REGRESSION_LOGISTIC

joseferson da silva barreto

2022-12-06

Obejetivo

O objetivo desse artigo é executar um modelo de regresão logistico,com o intuíto de classificar de acordo com seus atributos se o paciente com insuficiência cardiaca irá vir ou não à óbito.

Metódologia

Para esta análise foi utulizado o software RStudio , junto com as linguagens de programação R e python , para balanceamento do banco de dados foi utilizado o oversampling com o método **Smote** que é um dos mais utilizados quando temos um conjunto de dados com poucas observações, para verificação de associação entre variáveis categoricas foi utilizado o teste qui-quadrado e para as númericas como tivemos a rejeição de normalidade ,foi utilizado o teste Man-whitney

Introdução

Pesquisas indicam que a insuficiencia cardíaca afeta cerca de 65 milhões de pessoas no mundo„ ela é dividida em 2 tipos : insuficiência cardiáca com fração de ejeção reduzida e insuficiência cardiaca com fração de ejeção preservada.

FATORES DE RISCO Algumas pessoas tem maior probabilidade que outras de desenvolver insuficiência cardíaca. Ninguém pode prever com certeza quem irá desenvolvê-la. Estar ciente dos fatores de risco e ver um médico para tratamento precoce são boas estratégias para o controle da insuficiência cardíaca. Fatores de risco de insuficiência cardíaca incluem:

- Pressão alta (hipertensão)
- Ataque cardíaco (infarto do miocárdio)
- Válvulas cardíacas anormais
- Aumento do coração (cardiomiopatia)
- Histórico familiar de doença cardíaca
- Diabetes

#Treze (13) características clínicas:

- age: idade do paciente (anos)
- anemia: diminuição de glóbulos vermelhos ou hemoglobina (booleano)
- high blood pressure: se o paciente tiver hipertensão (booleano)
- creatinine phosphokinase (CPK): nível da enzima CPK no sangue (mcg/L)
- diabetes: se o paciente tem diabetes (booleano)
- ejection fraction: porcentagem de sangue que sai do coração a cada contração (porcentagem)

- platelets: plaquetas no sangue (quiloplaquetas/ mL)
- sex: mulher ou homem (binário)
- serum creatinin: nível de creatinina sérica no sangue (mg/dL)
- serum sodium: nível de sódio sérico no sangue (mEq/L)
- smoking: se o paciente fuma ou não (booleano)
- time: período de acompanhamento (dias)
- death event [alvo]: se o paciente faleceu durante o período de acompanhamento (booleano)

Carregando o Banco de Dados

```
rm(list = ls())
library(tidyverse)
dados<-read.csv("heart_failure_clinical_records_dataset.csv",sep = ",")
```

Análise exploratória e tratamento dos dados

Toda boa análise deve ser iniciada pela análise exploratória dos dados,vamos começar verificando se temos a presença de dados faltantes utilizando o comando abaixo

```
round(mean(is.na(dados))*100,10)
```

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

Como podemos obervar não temos a presença de nemhum dado faltante ,vamos verificar os tipos das nossas variáveis

```
#glimpse(dados)
```

Como podemos observar algumas classes estão como sendo inteiros ,ou seja,valores numericos ,vamos converte-las para fator

```
dados1<-dados|>   select(anaemia,diabetes,high_blood_pressure,sex,smoking,DEATH_EVENT
                 ) |>
mutate_if(is.integer,as.factor)

dados2<-dados|>   select(age,creatinine_phosphokinase,ejection_fraction,platelets,serum_creatinine,seru
                 )


dados<-cbind(dados2,dados1)




#glimpse(dados)
```

Agora vamos salvar o nosso nosso banco de dados

```
write.csv(dados,"dados_edit.csv",row.names = F,sep = ";")
```

Análise Exploratória

Finalmente após a primeira etapa de preparação dos dados vamos iniciar a análise exploratória de forma prática utilizando a linguagem python , para isso vamos utilizar os seguintes comandos

```
import pandas as pd

import numpy as np
import sweetviz as sv
import warnings


warnings.filterwarnings("ignore")


dados=pd.read_csv("dados1_edit.csv")

my_report = sv.analyze(dados) # cria o reporte e chama de my_report
```

```
#my_report.show_html()
```

```
msk = np.random.rand(len(dados)) < 0.7
train = dados[msk]
test = dados[~msk]

#train.head() # 80%
```

```
my_report = sv.compare([train, 'training set'], [test, 'testing set'])

#my_report.show_html()
```

```
#dados1=dados[0:11]
my_report = sv.compare_intra(dados, dados['DEATH_EVENT']==1,['morreu','não morreu'])
#my_report.show_html()
```

Clique aqui para ver o relatório https://josefersonbarreto.github.io/analise_exp/

Voltando para o R

Após a análise exploratória vamos voltar a utlizar a limguagem R,vamos observar as distribuições de nossas variaveis númericas

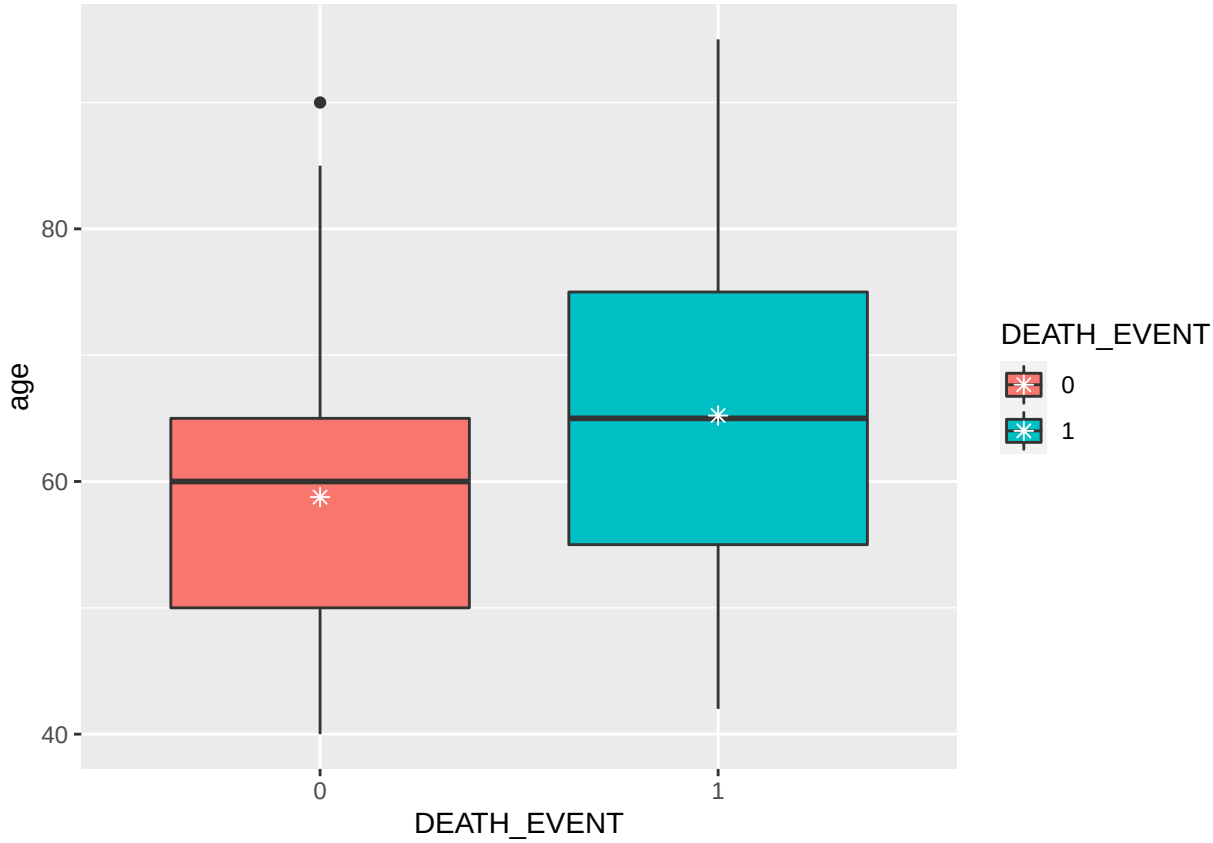
```
#dados ja balanceado
dados<- read.csv("dados_edit.csv",sep = ",")


library(tidyverse)
glimpse(dados)
```

```
## Rows: 299
## Columns: 13
## $ age                       <dbl> 75, 55, 65, 50, 65, 90, 75, 60, 65, 80, 75, 6~
## $ creatinine_phosphokinase <int> 582, 7861, 146, 111, 160, 47, 246, 315, 157, ~
## $ ejection_fraction        <int> 20, 38, 20, 20, 20, 40, 15, 60, 65, 35, 38, 2~
## $ platelets                <dbl> 265000, 263358, 162000, 210000, 327000, 20400~
## $ serum_creatinine         <dbl> 1.90, 1.10, 1.30, 1.90, 2.70, 2.10, 1.20, 1.1~
## $ serum_sodium             <int> 130, 136, 129, 137, 116, 132, 137, 131, 138, ~
## $ time                     <int> 4, 6, 7, 7, 8, 8, 10, 10, 10, 10, 10, 10, 11,~
## $ anaemia                  <int> 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, ~
## $ diabetes                 <int> 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, ~
## $ high_blood_pressure      <int> 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, ~
## $ sex                      <int> 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, ~
## $ smoking                  <int> 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, ~
## $ DEATH_EVENT              <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, ~
```

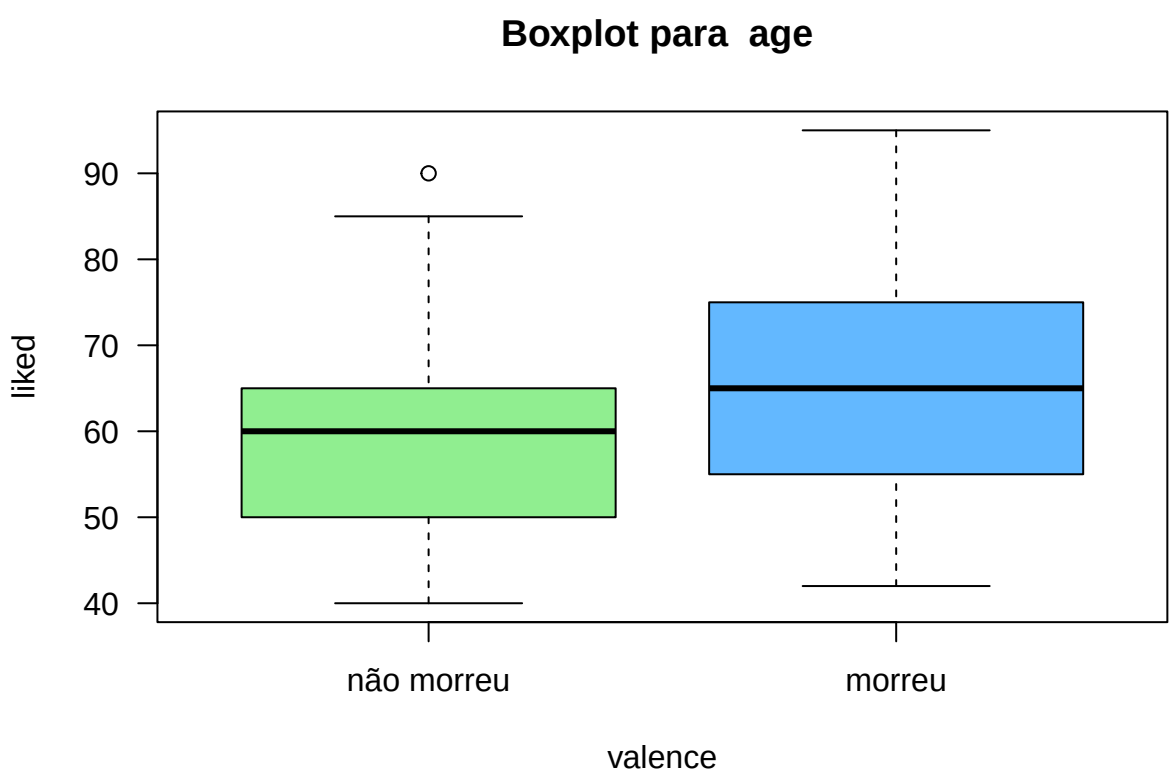
```
dados[8:13]<-dados[8:13] %>% mutate_if(is.integer,as.factor)
dados$age<-round(dados$age,0)



# qplot(dados$age,
#         main = "Histogram energy",
#         xlab = "valores",
#         ylab = "frequencias",
#         fill=I("orange"),
#         col=I("black")  ,
#           col="red",
#         fill="green",
#         alpha= .9) +
#      geom_density(alpha = 0.5)+
  library(plotly)

  ff<-ggplot(dados, aes(x = dados$age)) +

  geom_histogram(aes(y = ..density..),
                 binwidth = 1,     # Amplitude da classe
                 fill = 'dodgerblue',
                 color = 'black')+ # Linha de densidade

  stat_function(fun = dnorm, color='red', size = 2,
                args = list(mean = mean(dados$age),
                            sd = sd(dados$age)))+

  theme_light()
```

pelo gráfico acima podemos perceber que a variável **age(idade)** não aparenta seguir a distribuição normal,mas vamos fazer o teste para confirmar,ma antes vamos ver o boxplot

```
ggplot(dados, aes(x = DEATH_EVENT, y = age, fill = DEATH_EVENT)) +
  geom_boxplot() +
  stat_summary(fun = "mean", geom = "point", shape = 8,
               size = 2, color = "white")
```




```
boxplot (dados$age~dados$DEATH_EVENT,
        main = "Boxplot para  age ",
        xlab = "valence",
        ylab = "liked",
        las = 1,
        col = c ("light green", "steelblue1"),
        names = c("não morreu", "morreu")
        )
```




```
# ggplot(ds, aes(x = label, y = temperature, fill = label)) +
#   geom_boxplot() +
#   stat_summary(fun = "mean", geom = "point", shape = 8,
#                size = 2, color = "white")
```

```
dados$diferenaliked<-as.numeric(dados$DEATH_EVENT)-dados$age
shapiro.test(dados$diferenaliked)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  dados$diferenaliked
## W = 0.97753, p-value = 0.0001222
```

Logo, podemos afirma que não há evidências que a variável **age** siga distribuição normal, ou seja ,vamos utilizar o teste Mann-Whitney para verificar asociação entre nossa variável resposta com **age**

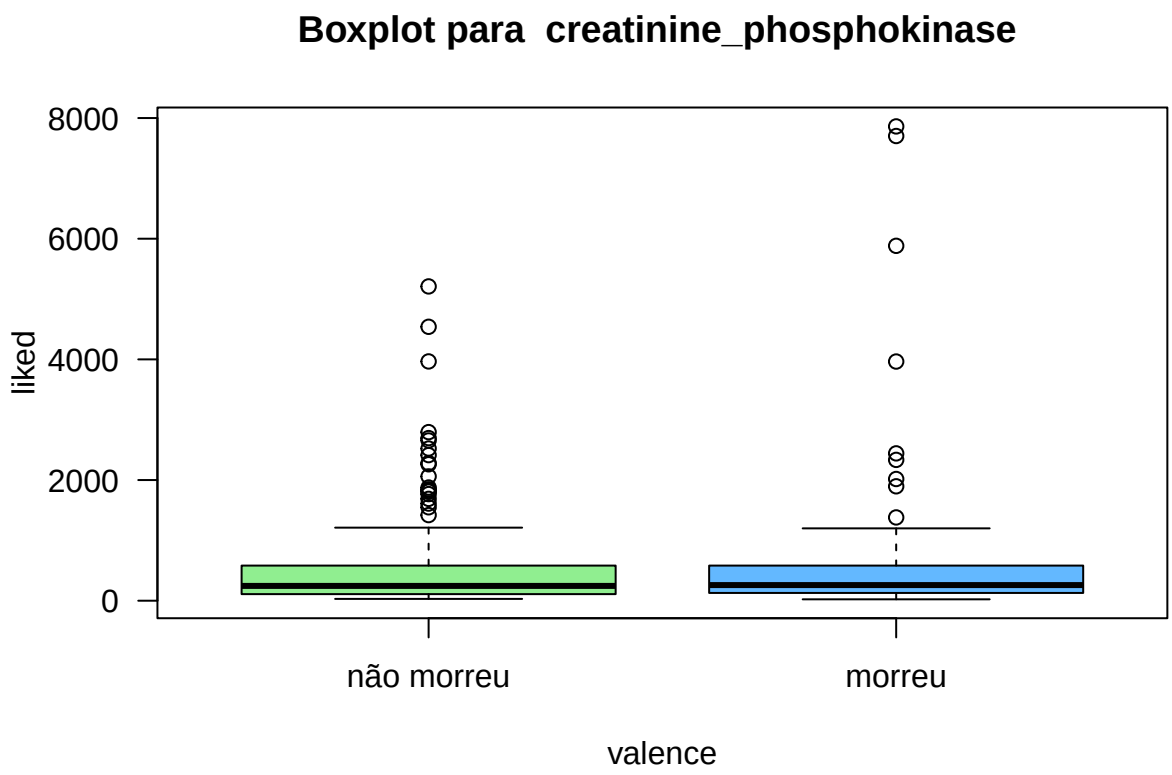
```
wilcox.test(dados$age~dados$DEATH_EVENT, data=dados,correct=T)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  dados$age by dados$DEATH_EVENT
## W = 7119, p-value = 0.0001648
## alternative hypothesis: true location shift is not equal to 0
```

Como o Pvalor $< 0{,}05$, rejeitamos a hipótese nula em favor da hipótese alternativa ,logo , a médiana da diferênça dos valores é realmente diferente de zero , nesse caso, podemos afirmar que existe associação significativa entre as variáveis .

creatinine_phosphokinase

```
boxplot (dados$creatinine_phosphokinase~dados$DEATH_EVENT,
        main = "Boxplot para  creatinine_phosphokinase ",
        xlab = "valence",
        ylab = "liked",
        las = 1,
        col = c ("light green", "steelblue1"),
        names = c("não morreu", "morreu")
        )
```



É possível verificar que temos a presença de outliers nessa variável

```
# ggplot(dados, aes(x = dados$creatinine_phosphokinase)) +
#
#   geom_histogram(aes(y = ..density..),
#                  binwidth = 1,     # Amplitude da classe
#                  fill = 'dodgerblue',
#                  color = 'black')+ # Linha de densidade
#
#   stat_function(fun = dnorm, color='red', size = 2,
```

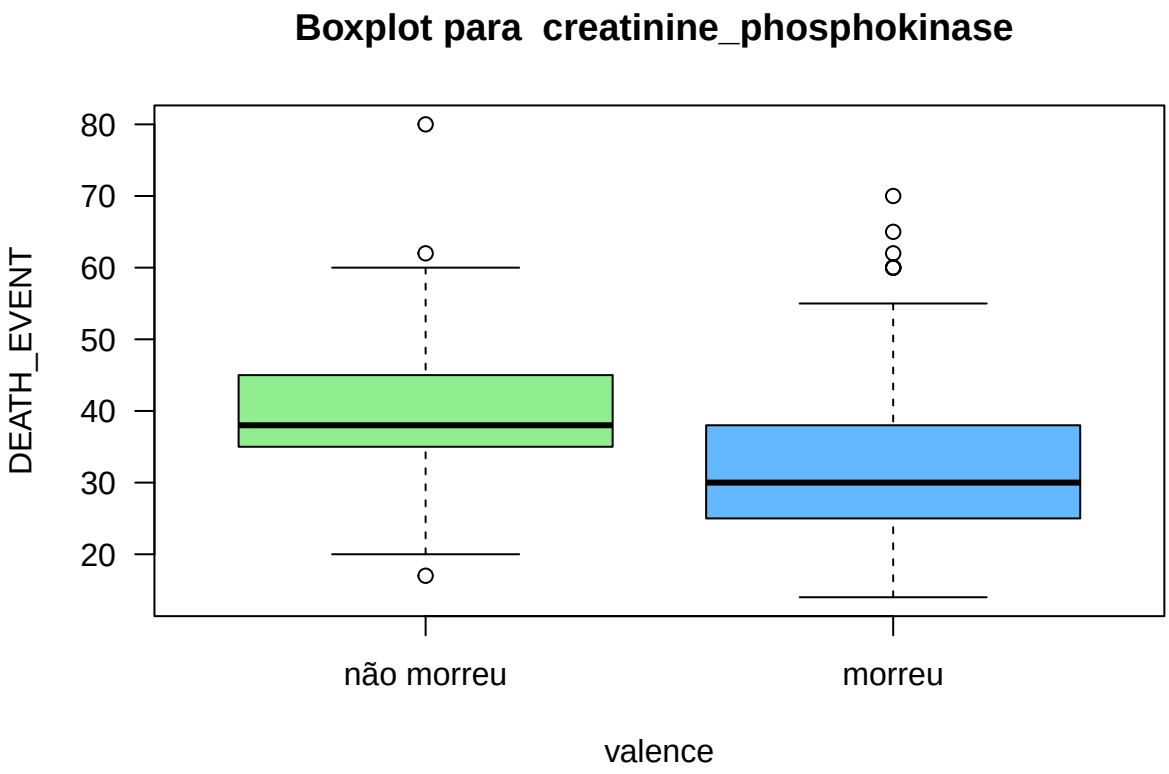
```
#                 args = list(mean = mean(dados$creatinine_phosphokinase),
#                             sd = sd(dados$creatinine_phosphokinase)))+
#
#   theme_light()

wilcox.test(dados$creatinine_phosphokinase~dados$DEATH_EVENT, data=dados,correct=T)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  dados$creatinine_phosphokinase by dados$DEATH_EVENT
## W = 9460, p-value = 0.684
## alternative hypothesis: true location shift is not equal to 0
```

Como Pvalor $>$ 0,05 ,rejeitamos a hipótese nula em favor da hipótese alternativa ,logo , a médiana da diferênça dos valores é realmente diferente de zero , nesse caso, podemos afirmar que existe associação significativa entre as variáveis .

ejection_fraction

```
boxplot (dados$ejection_fraction~dados$DEATH_EVENT,
        main = "Boxplot para  creatinine_phosphokinase ",
        xlab = "valence",
        ylab = "DEATH_EVENT",
        las = 1,
        col = c ("light green", "steelblue1"),
        names = c("não morreu", "morreu")
        )
```



Podemos perceber a presença de outliers, vamos verificar se a variável segue distribuição normal

```
dados$DEATH_EVENT<-as.numeric(dados$DEATH_EVENT)
options(scipen=999)
diferenaliked<-dados$DEATH_EVENT-dados$ejection_fraction
shapiro.test(diferenaliked)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  diferenaliked
## W = 0.95395, p-value = 0.00000004339
```

Como o Pvalor $< 0{,}05$, não rejeita-se a hipótese nula . Nesse sentido , temos que não existe normalidade entre as variáveis testadas , Logo ,o teste T independente não é o mais indacado , assim vamos utilizar o teste Wilcoxon:

```
wilcox.test(dados$ejection_fraction~dados$DEATH_EVENT, data=dados,correct=T)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  dados$ejection_fraction by dados$DEATH_EVENT
## W = 13176, p-value = 0.0000007368
## alternative hypothesis: true location shift is not equal to 0
```

Como Pvalor $>$ 0,05 ,rejeitamos a hipótese nula em favor da hipótese alternativa ,logo , a médiana da diferênça dos valores é realmente diferente de zero , nesse caso, podemos afirmar que existe associação significativa entre as variáveis .

fazendo o mesmo processo para as demais

```
#install.packages("gtsummary")
library(gtsummary)


wilcox.test(dados$platelets~dados$DEATH_EVENT, data=dados,correct=T)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  dados$platelets by dados$DEATH_EVENT
## W = 10300, p-value = 0.4256
## alternative hypothesis: true location shift is not equal to 0
```

```
wilcox.test(dados$serum_creatinine~dados$DEATH_EVENT, data=dados,correct=T)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  dados$serum_creatinine by dados$DEATH_EVENT
## W = 5298, p-value = 0.0000000001581
## alternative hypothesis: true location shift is not equal to 0
```

```
wilcox.test(dados$serum_sodium~dados$DEATH_EVENT, data=dados,correct=T)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  dados$serum_sodium by dados$DEATH_EVENT
## W = 12262, p-value = 0.0002928
## alternative hypothesis: true location shift is not equal to 0
```

```
wilcox.test(dados$time~dados$DEATH_EVENT, data=dados,correct=T)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  dados$time by dados$DEATH_EVENT
## W = 16288, p-value < 0.00000000000000022
## alternative hypothesis: true location shift is not equal to 0
```

```
variaveis<-c("platelets","serum_creatinine","serum_sodium","time")
valor_test_W<-c("0.6833","0.00000005241","0.0008524","0.00000000000002719")

data.frame(cbind(variaveis,valor_test_W))
```

```
##          variaveis         valor_test_W
## 1        platelets               0.6833
## 2 serum_creatinine        0.00000005241
## 3     serum_sodium            0.0008524
## 4             time 0.000000000000002719
```

```
dados$DEATH_EVENT<-as.factor(dados$DEATH_EVENT)
```

Apenas a variável platelets apresenta pvalor maior que 0,05 ,logo, ela não possui associação com a variável resposta , agora vamos verificar associação entre nossas variáveis categóricas e a nossa variável resposta:

Teste Qui-Quadrado para as variáveis resposta

O teste Qui-quadrado é um teste não-paramétrico utilizado, basicamente, para três finalidades específicas, que são:

Verificar as distribuições de probabilidades de cada categoria de uma variável em relação a um valor teórico esperado (aderência),

Verificar se as distribuições das categorias são as mesmas para diferentes subpopulações de interesse (homogeneidade)

Verificar se duas variáveis categóricas são independentes (independência).

Apesar das diferenças em relação às perguntas de pesquisa, o sistema matemático é o mesmo.

Hipóteses H_0 = não há asociação entre as vareiáveis instrumentalness e liked , para $Pvalor > 0,05$

H_1 = há asociação entre as vareiáveis instrumentalness e liked, para $pvalor \leq 0,05$

Antes temos que transformar a varíavel instrumentalness em intervalo :

```
library(tidyverse)
dados3<-dados |>
  dplyr::select(where(is.factor))

#glimpse(dados)


chisq.test(dados3$anaemia,dados3$DEATH_EVENT)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  dados3$anaemia and dados3$DEATH_EVENT
## X-squared = 1.0422, df = 1, p-value = 0.3073
```

```
chisq.test(dados3$diabetes,dados3$DEATH_EVENT)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  dados3$diabetes and dados3$DEATH_EVENT
## X-squared = 0.00000000000000000000000000021617, df = 1, p-value = 1
```

```
chisq.test(dados3$high_blood_pressure,dados3$DEATH_EVENT)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  dados3$high_blood_pressure and dados3$DEATH_EVENT
## X-squared = 1.5435, df = 1, p-value = 0.2141
```

```
chisq.test(dados3$DEATH_EVENT,dados3$sex)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  dados3$DEATH_EVENT and dados3$sex
## X-squared = 0, df = 1, p-value = 1
```

```
chisq.test(dados3$smoking,dados3$DEATH_EVENT)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  dados3$smoking and dados3$DEATH_EVENT
## X-squared = 0.0073315, df = 1, p-value = 0.9318
```

```
library(gmodels)

# CrossTable(dados3$DEATH_EVENT,dados1$sex,
#            expected = T, prop.r = F, prop.c = F, prop.t = T, prop.chisq = F,
#            chisq = T, fisher = T,
#            format = "SPSS")


chisq.test(dados$anaemia,dados$DEATH_EVENT)
```

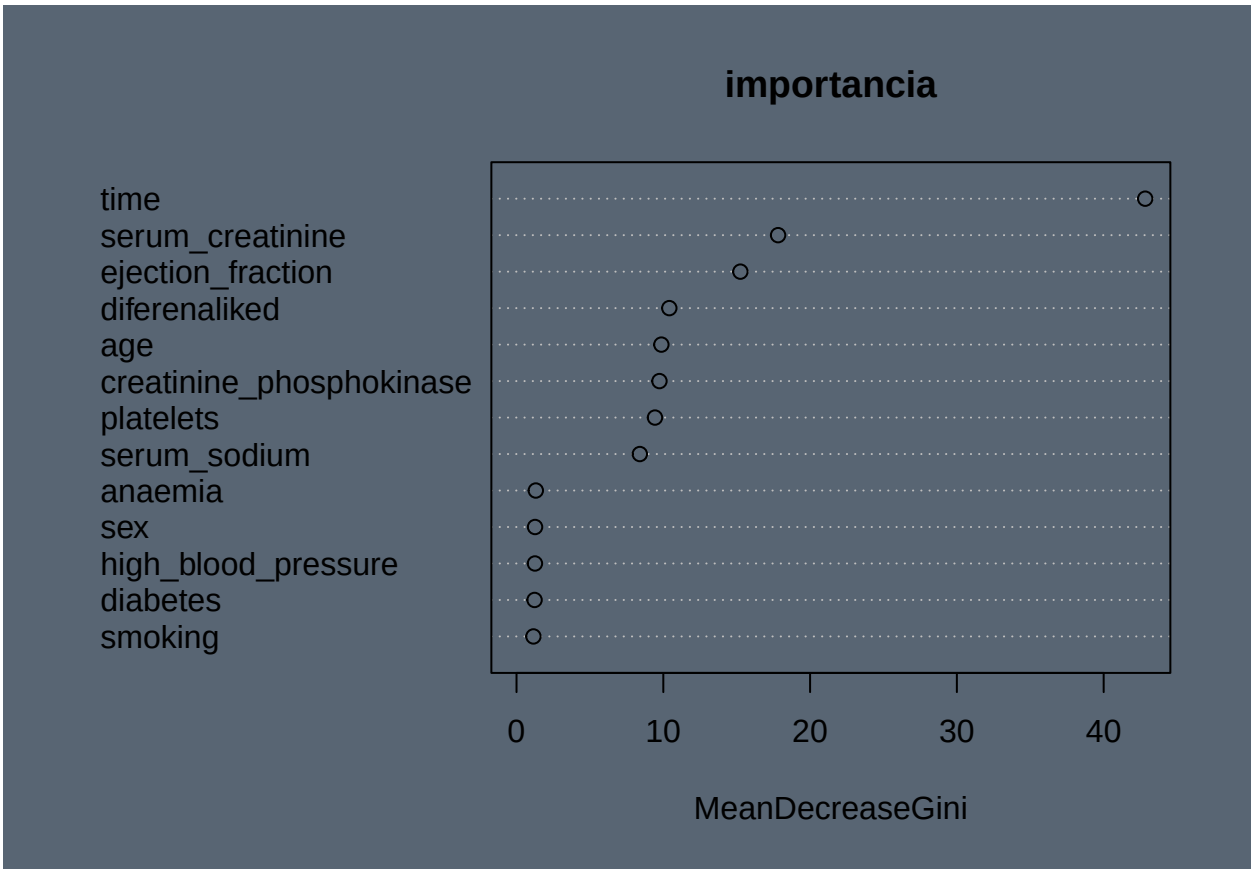
```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  dados$anaemia and dados$DEATH_EVENT
## X-squared = 1.0422, df = 1, p-value = 0.3073
```

```
#factor(dados$anaemia,levels = c("no","yes"),labels = c("0","1"))
```

Pelo teste qui-quadrado nenhuma variável categórica apresentou

```
library(randomForest)



importancia  = randomForest(DEATH_EVENT~ ., data = dados)
```

```
col = importance(importancia)
options(scipen=999)
par(bg = '#586573')
varImpPlot(importancia)
```



Como vimos as variáveis categoricas apresentam uma baixa importancia para o modelo,mas por enquanto vamos mante-las ,nosso próximo passo é converter as variáveis númericas para variáveis dammies , para isso temos que primeiro convertelas para variáveis categóricas:

```
dados4<-dados |>
  dplyr::select(where(is.numeric))

# as variaveis que serão convertidas são




dados5<-dados

dados5<-dados |> arrange(dados$age)
FX_age <- cut(dados5$age,
                   breaks=c(-Inf,50, 60, 70,80, Inf),
                   labels=c("ate50","50_60","60_70","70_80", "80mais"))


dados5$FX_age<-FX_age
```

```r
# faazendo o mesmo para variável creatinine_phosphokinase

dados5<-dados5 |> arrange(dados5$creatinine_phosphokinase)



FX_creat <- cut(dados5$creatinine_phosphokinase,
                  breaks=c(-Inf,100,200,400, 800, Inf),
                  labels=c("ate100","100_200","200_400","400_800","800mais"))

dados5$FX_creat<-FX_creat




#faazendo para variavel ejection_fraction

dados5<-dados5 |> arrange(dados5$ejection_fraction)



FX_ejec <- cut(dados5$ejection_fraction,
                  breaks=c(-Inf,30,40, Inf),
                  labels=c("ate30","30_40","40mais"))


dados5$FX_ejec<-FX_ejec



# faazendo para variavel platelets


dados5<-dados5 |> arrange(dados5$platelets)


FX_plat <- cut(dados5$platelets,
                  breaks=c(-Inf,200000,250000,289000, Inf),
                  labels=c("ate200000","200000_250000", "250000_289000","289000mais"))




# faazendo para variavel serum_creatinine



dados5<-dados5 |> arrange(dados5$serum_creatinine)

FX_serum <- cut(dados5$serum_creatinine,
                  breaks=c(-Inf,0.90,1.10,1.83, Inf),
                  labels=c("ate0.90","0.90_1.10", "1.10_1.83","1.83mais"))
```

```
dados5$FX_serum<-FX_serum
# faazendo para variavel serum_sodium

dados5<-dados5 |> arrange(dados5$serum_sodium)


FX_serum_so <- cut(dados5$serum_sodium,
                   breaks=c(-Inf,134,137, Inf),
                   labels=c("ate134","134_137","137mais"))



dados5$FX_serum_so<-FX_serum_so

# fazendo para variavel time


dados5<-dados5 |> arrange(dados5$time)


FX_time <- cut(dados5$time,
                   breaks=c(-Inf, 55,103,190, Inf),
                   labels=c("ate55","55_103","103_190","190mais"))


dados5$FX_time<-FX_time


dados5<-dados5[-14]


alvo= dados5[6]


#write.csv(dados5,"dados_p_dammies.csv",row.names = F,sep = ";")
```

O proximo passo é transformar as variaveis em dammy

```
dados<-read.csv("dados_p_dammies.csv")

library(tidyverse)

#dados[1:6]<-as.character(dados[1:6])

dados$DEATH_EVENT<-factor(dados$DEATH_EVENT,levels = c("1","2") ,labels = c("0","1"))
#dados[1:13]<-dados|>
#mutate_if(factor)

cols=colnames(dados)
```

```r
dados$anaemia<-as.factor(dados$anaemia)

dados$diabetes<-as.factor(dados$diabetes)

dados$high_blood_pressure<-as.factor(dados$high_blood_pressure)


dados$sex<-as.factor(dados$sex)


dados$smoking<-as.factor(dados$smoking)

dados$DEATH_EVENT<-as.factor(dados$DEATH_EVENT)

dados$FX_age<-as.factor(dados$FX_age)


dados$FX_creat<-as.factor(dados$FX_creat)


dados$FX_ejec<-as.factor(dados$FX_ejec)


dados$FX_plat<-as.factor(dados$FX_plat)

dados$FX_serum<-as.factor(dados$FX_serum)


dados$FX_serum_so<-as.factor(dados$FX_serum_so)


dados$FX_time<-as.factor(dados$FX_time)

#glimpse(dados)



#table(dados$anaemia,dados$DEATH_EVENT)


#round(prop.table(table(dados$anaemia,dados$DEATH_EVENT)),2)



chisq.test(dados$DEATH_EVENT,dados$anaemia)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  dados$DEATH_EVENT and dados$anaemia
## X-squared = 1.0422, df = 1, p-value = 0.3073
```

Criando as variavies dammys

```
library(fastDummies)
### variaveis que vamos criar as dummies
#X = data.frame( FX_GLUCOSE, FX_PREGNANT, FX_PRESSURE, FX_INSULIN, FX_MASS, FX_TRICEPS, FX_PEDIGREE, FX

alvo=dados$DEATH_EVENT
x=dados[-1:-13]
## Aplicando a funcao que vai gerar todas as dummies
dum2 <- dummy_cols(x, remove_selected_columns = T, remove_most_frequent_dummy = F) # dummies mod1

### Novo dataset com as dummies
bd_dum = data.frame( alvo,  dum2  )
```

Finalmente temos nosso dataset criado ,agora podemos criar o nosso modelo, vamos criar nosso primeiro modelo que servirá como base

Balanceamento do Banco de Dados

Como nossos dados não estão com uma proporção equivalente em nossa variável target então vamos reaalizar o balanciamento dos nossos dados , para balancear nosso banco de dados vamos utilizar o **oversampling** e um dos metodos mais utilizados em ciência de dados que é o metodo **SMOTE** que basicamente gera novas observações de forma randomizada para o nosso modelo, como temos poucas observações este método é uma boa opção .

```
# como temos poucas observações vamos usar o metodo oversampling metodo smoote




# Seed
set.seed(301)

# Pacote
#install.packages("DMwR")
install.packages( "C:/Users/joseferson/Documents/joseferson barreto/DMwR_0.4.1.tar.gz", repos=NULL, typ
library(DMwR)

# SMOTE - Synthetic Minority Oversampling Technique

dados_treino_balanceados <- SMOTE(alvo ~ ., bd_dum, perc.over = 400, perc.under = 100)
#
# #teste<-round(SMOTE(DEATH_EVENT ~ ., dados_treino, perc.over = 1, perc.under = 1),0)
# # Checando o balanceamento de classe da variável target
prop.table(table(dados_treino_balanceados$alvo)) * 100
```

```
##
##        0        1
## 44.44444 55.55556
```

```
#
```

Podemos perceber que o nosso modelo agora está melhor balanceado.

Gerendo os Dados de Treino e Teste

```
#
# teste<-dados_treino_balanceados |> arrange(DEATH_EVENT)
#
#
# teste<-dados_treino_balanceados |> arrange(dados_treino_balanceados$FX_age_50_60)
#
#
set.seed(100)
train <- sample(nrow(dados_treino_balanceados), 0.70*nrow(dados_treino_balanceados), replace = FALSE)
TrainSet <- dados_treino_balanceados[train,]
TestSet <- dados_treino_balanceados[-train,]
#

table(TrainSet$alvo)
```

```
##
##   0   1
## 268 336
```

```
#
mod1<- glm(alvo ~ .
           ,family = binomial(link='logit') ,na.action = na.fail,
           data = TrainSet
)


summary(mod1)
```

```
##
## Call:
## glm(formula = alvo ~ ., family = binomial(link = "logit"), data = TrainSet,
##     na.action = na.fail)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.7319  -0.2612   0.0310   0.2391   2.5212
##
## Coefficients: (7 not defined because of singularities)
##                       Estimate Std. Error z value             Pr(>|z|)
## (Intercept)             5.3908     0.9800   5.501   0.0000000377986615 ***
## FX_age_50_60           -0.1997     0.4481  -0.446              0.65587
## FX_age_60_70           -1.3110     0.5517  -2.377              0.01748 *
## FX_age_70_80            1.3147     0.5485   2.397              0.01653 *
## FX_age_80mais           1.9234     0.9222   2.086              0.03701 *
## FX_age_ate50                NA         NA      NA                   NA
## FX_creat_100_200        1.7928     0.6908   2.595              0.00945 **
## FX_creat_200_400        1.3001     0.6485   2.005              0.04500 *
## FX_creat_400_800        1.1026     0.6437   1.713              0.08674 .
## FX_creat_800mais        1.2193     0.6979   1.747              0.08061 .
## FX_creat_ate100             NA         NA      NA                   NA
```

```
## FX_ejec_30_40          -2.8343     0.4263  -6.648   0.0000000000296136 ***
## FX_ejec_40mais         -2.6921     0.4401  -6.117   0.0000000009554704 ***
## FX_ejec_ate30              NA         NA      NA                    NA
## FX_plat_200000_250000  -0.4063     0.5263  -0.772               0.44006
## FX_plat_250000_289000  -1.1059     0.5302  -2.086               0.03701 *
## FX_plat_289000mais      0.1067     0.5173   0.206               0.83663
## FX_plat_ate200000          NA         NA      NA                    NA
## FX_serum_0.90_1.10      1.0621     0.5250   2.023               0.04307 *
## FX_serum_1.10_1.83      2.0636     0.5278   3.910   0.0000923839641963 ***
## FX_serum_1.83mais       2.0036     0.6554   3.057               0.00223 **
## FX_serum_ate0.90           NA         NA      NA                    NA
## FX_serum_so_134_137    -1.5582     0.4920  -3.167               0.00154 **
## FX_serum_so_137mais    -1.4050     0.4584  -3.065               0.00217 **
## FX_serum_so_ate134         NA         NA      NA                    NA
## FX_time_103_190        -5.5066     0.7050  -7.810   0.0000000000000057 ***
## FX_time_190mais        -7.9781     0.8395  -9.504 < 0.0000000000000002 ***
## FX_time_55_103         -5.1629     0.6959  -7.420   0.0000000000001175 ***
## FX_time_ate55              NA         NA      NA                    NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 829.65  on 603  degrees of freedom
## Residual deviance: 279.75  on 582  degrees of freedom
## AIC: 323.75
##
## Number of Fisher Scoring iterations: 7
```

É notório que algumas variáveis preditoras apresentaram valores ausentes em seus resultados ,vamos usar a função **step** com base no modelo **mod1** que ajuda a encontrar o melhor modelo ,

```
# modelo step
#step(mod1)
#
#
#
mod2<-glm(formula = alvo ~ FX_age_60_70 + FX_age_70_80 + FX_age_80mais +
      FX_creat_100_200 + FX_creat_200_400 + FX_creat_400_800 +
      FX_creat_800mais + FX_ejec_30_40 + FX_ejec_40mais + FX_plat_200000_250000 +
      FX_plat_250000_289000 + FX_plat_289000mais + FX_serum_0.90_1.10 +
      FX_serum_1.10_1.83 + FX_serum_1.83mais + FX_serum_so_134_137 +
      FX_serum_so_137mais + FX_time_103_190 + FX_time_190mais +
      FX_time_55_103, family = binomial(link = "logit"), data = TrainSet,
    na.action = na.fail)
#
#
summary(mod2)
```

```
##
## Call:
## glm(formula = alvo ~ FX_age_60_70 + FX_age_70_80 + FX_age_80mais +
##     FX_creat_100_200 + FX_creat_200_400 + FX_creat_400_800 +
```

```
##     FX_creat_800mais + FX_ejec_30_40 + FX_ejec_40mais + FX_plat_200000_250000 +
##     FX_plat_250000_289000 + FX_plat_289000mais + FX_serum_0.90_1.10 +
##     FX_serum_1.10_1.83 + FX_serum_1.83mais + FX_serum_so_134_137 +
##     FX_serum_so_137mais + FX_time_103_190 + FX_time_190mais +
##     FX_time_55_103, family = binomial(link = "logit"), data = TrainSet,
##     na.action = na.fail)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.7438  -0.2662   0.0325   0.2401   2.5615
##
## Coefficients:
##                       Estimate Std. Error z value             Pr(>|z|)
## (Intercept)             5.2803     0.9462   5.581  0.00000002396665413 ***
## FX_age_60_70           -1.2005     0.4930  -2.435             0.014891 *
## FX_age_70_80            1.4236     0.4914   2.897             0.003768 **
## FX_age_80mais           2.0369     0.8843   2.303             0.021260 *
## FX_creat_100_200        1.8269     0.6851   2.667             0.007662 **
## FX_creat_200_400        1.3373     0.6418   2.083             0.037208 *
## FX_creat_400_800        1.1423     0.6367   1.794             0.072783 .
## FX_creat_800mais        1.2209     0.6961   1.754             0.079446 .
## FX_ejec_30_40          -2.8266     0.4249  -6.652  0.00000000002881731 ***
## FX_ejec_40mais         -2.6905     0.4405  -6.108  0.00000000100624292 ***
## FX_plat_200000_250000  -0.4010     0.5255  -0.763             0.445446
## FX_plat_250000_289000  -1.1052     0.5289  -2.090             0.036657 *
## FX_plat_289000mais      0.1083     0.5176   0.209             0.834265
## FX_serum_0.90_1.10      1.0444     0.5241   1.993             0.046306 *
## FX_serum_1.10_1.83      2.0293     0.5220   3.888             0.000101 ***
## FX_serum_1.83mais       1.9593     0.6460   3.033             0.002422 **
## FX_serum_so_134_137    -1.5618     0.4907  -3.183             0.001458 **
## FX_serum_so_137mais    -1.4262     0.4555  -3.131             0.001741 **
## FX_time_103_190        -5.4908     0.7043  -7.796  0.00000000000000639 ***
## FX_time_190mais        -7.9853     0.8409  -9.496 < 0.0000000000000002 ***
## FX_time_55_103         -5.1782     0.6966  -7.434  0.00000000000010551 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 829.65  on 603  degrees of freedom
## Residual deviance: 279.94  on 583  degrees of freedom
## AIC: 321.94
##
## Number of Fisher Scoring iterations: 7
```

```
#
```

O modelo **mod2** foi o modelo encontrado pela função step com base no modelo 1,vemos que algumas variáveis não forma seguinificativas para o modelo, vamos criar um terceiro modelo que será baseado no modelo **mod1** sem as variáveis que apresentaram valores ausentes

```
#colnames(TrainSet)
```

```r
mod3<-glm(formula =  alvo~  +FX_age_50_60+   FX_age_60_70+FX_age_70_80+FX_age_80mais +       FX_creat_
  FX_creat_400_800+ FX_creat_800mais+FX_ejec_30_40+
 FX_ejec_40mais         +FX_plat_200000_250000 +FX_plat_250000_289000+
 FX_plat_289000mais+     FX_serum_0.90_1.10+    FX_serum_1.10_1.83+
 FX_serum_1.83mais      +FX_serum_so_134_137+   FX_serum_so_137mais+
   FX_time_103_190+  FX_time_190mais+  FX_time_55_103,family = binomial(link = "logit"), data = TrainSe
    na.action = na.fail)



summary(mod3)
```

```
##
## Call:
## glm(formula = alvo ~ +FX_age_50_60 + FX_age_60_70 + FX_age_70_80 +
##     FX_age_80mais + FX_creat_100_200 + FX_creat_200_400 + FX_creat_400_800 +
##     FX_creat_800mais + FX_ejec_30_40 + FX_ejec_40mais + FX_plat_200000_250000 +
##     FX_plat_250000_289000 + FX_plat_289000mais + FX_serum_0.90_1.10 +
##     FX_serum_1.10_1.83 + FX_serum_1.83mais + FX_serum_so_134_137 +
##     FX_serum_so_137mais + FX_time_103_190 + FX_time_190mais +
##     FX_time_55_103, family = binomial(link = "logit"), data = TrainSet,
##     na.action = na.fail)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.7319  -0.2612   0.0310   0.2391   2.5212
##
## Coefficients:
##                       Estimate Std. Error z value             Pr(>|z|)
## (Intercept)             5.3908     0.9800   5.501   0.0000000377986615 ***
## FX_age_50_60           -0.1997     0.4481  -0.446              0.65587
## FX_age_60_70           -1.3110     0.5517  -2.377              0.01748 *
## FX_age_70_80            1.3147     0.5485   2.397              0.01653 *
## FX_age_80mais           1.9234     0.9222   2.086              0.03701 *
## FX_creat_100_200        1.7928     0.6908   2.595              0.00945 **
## FX_creat_200_400        1.3001     0.6485   2.005              0.04500 *
## FX_creat_400_800        1.1026     0.6437   1.713              0.08674 .
## FX_creat_800mais        1.2193     0.6979   1.747              0.08061 .
## FX_ejec_30_40          -2.8343     0.4263  -6.648   0.0000000000296136 ***
## FX_ejec_40mais         -2.6921     0.4401  -6.117   0.0000000009554704 ***
## FX_plat_200000_250000  -0.4063     0.5263  -0.772              0.44006
## FX_plat_250000_289000  -1.1059     0.5302  -2.086              0.03701 *
## FX_plat_289000mais      0.1067     0.5173   0.206              0.83663
## FX_serum_0.90_1.10      1.0621     0.5250   2.023              0.04307 *
## FX_serum_1.10_1.83      2.0636     0.5278   3.910   0.0000923839641963 ***
## FX_serum_1.83mais       2.0036     0.6554   3.057              0.00223 **
## FX_serum_so_134_137    -1.5582     0.4920  -3.167              0.00154 **
## FX_serum_so_137mais    -1.4050     0.4584  -3.065              0.00217 **
## FX_time_103_190        -5.5066     0.7050  -7.810   0.0000000000000057 ***
## FX_time_190mais        -7.9781     0.8395  -9.504 < 0.0000000000000002 ***
## FX_time_55_103         -5.1629     0.6959  -7.420   0.0000000000001175 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 829.65  on 603  degrees of freedom
## Residual deviance: 279.75  on 582  degrees of freedom
## AIC: 323.75
##
## Number of Fisher Scoring iterations: 7
```

Percebe-se que no modelo **mod3** várias variáveis não apresentaram significância para o modelo,logo ,vamos criar o modelo **mod4** sem essas variáveis

```
mod4<-glm(formula =  alvo~        FX_creat_100_200+FX_creat_200_400  +
  FX_creat_400_800+ FX_creat_800mais+FX_ejec_30_40+
 FX_ejec_40mais          +FX_plat_250000_289000+
 FX_plat_289000mais+     FX_serum_0.90_1.10+    FX_serum_1.10_1.83+
 FX_serum_1.83mais      +FX_serum_so_134_137+   FX_serum_so_137mais+
   FX_time_103_190+  FX_time_190mais+  FX_time_55_103,family = binomial(link = "logit"), data = TrainSe
    na.action = na.fail)



summary(mod4)
```

```
##
## Call:
## glm(formula = alvo ~ FX_creat_100_200 + FX_creat_200_400 + FX_creat_400_800 +
##     FX_creat_800mais + FX_ejec_30_40 + FX_ejec_40mais + FX_plat_250000_289000 +
##     FX_plat_289000mais + FX_serum_0.90_1.10 + FX_serum_1.10_1.83 +
##     FX_serum_1.83mais + FX_serum_so_134_137 + FX_serum_so_137mais +
##     FX_time_103_190 + FX_time_190mais + FX_time_55_103, family = binomial(link = "logit"),
##     data = TrainSet, na.action = na.fail)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.5157  -0.3178   0.0449   0.2638   2.4989
##
## Coefficients:
##                       Estimate Std. Error z value             Pr(>|z|)
## (Intercept)            4.65224    0.83545   5.569  0.00000002568566214 ***
## FX_creat_100_200       1.66080    0.62525   2.656             0.007902 **
## FX_creat_200_400       1.27842    0.61470   2.080             0.037550 *
## FX_creat_400_800       1.45980    0.60769   2.402             0.016296 *
## FX_creat_800mais       1.31196    0.62689   2.093             0.036365 *
## FX_ejec_30_40         -2.50488    0.38266  -6.546  0.00000000005910508 ***
## FX_ejec_40mais        -2.58526    0.42189  -6.128  0.00000000089087483 ***
## FX_plat_250000_289000 -0.78920    0.40273  -1.960             0.050040 .
## FX_plat_289000mais     0.03489    0.40528   0.086             0.931400
## FX_serum_0.90_1.10     1.24561    0.49926   2.495             0.012599 *
## FX_serum_1.10_1.83     2.22869    0.47617   4.680  0.00000286279035521 ***
## FX_serum_1.83mais      2.11971    0.58780   3.606             0.000311 ***
## FX_serum_so_134_137   -1.94071    0.45742  -4.243  0.00002207791860359 ***
## FX_serum_so_137mais   -1.52884    0.42775  -3.574             0.000351 ***
```

```
## FX_time_103_190       -5.00757    0.60653  -8.256 < 0.0000000000000002 ***
## FX_time_190mais       -7.81326    0.76147 -10.261 < 0.0000000000000002 ***
## FX_time_55_103        -4.70866    0.60071  -7.839  0.00000000000000456 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 829.65  on 603  degrees of freedom
## Residual deviance: 306.61  on 587  degrees of freedom
## AIC: 340.61
##
## Number of Fisher Scoring iterations: 7
```

```
mod5<-glm(formula = alvo ~ FX_age_60_70 + FX_age_70_80 + FX_age_80mais +
    FX_creat_100_200 + FX_creat_200_400 + FX_creat_400_800 +
    FX_creat_800mais + FX_ejec_30_40 + FX_ejec_40mais + FX_plat_250000_289000 +
    FX_serum_0.90_1.10 + FX_serum_1.10_1.83 + FX_serum_1.83mais +
    FX_serum_so_134_137 + FX_serum_so_137mais + FX_time_103_190 +
    FX_time_190mais + FX_time_55_103, family = binomial(link = "logit"),
    data = TrainSet, na.action = na.fail)


summary(mod5)
```

```
##
## Call:
## glm(formula = alvo ~ FX_age_60_70 + FX_age_70_80 + FX_age_80mais +
##     FX_creat_100_200 + FX_creat_200_400 + FX_creat_400_800 +
##     FX_creat_800mais + FX_ejec_30_40 + FX_ejec_40mais + FX_plat_250000_289000 +
##     FX_serum_0.90_1.10 + FX_serum_1.10_1.83 + FX_serum_1.83mais +
##     FX_serum_so_134_137 + FX_serum_so_137mais + FX_time_103_190 +
##     FX_time_190mais + FX_time_55_103, family = binomial(link = "logit"),
##     data = TrainSet, na.action = na.fail)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.6462  -0.2642   0.0341   0.2601   2.5919
##
## Coefficients:
##                       Estimate Std. Error z value             Pr(>|z|)
## (Intercept)             5.2318     0.9352   5.594   0.0000000221583352 ***
## FX_age_60_70           -1.1472     0.4816  -2.382             0.017217 *
## FX_age_70_80            1.3930     0.4905   2.840             0.004512 **
## FX_age_80mais           1.9879     0.9007   2.207             0.027313 *
## FX_creat_100_200        1.8168     0.6632   2.739             0.006158 **
## FX_creat_200_400        1.3587     0.6268   2.168             0.030196 *
## FX_creat_400_800        1.2116     0.6194   1.956             0.050468 .
## FX_creat_800mais        1.2537     0.6610   1.897             0.057880 .
## FX_ejec_30_40          -2.8958     0.4214  -6.872   0.0000000000063521 ***
## FX_ejec_40mais         -2.7307     0.4420  -6.178   0.0000000006484618 ***
## FX_plat_250000_289000  -1.0102     0.4240  -2.383             0.017182 *
## FX_serum_0.90_1.10      1.1030     0.5080   2.171             0.029914 *
## FX_serum_1.10_1.83      1.9965     0.5195   3.843             0.000122 ***
```

```
## FX_serum_1.83mais       1.7862     0.5944   3.005             0.002657 **
## FX_serum_so_134_137    -1.6256     0.4822  -3.371             0.000749 ***
## FX_serum_so_137mais    -1.4215     0.4534  -3.136             0.001715 **
## FX_time_103_190        -5.4919     0.6935  -7.919   0.0000000000000024 ***
## FX_time_190mais        -8.0243     0.8341  -9.621 < 0.0000000000000002 ***
## FX_time_55_103         -5.1938     0.6824  -7.611   0.0000000000000271 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 829.65  on 603  degrees of freedom
## Residual deviance: 281.12  on 585  degrees of freedom
## AIC: 319.12
##
## Number of Fisher Scoring iterations: 7
```

Modelo 5 é o modelo step baseado no modelo 3 que é o modelo com todas as variáveis onde cada classe dammy é tratada como **k-1** removendo as colunas que estavam como NA no primeiro modelo.

Avaliação da Performance do modelo

O nosso modelo mod2 que é o nosso modelo final, o critério utilizado para seleção do modelo foi aquele que apresentou maiores resultados de sensibilidade e melhor equilíbrio no Gráfico das probabilidades,logo, o modelo 2 foi o escolhido .
Na seguência vamos fazer previsões usaremos a função predict e os atributos no dataset de teste , elém disso o tipo é response pois queremos a variável resposta.

```
#
#
#
# analisando a curva rock e acuracia

library(caret)
probs_logistica <- predict(mod2, TestSet ,type='response' )
pred_log_test <-ifelse( probs_logistica > 0.5, 1 , 0)
tab_test = table(pred_log_test,TestSet$alvo)
confusionMatrix( tab_test )
```

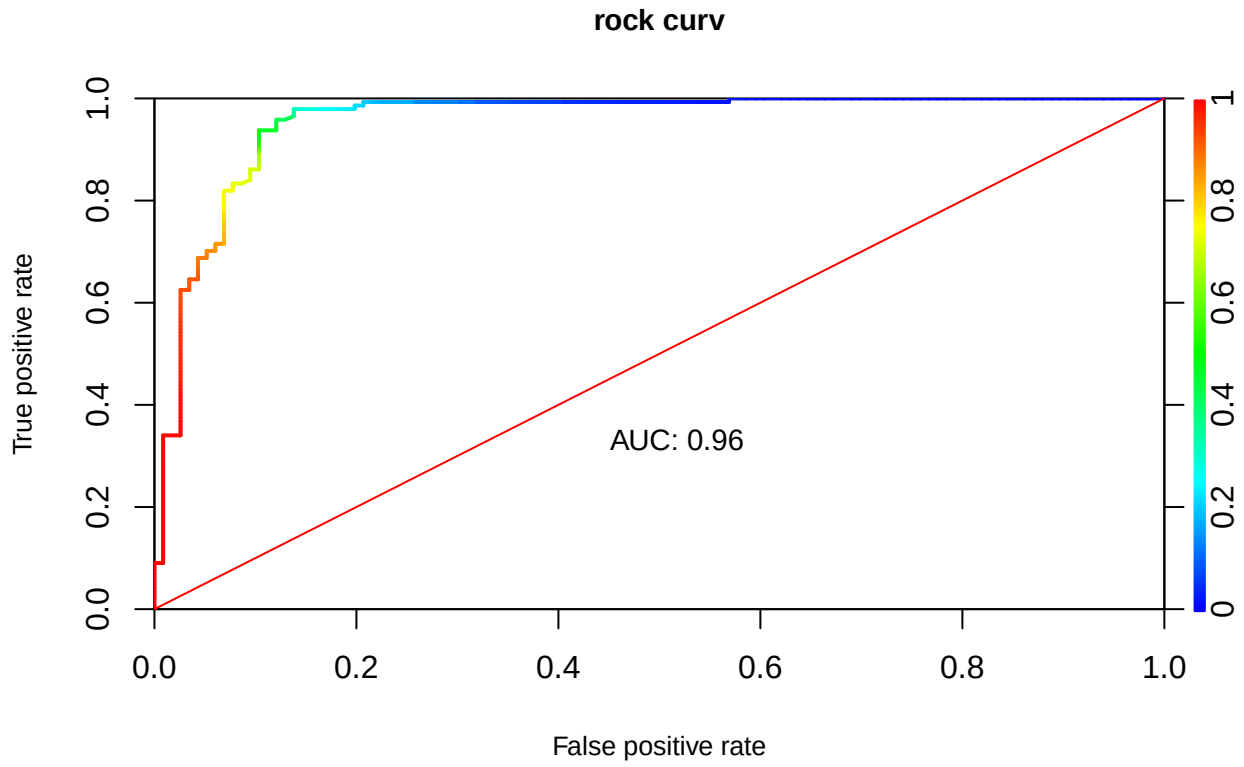
```
## Confusion Matrix and Statistics
##
##
## pred_log_test   0   1
##             0 104  12
##             1  12 132
##
##                Accuracy : 0.9077
##                  95% CI : (0.8658, 0.94)
##     No Information Rate : 0.5538
##     P-Value [Acc > NIR] : <0.0000000000000002
##
##                   Kappa : 0.8132
```

```
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.8966
##             Specificity : 0.9167
##          Pos Pred Value : 0.8966
##          Neg Pred Value : 0.9167
##              Prevalence : 0.4462
##          Detection Rate : 0.4000
##    Detection Prevalence : 0.4462
##       Balanced Accuracy : 0.9066
##
##        'Positive' Class : 0
##
```

Verificando a curva Rock

```
library(pROC)


pred_roc_reg_log <-
  dplyr::tibble(
    probs_logistica,
    "survived" = as.factor(as.numeric(TestSet$alvo))
  ) %>% arrange(desc(probs_logistica))



roc_reg_log <- pROC::roc(pred_roc_reg_log$survived , pred_roc_reg_log$probs_logistica, percent = TRUE)
```

```
library(ROCR)

df <- data.frame(pred_roc_reg_log)
pred <- prediction(as.numeric(pred_roc_reg_log$probs_logistica), as.numeric(pred_roc_reg_log$survived))

plot.roc.curve <- function(predictions, title.text){
  perf <- performance(predictions, "tpr", "fpr")
  plot(perf,colorize=TRUE,lty = 1, lwd = 2,
       main = title.text, cex.main = 0.9, cex.lab = 0.8,xaxs = "i", yaxs = "i")
  abline(0,1, col = "red")
  auc <- performance(predictions,"auc")
  auc <- unlist(slot(auc, "y.values"))
  auc <- round(auc,2)
  legend(0.4,0.4,legend = c(paste0("AUC: ",auc)), cex = 0.9, bty = "n", box.col = "white")

}

plot.roc.curve(pred,"rock curv")
```



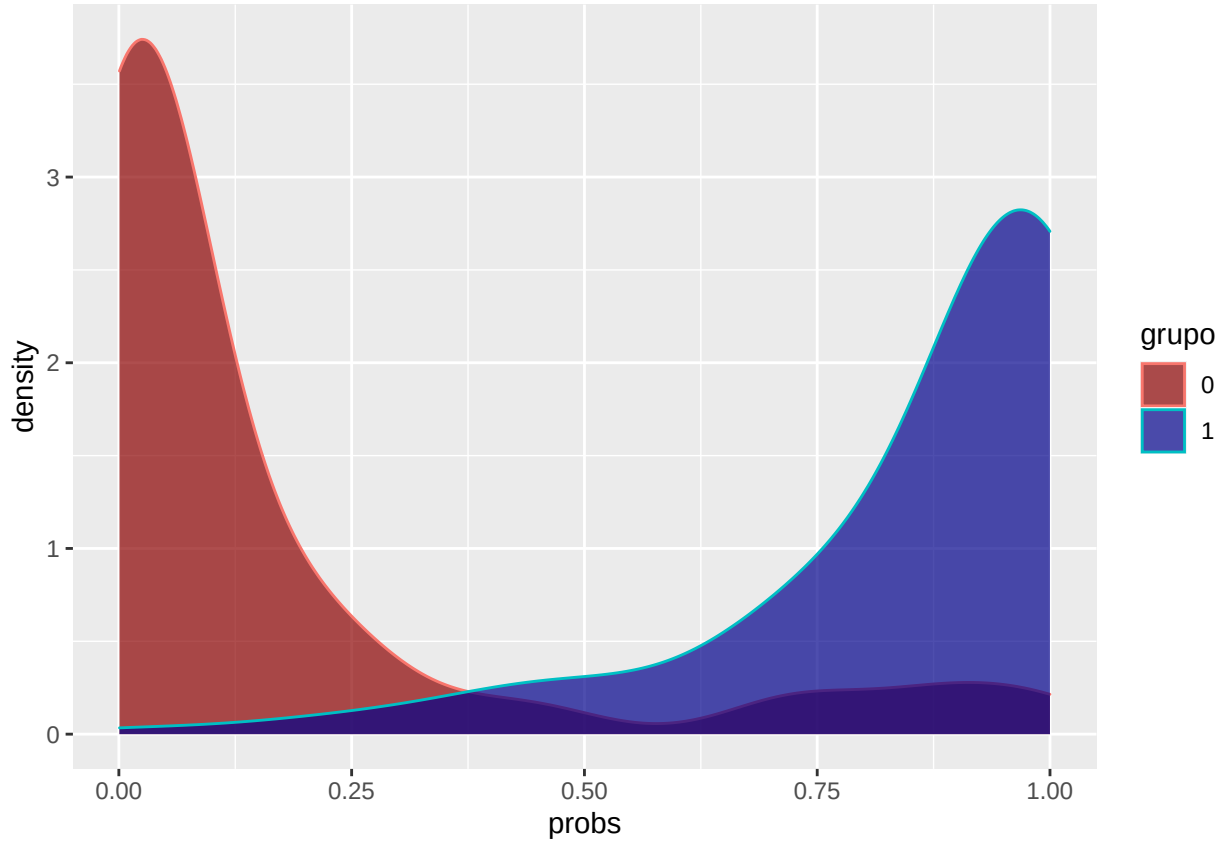

```
prop.table(table(TestSet$alvo))
```

```
##
##         0         1
## 0.4461538 0.5538462
```

Aqui podemos ver a curva ROC, onde podemos observar a relação entre a taxa de falsos positivos e a taxa de verdadeiros positivos, a linha vermelha é uma espécie de linha de corte, que representa 50% de precisão , a nossa meta é manter a linha preta no primeiro quadrante e o mais próximo de 1,logo,nosso modelo apresenta bons resultados.

Vamos observar os graficos das classificações

```
#### Grafico das probabilidades

library(ggplot2)
curvas = data.frame( probs = probs_logistica,  grupo = as.factor(TestSet$alvo) )
cols <- c("darkred", "darkblue")

ggplot(curvas, aes(x = probs, color = grupo)) +
  geom_density(alpha = 0.7, aes(x=probs, group=grupo, fill=grupo), adjust=2 ) +
  scale_fill_manual(values = cols)
```



as figuras 0 e 1) no gráfico acima mostra a Representação das curvas de distribuição de resultados para testes (testes 0 e 1) que visam classificar se o paciente morreu ou não morreu. Quanto mais distantes as curvas estiverem , melhor é a classificação feita pelo modelo,podemos perceber que o modelo **mod2** é capaz de realizar boas classificações.

Conclusão

Podemos perceber que o nosso modelo 2(mod2) ,o modelo step baseado no modelo 1 apresentou boas métricas ,sendo um dos principais candidatos a implementação, ou seja, é perceptível que os modelos de regressão logistica são bastante eficases quando temos variáveis dicotômicas e um pré-processamento adequado, um possível próximo passo para nosso trabalho séria realizar o deploy do modelo,já que o modelo final(mod2) apresentou boas métricas tanto na matriz de confusão quanto na curva ROC.

Referências

OQUE É INSUFICIÊNCIA CARDIÁCA? https://www.medtronic.com/br-pt/your-health/conditions/heart-failure.html . Acesso em :20/11/22

QUI-QUADRADO https://www.medtronic.com/br-pt/your-health/conditions/heart-failure.htmlAcesso em :20/11/22

APLICAÇÃO DE TESTES DE NORMALIDADE EM PUBLICAÇÕES NACIONAIS: LEVANTAMENTO BIBLIOGRÁFICO - Machado, A. F., de Almeida, A. C., Araújo, A. C., Ferrari, D., Lemes, Ítalo R., Faria, N. C. S., Lima, T. de S., & Fernandes, R. A. (2015). APLICAÇÃO DE TESTES DE NORMALIDADE

EM PUBLICAÇÕES NACIONAIS: LEVANTAMENTO BIBLIOGRÁFICO. Colloquium Vitae. ISSN: 1984-6436, 6(1), 01–10. Recuperado de https://revistas.unoeste.br/index.php/cv/article/view/1003

Hoo ZH, Candlish J, Teare D. What is an ROC curve? Emerg Med J. 2017;34(6):357-9. http://dx.doi.org/10.1136/emermed-2017-206735 PMid:28302644.

Polo TCF, Miot HA. Aplicações da curva ROC em estudos clínicos e experimentais. J Vasc Bras. 2020;19: e20200186. https://doi.org/10.1590/1677-5449.200186