Modelo Regressão - COLECISTECTOMIA

Aline Silva Medeiros, Christian Jacobsen Teixeira, Marcos de Morais Silva

08/05/2022

Introdução

Objetivo do Modelo

O tempo de permanência é um indicador do tempo que o paciente fica internado. Por conta de alguma complicação, uma internação que seria breve se torna algo mais complexo. Dessa forma, o custo médico do paciente aumenta consideralvemente.

O objetivo desse projeto é identificar a probabilidade que um determinado pode fica internado de acordo com CID de alta.

Acurácia do modelo

Devido a questão da generalização, evitando problemas de underfitting (muito erros do modelo) e overfitting (modelo sensível a outliers), uma acurácia em torno de 60% seria o ideal para esse modelo.

Dicionário de Dados

Para um melhor entendimento do modelo, segue uma definição dos atributos (variáveis) utilizados:

- CARATER: Eletivo ou Urgência (Categórica)
- AHRQ_DIAG_DTL_CATGY_CD: Código do Grupo CID (Categórica)
- Alta Complexidade: Classificação em Sim ou Não, ver Obs. (Categórica)
- IDADE: Idade do paciente (Numérica)
- GENERO: Sexo do paciente (Categórica)

OBSERVAÇÔES

Alta Complexidade - Acima de 7 dias

Fonte dos Dados

- Fonte: AORTA (OPTUM)
- Período: Jan/2018 - Jan/2022

Carga e Prepação do RStudio

Diretório de trabalho

```
setwd('C:/FCD/Projeto_Machine_Learning')
getwd()
```

```
## [1] "C:/FCD/Projeto_Machine_Learning"
```

Pacotes utilizados

```
library(ggplot2)
library(readxl)
library(dplyr)
library(forcats)
library(rmarkdown)
library(rcompanion)
library(Amelia)
library(caret)
library(ROSE)
library(ROCR)
```

Carga do dataset

```
#Carga dos dados
BD <- read_excel(file.choose())
View(BD)

#Filtrando as colunas que serão usadas no modelo

col_names <- c('CARATER','Classe1','Alta_Complexidade','IDADE', 'GENERO')
BD_MODELO <- BD[,col_names]

rm(col_names)

#Visualização dos dados
head(BD_MODELO)
```

```
## # A tibble: 6 x 5
##   CARATER  Classe1 Alta_Complexidade IDADE GENERO
##   <chr>    <chr>   <chr>             <dbl> <chr>
## 1 Elective Outros  N                    46 Feminino
## 2 Elective Outros  N                    29 Feminino
## 3 Elective Outros  N                    29 Feminino
## 4 Elective Outros  N                    41 Masculino
## 5 Elective Outros  N                    45 Feminino
## 6 Elective Outros  N                    30 Feminino
```

```
str(BD_MODELO)
```

```
## tibble [52,767 x 5] (S3: tbl_df/tbl/data.frame)
##  $ CARATER          : chr [1:52767] "Elective" "Elective" "Elective" "Elective" ...
##  $ Classe1          : chr [1:52767] "Outros" "Outros" "Outros" "Outros" ...
##  $ Alta_Complexidade: chr [1:52767] "N" "N" "N" "N" ...
##  $ IDADE            : num [1:52767] 46 29 29 41 45 30 25 47 47 39 ...
##  $ GENERO           : chr [1:52767] "Feminino" "Feminino" "Feminino" "Masculino" ...
```

```
summary(BD_MODELO)
```

```
##    CARATER            Classe1          Alta_Complexidade      IDADE
##  Length:52767       Length:52767       Length:52767       Min.   :  1.0
##  Class :character   Class :character   Class :character   1st Qu.: 36.0
##  Mode  :character   Mode  :character   Mode  :character   Median : 44.0
##                                                           Mean   : 45.9
```

```
##                                                       3rd Qu.: 55.0
##                                                       Max.   :105.0
##                                                       NA's   :1
##     GENERO
##  Length:52767
##  Class :character
##  Mode  :character
##
##
##
##
```

Pré-Processamento dos Dados

Como o modelo de Regressão Logística não aceita strings nas variáveis, devemos converter esse atributos como categórico. No R, essa categoria é Factor. Como temos poucos valores NAs, conforme observado na análise exploratória. Ao longo dessa etapa, ficou claro que as classes estão desbalanceadas no dataset do modelo. É necessário fazer um ajuste no dataset para balancear as classes e evitar que modelo fique tendencioso.

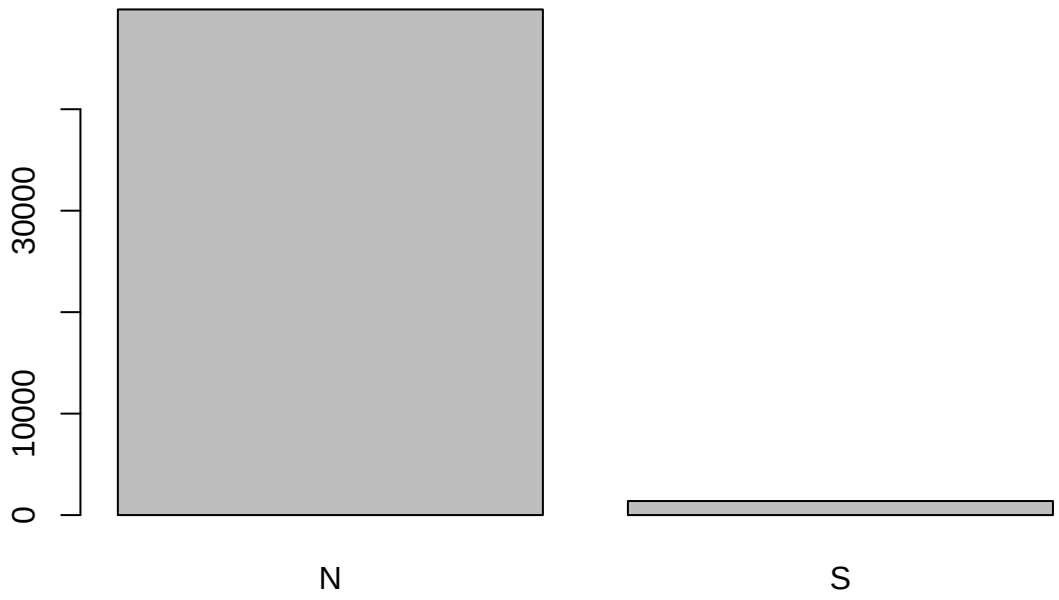
```
# Conversão em dados categóricos
BD_MODELO$CARATER <- as.factor(BD_MODELO$CARATER)
BD_MODELO$Classe1 <- as.factor(BD_MODELO$Classe1)
BD_MODELO$GENERO <- as.factor(BD_MODELO$GENERO)
BD_MODELO$Alta_Complexidade <- as.factor(BD_MODELO$Alta_Complexidade)

# Remoção do valores NAs
BD_MODELO2 <- na.omit(BD_MODELO)

#Confirmação da conversão dos dados
str(BD_MODELO2)
```

```
## tibble [51,218 x 5] (S3: tbl_df/tbl/data.frame)
##  $ CARATER          : Factor w/ 2 levels "Elective","Urgency": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Classe1          : Factor w/ 3 levels "Infeccioso","Oncológico",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ Alta_Complexidade: Factor w/ 2 levels "N","S": 1 1 1 1 1 1 1 1 1 1 ...
##  $ IDADE            : num [1:51218] 46 29 29 41 45 30 25 47 47 39 ...
##  $ GENERO           : Factor w/ 2 levels "Feminino","Masculino": 1 1 1 2 1 1 1 1 1 2 ...
##  - attr(*, "na.action")= 'omit' Named int [1:1549] 19198 24379 24385 24393 24410 24420 24421 24422 24
##   ..- attr(*, "names")= chr [1:1549] "19198" "24379" "24385" "24393" ...
```

```
#Verificação da classes para o modelo
round(prop.table(table(BD_MODELO2$Alta_Complexidade)),2)
```

```
##
##    N    S
## 0.97 0.03
```

```
plot(BD_MODELO2$Alta_Complexidade)
```



Versão 1 do Modelo

A versão 1 do modelo será feita sem o balaceamento de classes. Os dados foram divididos em treino e teste de maneira aleatória para garantir a generalização do modelo. Em regressão logística, o retorno das previsões em uma probabilidade de 0 a 1, nesse caso consideramos valores acima de 0,5 como uma classe.

O Modelo 1 apresenta uma acurácia acima de 90%, contudo ele errou muito os registros classificados como Alta Complexidade.

```
# Amostra de dados aleatória para divisão em treino e teste
amostra_dados <- sample(x = nrow(BD_MODELO2),
                        size = 0.8 * nrow(BD_MODELO2),
                        replace = FALSE)

# Dados de treino e teste
dados_treino <- BD_MODELO2[amostra_dados,]
dados_teste <- BD_MODELO2[-amostra_dados,]
size = 0.8 * nrow(BD_MODELO2)

# Versão 1 do modelo
modelo_1 <- glm(Alta_Complexidade ~ ., data = dados_treino, family = 'binomial')

# Dados de treino sem a variável target
dados_modelo_teste <- dados_teste[,-3]

previsoes <- predict(modelo_1, newdata = dados_modelo_teste, type = 'response')
```

```r
# Conversão do valores resultantes do modelo
resultado_modelo <- ifelse(previsoes>0.5, "S", "N")
real <- dados_teste$Alta_Complexidade

# Confusion Matrix e sumário do Modelo 1
summary(modelo_1)
```

```
##
## Call:
## glm(formula = Alta_Complexidade ~ ., family = "binomial", data = dados_treino)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7986  -0.1387  -0.0806  -0.0628   3.9123
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -5.904535   0.240067 -24.595   <2e-16 ***
## CARATERUrgency     3.448293   0.105845  32.579   <2e-16 ***
## Classe1Oncológico  0.035837   0.438482   0.082    0.935
## Classe1Outros     -1.917207   0.190524 -10.063   <2e-16 ***
## IDADE              0.041876   0.001986  21.089   <2e-16 ***
## GENEROMasculino    0.001606   0.068151   0.024    0.981
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 10200.7  on 40973  degrees of freedom
## Residual deviance:  7407.5  on 40968  degrees of freedom
## AIC: 7419.5
##
## Number of Fisher Scoring iterations: 8
```

```r
confusionMatrix(table(data=resultado_modelo, reference = real))
```

```
## Confusion Matrix and Statistics
##
##     reference
## data    N    S
##    N 9964  266
##    S    6    8
##
##                Accuracy : 0.9734
##                  95% CI : (0.9701, 0.9765)
##     No Information Rate : 0.9733
##     P-Value [Acc > NIR] : 0.4672
##
##                   Kappa : 0.0531
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9994
##             Specificity : 0.0292
```

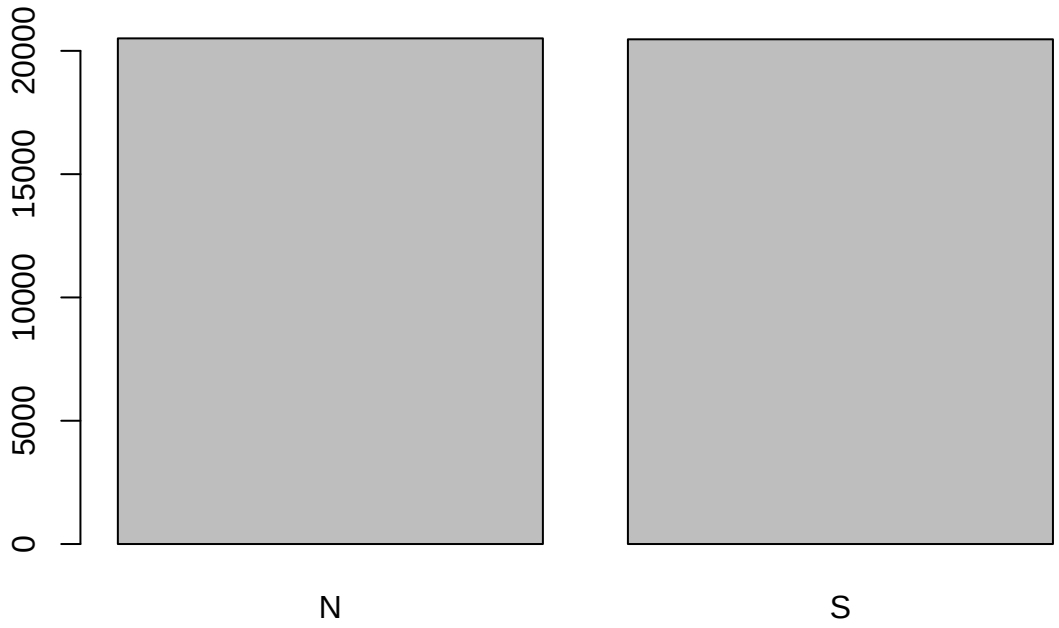
```
##          Pos Pred Value : 0.9740
##          Neg Pred Value : 0.5714
##              Prevalence : 0.9733
##          Detection Rate : 0.9727
##    Detection Prevalence : 0.9986
##       Balanced Accuracy : 0.5143
##
##        'Positive' Class : N
##
```

Versão 2 do Modelo

A versão 2 do modelo será feita com o balaceamento de classes, conforme gráfico abaixo. Os dados foram divididos em treino e teste de maneira aleatória para garantir a generalização do modelo. Em regressão logística, o retorno das previsões em uma probabilidade de 0 a 1, nesse caso consideramos valores acima de 0,5 como uma classe.

O Modelo 2 apresenta uma acurácia acima de 70%, contudo ele acertou mais registros classificados como Alta Complexidade em comparação com o Modelo 1

```
# Balanceamento das classes
treino2 <- ovun.sample(Alta_Complexidade~.,
                       data = dados_treino,
                       method = 'both',
                       N = size)

plot(treino2$data$Alta_Complexidade)
```



```r
# Versão 2 do modelo
modelo_2 <- glm(Alta_Complexidade~., data=treino2$data, family = 'binomial')
previsoes_2 <- predict(modelo_2, newdata = dados_modelo_teste, type = 'response')
resultado_modelo2 <- ifelse(previsoes_2>0.5, "S", "N")

# Confusion Matrix e sumário do Modelo 2
summary(modelo_2)
```

```
##
## Call:
## glm(formula = Alta_Complexidade ~ ., family = "binomial", data = treino2$data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9394  -0.4768  -0.2628   0.6599   2.6339
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -1.8069439  0.1505860 -11.999   <2e-16 ***
## CARATERUrgency     3.4124336  0.0303157 112.563   <2e-16 ***
## Classe1Oncológico  0.0375862  0.2348179   0.160   0.8728
## Classe1Outros     -1.8180092  0.1432386 -12.692   <2e-16 ***
## IDADE              0.0295649  0.0008357  35.379   <2e-16 ***
## GENEROMasculino    0.0697425  0.0302560   2.305   0.0212 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 56802  on 40973  degrees of freedom
## Residual deviance: 33751  on 40968  degrees of freedom
## AIC: 33763
##
## Number of Fisher Scoring iterations: 5
```

```r
confusionMatrix(table(data=resultado_modelo2, reference = real))
```

```
## Confusion Matrix and Statistics
##
##     reference
## data    N    S
##    N 7587   19
##    S 2383  255
##
##                Accuracy : 0.7655
##                  95% CI : (0.7572, 0.7737)
##     No Information Rate : 0.9733
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1331
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.76098
```

```
##             Specificity : 0.93066
##          Pos Pred Value : 0.99750
##          Neg Pred Value : 0.09666
##              Prevalence : 0.97325
##          Detection Rate : 0.74063
##    Detection Prevalence : 0.74248
##       Balanced Accuracy : 0.84582
##
##        'Positive' Class : N
##
```

Apresentação do Resultado

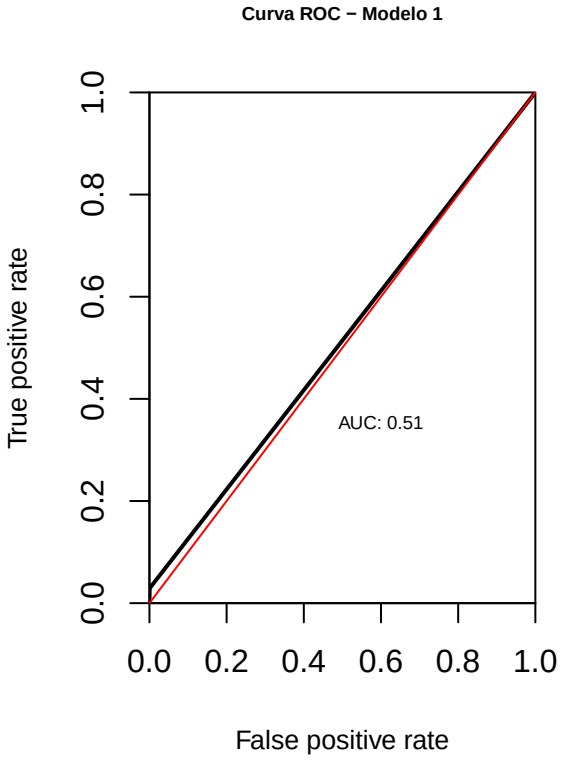
A curva ROC (Receiver Operating Characteristics) e AUC (Area Under the Curve) são apresentadas a seguir para análise da eficácia do modelo.

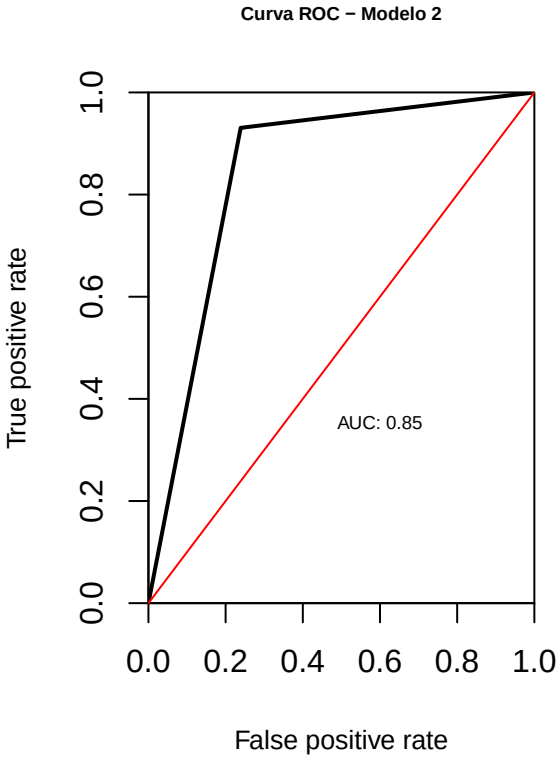
A linha na cor vermelha da imagem abaixo temos a curva AUC, que por sua vez representa 50% de precisão do modelo. Isso significa que seria a nossa linha de corte por assim dizer.

A linha preta representaa a curva ROC, que são as previsões feitas pelos modelos de Machine Learning. Quanto mais essas linhas estiverem próximas de 1 no eixo Y, melhor. Indica que o modelo tem uma alta acurácia. Linhas próximas ou abaixo do curva AUC indica que o modelo precisa de ajustes nos parâmetros. Ao analisar o gráfico abaixo, vemos que o modelo 2 teve uma performance muito superior ao 1, visto que este não teve a técnica de balanceamento de classes e por isso ele acabou não apredendendo de forma generalista as duas categorias apresentadas.

Sendo assim, o modelo 2 de regresssão logística apresentou o melhor resultado.

```
# Função para Plot ROC
plot.roc.curve <- function(predictions, title.text){
  perf <- performance(predictions, "tpr", "fpr")
  plot(perf,col = "black",lty = 1, lwd = 2,
       main = title.text, cex.main = 0.6, 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 = 0.6, bty = "n", box.col = "white")

}

# Plot Curva ROC
Teste_ROC <- dados_teste
Teste_ROC$Alta_Complexidade <- ifelse(Teste_ROC$Alta_Complexidade == "S", 1, 0)

df_modelo <- as.data.frame(previsoes_2)
df_modelo <- ifelse(previsoes_2>0.5, 1, 0)

df_modelo1 <- as.data.frame(previsoes)
df_modelo1 <- ifelse(previsoes>0.5, 1, 0)

predictions_ <- prediction(df_modelo, Teste_ROC$Alta_Complexidade)
predictions_1 <- prediction(df_modelo1, Teste_ROC$Alta_Complexidade)
par(mfrow = c(1, 2))
plot.roc.curve(predictions_1, title.text = "Curva ROC - Modelo 1")
plot.roc.curve(predictions_, title.text = "Curva ROC - Modelo 2")
```


Curva ROC − Modelo 1
True positive rate
False positive rate
AUC: 0.51
0.0
0.2
0.4
0.6
0.8
1.0


Curva ROC − Modelo 2
True positive rate
False positive rate
AUC: 0.85
0.0
0.2
0.4
0.6
0.8
1.0