Analise Base de Dados de CrÃ©dito

Luciano Teixeira

05 de novembro de 2018

# Processamento Base de Dados de Crédito

O propósio deste relatorio é analisar ma base de dados de crédito com 2000 registros, identificando qual dos clientes, ao solicitar um impréstimo, possui a maior chance de efetuar o pagamento ou não deste impréstimo, levando em consideração:

### 1 Idade

### 2 Renda

### 3 Historico Financeiro

Foi uilizado o seguinte algoritmos de Machine Learning para avaliar a mesma base:

### Árvore de Decisão

Para este algorítmo, foi adoto alguns procedimentos de ajustes, “ETL”, para eventuais correções, alterações, seja por conta de categorias de variaveis, seja por erro oriundos de intervenções manuais

### 1 Importação da base de dados, com o propósito de ler o arquivo csv para sua classificação;

### 2 Eliminação da coluna de clientid, pois não há propósito de categoria ou calssificação desta coluna;

### 3 Substituição de valores negativos pela média de idade positiva da base, a fim de minimizar a interferencia nos dados;

### 4 Substituição de vaores nulos “NA” pela média da idade possitiva, a fim de minimizar a interferencia nos dados;

### 5 Efetuado o nivelamento da escala, por exemplo, entre a idade e a renda, pois a valor da renda em escala comparado à idade, é muito maior, sendo assim, a aprendizagem não é eficiente;

### 6 O Encode da Classe ou transformação do atributos categoricos em discretos, é fundamental pois diversas bibliotecas não aceitam como entrada, atributos caegóricos.

### 7 Divisão da base em dados de treinamento e dados de teste.

### 8 DataSet: <https://www.kaggle.com/macchi57/dataset/downloads/dataset.zip/1>

# Árvore de Decisão

## Importando a Base de Dados

base = read.csv('credit\_data.csv')

## Eliminando coluna clientid

base$clientid = NULL

## Preencher os valores negativos com a média dos valores positivos da coluna Age

base$age = ifelse(base$age < 0, 40.92, base$age)

## Preencher os valores nulos

base$age = ifelse(is.na(base$age), mean(base$age, na.rm = TRUE), base$age)

## Executando o Escalonamento, transformando os atributos numéricos na mesma escala

base[, 1:3] = scale(base[, 1:3])

## Encode da classe

base$default = factor(base$default, levels = c(0,1))

## Executando a divisão da base de dados entre treinamento e teste

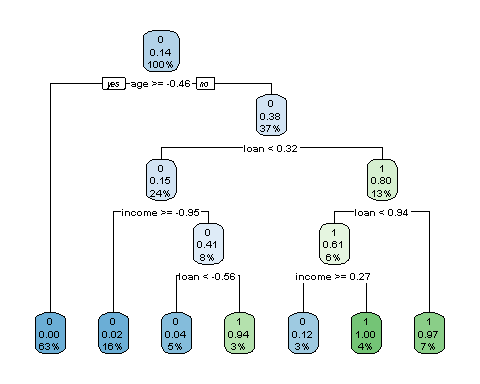
library(caTools)  
set.seed(1)  
divisao = sample.split(base$income, SplitRatio = 0.75)  
base\_treinamento = subset(base, divisao == TRUE)  
base\_teste = subset(base, divisao == FALSE)

## Executando a primeira avaliação atraves da criação do classificador

library(rpart)  
classificador = rpart(formula = default ~ ., data = base\_treinamento)  
print(classificador)

## n= 1500   
##   
## node), split, n, loss, yval, (yprob)  
## \* denotes terminal node  
##   
## 1) root 1500 210 0 (0.86000000 0.14000000)   
## 2) age>=-0.4569664 946 0 0 (1.00000000 0.00000000) \*  
## 3) age< -0.4569664 554 210 0 (0.62093863 0.37906137)   
## 6) loan< 0.3190762 356 52 0 (0.85393258 0.14606742)   
## 12) income>=-0.953323 240 5 0 (0.97916667 0.02083333) \*  
## 13) income< -0.953323 116 47 0 (0.59482759 0.40517241)   
## 26) loan< -0.5571763 69 3 0 (0.95652174 0.04347826) \*  
## 27) loan>=-0.5571763 47 3 1 (0.06382979 0.93617021) \*  
## 7) loan>=0.3190762 198 40 1 (0.20202020 0.79797980)   
## 14) loan< 0.9402764 95 37 1 (0.38947368 0.61052632)   
## 28) income>=0.2655176 42 5 0 (0.88095238 0.11904762) \*  
## 29) income< 0.2655176 53 0 1 (0.00000000 1.00000000) \*  
## 15) loan>=0.9402764 103 3 1 (0.02912621 0.97087379) \*

library(rpart.plot)  
rpart.plot(classificador)



## Executando a verificação do índice de acertos

previsoes = predict(classificador, newdata = base\_teste[-4], type = 'class')  
matriz\_confusao = table(base\_teste[, 4], previsoes)  
print(matriz\_confusao)

## previsoes  
## 0 1  
## 0 420 7  
## 1 11 62

library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

confusao\_ad = confusionMatrix(matriz\_confusao)  
confusionMatrix(matriz\_confusao)

## Confusion Matrix and Statistics  
##   
## previsoes  
## 0 1  
## 0 420 7  
## 1 11 62  
##   
## Accuracy : 0.964   
## 95% CI : (0.9437, 0.9785)  
## No Information Rate : 0.862   
## P-Value [Acc > NIR] : 1.534e-14   
##   
## Kappa : 0.8523   
## Mcnemar's Test P-Value : 0.4795   
##   
## Sensitivity : 0.9745   
## Specificity : 0.8986   
## Pos Pred Value : 0.9836   
## Neg Pred Value : 0.8493   
## Prevalence : 0.8620   
## Detection Rate : 0.8400   
## Detection Prevalence : 0.8540   
## Balanced Accuracy : 0.9365   
##   
## 'Positive' Class : 0   
##

Abaixo, seguem os dados de acuracidade do algorítimo de árvore de decisão de acordo como a conerencia da matriz de confusão.

# Árvore de Decisão

print(confusao\_ad)

## Confusion Matrix and Statistics  
##   
## previsoes  
## 0 1  
## 0 420 7  
## 1 11 62  
##   
## Accuracy : 0.964   
## 95% CI : (0.9437, 0.9785)  
## No Information Rate : 0.862   
## P-Value [Acc > NIR] : 1.534e-14   
##   
## Kappa : 0.8523   
## Mcnemar's Test P-Value : 0.4795   
##   
## Sensitivity : 0.9745   
## Specificity : 0.8986   
## Pos Pred Value : 0.9836   
## Neg Pred Value : 0.8493   
## Prevalence : 0.8620   
## Detection Rate : 0.8400   
## Detection Prevalence : 0.8540   
## Balanced Accuracy : 0.9365   
##   
## 'Positive' Class : 0   
##