

Decision Tree - Titanic DataBase

Vinicius Capozzi

Nesse arquivo, veremos um exemplo de como uma decision tree(árvore de decisão) consegue identificar padrões com base nas observações da base de treino e assim classificar quais pessoas sobreviveram ou não.

Vamos trabalhar com a base titanic, que contém as seguintes variáveis e características:

```
load("C:/Users/Vinicius/Desktop/MBA/Outros Modelos de Machine Learning/.RData")
titanic %>% head
```

```
##   Survived Pclass    Sex      Age SibSp Parch   Fare Embarked
## 1         N      3  male 22.00000     1     0  7.2500         S
## 2         Y      1 female 38.00000     1     0 71.2833         C
## 3         Y      3 female 26.00000     0     0  7.9250         S
## 4         Y      1 female 35.00000     1     0 53.1000         S
## 5         N      3  male 35.00000     0     0  8.0500         S
## 6         N      3  male 29.69912     0     0  8.4583         Q
```

```
titanic %>% str
```

```
## 'data.frame':   891 obs. of  8 variables:
## $ Survived: Factor w/ 2 levels "N","Y": 1 2 2 2 1 1 1 1 2 2 ...
## $ Pclass  : int   3 1 3 1 3 3 1 3 3 2 ...
## $ Sex     : Factor w/ 2 levels "female","male": 2 1 1 1 2 2 2 1 1 ...
## $ Age     : num   22 38 26 35 35 ...
## $ SibSp   : int   1 1 0 1 0 0 0 3 0 1 ...
## $ Parch   : int   0 0 0 0 0 0 0 1 2 0 ...
## $ Fare    : num   7.25 71.28 7.92 53.1 8.05 ...
## $ Embarked: Factor w/ 3 levels "C","Q","S": 3 1 3 3 3 2 3 3 3 1 ...
```

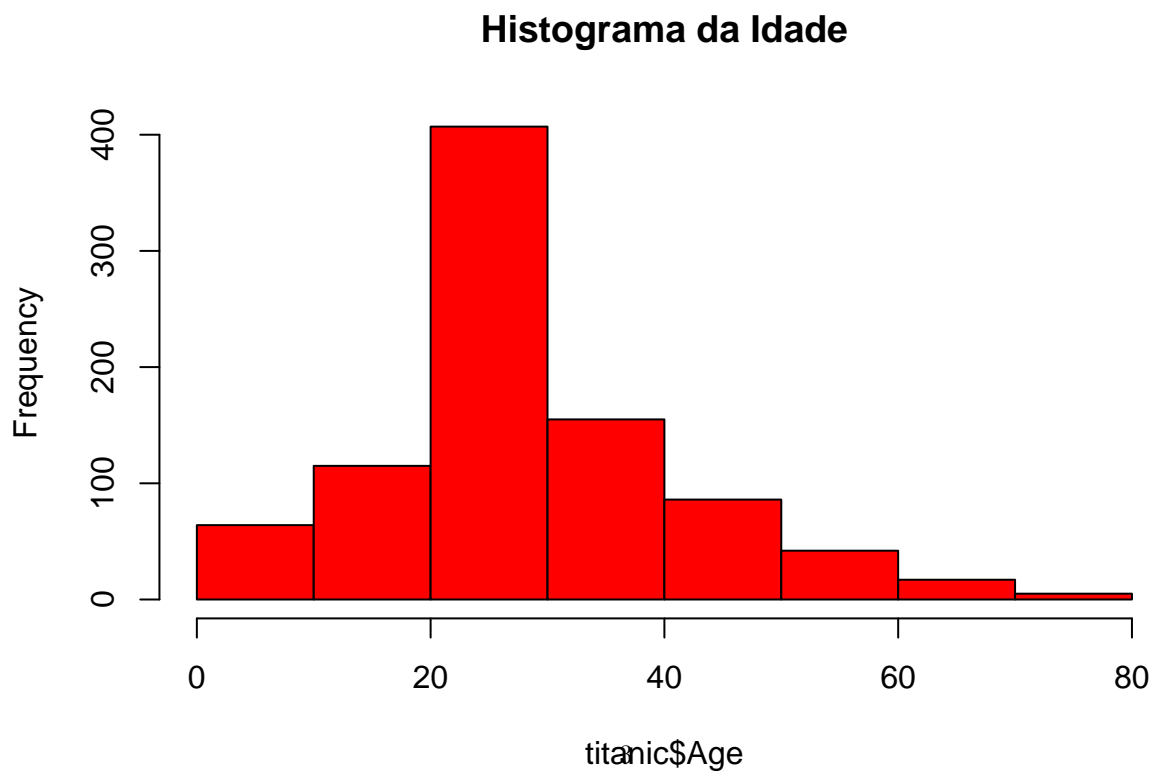
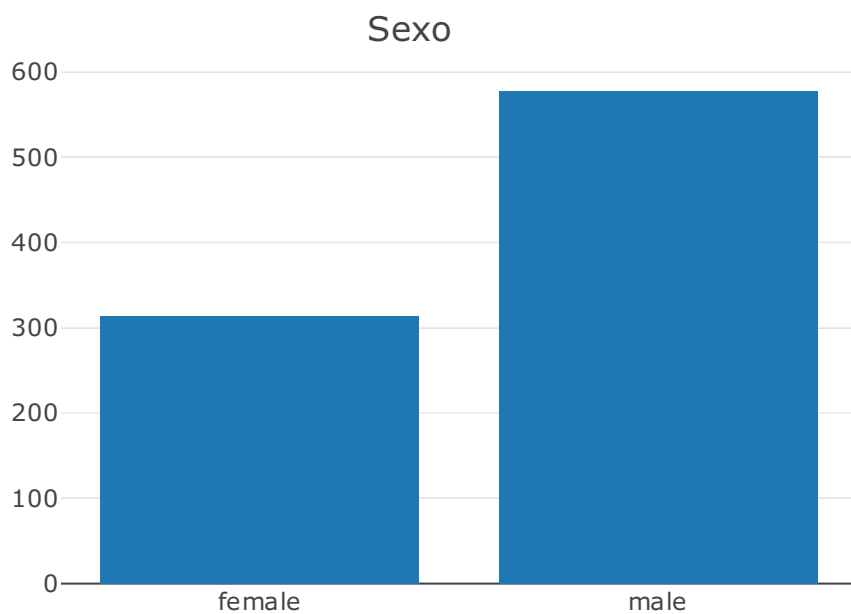
Com a função summary, vemos algumas estatísticas descritivas da nossa base de dados que podem nos ajudar a entender melhor como as observações estão distribuídas.

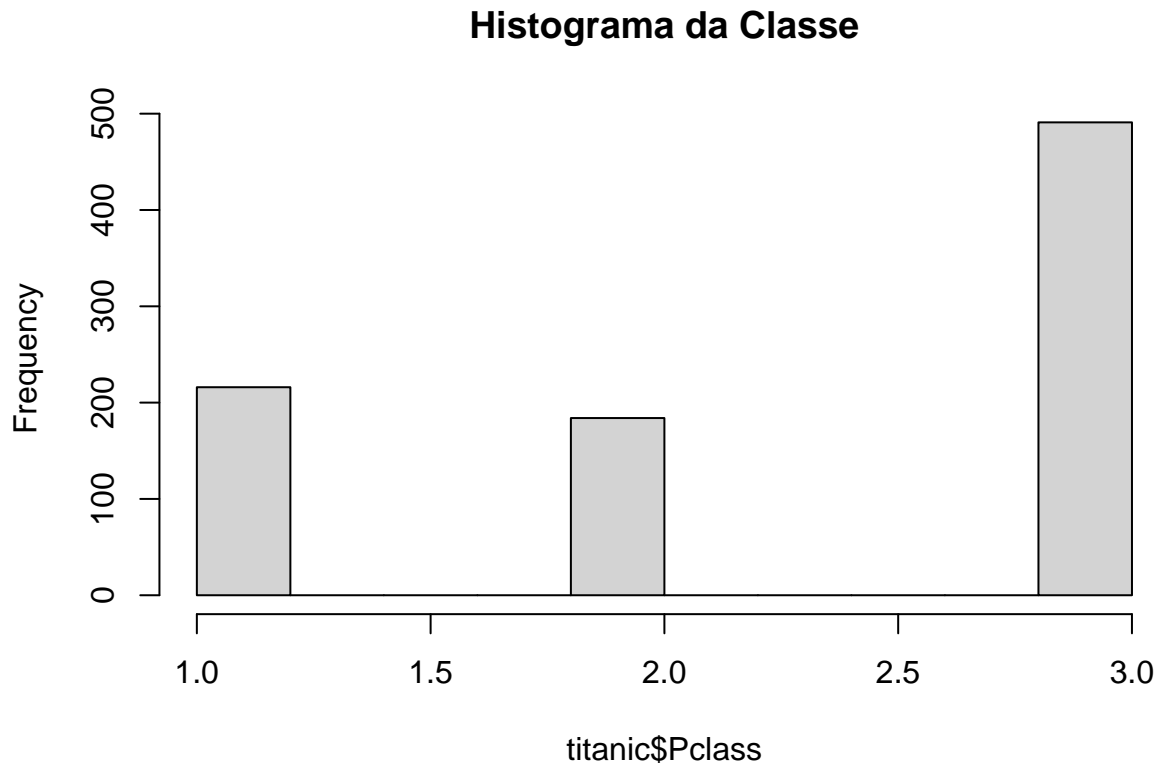
```
summary(titanic)
```

```
##   Survived      Pclass      Sex      Age      SibSp
## N:549   Min.    :1.000   female:314   Min.    : 0.42   Min.    :0.000
## Y:342   1st Qu.:2.000   male  :577   1st Qu.:22.00   1st Qu.:0.000
##         Median :3.000                     Median :29.70   Median :0.000
##         Mean   :2.309                     Mean   :29.70   Mean   :0.523
```

##	3rd Qu.:	3.000	3rd Qu.:	35.00	3rd Qu.:	1.000
##	Max.	:3.000	Max.	:80.00	Max.	:8.000
##	Parch	Fare	Embarked			
##	Min.	:0.0000	Min.	: 0.00	C:	168
##	1st Qu.:	0.0000	1st Qu.:	7.91	Q:	77
##	Median	:0.0000	Median	: 14.45	S:	646
##	Mean	:0.3816	Mean	: 32.20		
##	3rd Qu.:	0.0000	3rd Qu.:	31.00		
##	Max.	:6.0000	Max.	:512.33		

Em seguida, plotaremos alguns gráficos que irão nos dar uma visão melhor sobre as variáveis da nossa base de dados. As variáveis que achei mais interessante destacar foram Sexo, Idade e o número da Classe do passageiro.



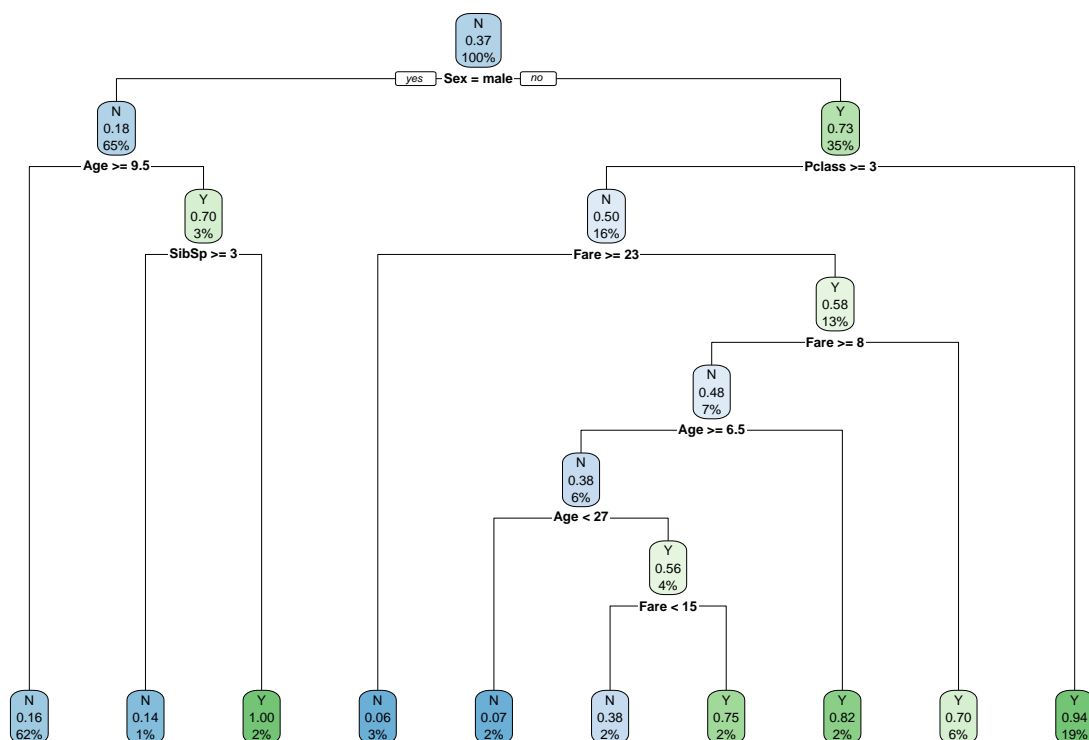


Com a análise exploratória realizada, podemos começar a trabalhar com a construção de algoritmo. Primeiramente, separamos a base de dados em teste e treino para que possamos ver se o nosso modelo está conseguindo classificar corretamente as observações. Depois geramos a variável “arvore” que irá conter a nossa árvore de decisão. Com base no primeiro argumento que passamos na função, no caso se a pessoa é uma sobrevivente ou não, o algoritmo irá testar qual das outras variáveis melhor se relaciona com o fato da pessoa ter sobrevivido, como o sexo, a idade, a classe.

Depois, plotamos a árvore par ver o resultado que ela apresentou.

```
set.seed(123)
bool_treino <- stats::runif(dim(titanic)[1])>.25

treino <- titanic[bool_treino,]
teste  <- titanic[!bool_treino,]
set.seed(123)
arvore <- rpart(Survived ~ Pclass + Sex + Age + SibSp + Parch + Fare + Embarked,
                data=treino,
                parms = list(split = 'gini'),
                method='class'
)
rpart.plot::rpart.plot(arvore
)
```



```

p_treino = stats::predict(arvore, treino)
c_treino = base::factor(ifelse(p_treino[,2]>.5, "Y", "N"))
p_teste = stats::predict(arvore, teste)
c_teste = base::factor(ifelse(p_teste[,2]>.5, "Y", "N"))

tab <- table(c_treino, treino$Survived)
acc <- (tab[1,1]+tab[2,2])/nrow(treino)
sprintf('Acurácia na base de treino: %s ', percent(acc))

```

```
## [1] "Acurácia na base de treino: 85% "
```

```

tab <- table(c_teste, teste$Survived)
acc <- (tab[1,1]+tab[2,2])/nrow(teste)
sprintf('Acurácia na base de teste: %s ', percent(acc))

```

```
## [1] "Acurácia na base de teste: 83% "
```

Com o modelo criado, podemos verificar a acurácia (número de acertos / número de tentativas) que o modelo possui de diversas maneiras. Iremos assumir que se a observação foi classificada com uma chance maior de 50% de sobreviver a pessoa será um sobrevivente, e caso seja menor de 50% a pessoa não será um sobrevivente.

```

prob = predict(arvore, titanic)
class = prob[,2]>.5
tab <- table(class, titanic$Survived)
tab

```

```

##
## class      N   Y
##  FALSE 519 107
##   TRUE   30 235

```

```

df <- as.data.frame(prob)
survived <- filter(df, Y>0.5)
not_survived <- filter(df, Y<0.5)
acc <- (tab[1,1] + tab[2,2]) / sum(tab)
print(paste0('O modelo apresentou ', nrow(survived), ' sobreviventes'))

```

```

## [1] "O modelo apresentou 265 sobreviventes"

```

```

print(paste0('O modelo apresentou ', nrow(not_survived), ' não sobreviventes'))

```

```

## [1] "O modelo apresentou 626 não sobreviventes"

```

```

print(paste0('A acurácia foi de:', acc))

```

```

## [1] "A acurácia foi de:0.846240179573513"

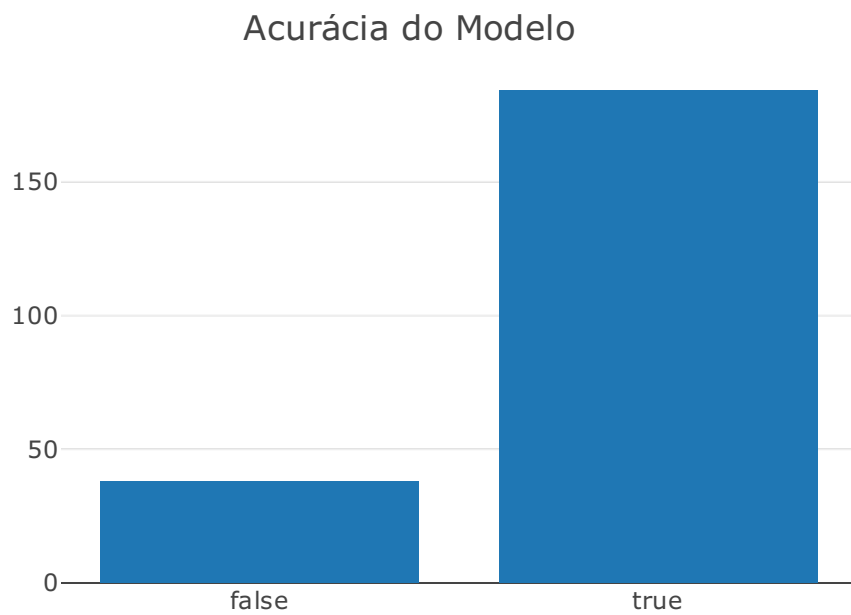
```

Também vemos no histograma abaixo a quantidade de acertos representados pela barra “true” e a quantidade de erros representada pela barra “false”

```

df5 <- as.data.frame(p_teste)
comp <- data.frame(df5)
comp['verif'] <- teste$Survived
comp$Y[comp$Y >= 0.5] <- 'Y'
comp$Y[comp$Y <= 0.5] <- 'N'
comp['acertos'] <- comp$Y == comp$verif
graf6 <- plot_ly(x=comp$acertos, type = 'histogram') %>% layout(title = 'Acurácia do Modelo')
graf6

```



Desse modo, concluímos a construção do algoritmo que é capaz de classificar futuras observações, nós dando a probabilidade de sobrevivência de determinado indivíduo com base nas suas características.