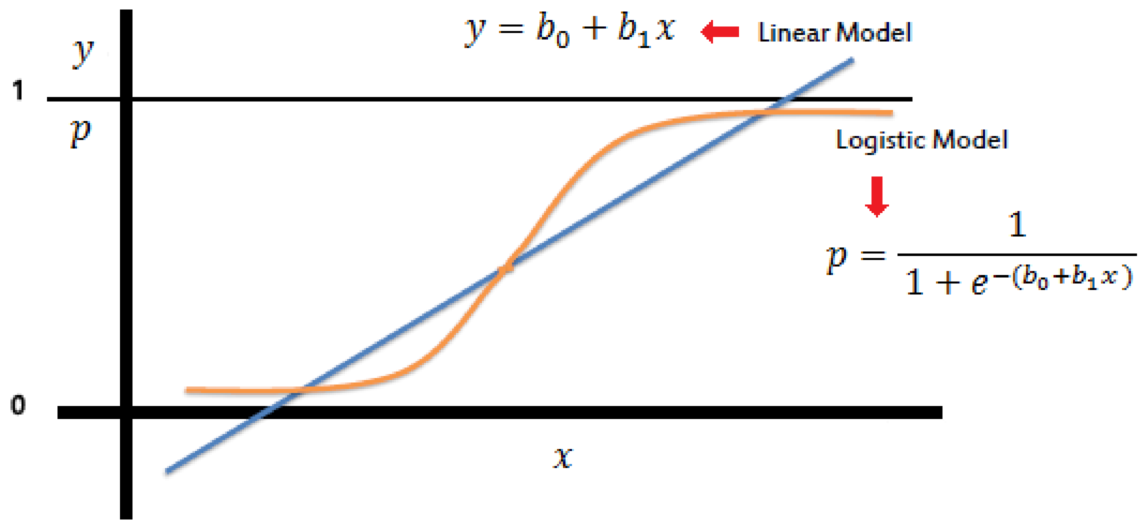
No significant null (missing) values or outliers are evident in the dataset, indicating that the data has likely been pre-processed and cleaned.

1.4. DATA MINING

1.4.1. The application of Data Mining Techniques

- **Logistic Regression**

Logistic regression is a widely used method for binary classification in data mining. It provides a probabilistic framework to model relationships between input features and categorical outcomes, making it suitable for various applications such as disease diagnosis. Therefore, by leveraging logistic regression, a binary classification model can be developed to estimate the probability of death events (DEATH_EVENTS) based on independent variables. The aim of using logistic regression to this dataset is to build a predictive model that can assist in identifying high risk patients and guiding personalized interventions to improve patient outcomes in the context of heart failure management.



1.4.2. Test and Train Data Split

Data must be divided into training and testing sets to use logistic regression on the UCI Heart Failure Clinical Records dataset. The dataset is split into two subsets at random for this process: one is used to train the logistic regression model (Training Model), and the other is used to assess how well it performs (Testing Model). To build the models, we have utilized 70% of the data set for training and 30% for testing.

The training set is used to fit the logistic regression model by learning the relationships between the independent variables and the dependent variable (DEATH_EVENT).

The model's ability to predict death events among patients with heart failure can then be evaluated by calculating its accuracy, precision, recall, and other performance metrics. By training the logistic regression model on one subset of the data and testing it on a different subset, this split strategy guarantees that the model's predictive power is reliably evaluated.

1.4.3. Use of Visualization Tools available in R

Using visualization tools for logistic regression in data mining enhances data exploration and interpretation in R-programming. Packages like "epitools," "tidyverse," "ggally," and "RColorBrewer" can be used to create scatter plots, matrix plots, and network graphs, which allow users to explore and modify visualizations dynamically. These tools can be used to visually represent logistic regression.

1.5. IMPLEMENTATION IN R

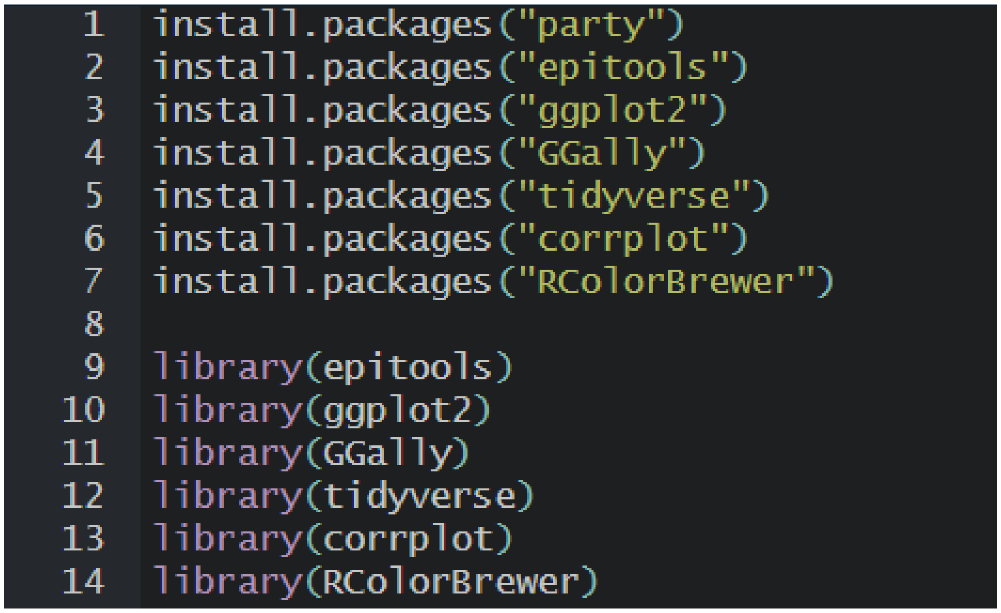
Using R to implement data mining techniques provides a flexible platform with many packages for exploring and analyzing large, complex datasets. R provides strong logistic regression tools, and its leveraging packages improve interactive exploration and visualization. For analysts and data scientists, its versatility and range of packages make data mining tasks efficient and productive.

1.5.1. Explanation of the experimental procedure and visualization of the results

Initially, before we proceed with the task, it is essential to install the required R-packages to get started with applying logistic regression to the Heart Failure Clinical Records dataset.

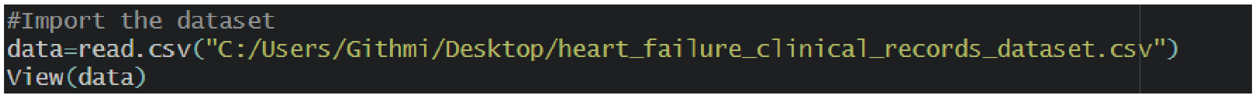
- **readr** : Offering effective data file reading and parsing tools for R programming.
- **party** : Tools for recursive partitioning, like decision trees and random trees, are included in this package. Regression and classification issues benefit from it.
- **epitools** : A software application for epidemiological computation and visualization is called Epitools. It has tools for producing common graphs used in epidemiology as well as statistics related to the field.
- **tidyverse** : A group of packages (such as ggplot2 and dplyr) for data manipulation and visualization.
- **RColorBrewer** : Offers color schemes that are appropriate for data visualization, improving the plots' visual appeal.
- **corrplot** : Used to display correlation matrices. It offers features with numerous customizable options for making correlation charts.
- **GGally** : Adds functions to ggplot2 to enable the creation of complex graphs like correlation plots, density plots, and scatterplot matrices. It works wonders for examining the relationships between a variety of variables.

Step 1 : The packages mentioned above can be installed and activated as follows.

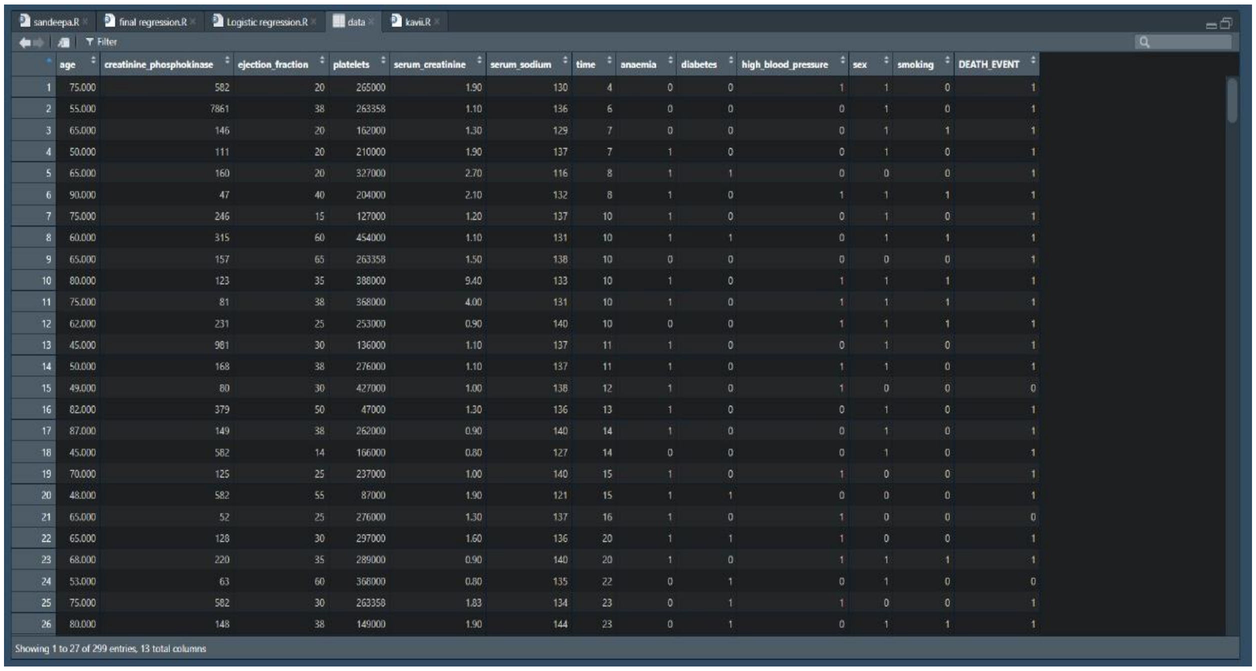


```r
install.packages("party")
install.packages("epitools")
install.packages("ggplot2")
install.packages("GGally")
install.packages("tidyverse")
install.packages("corrplot")
install.packages("RColorBrewer")

library(epitools)
library(ggplot2)
library(GGally)
library(tidyverse)
library(corrplot)
library(RColorBrewer)
```

Step 2 : The data set should be imported and viewed in R.



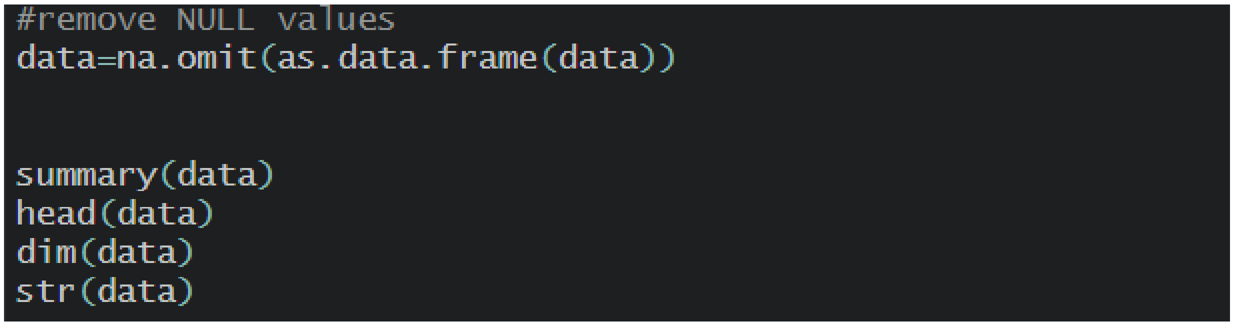
```r
#Import the dataset
data=read.csv("C:/Users/Githmi/Desktop/heart_failure_clinical_records_dataset.csv")
View(data)
```



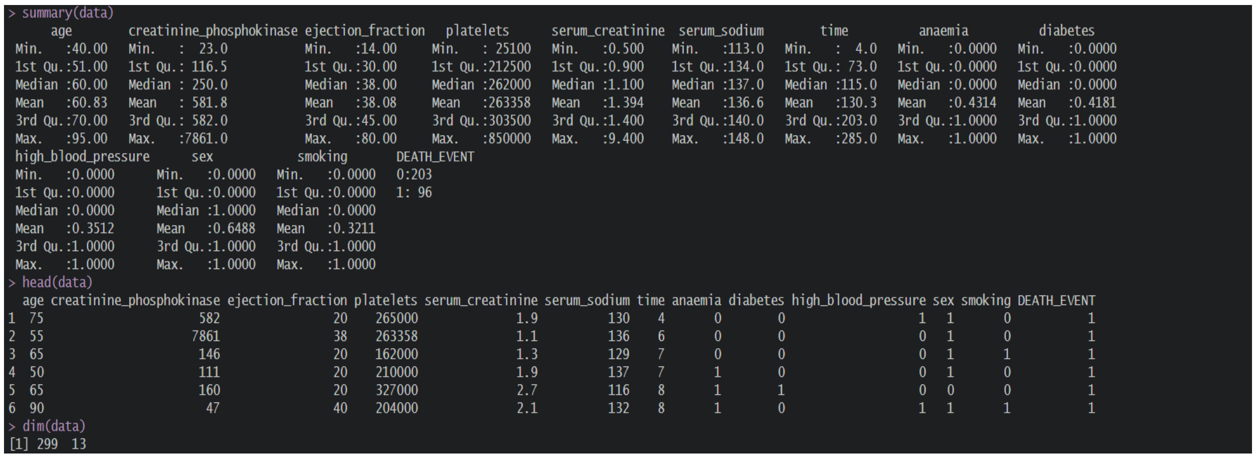
	age	creatinine_phosphokinase	ejection_fraction	platelets	serum_creatinine	serum_sodium	time	anaemia	diabetes	high_blood_pressure	sex	smoking	DEATH_EVENT
1	75.000	582	20	265000	1.90	130	4	0	0	1	1	0	1
2	55.000	7861	38	263358	1.10	136	6	0	0	0	1	0	1
3	65.000	146	20	162000	1.30	129	7	0	0	0	1	1	1
4	50.000	111	20	210000	1.90	137	7	1	0	0	1	0	1
5	65.000	160	20	327000	2.70	116	8	1	1	0	0	0	1
6	90.000	47	40	204000	2.10	132	8	1	0	1	1	1	1
7	75.000	246	15	127000	1.20	137	10	1	0	0	1	0	1
8	60.000	315	60	454000	1.10	131	10	1	1	0	1	1	1
9	65.000	157	65	263358	1.50	138	10	0	0	0	0	0	1
10	80.000	123	35	388000	9.40	133	10	1	0	1	1	1	1
11	75.000	81	38	368000	4.00	131	10	1	0	1	1	1	1
12	62.000	231	25	253000	0.90	140	10	0	0	1	1	1	1
13	45.000	981	30	136000	1.10	137	11	1	0	0	1	0	1
14	50.000	168	38	276000	1.10	137	11	1	0	1	1	0	1
15	49.000	80	30	427000	1.00	138	12	1	0	1	0	0	0
16	82.000	379	50	47000	1.30	136	13	1	0	0	1	0	1
17	87.000	149	38	262000	0.90	140	14	1	0	0	1	0	1
18	45.000	582	14	166000	0.80	127	14	0	0	0	1	0	1
19	70.000	125	25	237000	1.00	140	15	1	0	1	0	0	1
20	48.000	582	55	87000	1.90	121	15	1	1	0	0	0	1
21	65.000	52	25	276000	1.30	137	16	1	0	1	0	0	0
22	65.000	128	30	297000	1.60	136	20	1	1	1	0	0	1
23	68.000	220	35	289000	0.90	140	20	1	0	1	1	1	1
24	53.000	63	60	368000	0.80	135	22	0	1	0	1	0	0
25	75.000	582	30	263358	1.83	134	23	0	1	1	0	0	1
26	80.000	148	38	149000	1.90	144	23	0	1	0	1	1	1

Showing 1 to 27 of 299 entries, 13 total columns

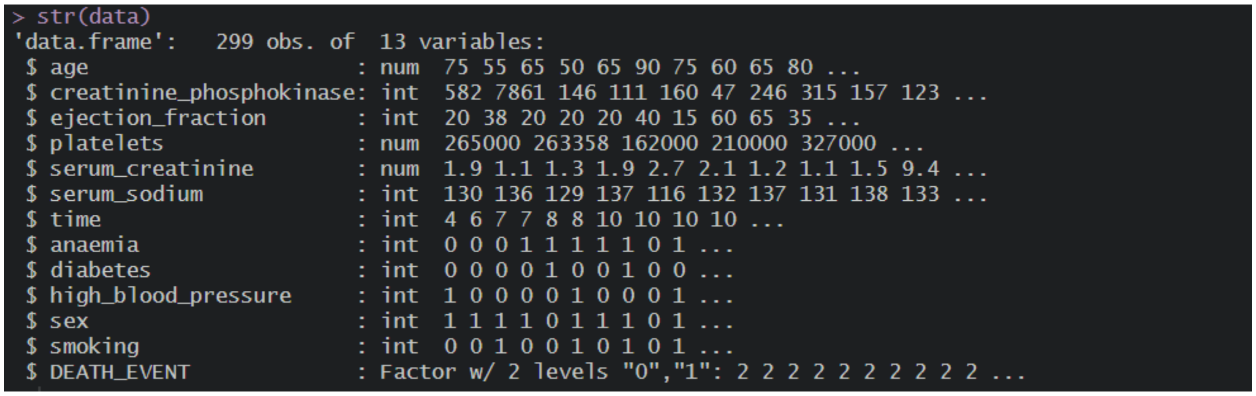
Step 3 : Check for null values and remove if there are any existing null values in the dataset. Next, the dataset can be further explored using head (), dim(), str() and summary() functions as shown below.



```
#remove NULL values
data=na.omit(as.data.frame(data))


summary(data)
head(data)
dim(data)
str(data)
```

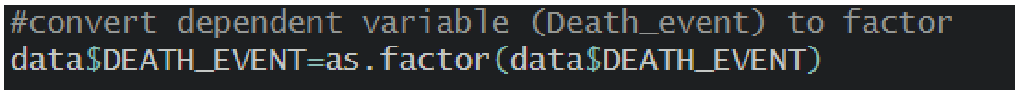


```
> summary(data)
      age        creatinine_phosphokinase ejection_fraction   platelets      serum_creatinine  serum_sodium        time          anaemia          diabetes     
 Min.   :40.00   Min.   :  23.0           Min.   :14.00     Min.   : 25100   Min.   :0.500    Min.   :113.0   Min.   :  4.0   Min.   :0.0000   Min.   :0.0000  
 1st Qu.:51.00   1st Qu.: 116.5           1st Qu.:30.00     1st Qu.:212500   1st Qu.:0.900    1st Qu.:134.0   1st Qu.: 73.0   1st Qu.:0.0000   1st Qu.:0.0000  
 Median :60.00   Median : 250.0           Median :38.00     Median :262000   Median :1.100    Median :137.0   Median :115.0   Median :0.0000   Median :0.0000  
 Mean   :60.83   Mean   : 581.8           Mean   :38.08     Mean   :263358   Mean   :1.394    Mean   :136.6   Mean   :130.3   Mean   :0.4314   Mean   :0.4181  
 3rd Qu.:70.00   3rd Qu.: 582.0           3rd Qu.:45.00     3rd Qu.:303500   3rd Qu.:1.400    3rd Qu.:140.0   3rd Qu.:203.0   3rd Qu.:1.0000   3rd Qu.:1.0000  
 Max.   :95.00   Max.   :7861.0           Max.   :80.00     Max.   :850000   Max.   :9.400    Max.   :148.0   Max.   :285.0   Max.   :1.0000   Max.   :1.0000  
 high_blood_pressure      sex            smoking       DEATH_EVENT
 Min.   :0.0000      Min.   :0.0000   Min.   :0.0000   0:203      
 1st Qu.:0.0000      1st Qu.:0.0000   1st Qu.:0.0000   1: 96      
 Median :0.0000      Median :1.0000   Median :0.0000              
 Mean   :0.3512      Mean   :0.6488   Mean   :0.3211              
 3rd Qu.:1.0000      3rd Qu.:1.0000   3rd Qu.:1.0000              
 Max.   :1.0000      Max.   :1.0000   Max.   :1.0000              
> head(data)
  age creatinine_phosphokinase ejection_fraction platelets serum_creatinine serum_sodium time anaemia diabetes high_blood_pressure sex smoking DEATH_EVENT
1  75                      582                20    265000              1.9          130    4       0        0                   1   1       0           1
2  55                     7861                38    263358              1.1          136    6       0        0                   0   1       0           1
3  65                      146                20    162000              1.3          129    7       0        0                   0   1       1           1
4  50                      111                20    210000              1.9          137    7       1        0                   0   1       0           1
5  65                      160                20    327000              2.7          116    8       1        1                   0   0       0           1
6  90                       47                40    204000              2.1          132    8       1        0                   1   1       1           1
> dim(data)
[1] 299  13
```



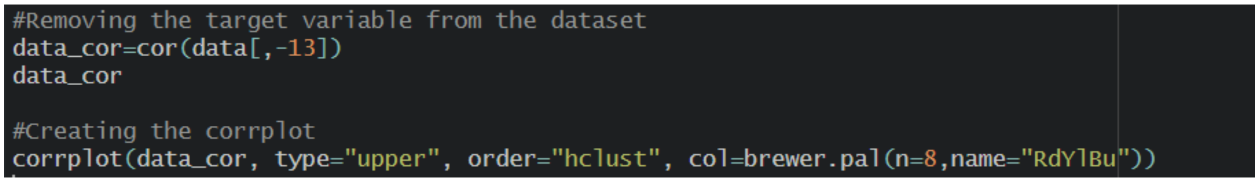
```
> str(data)
'data.frame':	299 obs. of  13 variables:
 $ age                     : num  75 55 65 50 65 90 75 60 65 80 ...
 $ creatinine_phosphokinase: int  582 7861 146 111 160 47 246 315 157 123 ...
 $ ejection_fraction       : int  20 38 20 20 20 40 15 60 65 35 ...
 $ platelets               : num  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 ...
 $ time                    : int  4 6 7 7 8 8 10 10 10 10 ...
 $ anaemia                 : int  0 0 0 1 1 1 1 1 0 1 ...
 $ diabetes                : int  0 0 0 0 1 0 0 1 0 0 ...
 $ high_blood_pressure     : int  1 0 0 0 0 1 0 0 0 1 ...
 $ sex                     : int  1 1 1 1 0 1 1 1 0 1 ...
 $ smoking                 : int  0 0 1 0 0 1 0 1 0 1 ...
 $ DEATH_EVENT             : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
```

Step 4 : Convert the target / dependent variable (DEATH_EVENT) into a factor.



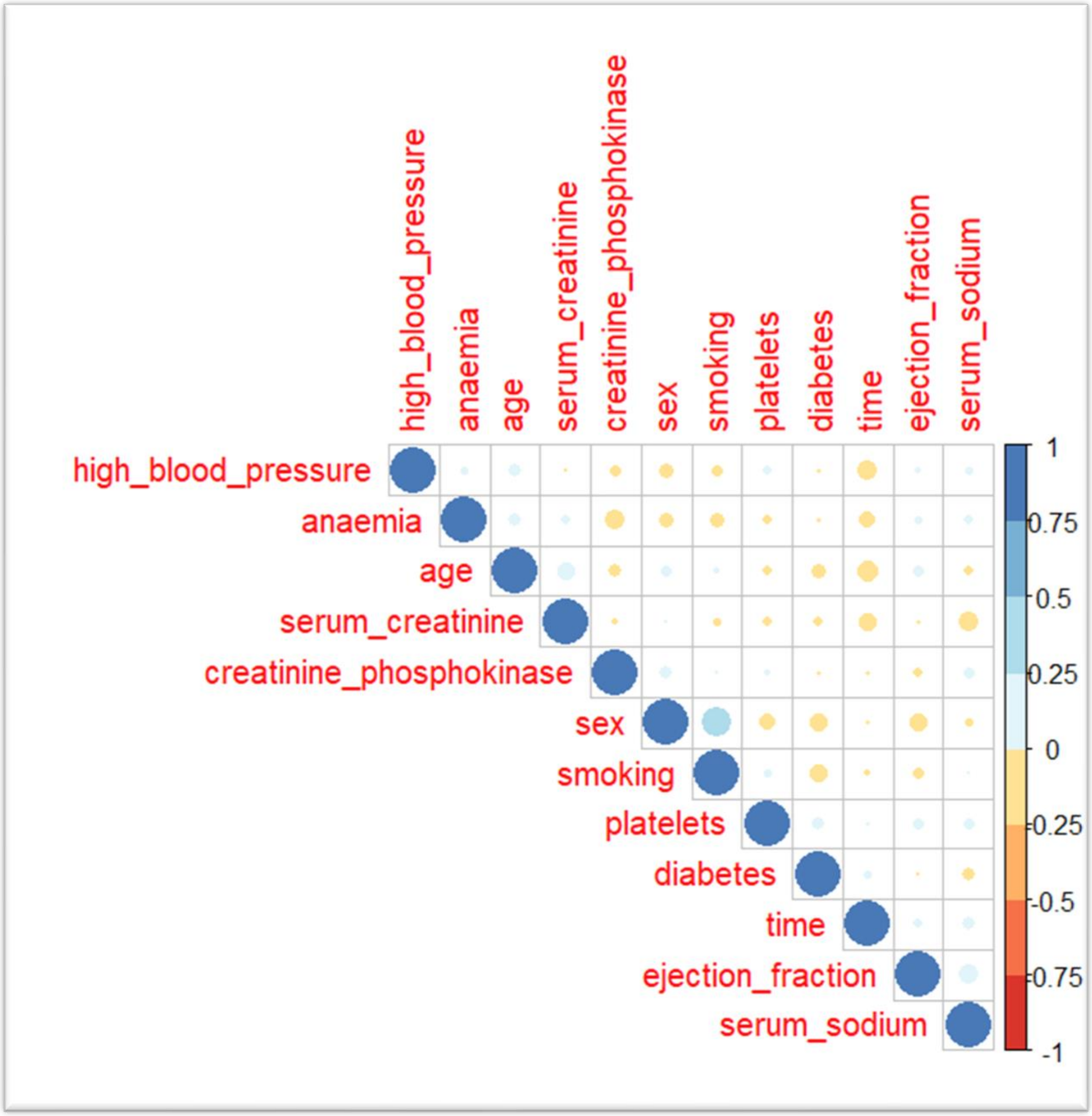
```
#convert dependent variable (Death_event) to factor
data$DEATH_EVENT=as.factor(data$DEATH_EVENT)
```

Step 5 : Creating a corrplot to identify the relationship between the independent variables and the target variable. In order to create the corrplot, the target variable (DEATH_EVENT) should be removed from the dataset.



```
#Removing the target variable from the dataset
data_cor=cor(data[,-13])
data_cor

#Creating the corrplot
corrplot(data_cor, type="upper", order="hclust", col=brewer.pal(n=8,name="RdYlBu"))
```

The corrplot generated is depicted below.

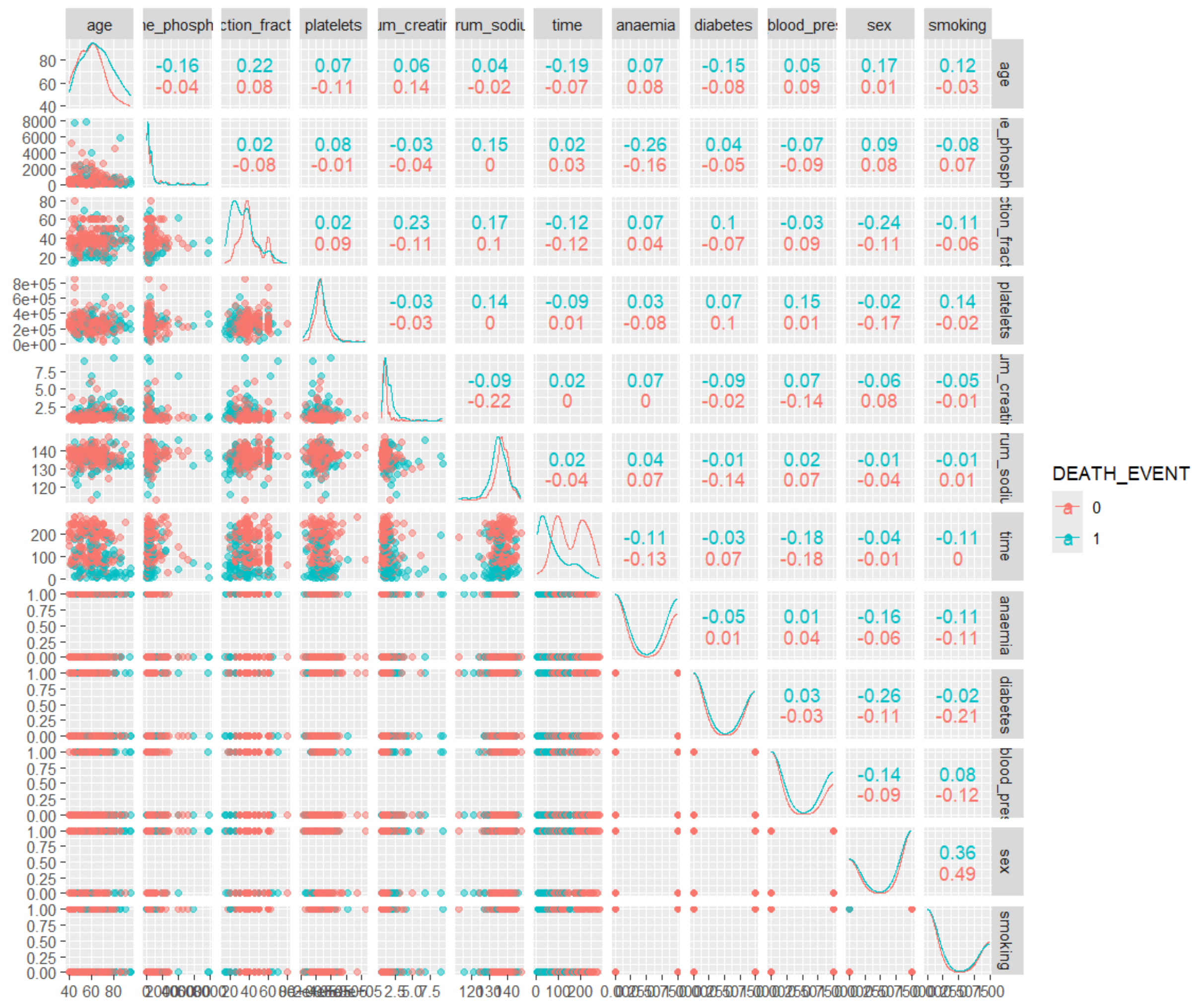


Step 6 : Creating the ggplot

A function named `ggpairs()` in the `GGally` package in R creates a matrix of scatterplots and histograms for each pair of variables in a dataset. This plot matrix can be used to investigate correlations between various variables in a dataset, spot trends, and find any outliers or anomalous observations.
Here are some typical applications for `ggpairs()`:

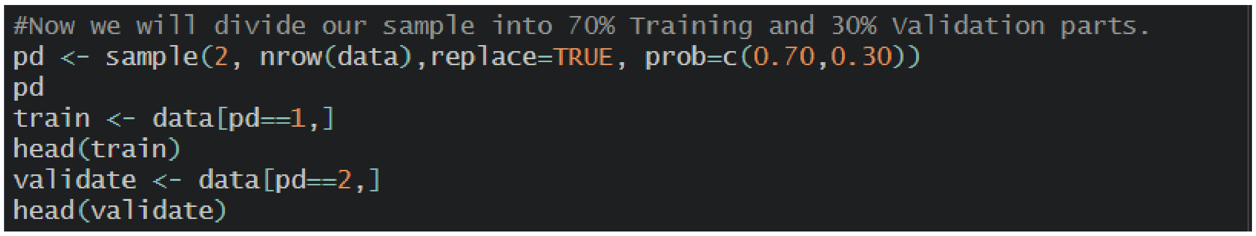
1. Investigative Information Analysis
2. Analysis of Correlations
3. Identifying Outliers
4. Modifiable Conversion
5. Model Construction and Feature Determination

All things considered, `ggpairs() is a helpful tool for learning about the structure of your dataset, comprehending the relationships between variables, and drawing well-informed conclusions from tasks involving data analysis and modeling.

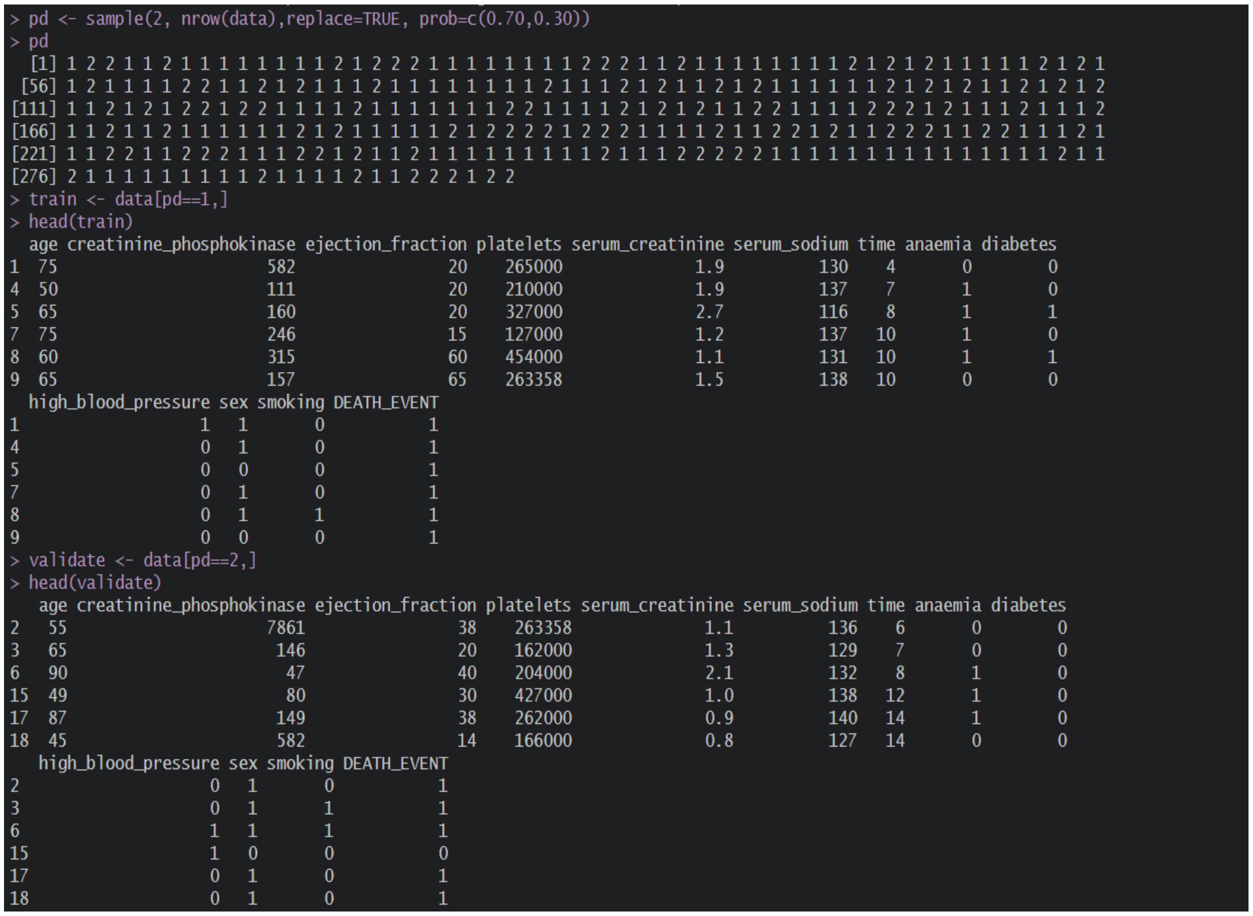


Step 7 : Testing and training data split

We have divided the dataset into two sections, such as the training model and the testing model, in order to proceed with building the logistic regression model. 30 % of the dataset was used for testing and 70% of it was used for training the models.



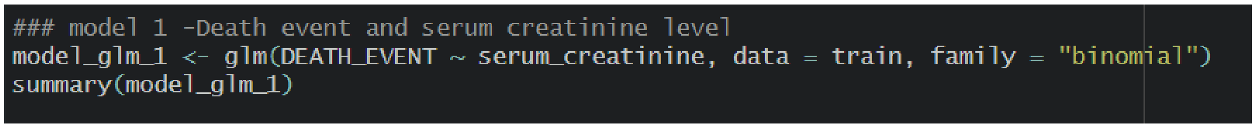
```
#Now we will divide our sample into 70% Training and 30% Validation parts.
pd <- sample(2, nrow(data),replace=TRUE, prob=c(0.70,0.30))
pd
train <- data[pd==1,]
head(train)
validate <- data[pd==2,]
head(validate)
```



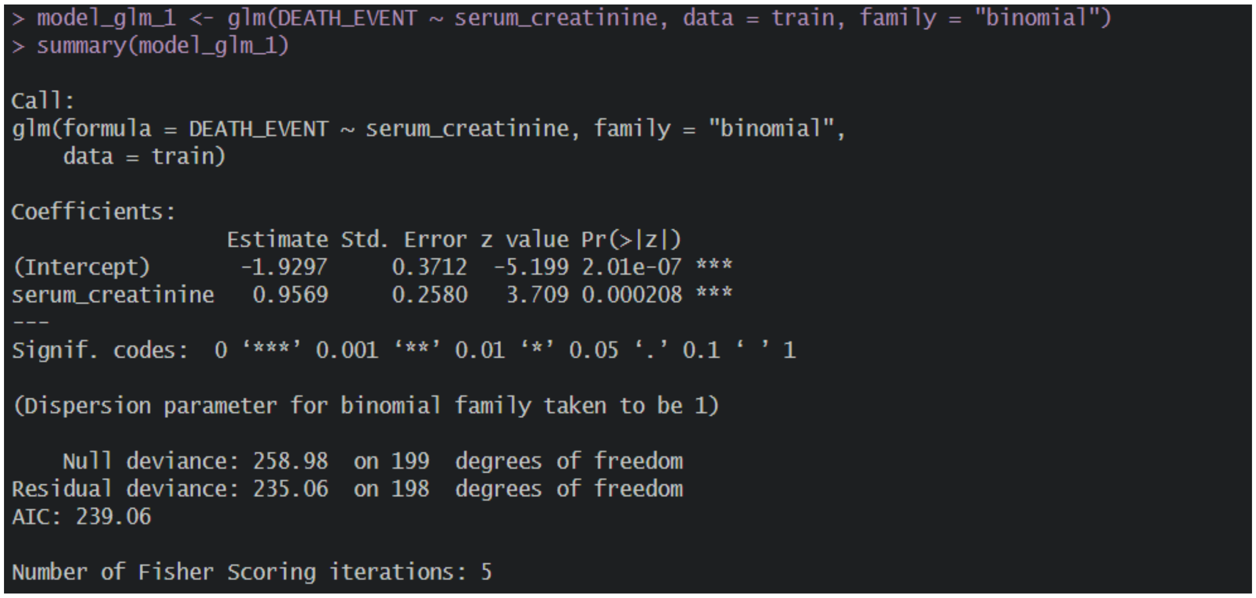
```
> pd <- sample(2, nrow(data),replace=TRUE, prob=c(0.70,0.30))
> pd
  [1] 1 2 2 1 1 2 1 1 1 1 1 1 1 1 2 1 2 2 2 1 1 1 1 1 1 1 1 2 2 2 1 1 2 1 1 1 1 1 1 1 1 2 1 2 1 2 1 1 1 1 1 2 1 2 1
 [56] 1 2 1 1 1 1 2 2 1 1 2 1 2 1 1 1 2 1 1 1 1 1 1 1 1 2 1 1 1 2 1 2 1 1 2 1 2 1 1 1 1 1 2 1 2 1 2 1 1 2 1 2 1 2
[111] 1 1 2 1 2 1 2 2 1 2 2 1 1 1 1 2 1 1 1 1 1 1 1 2 2 1 1 1 1 2 1 2 1 2 1 1 2 2 1 1 1 1 2 2 2 1 2 1 1 1 2 1 1 1 2
[166] 1 1 2 1 1 2 1 1 1 1 1 1 2 1 2 1 1 1 1 1 2 1 2 2 2 2 1 2 2 2 1 1 1 1 2 1 1 2 2 1 2 1 1 2 2 2 1 1 2 2 1 1 1 2 1
[221] 1 1 2 2 1 1 2 2 2 1 1 1 2 2 1 2 1 1 2 1 1 1 1 1 1 1 1 1 2 1 1 1 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1
[276] 2 1 1 1 1 1 1 1 1 1 2 1 1 1 1 2 1 1 2 2 2 1 2 2
> train <- data[pd==1,]
> head(train)
  age creatinine_phosphokinase ejection_fraction platelets serum_creatinine serum_sodium time anaemia diabetes
1  75                      582                20    265000              1.9          130    4       0        0
4  50                      111                20    210000              1.9          137    7       1        0
5  65                      160                20    327000              2.7          116    8       1        1
7  75                      246                15    127000              1.2          137   10       1        0
8  60                      315                60    454000              1.1          131   10       1        1
9  65                      157                65    263358              1.5          138   10       0        0
  high_blood_pressure sex smoking DEATH_EVENT
1                   1   1       0           1
4                   0   1       0           1
5                   0   0       0           1
7                   0   1       0           1
8                   0   1       1           1
9                   0   0       0           1
> validate <- data[pd==2,]
> head(validate)
   age creatinine_phosphokinase ejection_fraction platelets serum_creatinine serum_sodium time anaemia diabetes
2   55                     7861                38    263358              1.1          136    6       0        0
3   65                      146                20    162000              1.3          129    7       0        0
6   90                       47                40    204000              2.1          132    8       1        0
15  49                       80                30    427000              1.0          138   12       1        0
17  87                      149                38    262000              0.9          140   14       1        0
18  45                      582                14    166000              0.8          127   14       0        0
   high_blood_pressure sex smoking DEATH_EVENT
2                    0   1       0           1
3                    0   1       1           1
6                    1   1       1           1
15                   1   0       0           0
17                   0   1       0           1
18                   0   1       0           1
```

MODEL 1

Step 8 : Create a model for the logistic regression.

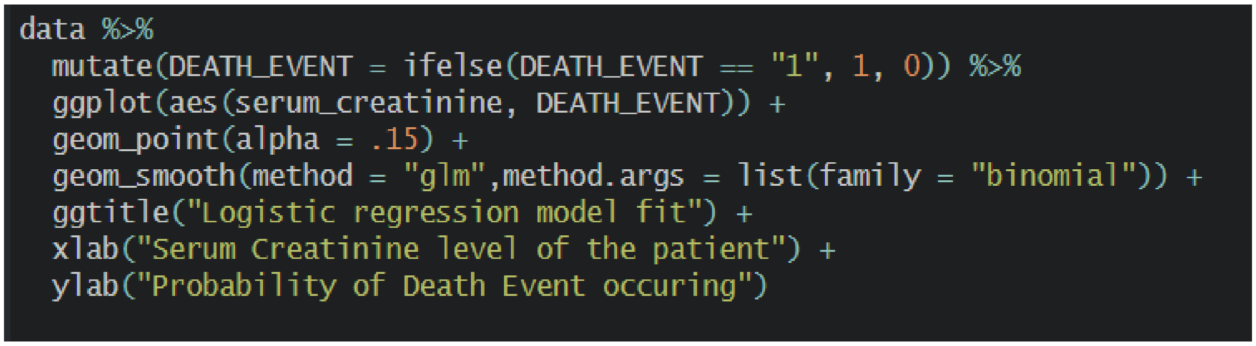


```
### model 1 -Death event and serum creatinine level
model_glm_1 <- glm(DEATH_EVENT ~ serum_creatinine, data = train, family = "binomial")
summary(model_glm_1)
```



```
> model_glm_1 <- glm(DEATH_EVENT ~ serum_creatinine, data = train, family = "binomial")
> summary(model_glm_1)

Call:
glm(formula = DEATH_EVENT ~ serum_creatinine, family = "binomial",
    data = train)

Coefficients:
                 Estimate Std. Error z value Pr(>|z|)    
(Intercept)       -1.9297     0.3712  -5.199 2.01e-07 ***
serum_creatinine   0.9569     0.2580   3.709 0.000208 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 258.98  on 199  degrees of freedom
Residual deviance: 235.06  on 198  degrees of freedom
AIC: 239.06

Number of Fisher Scoring iterations: 5
```

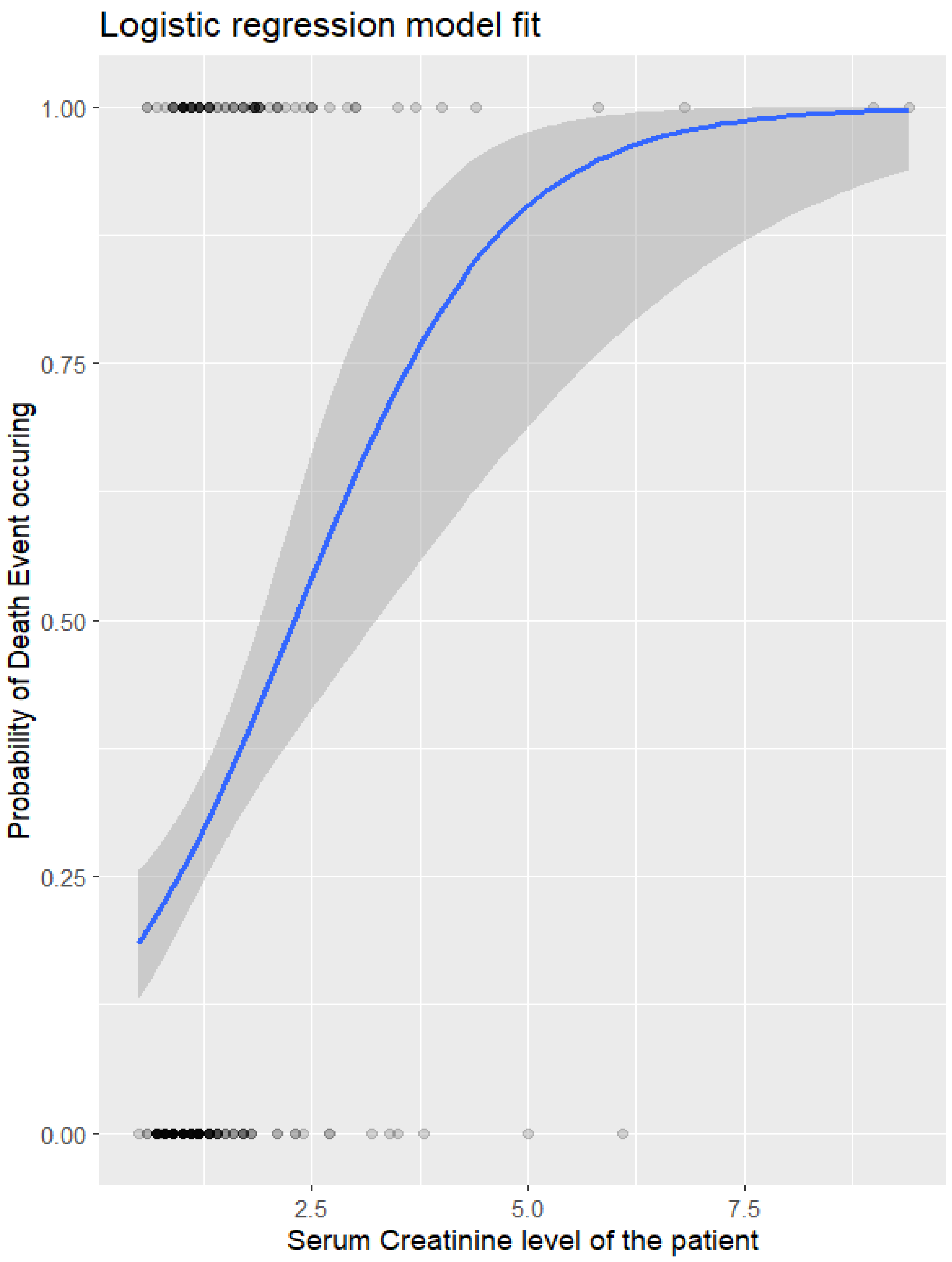
Step 9 : Plot the regression model



```
data %>%
  mutate(DEATH_EVENT = ifelse(DEATH_EVENT == "1", 1, 0)) %>%
  ggplot(aes(serum_creatinine, DEATH_EVENT)) +
  geom_point(alpha = .15) +
  geom_smooth(method = "glm",method.args = list(family = "binomial")) +
  ggtitle("Logistic regression model fit") +
  xlab("Serum Creatinine level of the patient") +
  ylab("Probability of Death Event occuring")
```

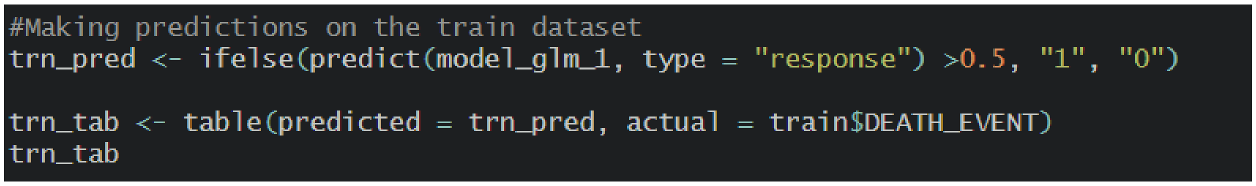
The output contains a summary of a R software fit for a binomial generalized linear model (glm). The model looks at the relationship (correlation) between a predictor variable called serum_creatinine and a binary outcome called {DEATH_EVENT}.

The important findings to consider are:
* **Coefficients :** For {serum_creatinine}, the coefficient is 0.96 and the intercept is -1.93.
* **p-values:** There are five Fisher Scoring iterations, indicating that the intercept and {serum_creatinine} variables are unlikely to be the result of chance because their p-values are statistically significant (less than 0.05). This is the quantity of iterations needed to determine the greatest probability estimates for the parameters of the model.

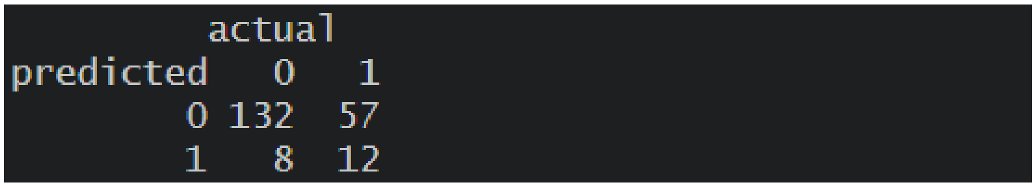


The higher the serum creatinine level the higher the death rate of the patients.

Step 10 : Making predictions for the trained dataset.

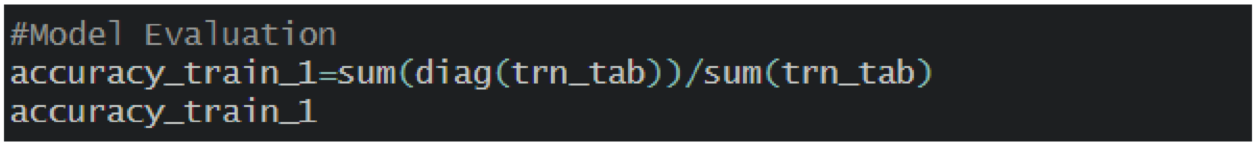


```
#Making predictions on the train dataset
trn_pred <- ifelse(predict(model_glm_1, type = "response") >0.5, "1", "0")

trn_tab <- table(predicted = trn_pred, actual = train$DEATH_EVENT)
trn_tab
```

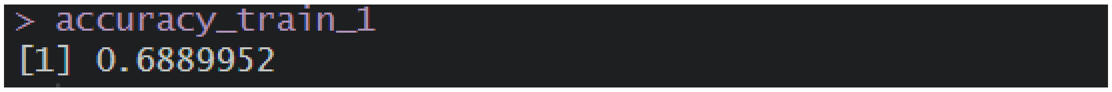


```
          actual
predicted    0   1
        0 132  57
        1   8  12
```

Step 11 : Model Evaluation



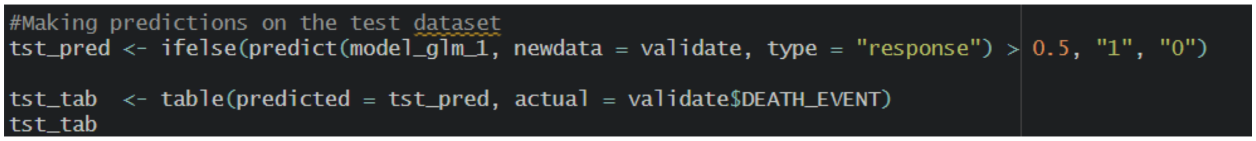
```
#Model Evaluation
accuracy_train_1=sum(diag(trn_tab))/sum(trn_tab)
accuracy_train_1
```



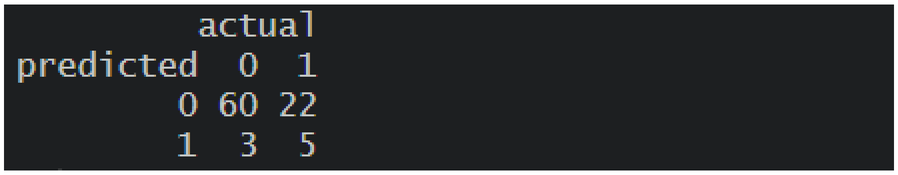
```
> accuracy_train_1
[1] 0.6889952
```

The accuracy of this is shown as 68.9%.

Step 12 : Making predictions for the test dataset

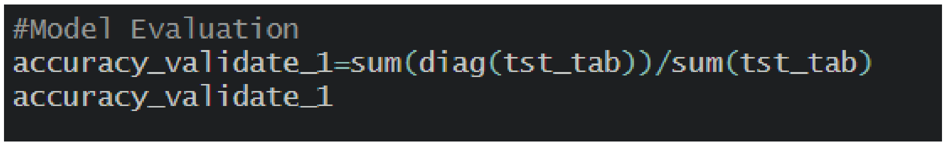


```
#Making predictions on the test dataset
tst_pred <- ifelse(predict(model_glm_1, newdata = validate, type = "response") > 0.5, "1", "0")

tst_tab  <- table(predicted = tst_pred, actual = validate$DEATH_EVENT)
tst_tab
```

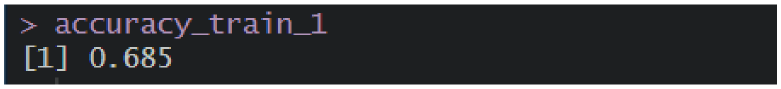


```
          actual
predicted  0  1
        0 60 22
        1  3  5
```

Step 13 : Assessing test model's accuracy level

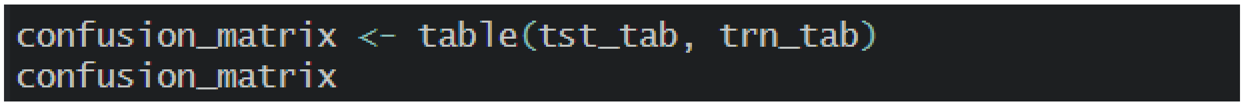


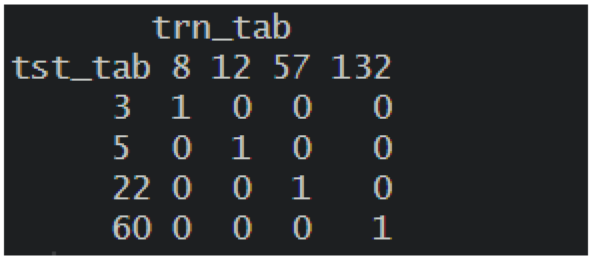
```
#Model Evaluation
accuracy_validate_1=sum(diag(tst_tab))/sum(tst_tab)
accuracy_validate_1
```


```
> accuracy_train_1
[1] 0.685
```

As shown in the above results, the accuracy is 68.5%.

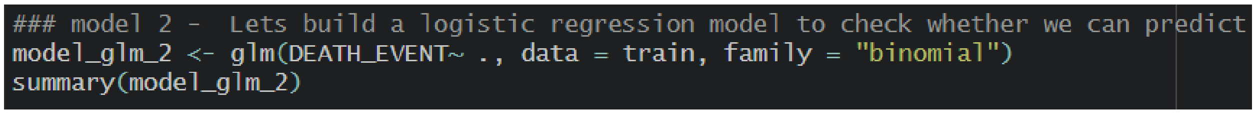
Step 14 : Evaluating the trained model using the confusion matrix.

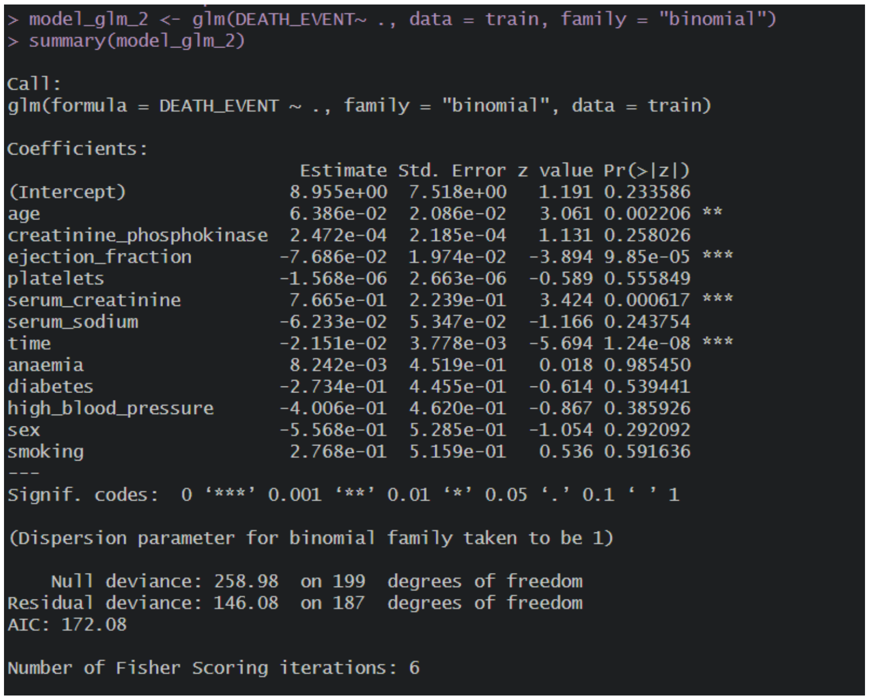

```
confusion_matrix <- table(tst_tab, trn_tab)
confusion_matrix
```


```
        trn_tab
tst_tab 8 12 57 132
     3  1  0  0   0
     5  0  1  0   0
     22 0  0  1   0
     60 0  0  0   1
```

MODEL 2

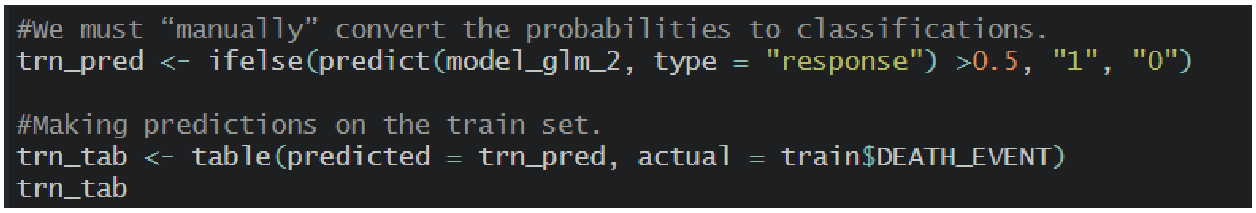
Step 15 : Create a second model to predict the probability of the DEATH_EVENT occurring based on other independent variables.


```
### model 2 -  Lets build a logistic regression model to check whether we can predict
model_glm_2 <- glm(DEATH_EVENT~ ., data = train, family = "binomial")
summary(model_glm_2)
```

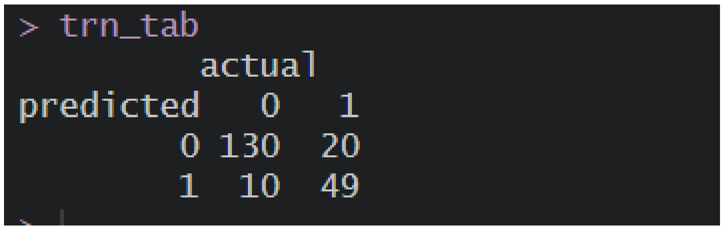

```
> model_glm_2 <- glm(DEATH_EVENT~ ., data = train, family = "binomial")
> summary(model_glm_2)

Call:
glm(formula = DEATH_EVENT ~ ., family = "binomial", data = train)

Coefficients:
                          Estimate Std. Error z value Pr(>|z|)    
(Intercept)              8.955e+00  7.518e+00   1.191 0.233586    
age                      6.386e-02  2.086e-02   3.061 0.002206 ** 
creatinine_phosphokinase  2.472e-04  2.185e-04   1.131 0.258026    
ejection_fraction       -7.686e-02  1.974e-02  -3.894 9.85e-05 ***
platelets               -1.568e-06  2.663e-06  -0.589 0.555849    
serum_creatinine         7.665e-01  2.239e-01   3.424 0.000617 ***
serum_sodium            -6.233e-02  5.347e-02  -1.166 0.243754    
time                    -2.151e-02  3.778e-03  -5.694 1.24e-08 ***
anaemia                  8.242e-03  4.519e-01   0.018 0.985450    
diabetes                -2.734e-01  4.455e-01  -0.614 0.539441    
high_blood_pressure     -4.006e-01  4.620e-01  -0.867 0.385926    
sex                     -5.568e-01  5.285e-01  -1.054 0.292092    
smoking                  2.768e-01  5.159e-01   0.536 0.591636    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 258.98  on 199  degrees of freedom
Residual deviance: 146.08  on 187  degrees of freedom
AIC: 172.08

Number of Fisher Scoring iterations: 6
```

Step 16 : Making predictions using logistic regression models.

Retrieve the predicted probabilities generated by the logistic regression model and compare each predicited probability to threshold value = 0.5. Next, assign a class label of "1" if the predicted probability is greater than or equal to 0.5, or a class label of "0" if it is less than 0.5.

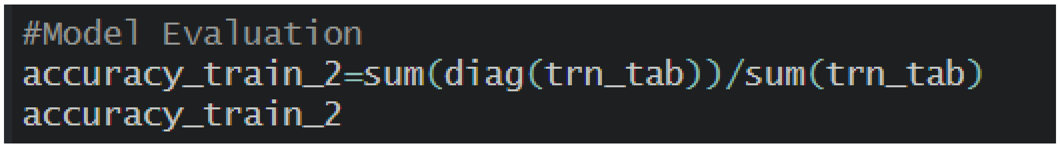


```
#We must "manually" convert the probabilities to classifications.
trn_pred <- ifelse(predict(model_glm_2, type = "response") >0.5, "1", "0")

#Making predictions on the train set.
trn_tab <- table(predicted = trn_pred, actual = train$DEATH_EVENT)
trn_tab
```

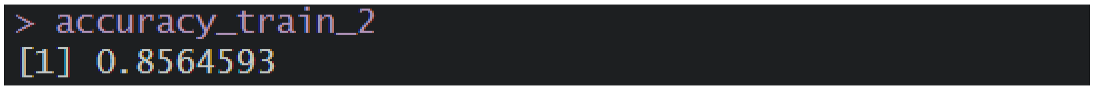


```
> trn_tab
         actual
predicted   0   1
        0 130  20
        1  10  49
```

Step 17 : Evaluate the trained model.



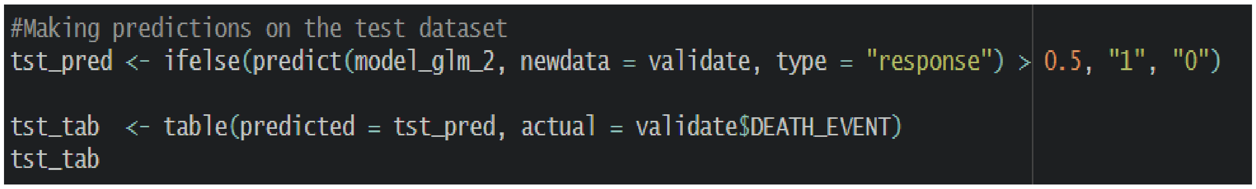
```
#Model Evaluation
accuracy_train_2=sum(diag(trn_tab))/sum(trn_tab)
accuracy_train_2
```



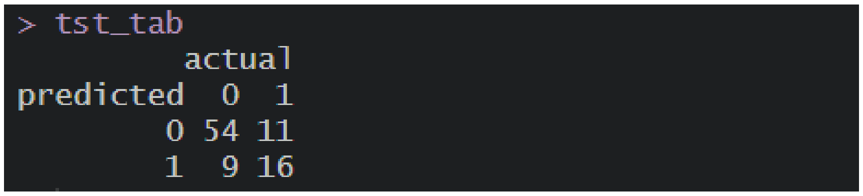
```
> accuracy_train_2
[1] 0.8564593
```

Accordingly, the accuracy of this model is 85.65%.

Step 18 : Making prediction for the trained dataset.

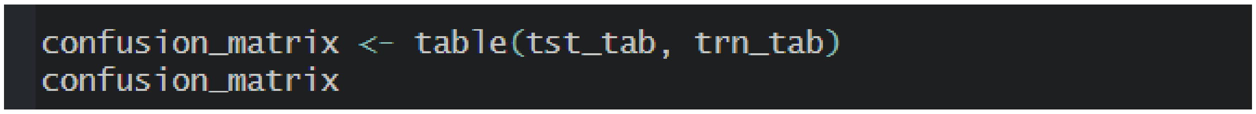


```
#Making predictions on the test dataset
tst_pred <- ifelse(predict(model_glm_2, newdata = validate, type = "response") > 0.5, "1", "0")

tst_tab  <- table(predicted = tst_pred, actual = validate$DEATH_EVENT)
tst_tab
```

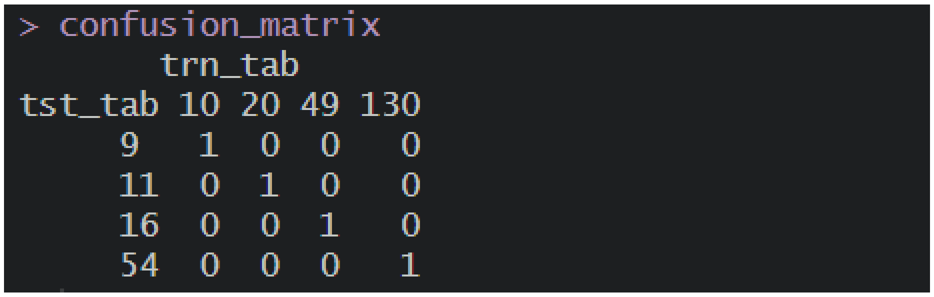


```
> tst_tab
         actual
predicted  0  1
        0 54 11
        1  9 16
```

Step 19 : Evaluating the dataset using the confusion matrix.

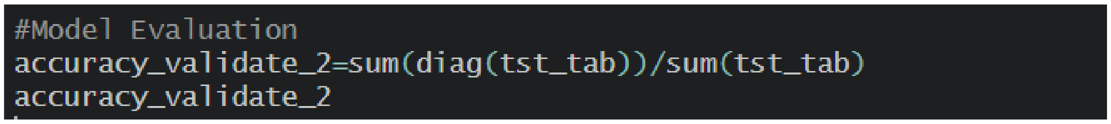


```
confusion_matrix <- table(tst_tab, trn_tab)
confusion_matrix
```

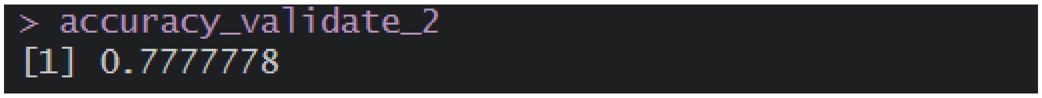


```
> confusion_matrix
       trn_tab
tst_tab 10 20 49 130
     9   1  0  0   0
     11  0  1  0   0
     16  0  0  1   0
     54  0  0  0   1
```

Step 20 : Assessing the trained model's accuracy level.



```
#Model Evaluation
accuracy_validate_2=sum(diag(tst_tab))/sum(tst_tab)
accuracy_validate_2
```



```
> accuracy_validate_2
[1] 0.7777778
```

Respectively, the accuracy of the model is 77.78%.

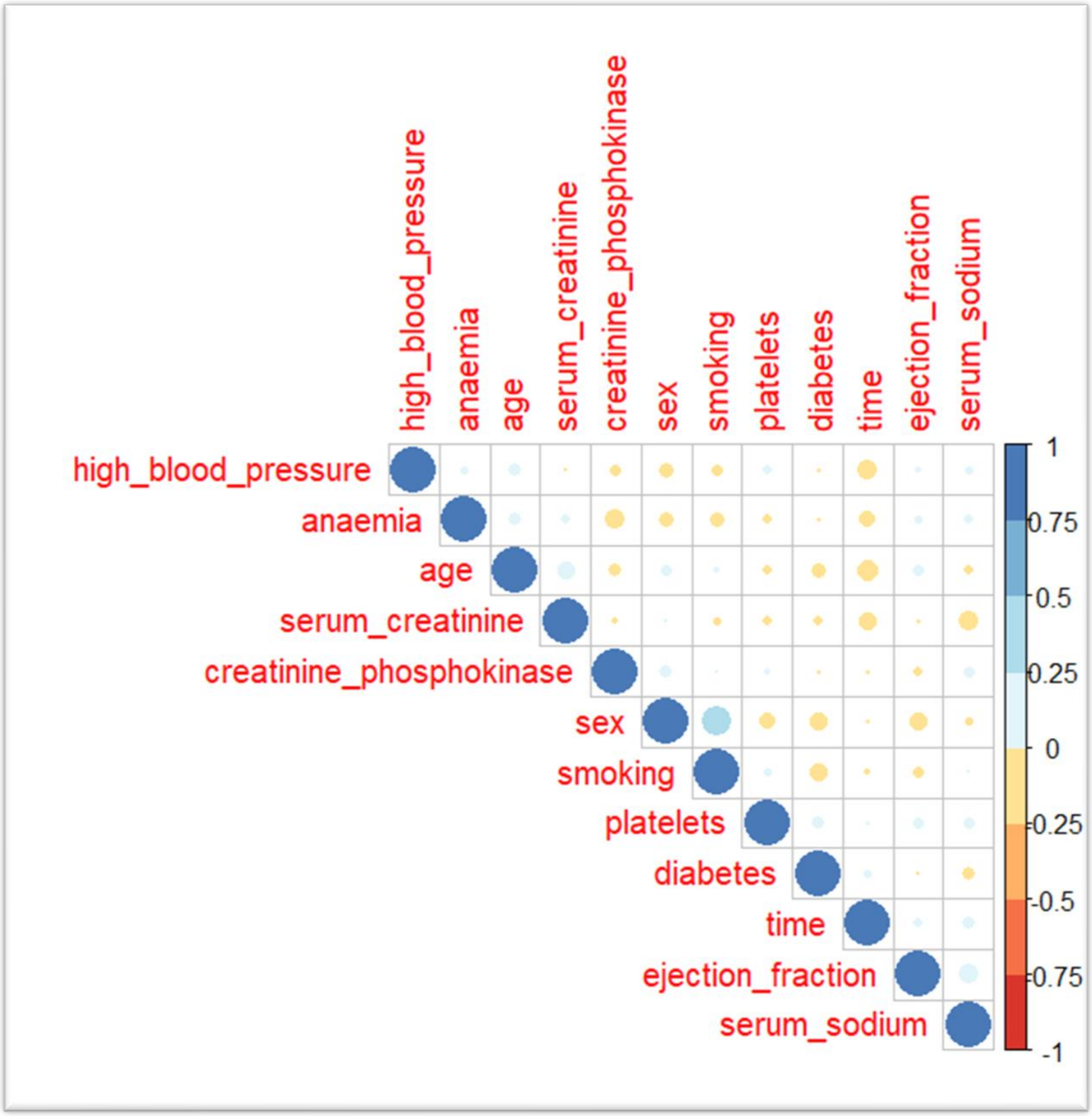
1.6. RESULTS ANALYSIS AND DISCUSSION

We will now proceed to review the analysis results derived from logistic regression and data mining techniques, followed by visualization using R programming. Our focus will be on summarizing the insights gained from the dataset analysis.

Result 1 : Corrplot

It shows the impact of various symptoms on the target variable (DEATH_EVENT). Each factor is listed on the left side of the chart, with factors with the greatest impact at the top.

The strength of the impact is measured by a number between -1 and 1. A positive number indicates a positive correlation with the target variable, while a negative number indicates a negative correlation. The rightmost side of the chart shows the impact visually, with a larger bar indicating a stronger impact.



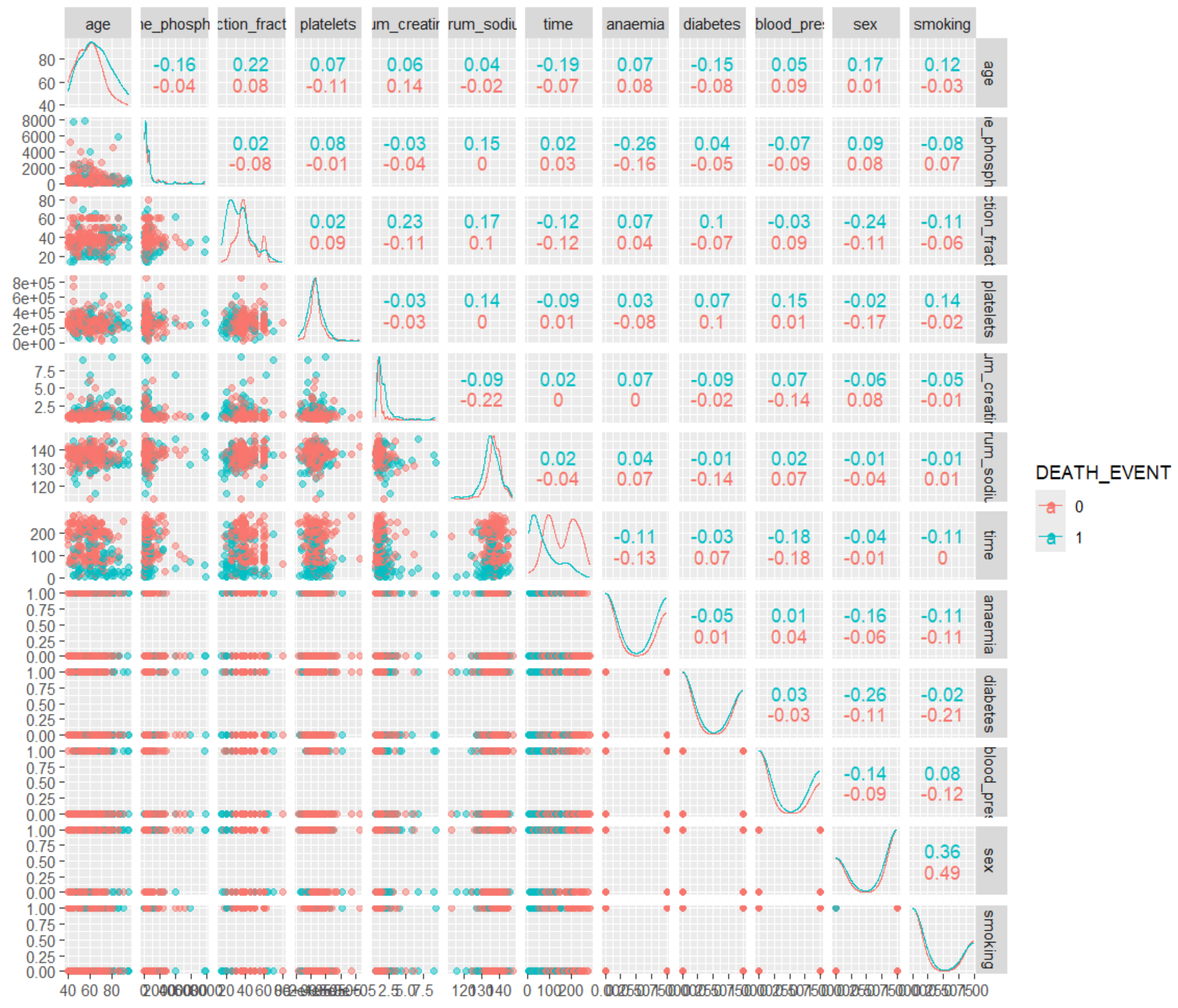
We have noticed the following after analyzing the chart.

- High blood pressure: This has the strongest positive correlation with the target variable according to the chart, with a value of 1.
- Ejection fraction: This also has a positive correlation, with a value of 0.75.
- Serum sodium: This has a negative correlation, with a value of -1.
- Time: This has a negative correlation, with a value of -0.5.
- Diabetes: This has a weak negative correlation, with a value close to -0.5.
- Age and Serum creatinine: These appear to have a positive correlation of around 0.5.
- Creatinine phosphokinase: This has a positive correlation of about 0.25.

- Sex, Smoking and Platelets: These factors appear to have a very weak impact, with values close to 0.

Overall, this means that higher values of these measures are associated with a higher value of the target variable. On the other hand, factors like serum sodium and time have a negative correlation, meaning higher values are associated with lower values of the target variable.

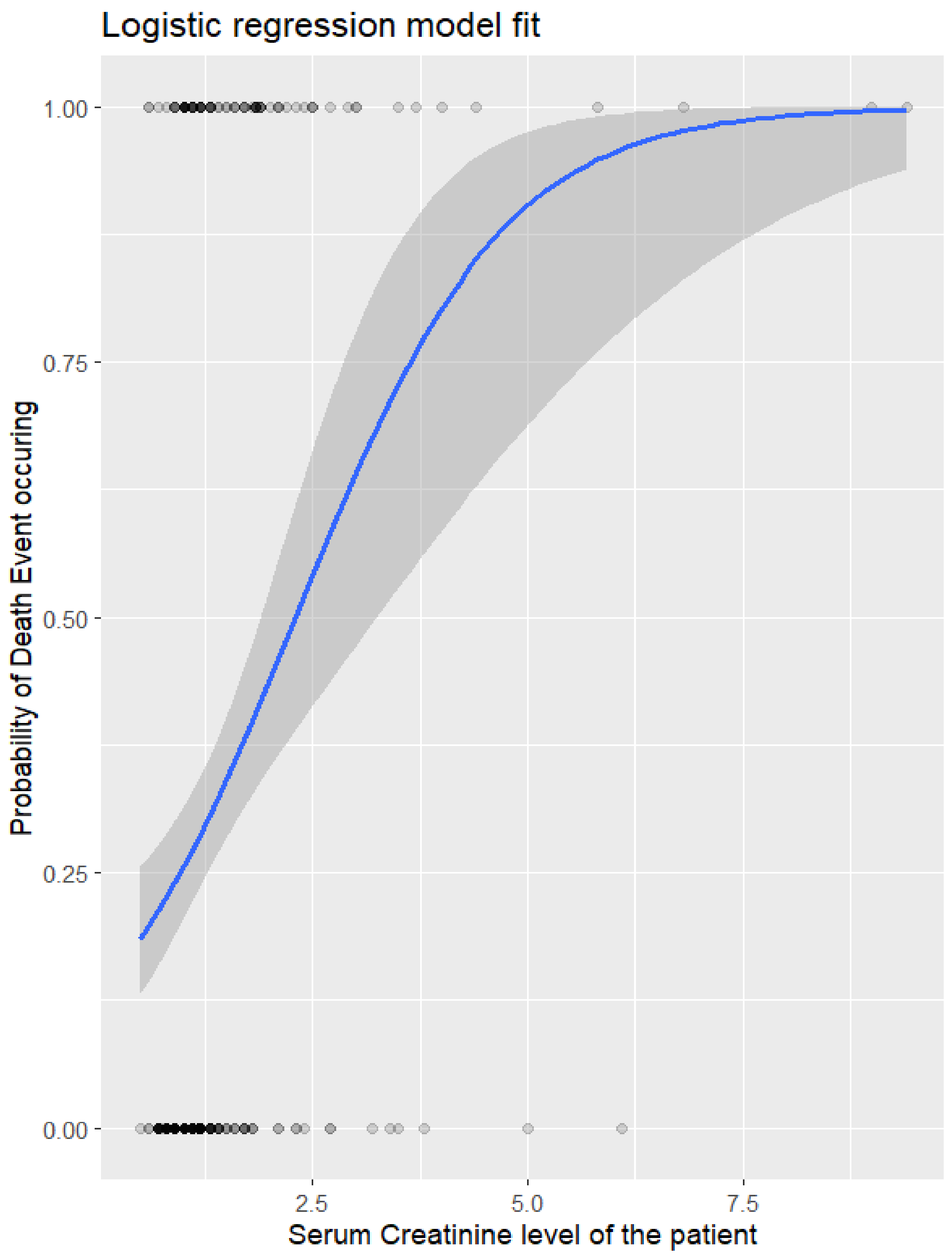
Result 2 : ggplot



The matrix scatter plot chart shows the results of multiple medical tests of people in different age groups under different variables.

It is important to note that this type of chart only shows average results. Individual results can vary depending on a person's health history and other factors.

Result 3 : Logistic Regression Model Fit Curve

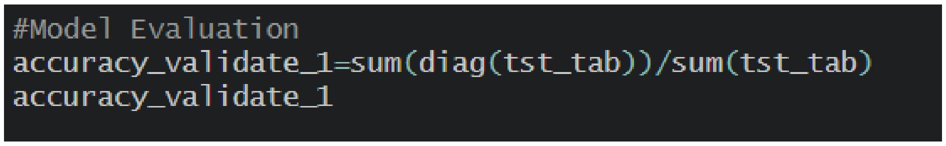


The above line graph is a depiction of a logistic regression model fit. It shows the relationship between serum creatinine level of the patient and the probability of a death event occurring.

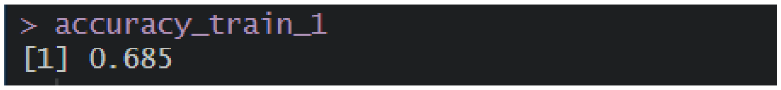
The patient's serum creatinine level is displayed on the graph's x-axis. The likelihood that a death event will occur is displayed on the y-axis. The graph's curve represents the expected likelihood that a death event would occur for a particular serum creatinine level.

For example, if a patient has a serum creatinine level of 2.5, the model predicts that the probability of a death event occurring is about 0.25. As the serum creatinine level increases, the predicted probability of a death event occurring also increases, which means the higher the serum creatinine level the higher the death rate of the patients.

Result 4 : Evaluation of the MODEL 1

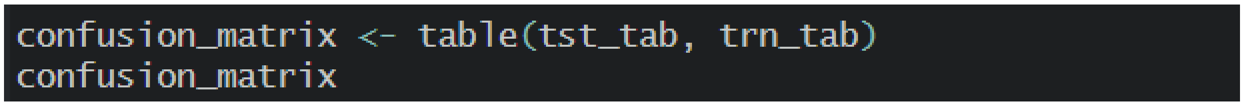


```
#Model Evaluation
accuracy_validate_1=sum(diag(tst_tab))/sum(tst_tab)
accuracy_validate_1
```

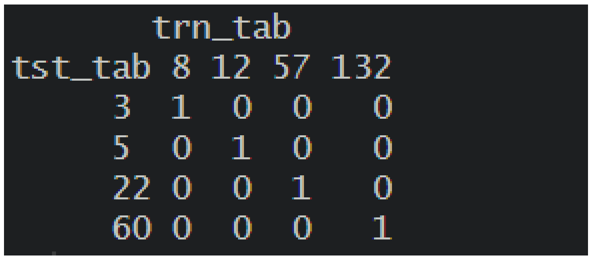


```
> accuracy_train_1
[1] 0.685
```

In machine learning, training accuracy refers to the proportion of correctly classified instances in the training data for the logistic regression model. The training data is the data used to fit the model. Here, a training accuracy of 68.5% means that the model correctly classified 68.5% of the instances in the training data.



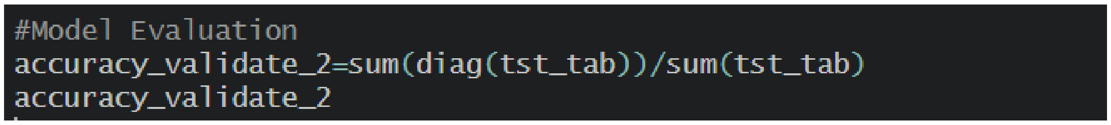
```
confusion_matrix <- table(tst_tab, trn_tab)
confusion_matrix
```



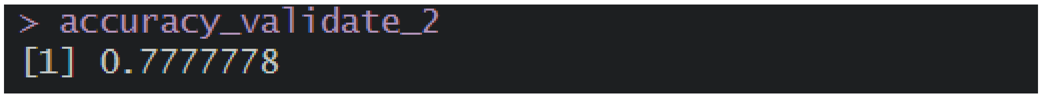
```
       trn_tab
tst_tab 8 12 57 132
     3  1  0  0   0
     5  0  1  0   0
     22 0  0  1   0
     60 0  0  0   1
```

The columns show the classes that the model predicted, while the rows show the actual classes of the data points. Overall, based on this confusion matrix, the model seems to perform well with a high number of correct predictions on the diagonal, but there is room for improvement in differentiating between the two classes.

Result 5 : Evaluation of MODEL 2

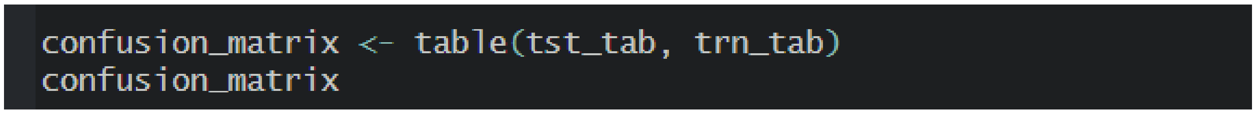


```
#Model Evaluation
accuracy_validate_2=sum(diag(tst_tab))/sum(tst_tab)
accuracy_validate_2
```

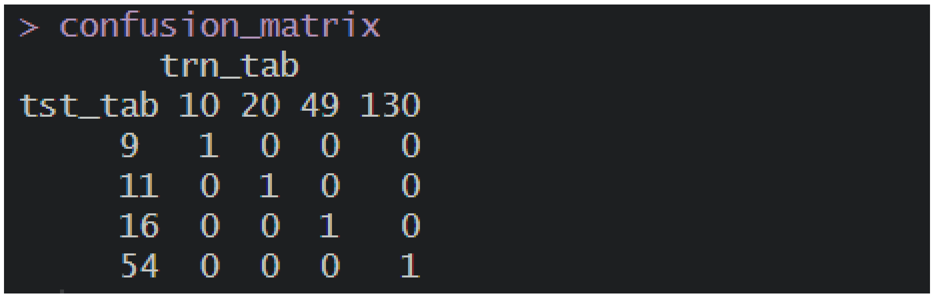


```
> accuracy_validate_2
[1] 0.7777778
```

In machine learning, training accuracy refers to the proportion of correctly classified instances in the testing data. The testing data is the data used to fit the model. Here, a testing accuracy of 77.78% means that the model correctly classified 77.78% of the instances in the testing data.



```
confusion_matrix <- table(tst_tab, trn_tab)
confusion_matrix
```

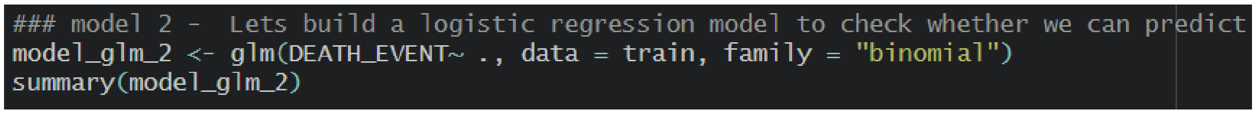


```
> confusion_matrix
       trn_tab
tst_tab 10 20 49 130
      9  1  0  0   0
     11  0  1  0   0
     16  0  0  1   0
     54  0  0  0   1
```

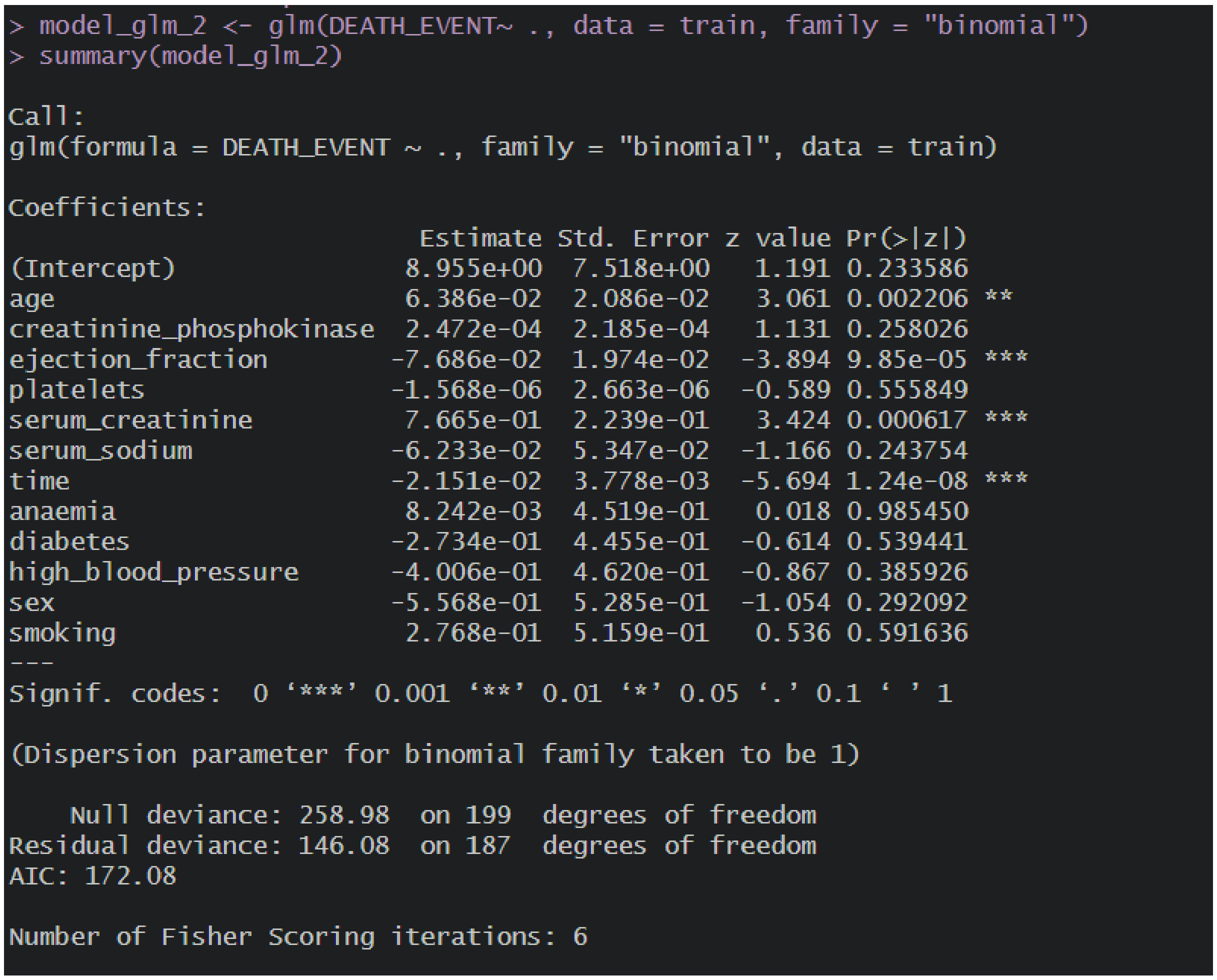
The columns show the classes that the model predicted, while the rows show the actual classes of the data points. Overall, based on this confusion matrix, suggests that the model seems to be over fitting the training data and generalizes poorly to test data. We can use techniques to regularize the model to improve generalizability.

Result 6 : Evaluating the Logistic Regression Model

For further understanding we must evaluate the logistic regression model obtained in order get better insights.



```
### model 2 -  Lets build a logistic regression model to check whether we can predict
model_glm_2 <- glm(DEATH_EVENT~ ., data = train, family = "binomial")
summary(model_glm_2)
```



```
> model_glm_2 <- glm(DEATH_EVENT~ ., data = train, family = "binomial")
> summary(model_glm_2)

Call:
glm(formula = DEATH_EVENT ~ ., family = "binomial", data = train)

Coefficients:
                           Estimate Std. Error z value Pr(>|z|)    
(Intercept)               8.955e+00  7.518e+00   1.191 0.233586    
age                       6.386e-02  2.086e-02   3.061 0.002206 ** 
creatinine_phosphokinase  2.472e-04  2.185e-04   1.131 0.258026    
ejection_fraction        -7.686e-02  1.974e-02  -3.894 9.85e-05 ***
platelets                -1.568e-06  2.663e-06  -0.589 0.555849    
serum_creatinine          7.665e-01  2.239e-01   3.424 0.000617 ***
serum_sodium             -6.233e-02  5.347e-02  -1.166 0.243754    
time                     -2.151e-02  3.778e-03  -5.694 1.24e-08 ***
anaemia                   8.242e-03  4.519e-01   0.018 0.985450    
diabetes                 -2.734e-01  4.455e-01  -0.614 0.539441    
high_blood_pressure      -4.006e-01  4.620e-01  -0.867 0.385926    
sex                      -5.568e-01  5.285e-01  -1.054 0.292092    
smoking                   2.768e-01  5.159e-01   0.536 0.591636    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 258.98  on 199  degrees of freedom
Residual deviance: 146.08  on 187  degrees of freedom
AIC: 172.08

Number of Fisher Scoring iterations: 6
```

The statistical analysis output, likely generated from R and focusing on logistic regression, displays coefficients and p-values for various predictor variables in the model. The predictor variables listed include age, sex, creatinine_phosphokinase, ejection_fraction, and serum_creatinine. The coefficients indicate the strength and direction of relationships with the target variable (often death_event in logistic regression). For instance, a coefficient of 0.75 for serum_creatinine suggests a positive relationship with the target variable. P-values, displayed alongside the coefficients, indicate statistical significance. In this analysis, most predictor variables have p-values below 0.05, indicating significant relationships with the target variable, except for anaemia and diabetes. Overall, the results suggest that factors like age, high serum creatinine levels, and low ejection fraction are associated with an increased likelihood of a death event.

1.7. CONCLUSION

In conclusion, the logistic regression analysis conducted on the Heart Failure Clinical Records dataset from UCI provides valuable insights into the factors influencing mortality among heart failure patients. Our findings indicate that age, serum creatinine levels, and ejection fraction are significant predictors of death events in this population. The identified relationships highlight the importance of these clinical variables in assessing and managing the prognosis of heart failure patients. Moving forward, these insights can inform personalized treatment strategies and interventions aimed at improving patient outcomes and reducing mortality rates in individuals diagnosed with heart failure. Further research and application of logistic regression modeling in cardiovascular health can contribute to enhanced risk stratification and more effective patient care.

REFERENCES

- https://archive.ics.uci.edu/dataset/519/heart+failure+clinical+records
- https://youtu.be/C4N3_XJJ-jU?si=_3Hmz5aJU0rusLyi
- https://youtu.be/XycruVLySDg?si=0yOIjnhGM0Ep09hs
- Chicco, D. and 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). Available at: https://doi.org/10.1186/s12911-020-1023-5.

EMBRACING DATA TO ILLUMINATE PATHS IN HEART HEALTH JOURNEY

Task 2 : Creating an interactive plotly dashboard to the Heart Failure Clinical records dataset based on Logistic Regression Analysis and R - programming.

CONTENTS

TASK 02 :

CREATING AN INTERACTIVE PLOTLY DASHBOARD TO THE HEART FAILURE CLINICAL RECORDS DATASET BASED ON LOGISTIC REGRESSION ANALYSIS AND R - PROGRAMMING.

2.1. INTRODUCTION

Heart failure is a global health challenge causing around 17 million deaths annually. To improve prognosis and build treatment plans, advanced strategies are needed. The Heart Failure Clinical Records dataset, used in this study, provides extensive clinical data for logistic regression to forecast survival rates for heart failure patients. This data can identify risk factors and create models for early diagnosis and personalized treatment. The combination of this dataset and logistic regression presents an opportunity to advance cardiovascular health knowledge and patient care.

This task forecasts patients heart failure predictions using logistic regression techniques, identifying correlations and trends to develop mitigation methods through a dynamic dashboard in plotly.

This dashboard represents the variation of the patients with different symptoms by using interactive graphs, tables, and map. The charts, graphs and tables of the dashboard is obtained using R codes.

2.2. DATA SET

The same dataset that was used in Task 2 (Logistic Regression) was used for further visualizations in the dashboard.

2.3. DASHBOARD DESIGN

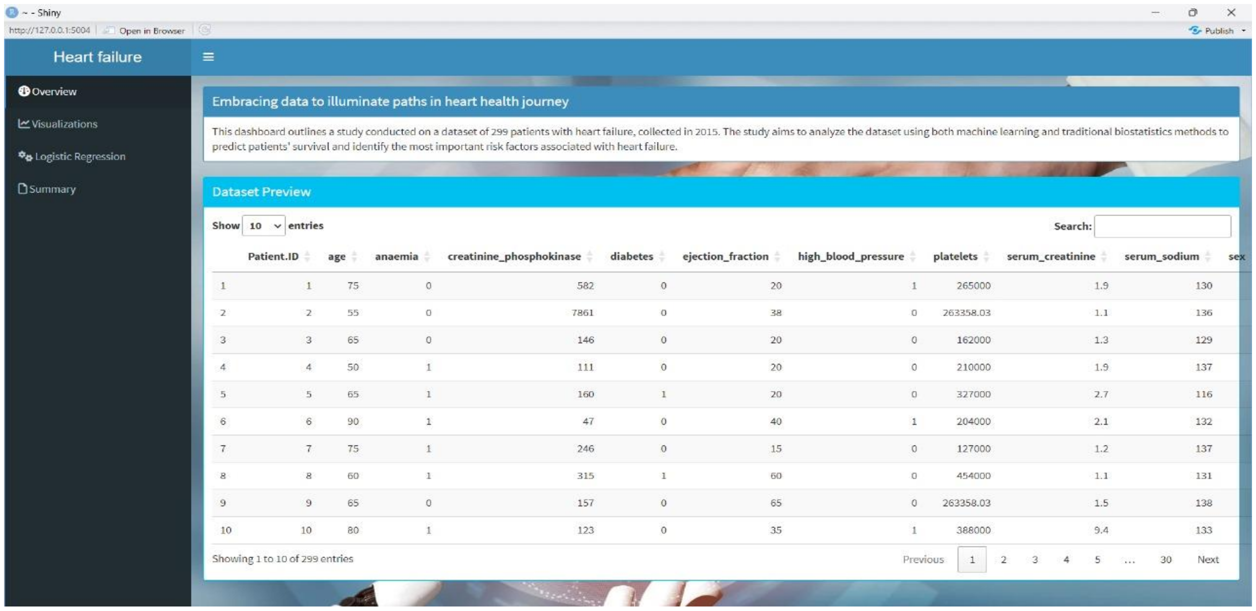
The interactive dashboard which was created, can be accessed by executing the R-program file attached along with the submission zip file document , as the created dashboard can only be published via commercial accounts.

Therefore, the screenshot images of the dashboard are attached below for reference.

The designed dashboard is shown below.

Overview :

This page includes the introduction of the dashboard and the dataset we used for the task.



Embracing data to illuminate paths in heart health journey

This dashboard outlines a study conducted on a dataset of 299 patients with heart failure, collected in 2015. The study aims to analyze the dataset using both machine learning and traditional biostatistics methods to predict patients' survival and identify the most important risk factors associated with heart failure.

Dataset Preview

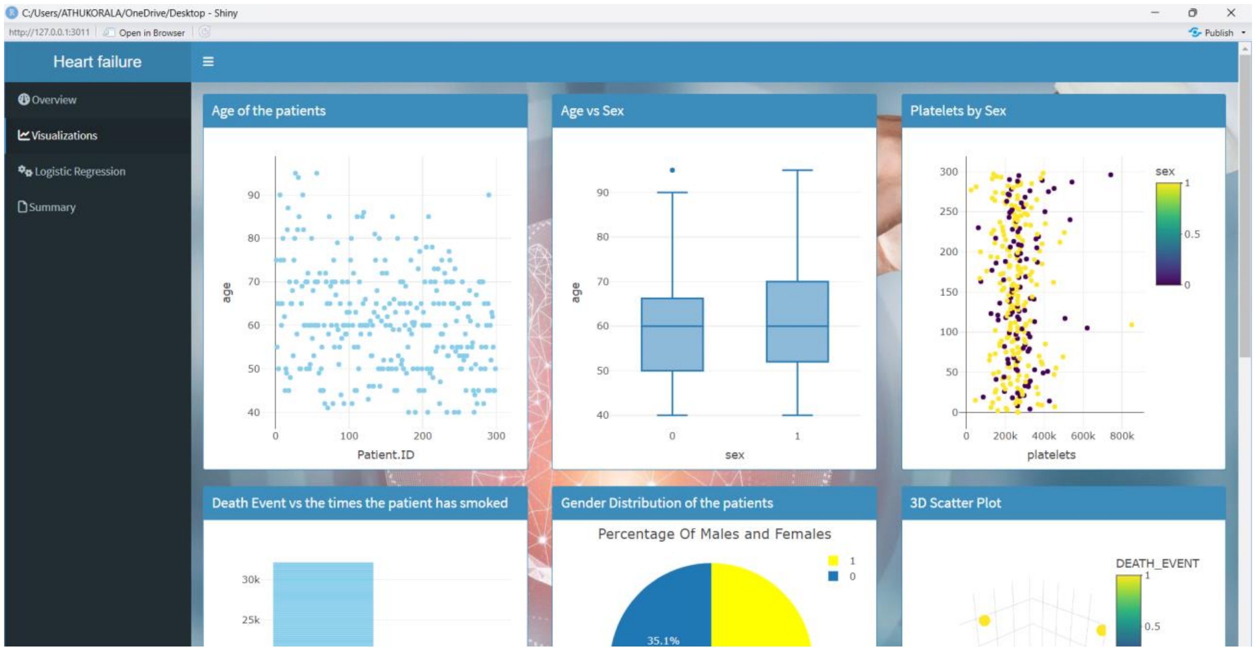
Show 10 entries Search:

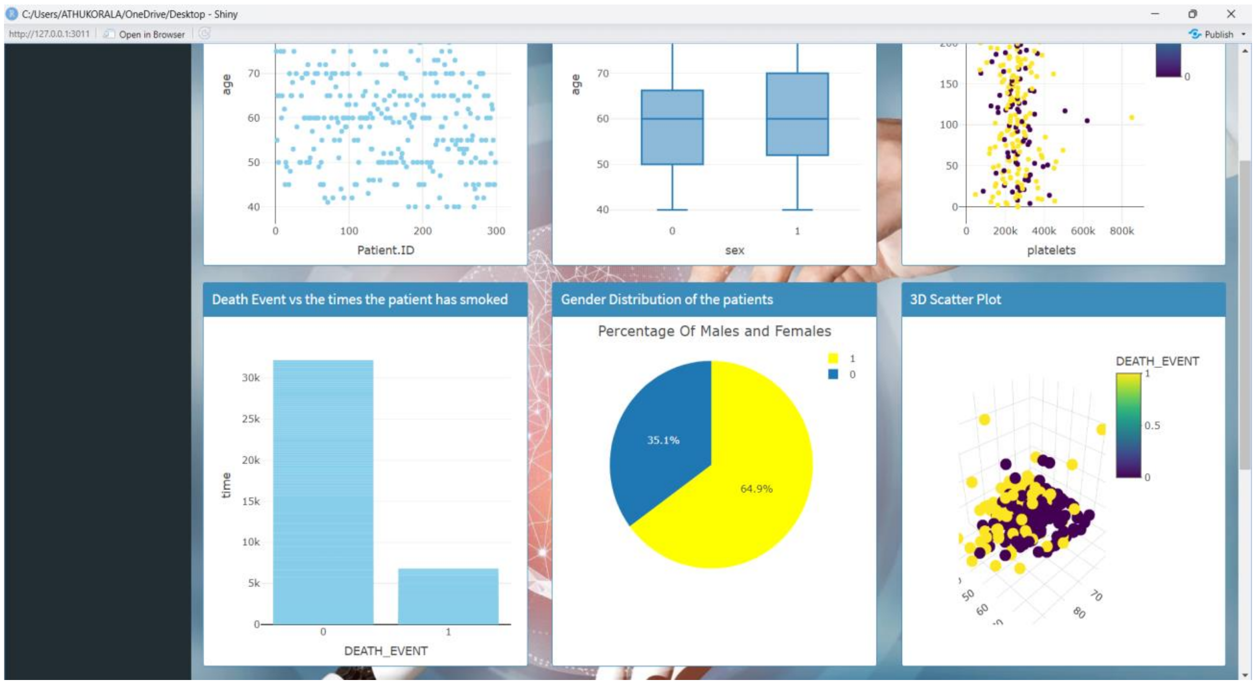
	Patient.ID	age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	platelets	serum_creatinine	serum_sodium	sex
1	1	75	0	582	0	20	1	265000	1.9	130	
2	2	55	0	7861	0	38	0	263358.03	1.1	136	
3	3	65	0	146	0	20	0	162000	1.3	129	
4	4	50	1	111	0	20	0	210000	1.9	137	
5	5	65	1	160	1	20	0	327000	2.7	116	
6	6	90	1	47	0	40	1	204000	2.1	132	
7	7	75	1	246	0	15	0	127000	1.2	137	
8	8	60	1	315	1	60	0	454000	1.1	131	
9	9	65	0	157	0	65	0	263358.03	1.5	138	
10	10	80	1	123	0	35	1	388000	9.4	133	

Showing 1 to 10 of 299 entries Previous 1 2 3 4 5 … 30 Next

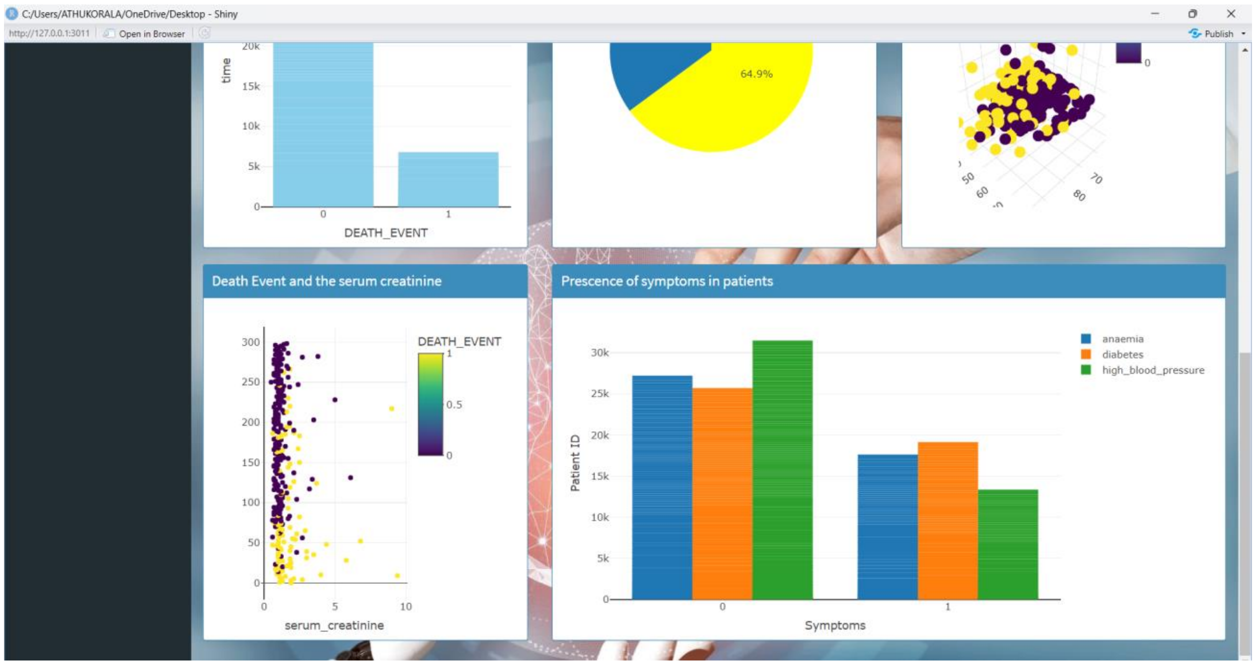
Visualizations :

This page includes all the visualizations related to the dataset.





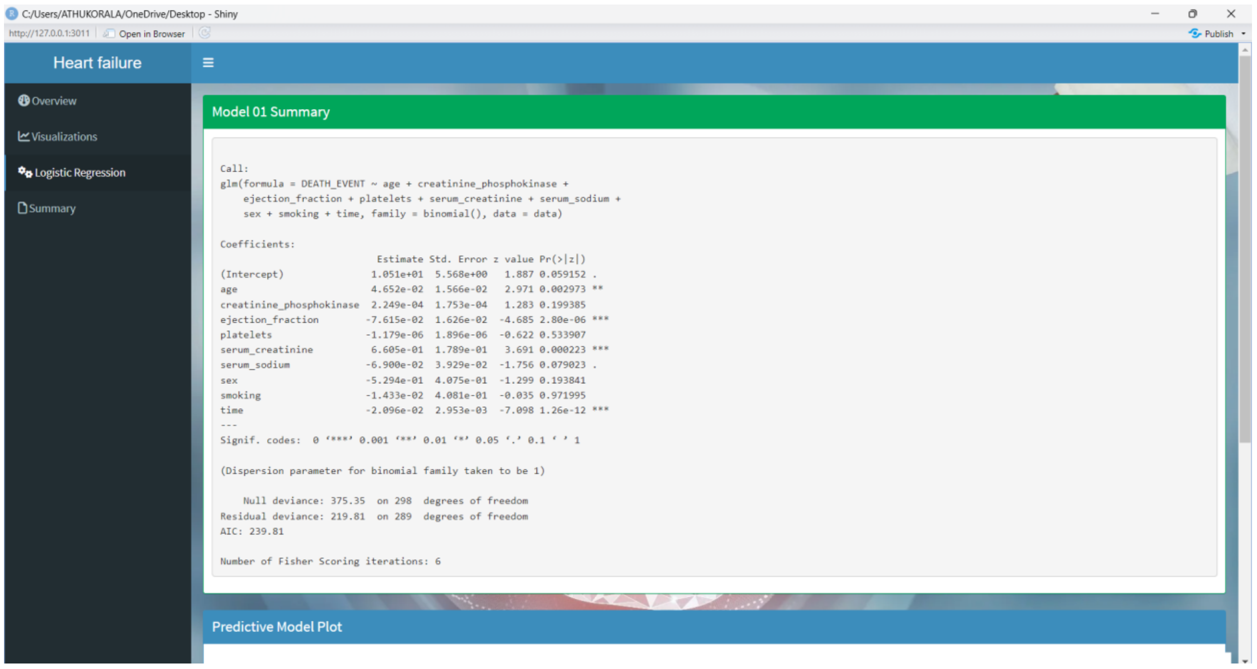
C:/Users/ATHUKORALA/OneDrive/Desktop - Shiny
http://127.0.0.1:3011
Open in Browser
Publish
Death Event vs the times the patient has smoked
Gender Distribution of the patients
Percentage Of Males and Females
3D Scatter Plot
age
Patient.ID
sex
platelets
time
DEATH_EVENT
35.1%
64.9%



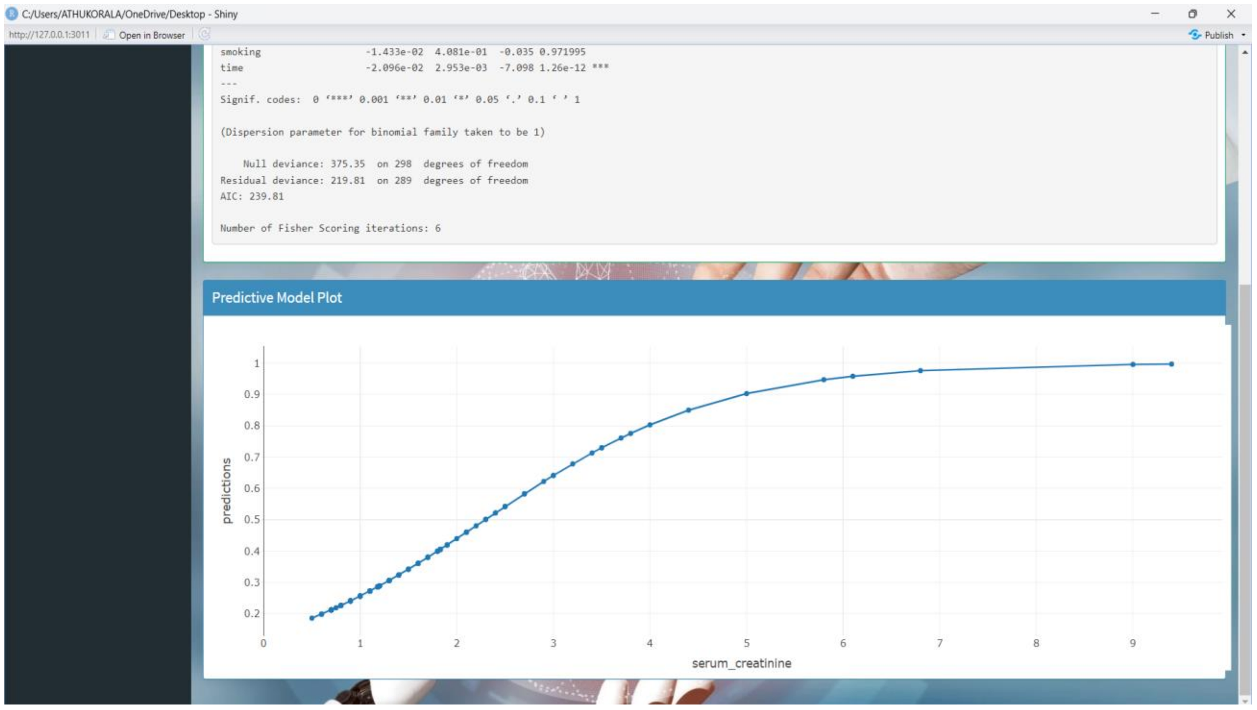
C:/Users/ATHUKORALA/OneDrive/Desktop - Shiny
http://127.0.0.1:3011
Open in Browser
Publish
Death Event and the serum creatinine
Prescence of symptoms in patients
DEATH_EVENT
serum_creatinine
Patient ID
Symptoms
anaemia
diabetes
high_blood_pressure

Logistic Regression :

This page includes the logistic regression performed for the dataset.

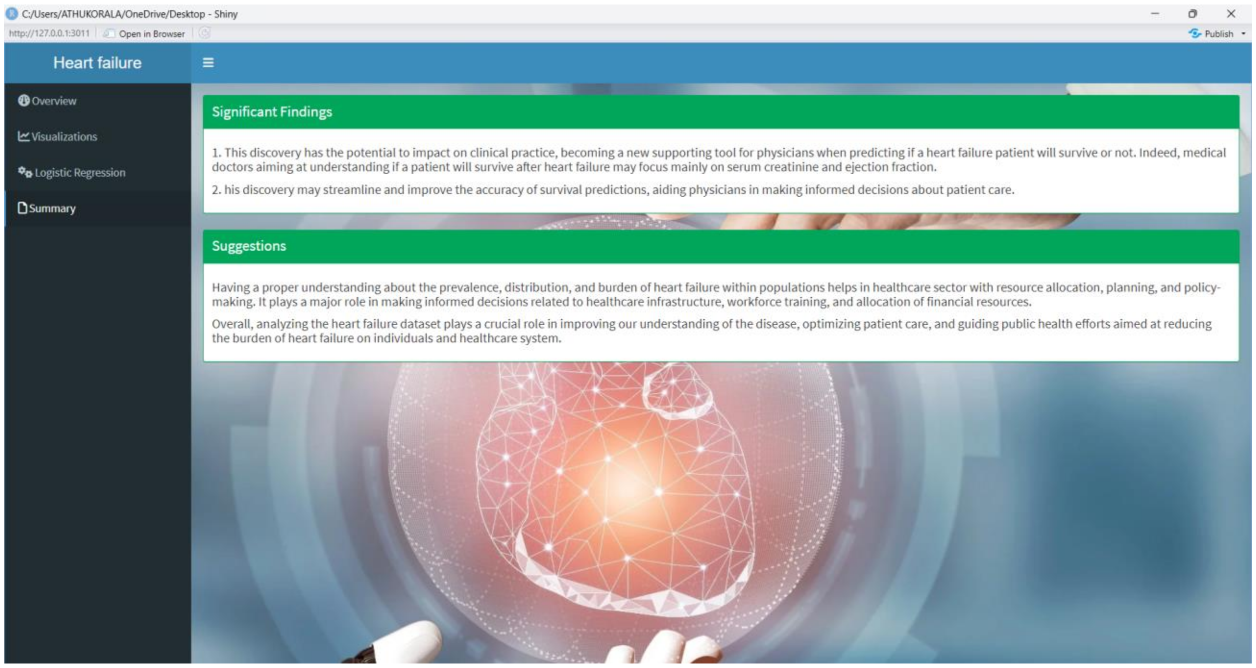


```
Call:
glm(formula = DEATH_EVENT ~ age + creatinine_phosphokinase +
    ejection_fraction + platelets + serum_creatinine + serum_sodium +
    sex + smoking + time, family = binomial(), data = data)

Coefficients:
                          Estimate Std. Error z value Pr(>|z|)
(Intercept)              1.051e+01  5.568e+00   1.887 0.059152 .
age                      4.652e-02  1.566e-02   2.971 0.002973 **
creatinine_phosphokinase 2.249e-04  1.753e-04   1.283 0.199385
ejection_fraction       -7.615e-02  1.626e-02  -4.685 2.80e-06 ***
platelets               -1.179e-06  1.896e-06  -0.622 0.533907
serum_creatinine         6.605e-01  1.789e-01   3.691 0.000223 ***
serum_sodium            -6.900e-02  3.929e-02  -1.756 0.079023 .
sex                     -5.294e-01  4.075e-01  -1.299 0.193841
smoking                 -1.433e-02  4.081e-01  -0.035 0.971995
time                    -2.096e-02  2.953e-03  -7.098 1.26e-12 ***
---
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: 219.81  on 289  degrees of freedom
AIC: 239.81

Number of Fisher Scoring iterations: 6
```



```
smoking                 -1.433e-02  4.081e-01  -0.035 0.971995
time                    -2.096e-02  2.953e-03  -7.098 1.26e-12 ***
---
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: 219.81  on 289  degrees of freedom
AIC: 239.81

Number of Fisher Scoring iterations: 6
```

Summary :
This page includes the significant findings and the analysis of the dataset.



Significant Findings

1. This discovery has the potential to impact on clinical practice, becoming a new supporting tool for physicians when predicting if a heart failure patient will survive or not. Indeed, medical doctors aiming at understanding if a patient will survive after heart failure may focus mainly on serum creatinine and ejection fraction.

2. his discovery may streamline and improve the accuracy of survival predictions, aiding physicians in making informed decisions about patient care.

Suggestions

Having a proper understanding about the prevalence, distribution, and burden of heart failure within populations helps in healthcare sector with resource allocation, planning, and policy-making. It plays a major role in making informed decisions related to healthcare infrastructure, workforce training, and allocation of financial resources.

Overall, analyzing the heart failure dataset plays a crucial role in improving our understanding of the disease, optimizing patient care, and guiding public health efforts aimed at reducing the burden of heart failure on individuals and healthcare system.

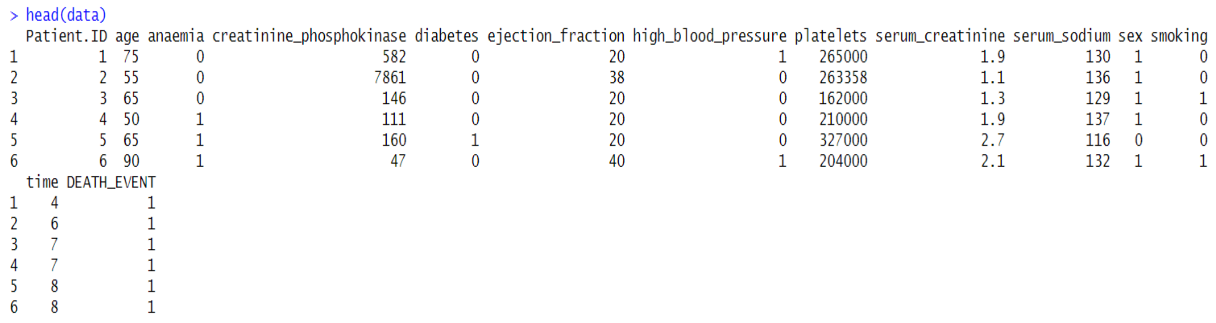
2.4. DASHBOARD IMPLEMENTATION

The dashboard was implemented using R – Programming Language, as shown below.

Step 1 :
Installing the relevant packages and importing the heart failure dataset as “data” using the read.csv package and to further analyze the dataset we used the head() function.

```
# Load required libraries
install.packages("plotly")
install.packages("shiny")
install.packages("shinyWidgets")
install.packages("dplyr")
install.packages("shinydashboard")
install.packages("DT")

library(shinydashboard)
library(DT)
library(plotly)
library(shiny)
library(shinyWidgets)
library(dplyr)
```

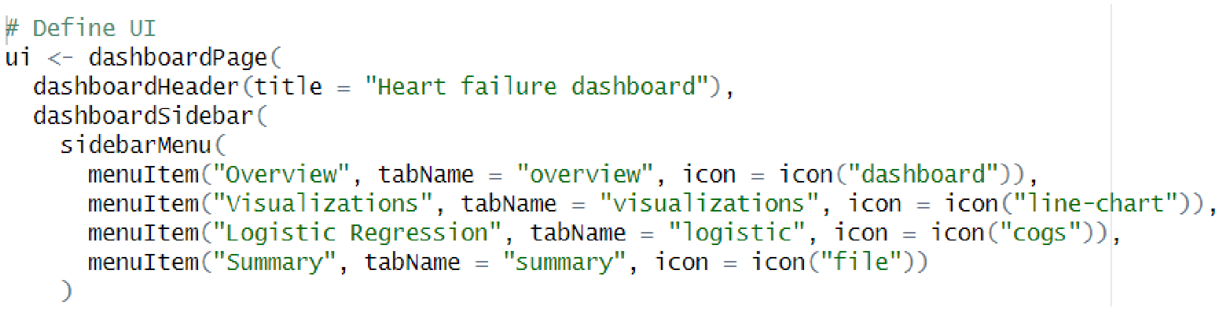
```
data <- read.csv("C:\\Users\\ATHUKORALA\\OneDrive\\Desktop\\heart_failure.csv")
data
head(data)
```



```
> head(data)
  Patient.ID age anaemia creatinine_phosphokinase diabetes ejection_fraction high_blood_pressure platelets serum_creatinine serum_sodium sex smoking
1          1  75       0                      582        0                20                   1    265000              1.9          130   1       0
2          2  55       0                     7861        0                38                   0    263358              1.1          136   1       0
3          3  65       0                      146        0                20                   0    162000              1.3          129   1       1
4          4  50       1                      111        0                20                   0    210000              1.9          137   1       0
5          5  65       1                      160        1                20                   0    327000              2.7          116   0       0
6          6  90       1                       47        0                40                   1    204000              2.1          132   1       1
  time DEATH_EVENT
1    4           1
2    6           1
3    7           1
4    7           1
5    8           1
6    8           1
```

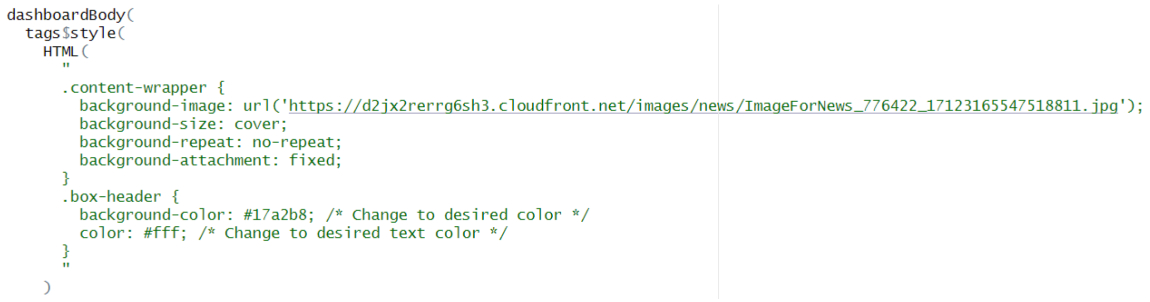
Step 2 :
Utilizing the shiny package in R, we may create a dashboard by utilizing the function ui(), which is also referred to as the user interface, to define the dashboard's structure.

The structure of the pages in the dataset is designed using the next section of the ui code. To Name the dashboard, dashboardHeader() code is used whilst to assign headings to the pages we use the sidebarMenu().



```
# Define UI
ui <- dashboardPage(
  dashboardHeader(title = "Heart failure dashboard"),
  dashdboardSidebar(
    sidebarMenu(
      menuItem("Overview", tabName = "overview", icon = icon("dashboard")),
      menuItem("Visualizations", tabName = "visualizations", icon = icon("line-chart")),
      menuItem("Logistic Regression", tabName = "logistic", icon = icon("cogs")),
      menuItem("Summary", tabName = "summary", icon = icon("file"))
    )
```

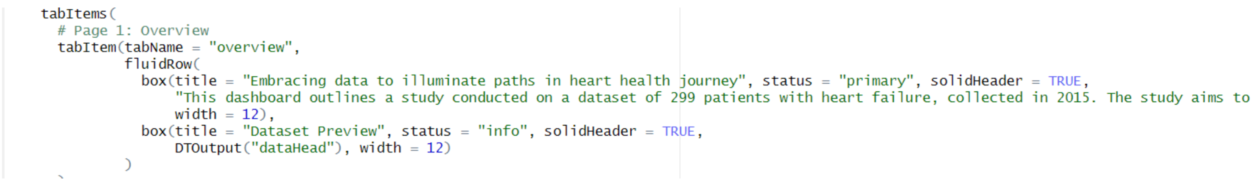
Step 3 :
The following code is used to insert a background picture to the dashboard.



```
dashboardBody(
  tags$style(
    HTML(
      "
      .content-wrapper {
        background-image: url('https://d2jx2rerrg6sh3.cloudfront.net/images/news/ImageForNews_776422_17123165547518811.jpg');
        background-size: cover;
        background-repeat: no-repeat;
        background-attachment: fixed;
      }
      .box-header {
        background-color: #17a2b8; /* Change to desired color */
        color: #fff; /* Change to desired text color */
      }
      "
    )
```

Step 4 :

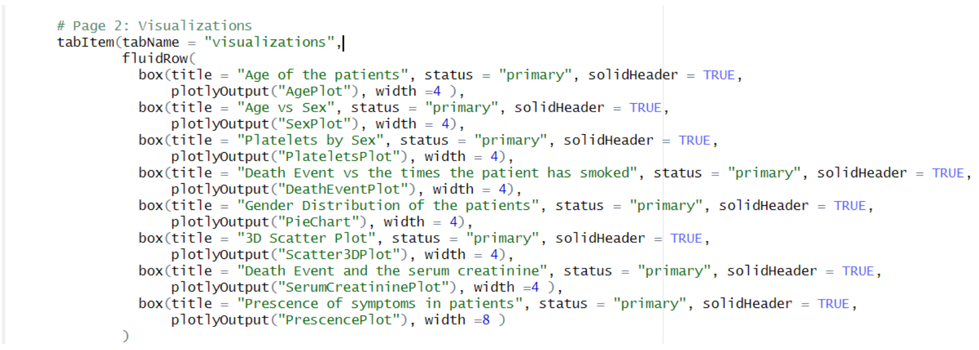
The Overview page was created as follows using R codes.



```
  tabItems(
    # Page 1: Overview
    tabItem(tabName = "overview",
            fluidRow(
              box(title = "Embracing data to illuminate paths in heart health journey", status = "primary", solidHeader = TRUE,
                  "This dashboard outlines a study conducted on a dataset of 299 patients with heart failure, collected in 2015. The study aims to
                  width = 12),
              box(title = "Dataset Preview", status = "info", solidHeader = TRUE,
                  DTOutput("dataHead"), width = 12)
            )
```

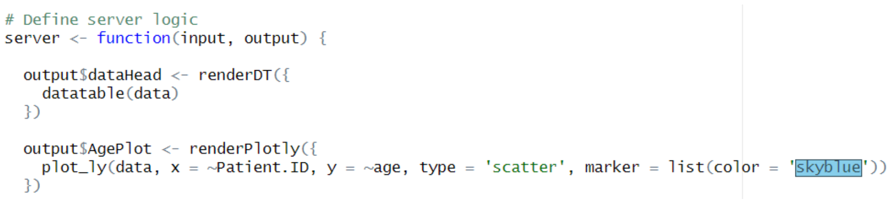
Step 5 :

Creating the Visualizations page. This code includes, renaming, positioning, sizing the plots as well.



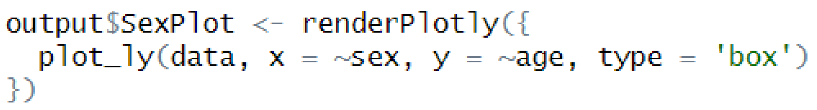
```
    # Page 2: Visualizations
    tabItem(tabName = "visualizations",
            fluidRow(
              box(title = "Age of the patients", status = "primary", solidHeader = TRUE,
                  plotlyOutput("AgePlot"), width =4 ),
              box(title = "Age vs Sex", status = "primary", solidHeader = TRUE,
                  plotlyOutput("SexPlot"), width = 4),
              box(title = "Platelets by Sex", status = "primary", solidHeader = TRUE,
                  plotlyOutput("PlateletsPlot"), width = 4),
              box(title = "Death Event vs the times the patient has smoked", status = "primary", solidHeader = TRUE,
                  plotlyOutput("DeathEventPlot"), width = 4),
              box(title = "Gender Distribution of the patients", status = "primary", solidHeader = TRUE,
                  plotlyOutput("PieChart"), width = 4),
              box(title = "3D Scatter Plot", status = "primary", solidHeader = TRUE,
                  plotlyOutput("Scatter3DPlot"), width = 4),
              box(title = "Death Event and the serum creatinine", status = "primary", solidHeader = TRUE,
                  plotlyOutput("SerumCreatininePlot"), width =4 ),
              box(title = "Prescence of symptoms in patients", status = "primary", solidHeader = TRUE,
                  plotlyOutput("PrescencePlot"), width =8 )
            )
```

To create a scatter plot based on the age of the patients and the patient ID, the following code was used.



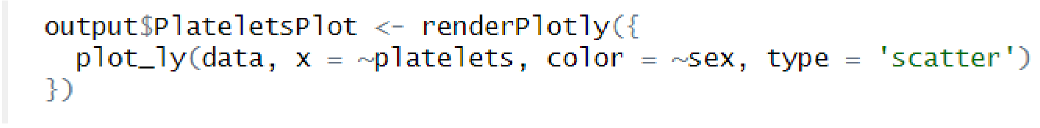
```
# Define server logic
server <- function(input, output) {

  output$dataHead <- renderDT({
    datatable(data)
  })

  output$AgePlot <- renderPlotly({
    plot_ly(data, x = ~Patient.ID, y = ~age, type = 'scatter', marker = list(color = 'skyblue'))
  })
```

To analyze the age distribution based on the gender of the patients, we created a box plot using the code given below [0 = Females, 1 = Male].



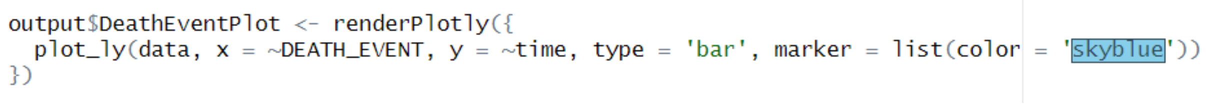
```
  output$SexPlot <- renderPlotly({
    plot_ly(data, x = ~sex, y = ~age, type = 'box')
  })
```

To analyze the platelets of the patients based on the gender of the patients, we created a scatter plot using the code given below [0 = Females, 1 = Male].



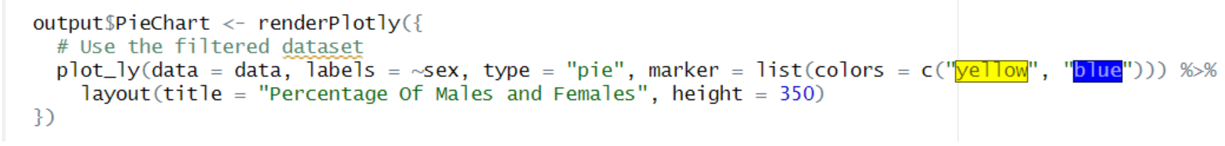
```
output$PlateletsPlot <- renderPlotly({
  plot_ly(data, x = ~platelets, color = ~sex, type = 'scatter')
})
```

To analyze the occurrence of the death event based on the smoked times of the patients, we created a bar plot using the code given below [0 = Not Survived, 1 = Survived].



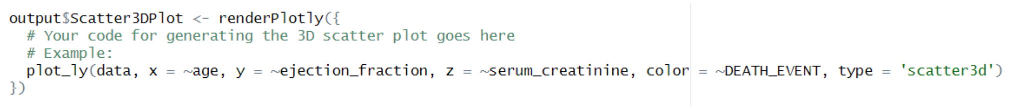
```
output$DeathEventPlot <- renderPlotly({
  plot_ly(data, x = ~DEATH_EVENT, y = ~time, type = 'bar', marker = list(color = 'skyblue'))
})
```

To analyze the proportion of the gender, we created a pie chart using the code given below [Female / Blue = 0, Male / Yellow = 1].



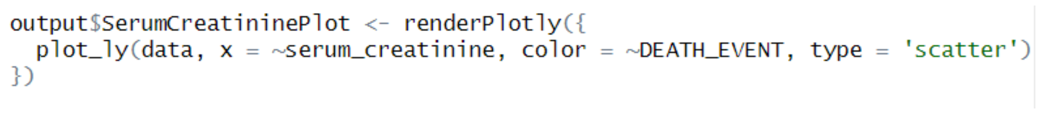
```
output$PieChart <- renderPlotly({
  # Use the filtered dataset
  plot_ly(data = data, labels = ~sex, type = "pie", marker = list(colors = c("yellow", "blue"))) %>%
    layout(title = "Percentage Of Males and Females", height = 350)
})
```

A scatter plot is typically used to examine how two variables are related to one another. We are able to examine the relationship between three variables by utilizing a 3D scatter plot. Consequently, in order to forecast the likelihood that the death event will occur, we employed the ejection fraction and the serum creatine parameter. We used the code provided below to create the scatter plot.



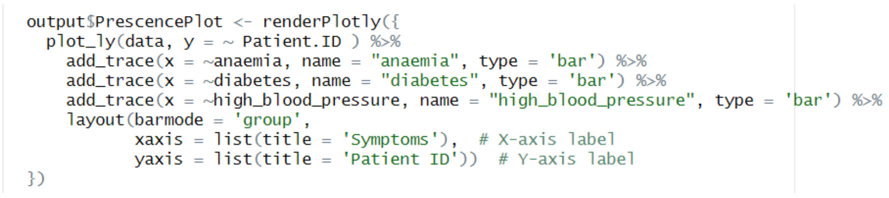
```
output$Scatter3DPlot <- renderPlotly({
  # Your code for generating the 3D scatter plot goes here
  # Example:
  plot_ly(data, x = ~age, y = ~ejection_fraction, z = ~serum_creatinine, color = ~DEATH_EVENT, type = 'scatter3d')
})
```

To analyze the serum creatine levels of the patients and the probability of the occurrence of the death event of patients we created a scatter plot using the code given below [0 = Not Survived, 1 = Survived].



```
output$SerumCreatininePlot <- renderPlotly({
  plot_ly(data, x = ~serum_creatinine, color = ~DEATH_EVENT, type = 'scatter')
})
```

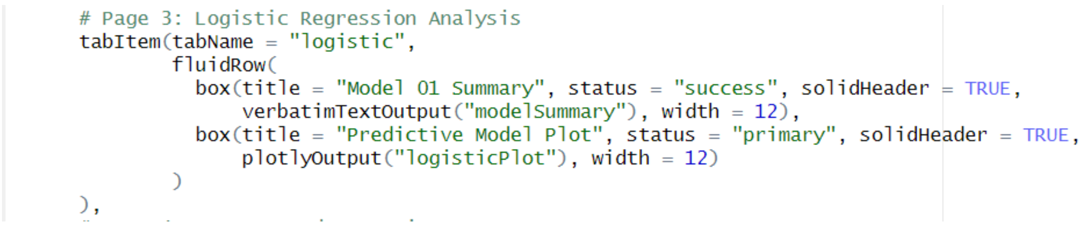
Thereafter, we created a multiple bar chart to analyze the presence of certain symptoms of the patients based on the patient ID. The symptoms are anemia, diabetes, high blood pressure. The code is given as below.



```
output$PrescencePlot <- renderPlotly({
  plot_ly(data, y = ~ Patient.ID ) %>%
    add_trace(x = ~anaemia, name = "anaemia", type = 'bar') %>%
    add_trace(x = ~diabetes, name = "diabetes", type = 'bar') %>%
    add_trace(x = ~high_blood_pressure, name = "high_blood_pressure", type = 'bar') %>%
    layout(barmode = 'group',
           xaxis = list(title = 'Symptoms'),  # X-axis label
           yaxis = list(title = 'Patient ID'))  # Y-axis label
})
```

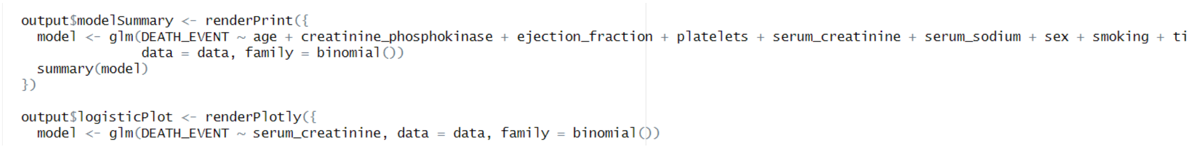
Step 6 :

Creating the Logistic Regression page.



```
# Page 3: Logistic Regression Analysis
tabItem(tabName = "logistic",
        fluidRow(
          box(title = "Model 01 Summary", status = "success", solidHeader = TRUE,
              verbatimTextOutput("modelSummary"), width = 12),
          box(title = "Predictive Model Plot", status = "primary", solidHeader = TRUE,
              plotlyOutput("logisticPlot"), width = 12)
        )
),
```

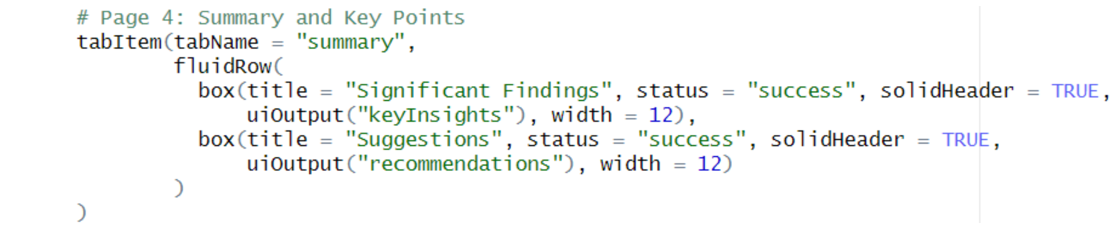
In order to make the dashboard more informative, we added the generated logistic regression curve to this page. The dashboard was made using the dataset that we used for the logistic regression study. We used the code provided below to produce the plot.



```
output$modelSummary <- renderPrint({
  model <- glm(DEATH_EVENT ~ age + creatinine_phosphokinase + ejection_fraction + platelets + serum_creatinine + serum_sodium + sex + smoking + ti
               data = data, family = binomial())
  summary(model)
})

output$logisticPlot <- renderPlotly({
  model <- glm(DEATH_EVENT ~ serum_creatinine, data = data, family = binomial())
```

Step 7 :

The final page of the dash board is created.



```
# Page 4: Summary and Key Points
tabItem(tabName = "summary",
        fluidRow(
          box(title = "Significant Findings", status = "success", solidHeader = TRUE,
              uiOutput("keyInsights"), width = 12),
          box(title = "Suggestions", status = "success", solidHeader = TRUE,
              uiOutput("recommendations"), width = 12)
        )
)
```

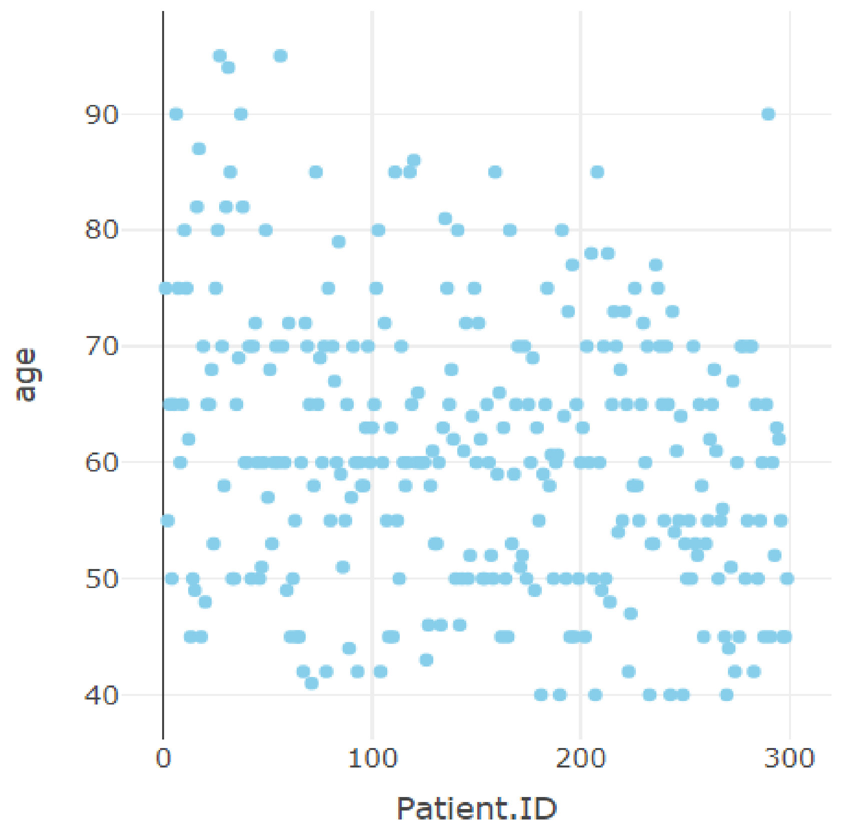
Lastly, the dashboard can be created as shown above using the R- Programming Language.

2.5. DATA ANALYSIS

In order to obtain insightful knowledge, it is necessary to analyze the data and visualizations on our dashboard.

The analysis of the data and the visualizations present in our dashboard are as follows.

Age of the patients

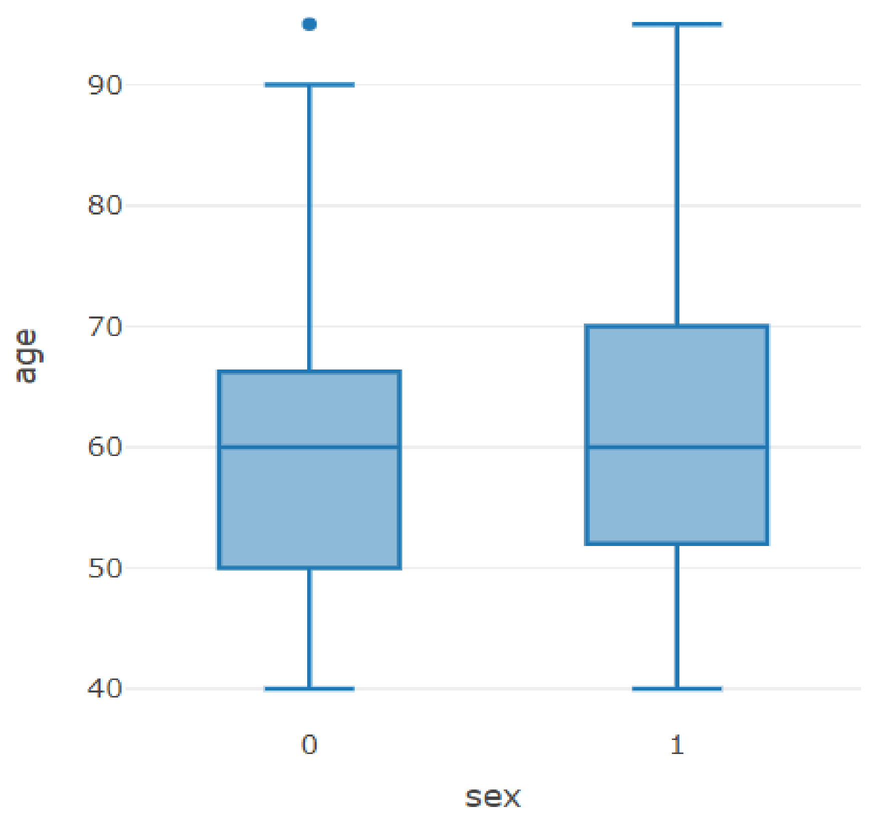


It's important to note that scatter plots are useful for visualizing trends and one point on the scatter plot shows each patients age according to its Patient ID.

The x-axis shows the Patient ID and the y-axis shows the age.

There does not appear to be a strong linear relationship between age and the number of patients. There seems to be a spread of data points across the entire age range, with a concentration of data points in the middle range (possibly between 40 and 60 years old).

Age vs Sex



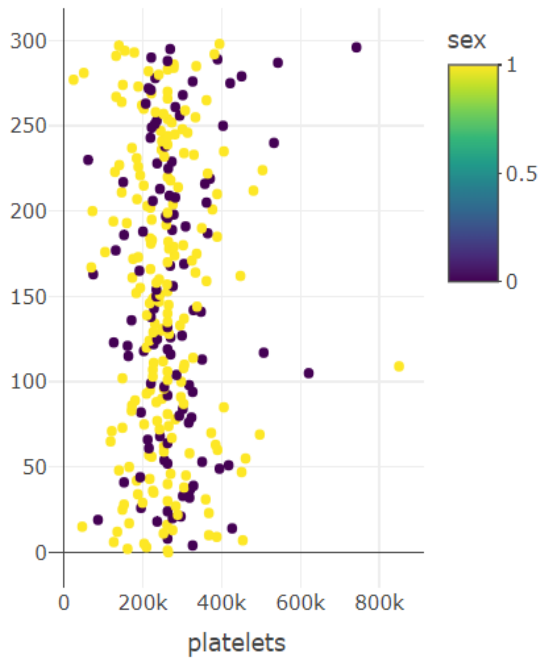
This boxplot shows the gender distribution according to the age of the patients. The x- axis shows the Gender [0 = Females, 1 = Male] and the y-axis shows the age of the patient.

The mean is 60 years to both males and females and there are outliers to be found in the females distribution. The spread of the ages seems to be wider for Males, as the whiskers are longer than the females boxplot whiskers. Both the boxplots are positively skewed. The whisker on the lower end of the box is shorter than the whisker on the upper end, then the distribution is positively skewed. This means there are more data points towards the lower values and a longer tail towards the higher values.

The boxplot that depicts the female distribution clearly shows the following characteristics.
Minimum = 40, Lower Quartile (Q1) = 50, Median (Q2) = 60, and Maximum = 90

The boxplot that depicts the male distribution shows the following characteristics.
Minimum = 40, Median (Q2) = 60, Upper Quartile (Q3) = 70, and Maximum = 90

Platelets by Sex

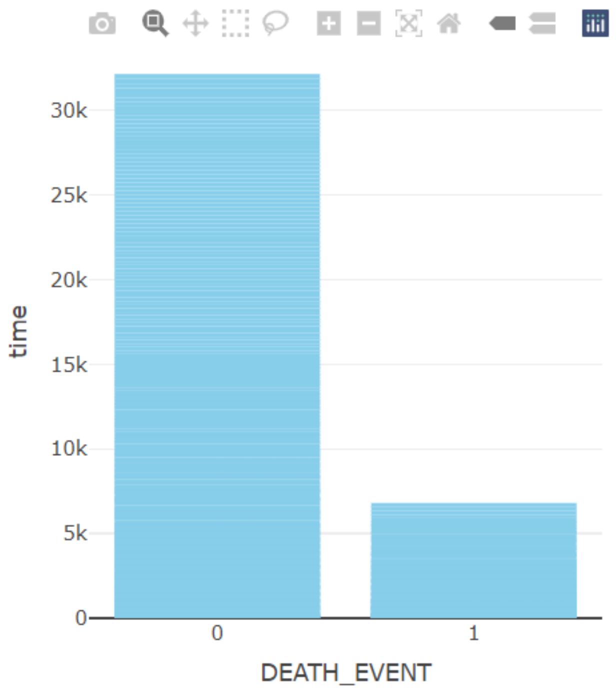


This scatter plot shows the gender distribution [0 = Females, 1 = Male] according to the platelets count.

The x- axis shows the Number of Platelets and each point on the scatter plot shows the individual platelets count of the patient according to the patient's gender.

The chart shows that the average platelets count is higher for the males than the females. The platelet count for males is around 275, while the platelet count for females is around 225.

Death Event vs the times the patient has smoked

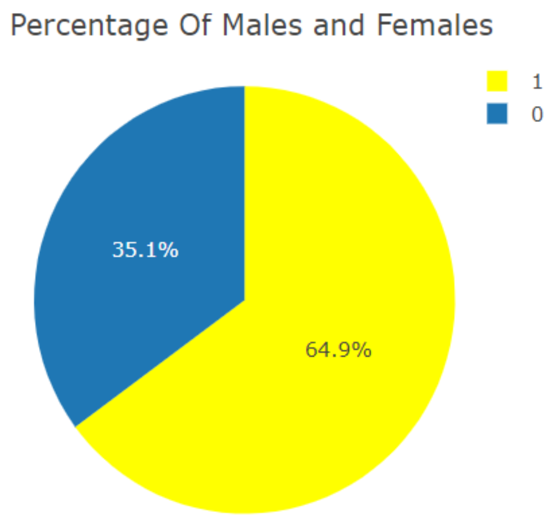


The bar graph shows the relationship between the death event distribution [0 = Not Survived, 1 = Suvived] and the smoking habit of the patient.

The y- axis shows the number of times the patient has done smoking and the x-axis shows the death event.

Accordingly, the death rate of the patients who has the smoking habit is comparatively higher than the survival rate.

Gender Distribution of the patients

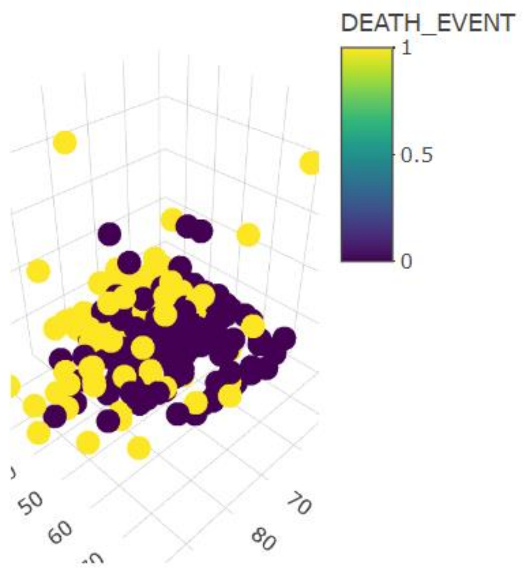


The pie chart shows the gender distribution of the patients in the dataset.

The females are depicted in the color Blue = 0, while the males are represented from the color Yellow = 1.

By observing the chart, it is clear that the Male percentage (64.9%) is higher than the Females percentage (35.1%).

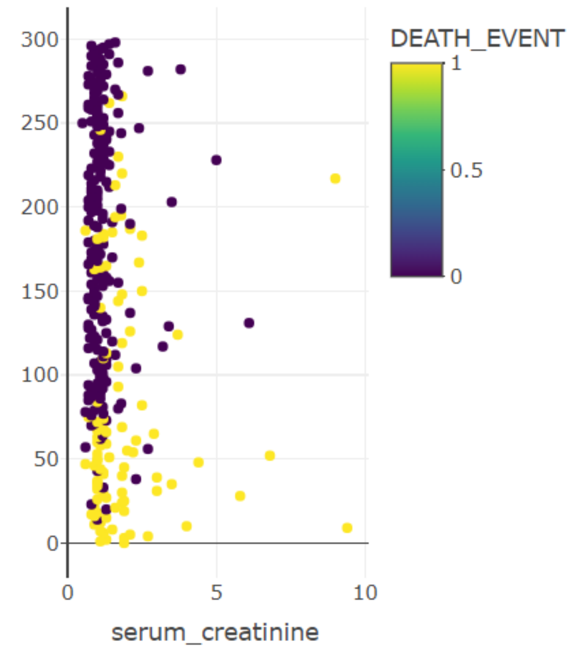
3D Scatter Plot



The 3D scatter plot depicted here, was used to forecast the likelihood that the death event [1 = Survived, 0 = Not Survived] will occur, with the ejection fraction and the serum creatinine parameter.

The x-axis shows the age of the patient, y-axis shows the ejection fraction and z-axis shows the serum creatinine.

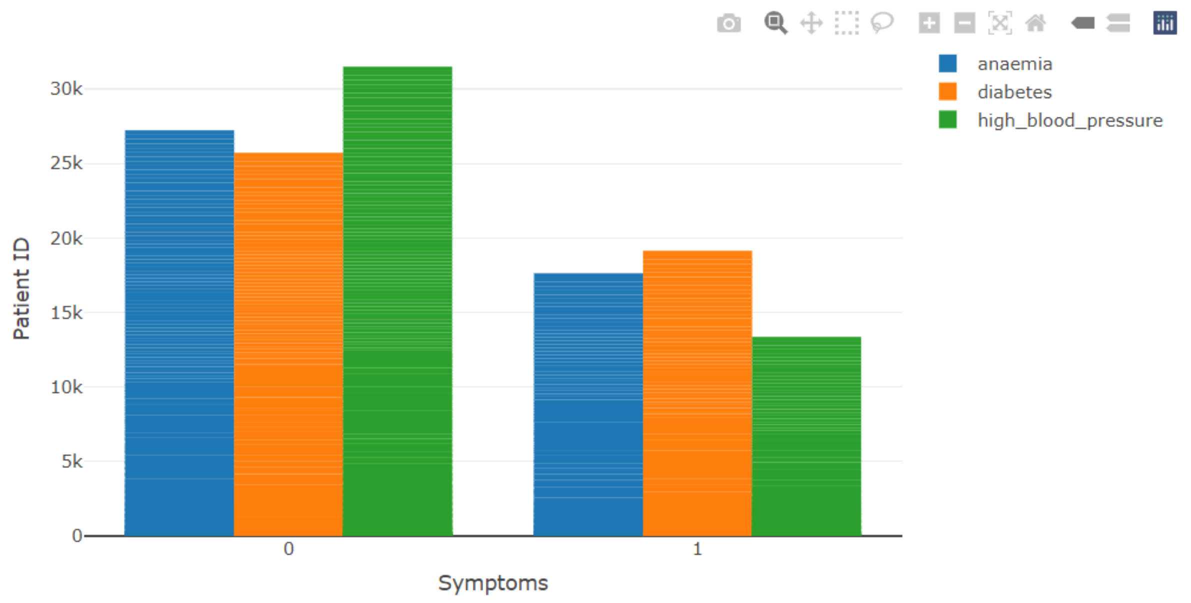
Death Event and the serum creatinine



The scatter plot depicted here, was used to forecast the likelihood that the death event [1 = Survived, 0 = Not Survived] will occur, with the serum creatinine parameter.

The x-axis shows the Serum Creatinine parameter and the y-axis shows the number of patients.

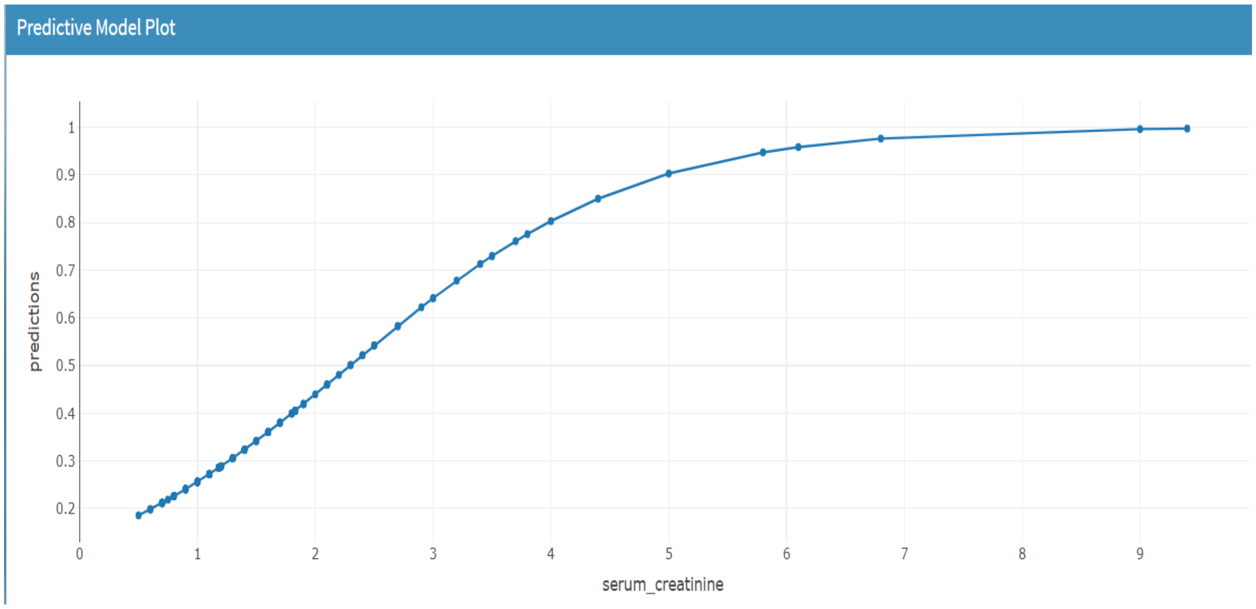
Prescence of symptoms in patients



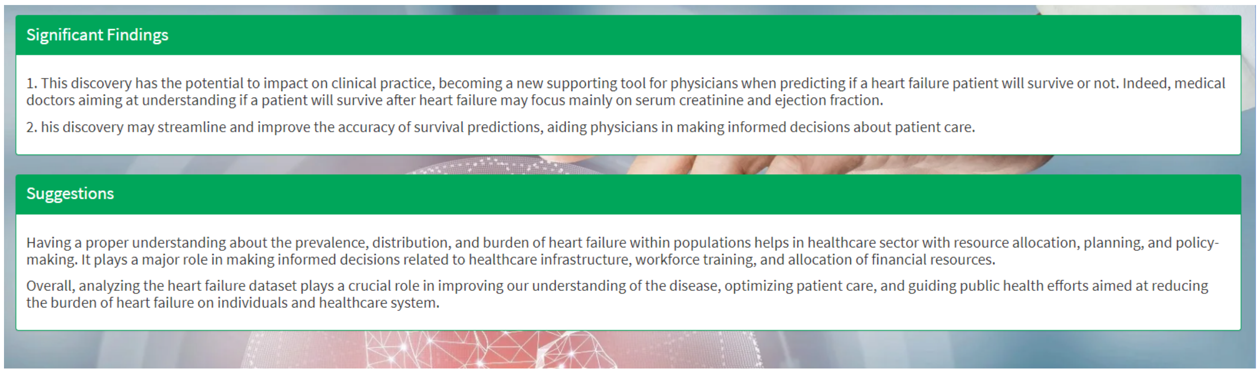
The multiple bar chart we created helps to analyze the presence of certain symptoms [0 = Absence, 1 = Presence] of the patients based on the patient ID. The symptoms are Anemia, Diabetes, High Blood Pressure.

The x-axis shows the Symptoms and the y-axis shows the number of patients according to their patient ID.

Overall, the absence of the symptoms is higher than the presence of the symptoms. The presence of Symptoms according to their abundance are Diabetes, Anaemia and High Blood Pressure respectively while the absence of symptoms according to their abundance are High Blood Pressure, Aneamia and Diabetes respectively.



The Logistic Regression Curve obtained from the Regression Analysis for the Task 2 is depicted on the dashboard. The analysis on this curve can be referred in page 58.



Significant Findings

1. This discovery has the potential to impact on clinical practice, becoming a new supporting tool for physicians when predicting if a heart failure patient will survive or not. Indeed, medical doctors aiming at understanding if a patient will survive after heart failure may focus mainly on serum creatinine and ejection fraction.

2. his discovery may streamline and improve the accuracy of survival predictions, aiding physicians in making informed decisions about patient care.

Suggestions

Having a proper understanding about the prevalence, distribution, and burden of heart failure within populations helps in healthcare sector with resource allocation, planning, and policy-making. It plays a major role in making informed decisions related to healthcare infrastructure, workforce training, and allocation of financial resources.

Overall, analyzing the heart failure dataset plays a crucial role in improving our understanding of the disease, optimizing patient care, and guiding public health efforts aimed at reducing the burden of heart failure on individuals and healthcare system.

After analyzing the visualizations and data, the significant observations and the insights obtained are as shown above and they are published in the Summary page of the Dashboard.

2.6. CONCLUSION

In conclusion, the development of the logistic regression dashboard using the Heart Failure Clinical Records dataset from UCI has provided valuable insights into predicting survival outcomes among heart failure patients. The dashboard effectively leverages an overview about the dataset, logistic regression modeling, dataset related visualizations to assess the impact of various clinical factors on the likelihood of death events. Through this project, we have identified key predictors such as age, serum creatinine levels, and ejection fraction that significantly influence patient outcomes. The dashboard not only facilitates real-time monitoring and analysis but also supports informed decision-making for healthcare professionals by highlighting critical risk factors. Moving forward, continued refinement and validation of the dashboard will enhance its utility in clinical settings, ultimately contributing to improved patient care and management strategies in the field of cardiovascular health.

REFERENCES

- **https://youtu.be/-lN_JNvFSyE?si=2RkaMGSF51yqd8A-**
- **https://youtu.be/_a4S4tq62OE?si=38VlveNvFaYuU1Nn**
- **https://youtu.be/WmofiOklux8?si=rba_YXvuIVvaef1a**