Heart Failure Prediction~HarvardX: PH125.9x(Choose your own)

Hrithik Muskan

08/01/2021

Index:

Dataset Description

Link: https://www.kaggle.com/andrewmvd/heart-failure-clinical-data

Introduction:

In 2015, heart failure affected about 40 million people globally. Overall, around 2% of adults have heart failure and in those over the age of 65, this increases to 6–10%.Above 75 years old, rates are greater than 10%.

Rates are predicted to increase.Increasing rates are mostly because of increasing lifespan, but also because of increased risk factors (hypertension, diabetes, dyslipidemia, and obesity) and improved survival rates from other types of cardiovascular disease (myocardial infarction, valvular disease, and arrhythmia).Heart failure is the leading cause of hospitalization in people older than 65.

Cardiovascular diseases (CVDs) are the number 1 cause of death globally, taking an estimated 17.9 million lives each year, which accounts for 31% of all deaths worldwide. Heart failure is a common event caused by CVDs and this dataset contains 12 features that can be used to predict mortality by heart failure.

Most cardiovascular diseases can be prevented by addressing behavioral risk factors such as tobacco use, unhealthy diet and obesity, physical inactivity and harmful use of alcohol using population-wide strategies.

People with cardiovascular disease or who are at high cardiovascular risk (due to the presence of one or more risk factors such as hypertension, diabetes, hyperlipidaemia or already established disease) need early detection and management wherein a machine learning model can be of great help.

Variables:

Let's see the variables along with the required setup. This is a Dataset from Davide Chicco, BMC Medical Informatics and Decision Making 20, 16 (2020)
Let's load the data and the required libraries.

```r
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: tidyverse
```

```
## -- Attaching packages ---------------------------------------------------------------------------------
```

```
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

```
## -- Conflicts ------------------------------------------------------------------------------------ tid
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: caret
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```
if(!require(ggplot2)) install.packages("ggplot2", repos = "http://cran.us.r-project.org")
if(!require(e1071)) install.packages("e1071", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: e1071

## Warning: package 'e1071' was built under R version 4.0.3
```

```
library(tidyverse)
library(caret)
library(ggplot2)
data <- read.csv("https://storage.googleapis.com/kagglesdsdata/datasets/727551/1263738/heart_failure_cl
```

Summary:

Let's look up to the structure and summary of the data.

```
glimpse(data)
```

```
## Rows: 299
## Columns: 13
## $ age                      <dbl> 75, 55, 65, 50, 65, 90, 75, 60, 65, 80, 75...
## $ anaemia                  <int> 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, ...
## $ creatinine_phosphokinase <int> 582, 7861, 146, 111, 160, 47, 246, 315, 15...
## $ diabetes                 <int> 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, ...
## $ ejection_fraction        <int> 20, 38, 20, 20, 20, 40, 15, 60, 65, 35, 38...
## $ high_blood_pressure      <int> 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, ...
## $ platelets                <dbl> 265000, 263358, 162000, 210000, 327000, 20...
## $ serum_creatinine         <dbl> 1.90, 1.10, 1.30, 1.90, 2.70, 2.10, 1.20, ...
## $ serum_sodium             <int> 130, 136, 129, 137, 116, 132, 137, 131, 13...
## $ sex                      <int> 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, ...
## $ smoking                  <int> 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, ...
## $ time                     <int> 4, 6, 7, 7, 8, 8, 10, 10, 10, 10, 10, 10, ...
## $ DEATH_EVENT              <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
```

Variables:

age - Age

anaemia - Decrease of red blood cells or hemoglobin (boolean) (0:False, 1:True)

creatinine_phosphokinase - Level of the CPK enzyme in the blood (mcg/L)

diabetes - If the patient has diabetes (boolean) (0:False, 1:True)

ejection_fraction - Percentage of blood leaving the heart at each contraction (percentage)

high_blood_pressure - If the patient has hypertension (boolean) (0:False, 1:True)

platelets - Platelets in the blood (kiloplatelets/mL)

serum_creatinine - Level of serum creatinine in the blood (mg/dL)

serum_sodium - Level of serum sodium in the blood (mEq/L)

sex - Woman or Man (binary) (0: Woman, 1: Man)

smoking - If the patient smokes or not (boolean) (0:False, 1:True)

time - Follow-up period (days)

DEATH_EVENT - If the patient deceased during the follow-up period (boolean)

Goal:

Goal: We are going to predict mortality by heart failure based on the 12 features included in the data set. This can be used in assessing the severity of patients with cardiovascular diseases (CVDs) leading to Heart Failure.

Key Steps:

- Key Steps
  - Data Visualization
  - Data Preprocessing
  - Training Models
  - Accuracy of Models

Methods:

Data Cleaning

Let's see if there's any missing value in the data:

```r
any(is.na(data)) # returns True if there is any missing value in the entire dataframe
```

```
## [1] FALSE
```

Data Exploration

Let's see a bit of stats and summary of the data:

```r
class(data)
```

```
## [1] "data.frame"
```

```r
dim(data)
```

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

It has 299 rows and 13 columns.

```r
summary(data)
```

```
##       age           anaemia       creatinine_phosphokinase    diabetes     
##  Min.   :40.00   Min.   :0.0000   Min.   :  23.0           Min.   :0.0000  
##  1st Qu.:51.00   1st Qu.:0.0000   1st Qu.: 116.5           1st Qu.:0.0000  
##  Median :60.00   Median :0.0000   Median : 250.0           Median :0.0000  
##  Mean   :60.83   Mean   :0.4314   Mean   : 581.8           Mean   :0.4181  
##  3rd Qu.:70.00   3rd Qu.:1.0000   3rd Qu.: 582.0           3rd Qu.:1.0000  
##  Max.   :95.00   Max.   :1.0000   Max.   :7861.0           Max.   :1.0000  
##  ejection_fraction high_blood_pressure   platelets      serum_creatinine
##  Min.   :14.00     Min.   :0.0000      Min.   : 25100   Min.   :0.500   
##  1st Qu.:30.00     1st Qu.:0.0000      1st Qu.:212500   1st Qu.:0.900   
##  Median :38.00     Median :0.0000      Median :262000   Median :1.100   
##  Mean   :38.08     Mean   :0.3512      Mean   :263358   Mean   :1.394   
##  3rd Qu.:45.00     3rd Qu.:1.0000      3rd Qu.:303500   3rd Qu.:1.400   
##  Max.   :80.00     Max.   :1.0000      Max.   :850000   Max.   :9.400   
##   serum_sodium        sex            smoking            time      
##  Min.   :113.0   Min.   :0.0000   Min.   :0.0000   Min.   :  4.0  
##  1st Qu.:134.0   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.: 73.0  
##  Median :137.0   Median :1.0000   Median :0.0000   Median :115.0  
##  Mean   :136.6   Mean   :0.6488   Mean   :0.3211   Mean   :130.3  
##  3rd Qu.:140.0   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:203.0  
##  Max.   :148.0   Max.   :1.0000   Max.   :1.0000   Max.   :285.0  
##   DEATH_EVENT    
##  Min.   :0.0000  
##  1st Qu.:0.0000  
##  Median :0.0000  
##  Mean   :0.3211  
##  3rd Qu.:1.0000  
##  Max.   :1.0000
```

Here's a detailed view of data.

Data Visualization

Setup:

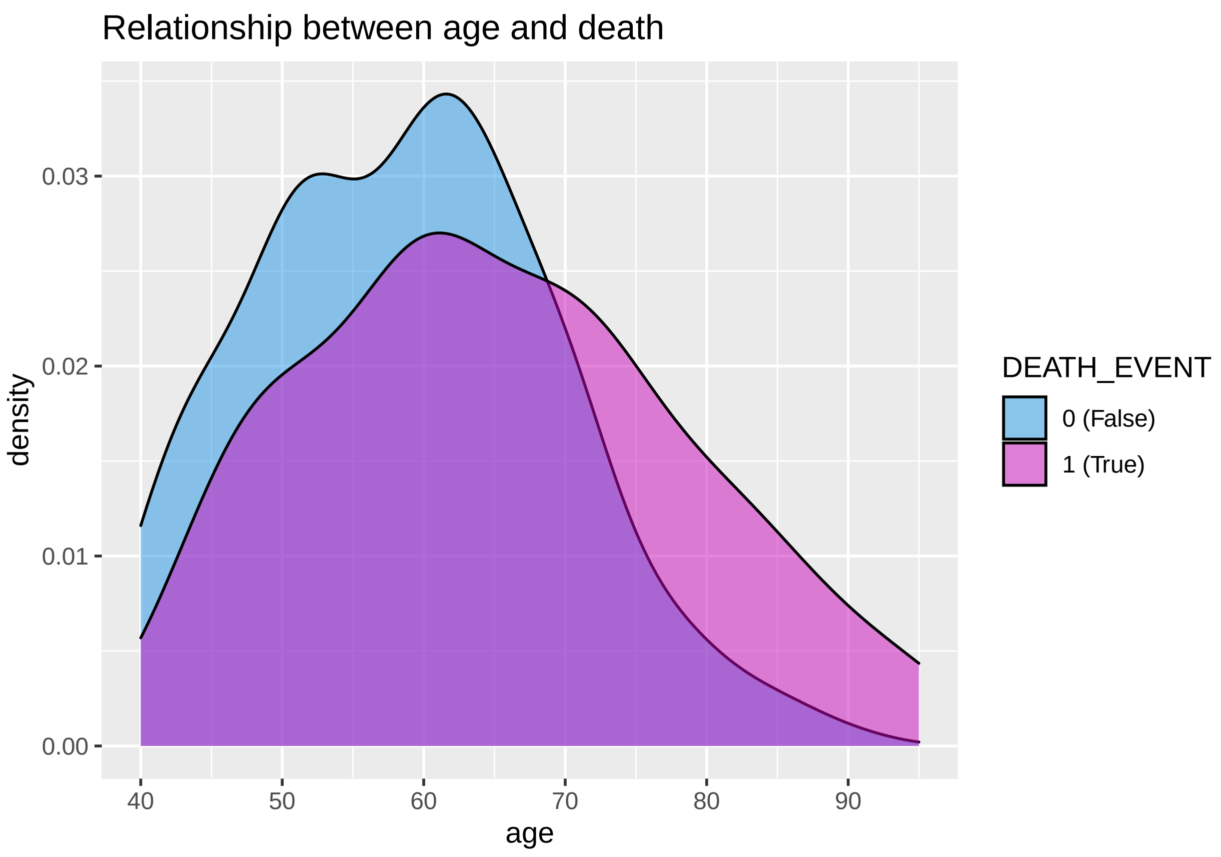
```r
# Changing datatype mutating some columns into factors for future usage
features = c("anaemia", "diabetes", "high_blood_pressure", "sex", "smoking", "DEATH_EVENT")

data_old <- data # let's store the original data.
data <- data %>%
  mutate_at(features, as.factor)
```

Some Important Visualizations:‘

Age vs Death

```
p1 <- ggplot(data, aes(x = age, fill = DEATH_EVENT))+ geom_density(alpha = 0.5) +scale_fill_manual(valu
p1
```

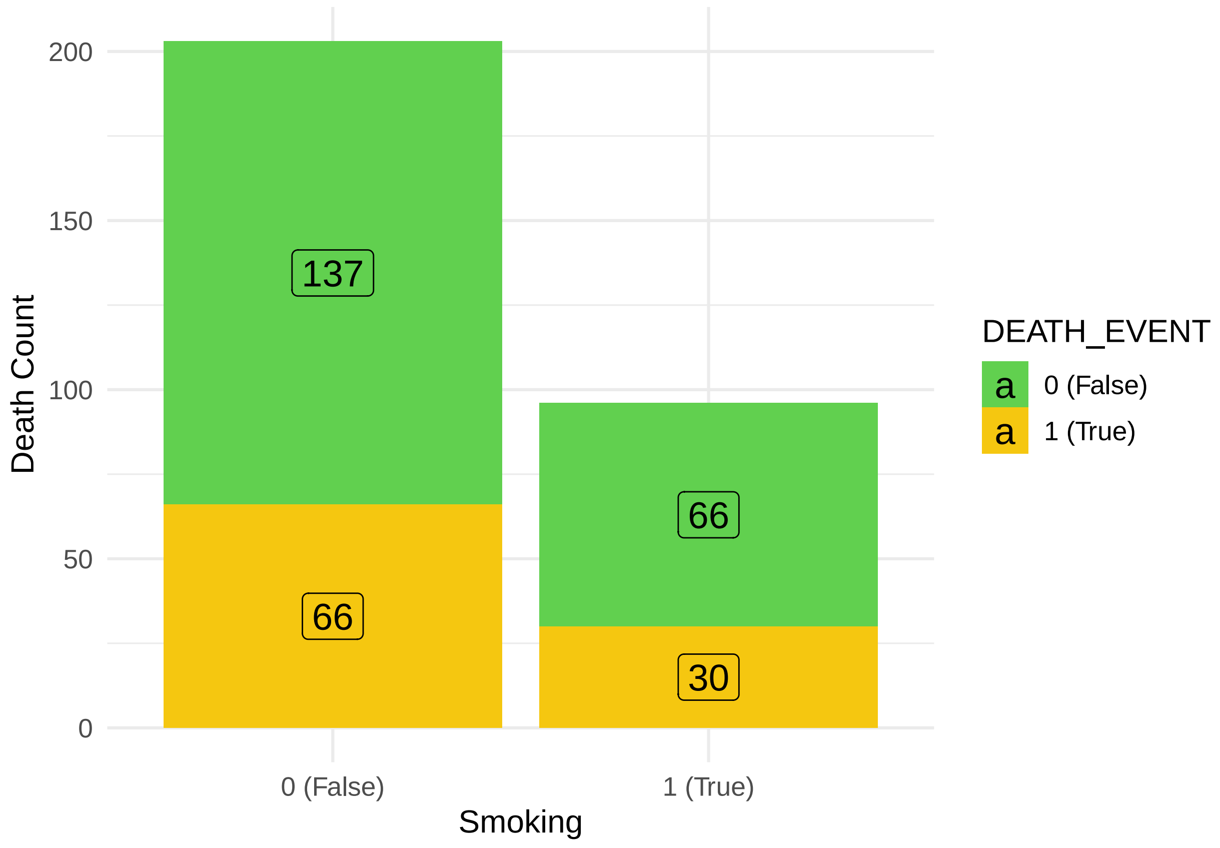


Insights:

1.Most of the patients are around 60-65 years old, and the number of patients decreased in a bell-shaped pattern around that age. 2.With the younger the age, the more difficult it is to die; the probability density reverses after the age of just under 68.

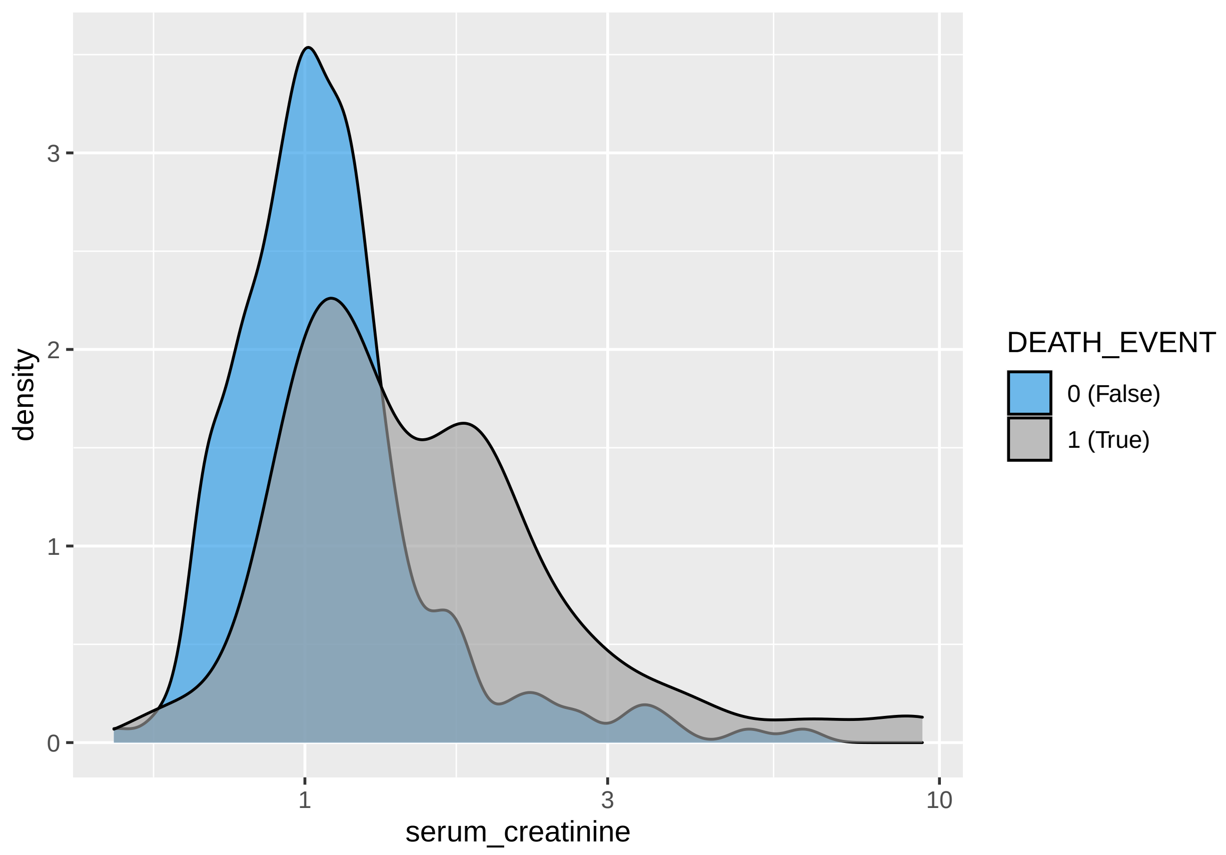
Smoking vs Death

```
p2<- ggplot(data, aes(x = smoking, fill = DEATH_EVENT)) +
  geom_bar(stat = "count", position = "stack", show.legend = FALSE) +
  scale_x_discrete(labels  = c("0 (False)", "1 (True)")) +
  scale_fill_manual(values = c(3,7),
                    name = "DEATH_EVENT",
                    labels = c("0 (False)", "1 (True)")) +
  labs(x = "Smoking") + labs(y ="Death Count")+theme_minimal(base_size = 12)+
  geom_label(stat = "count", aes(label = ..count..), position = position_stack(vjust = 0.5),
             size = 5, show.legend = TRUE)
p2
```



Serum Creatinine vs Death

```
p3 <- ggplot(data, aes(x = serum_creatinine, fill = factor(DEATH_EVENT))) +
  geom_density(alpha = 0.64) +
  scale_fill_manual(values = c(4,8),
                    name = "DEATH_EVENT",
                    labels = c("0 (False)", "1 (True)"))+scale_x_log10()
p3
```



density
3
2
1
0
1
3
10
serum_creatinine
DEATH_EVENT
0 (False)
1 (True)

Data Preprocessing:

Spliting data into train and test

```
set.seed(1,sample.kind = "Rounding")
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

```
test_index <- createDataPartition(y = data$DEATH_EVENT, times = 1, p = 0.5, list = FALSE)
test <- data[test_index,]
train <- data[-test_index,]
head(train)
```

```
##    age anaemia creatinine_phosphokinase diabetes ejection_fraction
## 1   75       0                      582        0                20
## 4   50       1                      111        0                20
## 5   65       1                      160        1                20
## 7   75       1                      246        0                15
## 8   60       1                      315        1                60
## 11  75       1                       81        0                38
##    high_blood_pressure platelets serum_creatinine serum_sodium sex smoking time
## 1                    1    265000              1.9          130   1       0    4
## 4                    0    210000              1.9          137   1       0    7
## 5                    0    327000              2.7          116   0       0    8
## 7                    0    127000              1.2          137   1       0   10
## 8                    0    454000              1.1          131   1       1   10
## 11                   1    368000              4.0          131   1       1   10
##    DEATH_EVENT
## 1            1
## 4            1
## 5            1
## 7            1
## 8            1
## 11           1
```

Modeling Approach:

KNN

K-nearest neighbors (kNN) estimates the conditional probabilities in a similar way to bin smoothing. However, kNN is easier to adapt to multiple dimensions. Using kNN, for any point (x1,x2) for which we want an estimate of p(x1,x2) , we look for the k nearest points to (x1,x2) and take an average of the 0s and 1s associated with these points. We refer to the set of points used to compute the average as the neighborhood. Larger values of k result in smoother estimates, while smaller values of k result in more flexible and more wiggly estimates

```
suppressWarnings(set.seed(1,sample.kind = "Rounding")) # use set.seed(1) for R versions prior R 3.5
knn_fit <- knn3(DEATH_EVENT ~ ., data = test)
x <- as.matrix(train[,0:12])
```

```r
y=train$DEATH_EVENT
knn_fit <- knn3(x, y)
knn_fit <- knn3(DEATH_EVENT~ ., data = train, k=5)
y_hat_knn <- predict(knn_fit,test, type = "class")
confusionMatrix(data = y_hat_knn, reference = test$DEATH_EVENT)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 84 41
##          1 18  7
##
##                Accuracy : 0.6067
##                  95% CI : (0.5237, 0.6853)
##     No Information Rate : 0.68
##     P-Value [Acc > NIR] : 0.976469
##
##                   Kappa : -0.0351
##
##  Mcnemar's Test P-Value : 0.004181
##
##             Sensitivity : 0.8235
##             Specificity : 0.1458
##          Pos Pred Value : 0.6720
##          Neg Pred Value : 0.2800
##              Prevalence : 0.6800
##          Detection Rate : 0.5600
##    Detection Prevalence : 0.8333
##       Balanced Accuracy : 0.4847
##
##        'Positive' Class : 0
##
```

IT HAS AN ACCURACY OF 60% AND SENSITIVITY OF 0.82

Logistic Regression

Logistic regression analysis is a model used to predict and analyze the probability of occurrence of an event (in this case, DEATH_EVENT == 1). By estimating the parameters (intercept and regression coefficients) using the maximum likelihood method, it is possible to calculate the change in odds (the ratio of the probability that the event will happen to the probability that the event will not happen) when the values of the explanatory variables change.

Fitting the model

```r
lr <- glm(DEATH_EVENT ~ .,
          family=binomial(logit), data=train)
```

Prediction and Results

```r
pred <- as.factor(predict(lr, newdata=test, type="response") >= 0.5) %>%
  fct_recode("0" = "FALSE", "1" = "TRUE")
confusionMatrix(pred, test$DEATH_EVENT, positive = "1")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 94 19
##          1  8 29
##
##                Accuracy : 0.82
##                  95% CI : (0.749, 0.8779)
##     No Information Rate : 0.68
##     P-Value [Acc > NIR] : 8.52e-05
##
##                   Kappa : 0.5597
##
##  Mcnemar's Test P-Value : 0.05429
##
##             Sensitivity : 0.6042
##             Specificity : 0.9216
##          Pos Pred Value : 0.7838
##          Neg Pred Value : 0.8319
##              Prevalence : 0.3200
##          Detection Rate : 0.1933
##    Detection Prevalence : 0.2467
##       Balanced Accuracy : 0.7629
##
##        'Positive' Class : 1
##
```

Conclusion:

We have the accuracy rate of both the models and particularly Logistic Regression has accuracy of 82% which provides us the model to account for heart failure prediction

Thanks for reading!!

References:

1. https://en.wikipedia.org/wiki/Heart_failure
2. https://www.world-heart-federation.org/cvd-roadmaps/whf-global-roadmaps/heart-failure/