**AnaliseRiscoCredito.R**

Rafael

2019-04-15

*# Mini Projeto 2 - Analise de Risco de Crédito*  
  
*# Objetivo: Avaliar o risco de concessão de crédito a clientes de instituições financeiras.*

*# Neste mini-projeto, seu trabalho é avaliar o risco de concessão de crédito a clientes de instituições financeiras, tarefa cada vez mais comum atualmente!*

*# O dataset fornecido terá diversas variáveis e nem todas serão relevantes para construção do modelo, requer uma análise no dataset, conversão e normalização das variáveis e então construção do modelo.*   
  
*# Será necessário identificar as variáveis mais relevantes para a construção do modelo, bem como analisar o dataset realizar conversões e normalizações das variáveis e por fim, construir o melhor modelo preditivo possível.*  
  
*# Carregando o datset*  
dataset\_credit <- **read.csv**("credit\_dataset.csv", header = TRUE, sep = ",")  
  
*# Coluna credit.rating é nossa coluna target*  
**head**(dataset\_credit)

## credit.rating account.balance credit.duration.months  
## 1 1 1 18  
## 2 1 1 9  
## 3 1 2 12  
## 4 1 1 12  
## 5 1 1 12  
## 6 1 1 10  
## previous.credit.payment.status credit.purpose credit.amount savings  
## 1 3 2 1049 1  
## 2 3 4 2799 1  
## 3 2 4 841 2  
## 4 3 4 2122 1  
## 5 3 4 2171 1  
## 6 3 4 2241 1  
## employment.duration installment.rate marital.status guarantor  
## 1 1 4 1 1  
## 2 2 2 3 1  
## 3 3 2 1 1  
## 4 2 3 3 1  
## 5 2 4 3 1  
## 6 1 1 3 1  
## residence.duration current.assets age other.credits apartment.type  
## 1 4 2 21 2 1  
## 2 2 1 36 2 1  
## 3 4 1 23 2 1  
## 4 2 1 39 2 1  
## 5 4 2 38 1 2  
## 6 3 1 48 2 1  
## bank.credits occupation dependents telephone foreign.worker  
## 1 1 3 1 1 1  
## 2 2 3 2 1 1  
## 3 1 2 1 1 1  
## 4 2 2 2 1 2  
## 5 2 2 1 1 2  
## 6 2 2 2 1 2

**str**(dataset\_credit)

## 'data.frame': 1000 obs. of 21 variables:  
## $ credit.rating : int 1 1 1 1 1 1 1 1 1 1 ...  
## $ account.balance : int 1 1 2 1 1 1 1 1 3 2 ...  
## $ credit.duration.months : int 18 9 12 12 12 10 8 6 18 24 ...  
## $ previous.credit.payment.status: int 3 3 2 3 3 3 3 3 3 2 ...  
## $ credit.purpose : int 2 4 4 4 4 4 4 4 3 3 ...  
## $ credit.amount : int 1049 2799 841 2122 2171 2241 3398 1361 1098 3758 ...  
## $ savings : int 1 1 2 1 1 1 1 1 1 3 ...  
## $ employment.duration : int 1 2 3 2 2 1 3 1 1 1 ...  
## $ installment.rate : int 4 2 2 3 4 1 1 2 4 1 ...  
## $ marital.status : int 1 3 1 3 3 3 3 3 1 1 ...  
## $ guarantor : int 1 1 1 1 1 1 1 1 1 1 ...  
## $ residence.duration : int 4 2 4 2 4 3 4 4 4 4 ...  
## $ current.assets : int 2 1 1 1 2 1 1 1 3 4 ...  
## $ age : int 21 36 23 39 38 48 39 40 65 23 ...  
## $ other.credits : int 2 2 2 2 1 2 2 2 2 2 ...  
## $ apartment.type : int 1 1 1 1 2 1 2 2 2 1 ...  
## $ bank.credits : int 1 2 1 2 2 2 2 1 2 1 ...  
## $ occupation : int 3 3 2 2 2 2 2 2 1 1 ...  
## $ dependents : int 1 2 1 2 1 2 1 2 1 1 ...  
## $ telephone : int 1 1 1 1 1 1 1 1 1 1 ...  
## $ foreign.worker : int 1 1 1 2 2 2 2 2 1 1 ...

*# Verificar se existe dados missing no dataset.*  
**any**(**is.na**(dataset\_credit))

## [1] FALSE

################################################################################################################################################  
 ################## ETAPA 1: ANALISAR O DATASET PARA COMPRENDER AS VARIÁVEIS ##################  
################################################################################################################################################  
  
*# Identificando variáveis categóricas que não estão como factor e devemos realizar a transformação.*  
Variaveis\_Categoricas <- **c**('credit.rating', 'account.balance', 'previous.credit.payment.status',  
 'credit.purpose', 'savings', 'employment.duration', 'installment.rate',  
 'marital.status', 'guarantor', 'residence.duration', 'current.assets',  
 'other.credits', 'apartment.type', 'bank.credits', 'occupation',   
 'dependents', 'telephone', 'foreign.worker')  
  
*# As variáveis que não forem Categóricas, ou seja, variáveis numéricas que realmente representam valores e não categorias*  
*# deveremos realizar a normalização dos dados.*  
*# Os dados estão em escalas diferentes e para um melhor resultado do modelo, devemos normalizar todos na mesma escala.*  
Variaveis\_Numericas <- **c**("credit.duration.months", "age", "credit.amount")  
  
################################################################################################################################################  
 ################## ETAPA 2: CRIANDO FUNÇÕES PARA CONVERTER VARIÁVEIS CATEGÓRIAS E NORMALIZAÇÃO DE VARIÁVEIS NUMÉRICAS ##################   
################################################################################################################################################  
  
*# Função irá transformar cada uma das "colunas" informadas em factor e salvar novamente no dataset*  
func\_to.factors <- **function**(dataset, var\_categoricas){  
 **for** (n **in** var\_categoricas){  
 dataset[[n]] <- **as.factor**(dataset[[n]])  
 }  
 **return**(dataset)  
}  
  
*# Função irá realizar a normalização de cada uma das "colunas" informadas e salvar novamente no dataset*  
func\_scale.features <- **function**(dataset, vars\_to\_scale){  
 **for** (n **in** vars\_to\_scale){  
 dataset[[n]] <- **scale**(dataset[[n]], center=T, scale=T) *# Função scale está sendo utilizada para realizar a normalização dos dados*  
 }  
 **return**(dataset)  
}  
  
  
*# Realizando a transformação dos dados em variáveis categóricas*  
dataset\_credit\_transformed <- **func\_to.factors**(dataset = dataset\_credit, var\_categoricas = Variaveis\_Categoricas)  
  
*# Realizando a padronização/normalização dos dados numéricos*  
*# Obs. O dataset a ser passado é o "dataset\_credit\_transformed" que já sofreu a transformação das variáveis categóricas*  
dataset\_credit\_transformed <- **func\_scale.features**(dataset = dataset\_credit\_transformed, vars\_to\_scale = Variaveis\_Numericas)  
  
**str**(dataset\_credit\_transformed)

## 'data.frame': 1000 obs. of 21 variables:  
## $ credit.rating : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...  
## $ account.balance : Factor w/ 3 levels "1","2","3": 1 1 2 1 1 1 1 1 3 2 ...  
## $ credit.duration.months : num [1:1000, 1] -0.241 -0.987 -0.738 -0.738 -0.738 ...  
## ..- attr(\*, "scaled:center")= num 20.9  
## ..- attr(\*, "scaled:scale")= num 12.1  
## $ previous.credit.payment.status: Factor w/ 3 levels "1","2","3": 3 3 2 3 3 3 3 3 3 2 ...  
## $ credit.purpose : Factor w/ 4 levels "1","2","3","4": 2 4 4 4 4 4 4 4 3 3 ...  
## $ credit.amount : num [1:1000, 1] -0.787 -0.167 -0.861 -0.407 -0.39 ...  
## ..- attr(\*, "scaled:center")= num 3271  
## ..- attr(\*, "scaled:scale")= num 2823  
## $ savings : Factor w/ 4 levels "1","2","3","4": 1 1 2 1 1 1 1 1 1 3 ...  
## $ employment.duration : Factor w/ 4 levels "1","2","3","4": 1 2 3 2 2 1 3 1 1 1 ...  
## $ installment.rate : Factor w/ 4 levels "1","2","3","4": 4 2 2 3 4 1 1 2 4 1 ...  
## $ marital.status : Factor w/ 3 levels "1","3","4": 1 2 1 2 2 2 2 2 1 1 ...  
## $ guarantor : Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 1 1 ...  
## $ residence.duration : Factor w/ 4 levels "1","2","3","4": 4 2 4 2 4 3 4 4 4 4 ...  
## $ current.assets : Factor w/ 4 levels "1","2","3","4": 2 1 1 1 2 1 1 1 3 4 ...  
## $ age : num [1:1000, 1] -1.2809 0.0403 -1.1048 0.3046 0.2165 ...  
## ..- attr(\*, "scaled:center")= num 35.5  
## ..- attr(\*, "scaled:scale")= num 11.4  
## $ other.credits : Factor w/ 2 levels "1","2": 2 2 2 2 1 2 2 2 2 2 ...  
## $ apartment.type : Factor w/ 3 levels "1","2","3": 1 1 1 1 2 1 2 2 2 1 ...  
## $ bank.credits : Factor w/ 2 levels "1","2": 1 2 1 2 2 2 2 1 2 1 ...  
## $ occupation : Factor w/ 4 levels "1","2","3","4": 3 3 2 2 2 2 2 2 1 1 ...  
## $ dependents : Factor w/ 2 levels "1","2": 1 2 1 2 1 2 1 2 1 1 ...  
## $ telephone : Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 1 1 ...  
## $ foreign.worker : Factor w/ 2 levels "1","2": 1 1 1 2 2 2 2 2 1 1 ...

*# Verificar se o número de colunas do novo dataset é igual ao anterior, garantindo assim que todas variáveis foram selecionadas.*  
**if** ( **ncol**(dataset\_credit\_transformed) **!=** **ncol**(dataset\_credit) ){  
 "VERIFICAR A ETAPA DA SELEÇÃO DAS VARIÁVEIS PARA TRANSFORMAÇÃO E NORMALIZAÇÃO.  
 EXISTEM VARIÁVEIS QUE FICARAM DE FORA."  
}  
  
################################################################################################################################################  
 ################## ETAPA 3: DIVIDINDO OS DADOS EM TREINO E TESTE ##################   
################################################################################################################################################  
  
*# Existem várias maneiras de realizar o split dos dados em treino e teste. Abaixo será utilizando o pacote caTools para o split.*  
  
*# Dividindo os dados utilizando o pacote CaTools.*  
*# install.packages("caTools")*  
**library**("caTools")  
  
*# 70% dos dados serão classificados como TRUE e 30% como FALSE na variavel amostra\_Dados,*  
*# Representando assim 70% -> Treino e 30% Teste.*  
amostra\_Dados <- **sample.split**(dataset\_credit\_transformed**$**credit.rating, SplitRatio = 0.70)  
dados\_treino <- **subset**(dataset\_credit\_transformed, amostra\_Dados **==** TRUE)  
dados\_teste <- **subset**(dataset\_credit\_transformed, amostra\_Dados **==** FALSE)  
  
################################################################################################################################################  
 ################## ETAPA 4: IDENTIFICANDO VARIÁVEIS MAIS RELEVANTES DO DATASET ##################  
################################################################################################################################################  
  
*# Importando o pacote caret para ser possível utilizar a função: varImp*  
**library**("caret")

## Loading required package: lattice

## Loading required package: ggplot2

*# Existem várias formas de realizar o levantamento das variáveis mais relevantes do dataset.*  
*# Abaixo irei utilizar o algoritmo RandomForest para obter essa informação.*  
*# Obs. Importante: Utilizar os dados de Treino*  
**library**(randomForest)

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##   
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':  
##   
## margin

variaveis\_relevantes\_rf <- **randomForest**(credit.rating **~** . ,  
 data = dados\_treino,  
 ntree = 100, nodesize = 10, importance = T)  
  
  
*# Visualizando o resultado das variáveis mais relevantes do dataset*  
variaveis\_relevantes\_rf

##   
## Call:  
## randomForest(formula = credit.rating ~ ., data = dados\_treino, ntree = 100, nodesize = 10, importance = T)   
## Type of random forest: classification  
## Number of trees: 100  
## No. of variables tried at each split: 4  
##   
## OOB estimate of error rate: 24.14%  
## Confusion matrix:  
## 0 1 class.error  
## 0 89 121 0.57619048  
## 1 48 442 0.09795918

**varImp**(variaveis\_relevantes\_rf)

## 0 1  
## account.balance 11.69154497 11.69154497  
## credit.duration.months 5.34454521 5.34454521  
## previous.credit.payment.status 4.45631693 4.45631693  
## credit.purpose -0.03114303 -0.03114303  
## credit.amount 0.77976956 0.77976956  
## savings 4.10942566 4.10942566  
## employment.duration 0.71465172 0.71465172  
## installment.rate 0.21776109 0.21776109  
## marital.status 0.65158589 0.65158589  
## guarantor 1.38076919 1.38076919  
## residence.duration -0.10501546 -0.10501546  
## current.assets 0.35850745 0.35850745  
## age 1.46245063 1.46245063  
## other.credits 0.70152604 0.70152604  
## apartment.type 0.90018017 0.90018017  
## bank.credits 0.31885325 0.31885325  
## occupation 0.59903145 0.59903145  
## dependents 0.99155253 0.99155253  
## telephone 0.32255518 0.32255518  
## foreign.worker 1.03771007 1.03771007

**varImpPlot**(variaveis\_relevantes\_rf)



*# Realizando O mesmo levantamento de variáveis mais relevantes do dataset, porém utilizando funções fornecidas pelo pacote Caret.*  
*# Obs: Esse algoritmo requer bastante processamento de CPU e pode levar + tempo para o resultado.*  
*# library("caret")*  
  
*# Criando uma função que será responsável por fazer a análise de variávéis relevantes.*  
  
*# Metodo de Cross-Validation divide o conjunto de dados em subconjuntos e cada subconjunto é mantido até*   
*# o fim do treinamento dos outros subconjuntos. Esse processo ocorre até que cada subconjunto tenha um resultado*  
*# e uma estimativa geral de precisão seja fornecida no final.*  
func\_feature\_selection <- **function** (n\_interacoes, var\_preditoras, var\_target){  
 ctrl\_rfe <- **rfeControl**(functions = rfFuncs, method = "cv", *# Cross-Validation Method*  
 verbose = FALSE, returnResamp = "all",  
 number = n\_interacoes) *# Realiza a divisão em 10-fold-cross validation*  
   
 result\_rfe <- **rfe**(x = var\_preditoras, y = var\_target,  
 sizes = 1**:**10,  
 rfeControl = ctrl\_rfe)  
}  
  
variaveis\_relevantes\_caret <- **func\_feature\_selection**(n\_interacoes = 20,   
 var\_preditoras = dados\_treino[,**-**1], *#Ignorando a 1º Coluna*  
 var\_target = dados\_treino[,1] *#Somente a 1º Coluna*  
 )  
  
  
*# Visualizando o resultado das variváveis mais relevantes do dataset utilizando pacote caret*  
variaveis\_relevantes\_caret

##   
## Recursive feature selection  
##   
## Outer resampling method: Cross-Validated (20 fold)   
##   
## Resampling performance over subset size:  
##   
## Variables Accuracy Kappa AccuracySD KappaSD Selected  
## 1 0.7001 0.0000 0.01116 0.0000   
## 2 0.7259 0.3030 0.07736 0.1982   
## 3 0.7502 0.3325 0.06772 0.2017   
## 4 0.7572 0.3939 0.06590 0.1808   
## 5 0.7657 0.4117 0.06715 0.1678   
## 6 0.7540 0.3729 0.06930 0.1895   
## 7 0.7530 0.3567 0.07438 0.2102   
## 8 0.7517 0.3414 0.08094 0.2377   
## 9 0.7655 0.3933 0.07796 0.2201   
## 10 0.7614 0.3757 0.07071 0.2156   
## 20 0.7701 0.3889 0.05375 0.1577 \*  
##   
## The top 5 variables (out of 20):  
## account.balance, credit.duration.months, savings, previous.credit.payment.status, credit.amount

**varImp**(variaveis\_relevantes\_caret)

## Overall  
## account.balance 24.2519499  
## credit.duration.months 10.9007776  
## savings 8.8907375  
## previous.credit.payment.status 7.7475251  
## credit.amount 4.2683456  
## guarantor 2.5869954  
## credit.purpose 2.2976688  
## current.assets 2.2545275  
## employment.duration 2.2404305  
## age 2.2250283  
## foreign.worker 2.1099268  
## telephone 2.0338105  
## apartment.type 1.3762314  
## other.credits 1.1565473  
## installment.rate 1.0435337  
## bank.credits 0.8045133  
## occupation 0.6908812  
## marital.status -0.3584222  
## dependents -0.3691804  
## residence.duration -0.8722916

*# Realizando a análise de variáveis mais importantes com uma segunda opção de algoritmo utilizando o pacote caret:*  
  
*# method = repeatedcv : Realiza a cross validation N vezes e o resultado final é a média do número de repetições.*  
*# number = Número de divisões a serem realizadas*  
*# repeats = Número de vezes que será executado*  
  
*# ?trainControl*  
*# ?train*  
  
fun\_train\_control <- **function**(metodo\_trainControl, number\_kfold, n\_repeats, formula, dataset, method\_train){  
 train\_control <- **trainControl**(method = metodo\_trainControl,   
 number = number\_kfold,   
 repeats = n\_repeats)  
   
 train\_result <- **train**(formula,   
 data = dataset,   
 method = method\_train,   
 trControl = train\_control)  
   
 **return**(train\_result)  
}  
  
variaveis\_relevantes\_trainControl <- **fun\_train\_control**(metodo\_trainControl = "repeatedcv",  
 number\_kfold = 10,  
 n\_repeats = 3,  
 formula = credit.rating **~** .,  
 dataset = dados\_treino,  
 method\_train = "glm")  
  
**varImp**(variaveis\_relevantes\_trainControl)

## glm variable importance  
##   
## only 20 most important variables shown (out of 38)  
##   
## Overall  
## account.balance3 100.00  
## savings4 63.93  
## credit.purpose4 58.27  
## credit.purpose3 49.47  
## credit.duration.months 46.81  
## previous.credit.payment.status3 46.79  
## employment.duration3 41.80  
## credit.purpose2 39.63  
## current.assets4 34.48  
## savings3 34.16  
## installment.rate4 26.15  
## foreign.worker2 26.13  
## bank.credits2 24.81  
## employment.duration2 21.71  
## residence.duration2 21.25  
## savings2 19.17  
## employment.duration4 16.93  
## apartment.type2 15.23  
## guarantor2 15.21  
## credit.amount 15.15

**plot**(**varImp**(variaveis\_relevantes\_trainControl))



*# RESULTADO: Análisando os 3 algoritmos utilizandos para identificar variáveis mais relevantes, é possível notar que ambos os*   
*# algoritmos apresentara quase as mesmas variáveis como sendo "relevantes".*   
*# A divergência entre um modelo e outro é ACEITÁVEL e sempre vai ocorrer. Cabe ao analista escolher quais são as variáveis com*  
*# maior relevancia e utiliza-las. Os algoritmos apenas auxiliam para essa tomada de decisão.*  
  
################################################################################################################################################  
 ################## ETAPA 5: CONSTRUINDO UM MODELO UTILIZANDO OS DADOS DE TREINO ##################   
################################################################################################################################################  
  
*# Durante essa etapa iremos realizar a construção de um modelo de classificação utilizando o algoritmo de regressão Logística,*  
*# que é utilizado e recomendado para fazer classificações de 2 possíveis OUTPUTs onde os dados "independentes" possuem*  
*# relações entre sí para realizar o output classificatório.*  
*# No caso da classificação que queremos realizar é se o cliente irá receber crédito/empréstimo da agência ou não.*  
  
modelo\_logistico <- **glm**("credit.rating ~ .", data = dados\_treino, family = **binomial**(link = "logit"))  
**summary**(modelo\_logistico)

##   
## Call:  
## glm(formula = "credit.rating ~ .", family = binomial(link = "logit"),   
## data = dados\_treino)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.6950 -0.6818 0.3436 0.6980 1.8590   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 1.48051 0.96881 1.528 0.126471   
## account.balance2 0.16702 0.25743 0.649 0.516484   
## account.balance3 1.82609 0.26682 6.844 7.71e-12 \*\*\*  
## credit.duration.months -0.43773 0.13518 -3.238 0.001203 \*\*   
## previous.credit.payment.status2 0.18817 0.36135 0.521 0.602554   
## previous.credit.payment.status3 1.25999 0.38928 3.237 0.001209 \*\*   
## credit.purpose2 -1.59354 0.57914 -2.752 0.005932 \*\*   
## credit.purpose3 -1.90412 0.55698 -3.419 0.000629 \*\*\*  
## credit.purpose4 -2.20085 0.54816 -4.015 5.95e-05 \*\*\*  
## credit.amount -0.16616 0.15222 -1.092 0.275001   
## savings2 0.48961 0.35884 1.364 0.172431   
## savings3 0.90603 0.38062 2.380 0.017292 \*   
## savings4 1.49858 0.34068 4.399 1.09e-05 \*\*\*  
## employment.duration2 0.43824 0.28526 1.536 0.124476   
## employment.duration3 1.01902 0.35159 2.898 0.003751 \*\*   
## employment.duration4 0.40651 0.33520 1.213 0.225231   
## installment.rate2 -0.18241 0.37656 -0.484 0.628099   
## installment.rate3 -0.35913 0.42383 -0.847 0.396799   
## installment.rate4 -0.67736 0.36868 -1.837 0.066172 .   
## marital.status3 0.25287 0.24785 1.020 0.307602   
## marital.status4 0.30820 0.39490 0.780 0.435128   
## guarantor2 0.39633 0.36158 1.096 0.273028   
## residence.duration2 -0.51923 0.34490 -1.505 0.132207   
## residence.duration3 -0.17254 0.38260 -0.451 0.652014   
## residence.duration4 0.02245 0.34637 0.065 0.948315   
## current.assets2 -0.29783 0.29891 -0.996 0.319076   
## current.assets3 -0.13247 0.28947 -0.458 0.647228   
## current.assets4 -1.21877 0.50738 -2.402 0.016303 \*   
## age 0.12894 0.12381 1.041 0.297666   
## other.credits2 0.28376 0.27262 1.041 0.297937   
## apartment.type2 0.31751 0.28929 1.098 0.272400   
## apartment.type3 0.44759 0.57899 0.773 0.439490   
## bank.credits2 -0.48964 0.28029 -1.747 0.080647 .   
## occupation2 -0.41832 0.68558 -0.610 0.541748   
## occupation3 -0.51814 0.65709 -0.789 0.430380   
## occupation4 -0.29902 0.70716 -0.423 0.672409   
## dependents2 -0.06288 0.29843 -0.211 0.833128   
## telephone2 0.21311 0.23820 0.895 0.370962   
## foreign.worker2 1.64707 0.89704 1.836 0.066341 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 855.21 on 699 degrees of freedom  
## Residual deviance: 613.17 on 661 degrees of freedom  
## AIC: 691.17  
##   
## Number of Fisher Scoring iterations: 5

previsao\_modelo <- **predict**(modelo\_logistico, dados\_teste, type = "response")  
  
*# O resultado do modelo de Regressão Logística consiste em "probabilidade" de um evento acontecer devido ao "response".*  
*# O resultado varia entre 0-1 e iremos realizar um arredondamento nos valores para os valores serem 0 ou 1,*  
*# para posteriormente ser possível realizar a comparação do resultado através do método confusionMatrix.*  
previsao\_modelo <- **round**(previsao\_modelo)  
previsao\_modelo

## 2 5 7 8 17 18 19 22 23 26 32 34 35 39 42 47 48 49   
## 1 1 1 1 1 1 1 1 0 1 1 1 0 1 1 1 1 1   
## 57 58 62 63 69 73 75 76 81 83 89 93 101 113 119 121 123 124   
## 1 1 1 1 1 1 1 1 1 0 1 0 1 1 1 0 1 1   
## 125 126 132 133 143 145 150 153 154 156 157 159 160 164 170 178 180 181   
## 1 1 1 1 1 1 0 1 1 1 0 0 1 1 1 1 1 1   
## 197 198 206 207 208 222 223 224 226 232 234 236 237 240 244 245 246 247   
## 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1   
## 248 249 252 253 271 273 274 276 282 286 287 288 290 291 298 305 306 307   
## 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1   
## 314 315 317 319 324 327 333 334 347 350 354 355 360 366 367 373 376 383   
## 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1   
## 388 395 396 398 399 400 401 402 406 407 409 410 412 414 430 431 432 436   
## 1 1 1 1 1 1 0 1 0 1 1 1 1 1 1 1 1 1   
## 442 453 454 455 457 459 460 461 462 463 466 469 472 474 475 483 485 490   
## 1 0 0 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1   
## 492 498 500 501 502 512 514 517 520 522 523 524 526 531 534 536 541 550   
## 1 1 1 1 1 1 0 1 1 0 1 1 1 1 1 0 0 1   
## 553 559 561 562 573 575 576 578 580 581 584 585 587 588 589 590 595 597   
## 1 1 1 1 1 1 0 0 1 0 0 0 0 0 1 1 0 1   
## 598 599 602 605 606 607 610 611 612 614 622 624 627 636 637 638 640 644   
## 1 1 1 0 1 1 0 0 1 0 0 1 1 0 1 0 1 0   
## 649 650 651 658 661 670 671 674 675 682 687 690 697 698 699 701 702 706   
## 0 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1   
## 707 711 713 714 717 718 719 724 735 736 737 738 739 740 741 743 750 754   
## 1 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1   
## 756 758 763 776 777 782 795 797 799 802 805 808 812 815 819 829 835 836   
## 0 0 1 0 1 1 1 1 0 0 1 0 0 1 1 0 1 1   
## 837 845 847 856 862 863 864 865 867 870 872 874 875 882 884 885 886 892   
## 1 1 0 0 0 1 1 0 0 0 1 1 0 1 0 0 0 1   
## 895 896 904 914 915 916 928 935 936 942 943 947 948 949 950 951 954 958   
## 1 1 1 0 0 1 0 1 1 0 1 1 1 0 1 0 1 1   
## 961 963 969 972 979 982 986 989 991 993 994 998   
## 1 0 0 0 1 1 1 0 0 1 1 1

*# Avaliando o resultado previsto pelo modelo criado com todas as variáveis*  
*# A avaliação é feita utilizando como referencia, ou seja, valores corretos, os dados de teste*  
  
*# Modelo criado com dados treino*  
*# Previsão feita com dados teste*  
*# E Avaliação feita com dados teste*  
**confusionMatrix**(**table**(data = previsao\_modelo, reference = dados\_teste[,1]), positive = '1')

## Confusion Matrix and Statistics  
##   
## reference  
## data 0 1  
## 0 41 30  
## 1 49 180  
##   
## Accuracy : 0.7367   
## 95% CI : (0.683, 0.7856)  
## No Information Rate : 0.7   
## P-Value [Acc > NIR] : 0.09179   
##   
## Kappa : 0.3328   
##   
## Mcnemar's Test P-Value : 0.04285   
##   
## Sensitivity : 0.8571   
## Specificity : 0.4556   
## Pos Pred Value : 0.7860   
## Neg Pred Value : 0.5775   
## Prevalence : 0.7000   
## Detection Rate : 0.6000   
## Detection Prevalence : 0.7633   
## Balanced Accuracy : 0.6563   
##   
## 'Positive' Class : 1   
##

*# Resultado: Obtivemos 78% de acertividade dos dados utilizando todas as variáveis em um modelo de regressão Logistica*  
  
################################################################################################################################################  
 ################## ETAPA 6: OTIMIZANDO O MODELO UTILIZANDO SOMENTE AS VARIÁVEIS MAIS RELEVANTES ##################  
################################################################################################################################################  
  
*# De acordo com o resultado da etapa 4, as variáveis mais relevantes são:*  
 *# account.balance | credit.duration.months | previous.credit.payment.status | credit.purpose | credit.amount | savings*  
  
modelo\_logistico\_v2 <- **glm**("credit.rating ~ account.balance + credit.duration.months + previous.credit.payment.status + credit.purpose + credit.amount + savings",  
 data = dados\_treino, family = **binomial**(link = "logit"))  
  
**summary**(modelo\_logistico\_v2)

##   
## Call:  
## glm(formula = "credit.rating ~ account.balance + credit.duration.months + previous.credit.payment.status + credit.purpose + credit.amount + savings",   
## family = binomial(link = "logit"), data = dados\_treino)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.7429 -0.8518 0.4145 0.7209 1.9862   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 0.74398 0.53004 1.404 0.160425   
## account.balance2 0.24397 0.23703 1.029 0.303355   
## account.balance3 1.78873 0.24875 7.191 6.44e-13 \*\*\*  
## credit.duration.months -0.50553 0.12177 -4.152 3.30e-05 \*\*\*  
## previous.credit.payment.status2 0.51472 0.31697 1.624 0.104409   
## previous.credit.payment.status3 1.27562 0.33604 3.796 0.000147 \*\*\*  
## credit.purpose2 -1.44024 0.50893 -2.830 0.004656 \*\*   
## credit.purpose3 -1.67364 0.49251 -3.398 0.000678 \*\*\*  
## credit.purpose4 -1.86981 0.48767 -3.834 0.000126 \*\*\*  
## credit.amount -0.04452 0.12295 -0.362 0.717265   
## savings2 0.39082 0.32459 1.204 0.228573   
## savings3 0.74332 0.35169 2.114 0.034552 \*   
## savings4 1.32447 0.31147 4.252 2.12e-05 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 855.21 on 699 degrees of freedom  
## Residual deviance: 658.69 on 687 degrees of freedom  
## AIC: 684.69  
##   
## Number of Fisher Scoring iterations: 5

previsao\_modelo\_v2 <- **predict**(modelo\_logistico\_v2, dados\_teste, type = "response")  
previsao\_modelo\_v2 <- **round**(previsao\_modelo\_v2)  
previsao\_modelo\_v2

## 2 5 7 8 17 18 19 22 23 26 32 34 35 39 42 47 48 49   
## 1 1 1 1 1 1 0 1 0 1 1 1 0 1 0 1 1 1   
## 57 58 62 63 69 73 75 76 81 83 89 93 101 113 119 121 123 124   
## 1 0 1 1 1 1 1 1 1 1 1 0 1 1 1 0 1 1   
## 125 126 132 133 143 145 150 153 154 156 157 159 160 164 170 178 180 181   
## 1 1 1 1 1 1 1 1 0 1 0 0 1 1 1 1 1 1   
## 197 198 206 207 208 222 223 224 226 232 234 236 237 240 244 245 246 247   
## 1 1 0 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1   
## 248 249 252 253 271 273 274 276 282 286 287 288 290 291 298 305 306 307   
## 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1   
## 314 315 317 319 324 327 333 334 347 350 354 355 360 366 367 373 376 383   
## 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1   
## 388 395 396 398 399 400 401 402 406 407 409 410 412 414 430 431 432 436   
## 1 1 1 1 1 0 0 1 0 1 1 1 1 1 1 1 1 1   
## 442 453 454 455 457 459 460 461 462 463 466 469 472 474 475 483 485 490   
## 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1   
## 492 498 500 501 502 512 514 517 520 522 523 524 526 531 534 536 541 550   
## 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 0 1 1   
## 553 559 561 562 573 575 576 578 580 581 584 585 587 588 589 590 595 597   
## 1 1 1 1 1 1 0 0 0 0 1 1 1 0 0 1 0 1   
## 598 599 602 605 606 607 610 611 612 614 622 624 627 636 637 638 640 644   
## 1 0 1 0 1 1 1 1 1 1 1 1 1 0 1 1 1 1   
## 649 650 651 658 661 670 671 674 675 682 687 690 697 698 699 701 702 706   
## 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1   
## 707 711 713 714 717 718 719 724 735 736 737 738 739 740 741 743 750 754   
## 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1   
## 756 758 763 776 777 782 795 797 799 802 805 808 812 815 819 829 835 836   
## 0 0 1 0 1 1 1 1 1 1 1 0 0 1 1 1 1 1   
## 837 845 847 856 862 863 864 865 867 870 872 874 875 882 884 885 886 892   
## 1 1 1 0 0 1 1 0 1 0 1 1 0 1 0 1 0 1   
## 895 896 904 914 915 916 928 935 936 942 943 947 948 949 950 951 954 958   
## 1 1 0 0 0 0 0 1 1 1 1 1 1 0 1 0 1 1   
## 961 963 969 972 979 982 986 989 991 993 994 998   
## 1 0 0 0 1 1 1 0 0 1 1 1

**confusionMatrix**(**table**(data = previsao\_modelo\_v2, reference = dados\_teste[,1]), positive = '1')

## Confusion Matrix and Statistics  
##   
## reference  
## data 0 1  
## 0 31 27  
## 1 59 183  
##   
## Accuracy : 0.7133   
## 95% CI : (0.6586, 0.7638)  
## No Information Rate : 0.7   
## P-Value [Acc > NIR] : 0.3321222   
##   
## Kappa : 0.2403   
##   
## Mcnemar's Test P-Value : 0.0008293   
##   
## Sensitivity : 0.8714   
## Specificity : 0.3444   
## Pos Pred Value : 0.7562   
## Neg Pred Value : 0.5345   
## Prevalence : 0.7000   
## Detection Rate : 0.6100   
## Detection Prevalence : 0.8067   
## Balanced Accuracy : 0.6079   
##   
## 'Positive' Class : 1   
##

*# Resultado: Obtivemos 76% de acertividade dos dados utilizando somente variáveis consideradas "mais importantes".*  
  
  
### Realizando um teste de otimização utilizando o algoritmo RandomForest  
**library**("rpart")  
  
modelo\_randomForest <- **rpart**("credit.rating ~ account.balance + credit.duration.months + previous.credit.payment.status + credit.purpose + credit.amount + savings", data = dados\_treino, method = "class")  
**summary**(modelo\_randomForest)

## Call:  
## rpart(formula = "credit.rating ~ account.balance + credit.duration.months + previous.credit.payment.status + credit.purpose + credit.amount + savings",   
## data = dados\_treino, method = "class")  
## n= 700   
##   
## CP nsplit rel error xerror xstd  
## 1 0.07777778 0 1.0000000 1.0000000 0.05773503  
## 2 0.02380952 3 0.7666667 0.7714286 0.05313496  
## 3 0.01785714 4 0.7428571 0.7571429 0.05278723  
## 4 0.01190476 8 0.6714286 0.7761905 0.05324914  
## 5 0.01000000 10 0.6476190 0.7619048 0.05290401  
##   
## Variable importance  
## account.balance savings   
## 38 17   
## credit.duration.months previous.credit.payment.status   
## 13 13   
## credit.amount credit.purpose   
## 12 7   
##   
## Node number 1: 700 observations, complexity param=0.07777778  
## predicted class=1 expected loss=0.3 P(node) =1  
## class counts: 210 490  
## probabilities: 0.300 0.700   
## left son=2 (379 obs) right son=3 (321 obs)  
## Primary splits:  
## account.balance splits as LLR, improve=41.842580, (0 missing)  
## savings splits as LLRR, improve=13.074830, (0 missing)  
## previous.credit.payment.status splits as LRR, improve=10.897600, (0 missing)  
## credit.duration.months < -0.3236637 to the right, improve=10.404880, (0 missing)  
## credit.amount < 3.017181 to the right, improve= 7.525106, (0 missing)  
## Surrogate splits:  
## savings splits as LLRR, agree=0.596, adj=0.118, (0 split)  
## previous.credit.payment.status splits as LLR, agree=0.576, adj=0.075, (0 split)  
## credit.purpose splits as RLRL, agree=0.570, adj=0.062, (0 split)  
## credit.duration.months < -0.448054 to the right, agree=0.556, adj=0.031, (0 split)  
##   
## Node number 2: 379 observations, complexity param=0.07777778  
## predicted class=1 expected loss=0.4591029 P(node) =0.5414286  
## class counts: 174 205  
## probabilities: 0.459 0.541   
## left son=4 (293 obs) right son=5 (86 obs)  
## Primary splits:  
## savings splits as LLRR, improve=11.418470, (0 missing)  
## credit.duration.months < 0.1324342 to the right, improve=11.185020, (0 missing)  
## previous.credit.payment.status splits as LLR, improve= 6.995410, (0 missing)  
## credit.amount < 0.3307949 to the right, improve= 6.215921, (0 missing)  
## credit.purpose splits as RLLL, improve= 2.643871, (0 missing)  
##   
## Node number 3: 321 observations  
## predicted class=1 expected loss=0.1121495 P(node) =0.4585714  
## class counts: 36 285  
## probabilities: 0.112 0.888   
##   
## Node number 4: 293 observations, complexity param=0.07777778  
## predicted class=0 expected loss=0.4744027 P(node) =0.4185714  
## class counts: 154 139  
## probabilities: 0.526 0.474   
## left son=8 (131 obs) right son=9 (162 obs)  
## Primary splits:  
## credit.duration.months < 0.1324342 to the right, improve=12.348060000, (0 missing)  
## previous.credit.payment.status splits as LLR, improve= 7.403372000, (0 missing)  
## credit.amount < 0.226287 to the right, improve= 6.402667000, (0 missing)  
## credit.purpose splits as RLLL, improve= 2.046601000, (0 missing)  
## account.balance splits as LR-, improve= 0.001413069, (0 missing)  
## Surrogate splits:  
## credit.amount < -0.22416 to the right, agree=0.788, adj=0.527, (0 split)  
## credit.purpose splits as LRRR, agree=0.594, adj=0.092, (0 split)  
## savings splits as RL--, agree=0.580, adj=0.061, (0 split)  
## previous.credit.payment.status splits as LRR, agree=0.556, adj=0.008, (0 split)  
##   
## Node number 5: 86 observations  
## predicted class=1 expected loss=0.2325581 P(node) =0.1228571  
## class counts: 20 66  
## probabilities: 0.233 0.767   
##   
## Node number 8: 131 observations, complexity param=0.02380952  
## predicted class=0 expected loss=0.3129771 P(node) =0.1871429  
## class counts: 90 41  
## probabilities: 0.687 0.313   
## left son=16 (114 obs) right son=17 (17 obs)  
## Primary splits:  
## credit.purpose splits as RLLL, improve=4.360646, (0 missing)  
## credit.duration.months < 2.205607 to the right, improve=2.532947, (0 missing)  
## previous.credit.payment.status splits as LLR, improve=2.402759, (0 missing)  
## credit.amount < -0.3621459 to the left, improve=1.253896, (0 missing)  
## savings splits as LR--, improve=1.241641, (0 missing)  
##   
## Node number 9: 162 observations, complexity param=0.01785714  
## predicted class=1 expected loss=0.3950617 P(node) =0.2314286  
## class counts: 64 98  
## probabilities: 0.395 0.605   
## left son=18 (108 obs) right son=19 (54 obs)  
## Primary splits:  
## previous.credit.payment.status splits as LLR, improve=4.8395060, (0 missing)  
## credit.amount < -0.940128 to the right, improve=2.5187650, (0 missing)  
## credit.duration.months < -0.7797616 to the right, improve=1.8583710, (0 missing)  
## account.balance splits as LR-, improve=1.1993080, (0 missing)  
## savings splits as RL--, improve=0.3910029, (0 missing)  
## Surrogate splits:  
## credit.purpose splits as RLLL, agree=0.673, adj=0.019, (0 split)  
## credit.amount < 0.0151455 to the left, agree=0.673, adj=0.019, (0 split)  
##   
## Node number 16: 114 observations, complexity param=0.01190476  
## predicted class=0 expected loss=0.2631579 P(node) =0.1628571  
## class counts: 84 30  
## probabilities: 0.737 0.263   
## left son=32 (91 obs) right son=33 (23 obs)  
## Primary splits:  
## savings splits as LR--, improve=1.6973870, (0 missing)  
## previous.credit.payment.status splits as LLR, improve=1.6635180, (0 missing)  
## credit.duration.months < 2.205607 to the right, improve=1.6176410, (0 missing)  
## credit.amount < -0.3621459 to the left, improve=0.6211646, (0 missing)  
## account.balance splits as RL-, improve=0.3438596, (0 missing)  
## Surrogate splits:  
## credit.duration.months < 2.744631 to the left, agree=0.807, adj=0.043, (0 split)  
##   
## Node number 17: 17 observations  
## predicted class=1 expected loss=0.3529412 P(node) =0.02428571  
## class counts: 6 11  
## probabilities: 0.353 0.647   
##   
## Node number 18: 108 observations, complexity param=0.01785714  
## predicted class=1 expected loss=0.4814815 P(node) =0.1542857  
## class counts: 52 56  
## probabilities: 0.481 0.519   
## left son=36 (99 obs) right son=37 (9 obs)  
## Primary splits:  
## credit.amount < -0.9434935 to the right, improve=2.6936030, (0 missing)  
## previous.credit.payment.status splits as LR-, improve=2.3441080, (0 missing)  
## account.balance splits as LR-, improve=2.0622900, (0 missing)  
## credit.duration.months < -0.821225 to the right, improve=1.4807050, (0 missing)  
## credit.purpose splits as LRRL, improve=0.5965142, (0 missing)  
## Surrogate splits:  
## credit.duration.months < -1.194396 to the right, agree=0.926, adj=0.111, (0 split)  
##   
## Node number 19: 54 observations  
## predicted class=1 expected loss=0.2222222 P(node) =0.07714286  
## class counts: 12 42  
## probabilities: 0.222 0.778   
##   
## Node number 32: 91 observations  
## predicted class=0 expected loss=0.2197802 P(node) =0.13  
## class counts: 71 20  
## probabilities: 0.780 0.220   
##   
## Node number 33: 23 observations, complexity param=0.01190476  
## predicted class=0 expected loss=0.4347826 P(node) =0.03285714  
## class counts: 13 10  
## probabilities: 0.565 0.435   
## left son=66 (14 obs) right son=67 (9 obs)  
## Primary splits:  
## previous.credit.payment.status splits as LLR, improve=3.4789510, (0 missing)  
## credit.purpose splits as -LLR, improve=1.1073780, (0 missing)  
## credit.duration.months < 0.5056053 to the right, improve=0.8876812, (0 missing)  
## credit.amount < -0.1090241 to the right, improve=0.8876812, (0 missing)  
## Surrogate splits:  
## credit.amount < -0.4719678 to the right, agree=0.739, adj=0.333, (0 split)  
##   
## Node number 36: 99 observations, complexity param=0.01785714  
## predicted class=0 expected loss=0.4848485 P(node) =0.1414286  
## class counts: 51 48  
## probabilities: 0.515 0.485   
## left son=72 (12 obs) right son=73 (87 obs)  
## Primary splits:  
## credit.amount < -0.8740577 to the left, improve=2.7648900, (0 missing)  
## previous.credit.payment.status splits as LR-, improve=2.3111240, (0 missing)  
## account.balance splits as LR-, improve=1.4160840, (0 missing)  
## credit.purpose splits as RRRL, improve=0.5259740, (0 missing)  
## credit.duration.months < -0.821225 to the right, improve=0.3996004, (0 missing)  
##   
## Node number 37: 9 observations  
## predicted class=1 expected loss=0.1111111 P(node) =0.01285714  
## class counts: 1 8  
## probabilities: 0.111 0.889   
##   
## Node number 66: 14 observations  
## predicted class=0 expected loss=0.2142857 P(node) =0.02  
## class counts: 11 3  
## probabilities: 0.786 0.214   
##   
## Node number 67: 9 observations  
## predicted class=1 expected loss=0.2222222 P(node) =0.01285714  
## class counts: 2 7  
## probabilities: 0.222 0.778   
##   
## Node number 72: 12 observations  
## predicted class=0 expected loss=0.1666667 P(node) =0.01714286  
## class counts: 10 2  
## probabilities: 0.833 0.167   
##   
## Node number 73: 87 observations, complexity param=0.01785714  
## predicted class=1 expected loss=0.4712644 P(node) =0.1242857  
## class counts: 41 46  
## probabilities: 0.471 0.529   
## left son=146 (17 obs) right son=147 (70 obs)  
## Primary splits:  
## previous.credit.payment.status splits as LR-, improve=2.3260700, (0 missing)  
## account.balance splits as LR-, improve=1.1109160, (0 missing)  
## credit.purpose splits as LLRL, improve=0.9381400, (0 missing)  
## credit.amount < 0.2282354 to the right, improve=0.8991790, (0 missing)  
## credit.duration.months < -0.821225 to the right, improve=0.2637845, (0 missing)  
##   
## Node number 146: 17 observations  
## predicted class=0 expected loss=0.2941176 P(node) =0.02428571  
## class counts: 12 5  
## probabilities: 0.706 0.294   
##   
## Node number 147: 70 observations  
## predicted class=1 expected loss=0.4142857 P(node) =0.1  
## class counts: 29 41  
## probabilities: 0.414 0.586

previsao\_randomForest <- **predict**(modelo\_randomForest, dados\_teste, type = "class")  
  
**confusionMatrix**(**table**(data = previsao\_randomForest, reference = dados\_teste[,1]), positive = '1')

## Confusion Matrix and Statistics  
##   
## reference  
## data 0 1  
## 0 33 26  
## 1 57 184  
##   
## Accuracy : 0.7233   
## 95% CI : (0.669, 0.7732)  
## No Information Rate : 0.7   
## P-Value [Acc > NIR] : 0.2072197   
##   
## Kappa : 0.2694   
##   
## Mcnemar's Test P-Value : 0.0009915   
##   
## Sensitivity : 0.8762   
## Specificity : 0.3667   
## Pos Pred Value : 0.7635   
## Neg Pred Value : 0.5593   
## Prevalence : 0.7000   
## Detection Rate : 0.6133   
## Detection Prevalence : 0.8033   
## Balanced Accuracy : 0.6214   
##   
## 'Positive' Class : 1   
##

*# Resultado : Obtivemos 77% de acertividade dos dados utilizando o modelo RandomForest*  
  
  
*# Resultado Final: De todos os modelos utilizados, O modelo com todas as variáveis de Regressão Logistica obteve um resultado melhor (78%) em comparação ao resultado quando selecionamos somente variáveis consideradas*  
*# "importantes" durante a análise.*  
  
################################################################################################################################################  
 ################## ETAPA 7: APRESENTANDO O RESULTADO DO MODELO FINAL ATRAVÉS DE GRÁFICOS DE CURVAS ROC ##################   
################################################################################################################################################  
  
*# Atribuindo o melhor modelo que obtivemos em uma nova variável.*  
best\_fit\_model <- modelo\_logistico  
best\_fit\_prediction <- **predict**(best\_fit\_model, dados\_teste[,**-**1], type = "response") *# Não passando a coluna que queremos prever durante a previsão.*  
  
  
*# Carregando Biblioteca de utilitários para construção de gráficos*  
**source**("plot\_utils.R")

## Loading required package: gplots

##   
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':  
##   
## lowess

*# Pacote útil para gerar curvas ROC*  
**library**("ROCR")  
  
previsoes <- **prediction**(best\_fit\_prediction, dados\_teste[,1])  
  
*# Acima da linha vermelha representa "boa Previsão".*   
*# Quanto mais alinhado ao canto superior esquerdo, melhor é o algoritmo e melhor foi a previsão.*  
**plot.roc.curve**(previsoes, title.text = "Curva ROC")



**plot.pr.curve**(previsoes, title.text = "Curva Precision/Recall")



## OBSERVAÇÕES FINAIS E IMPORTANTES:   
*# O MODELO E O RESULTADO AQUI APRESENTADO, PODEM SOFRER ANÁLISES E AJUSTES EM BUSCA DE MELHORAR A ASSERTIVIDADE DO MODELO E AUMENTAR A TAXA DE PERCENTUAL.*  
*# O FIM DA ANÁLISE E ACEITAÇÃO DO RESULTADO, VARIA DE ACORDO COM O QUE ESTÁ SE PREVENDO.*  
*# NO CASO DE CONCEDER CREDITO OU NÃO AO CLIENTE, ESTOU CONSIDERANDO QUE 78% DE ASSERTIVIDADE DO MEU MODELO PODE SER CONSIDERADO UMA TAXA BOA PARA*  
*# PREVISÕES E CONSIDERAÇÕES EM CONCEDER OU NÃO O EMPRÉSTIMO.*  
*# LEMBRANDO TAMBÉM QUE A DECISÃO FINAL DE CONCEDER OU NÃO, CABE A PESSOA FÍSICA FINAL, QUE IRÁ UTILIZAR A PREVISÃO DO ALGORITMO PARA LHE AUXILIAR*  
*# NA TOMADA DE DECISÃO.*