Avaliação de Risco de crédito

Avaliação de Risco de Crédito

Para esta análise, vamos usar um conjunto de dados German Credit Data, já devidamente limpo e organizado para a criação do modelo preditivo.

Todo o projecto será descrito de acordo com suas etapas.

Etapa 1 - Colectando os Dados

Aqui está a coleta de dados, neste caso um arquivo csv.

```r
# Colectando dados
credit.df <- read.csv("credit_dataset.csv", header = TRUE, sep = ",")
```

Etapa 2 - Normalizando os Dados

```r
## Convertendo as variáveis para o tipo factor
(categórica) to.factors <- function(df, variables){
  for (variable in variables){
    df[[variable]] <- as.factor(df[[variable]])
  }
  return(df)
}

## Normalização
scale.features <- function(df, variables){
  for (variable in variables){
    df[[variable]] <- scale(df[[variable]], center=T, scale=T)
  }
  return(df)
}

# Normalizando as variáveis
numeric.vars <- c("credit.duration.months", "age", "credit.amount")
credit.df <- scale.features(credit.df, numeric.vars)

# Variáveis do tipo fator
categorical.vars <- 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')

credit.df <- to.factors(df = credit.df, variables = categorical.vars)
```

Etapa 3 - Dividindo os dados em dados de treino e de teste

```
# Dividindo os dados em treino e teste - 60:40 ratio
indexes <- sample(1:nrow(credit.df), size = 0.6 * nrow(credit.df))
train.data <- credit.df[indexes,]
test.data <- credit.df[-indexes,]
```

Etapa 4 - Feature Selection

```
library(caret)
```

```
## Loading required package: lattice

## Loading required package: ggplot2
```

```
library(randomForest)
```

```
## randomForest 4.6-12

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

##
## Attaching package: 'randomForest'

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

```
# Função para seleção de variáveis
run.feature.selection <- function(num.iters=20, feature.vars, class.var){
  set.seed(10)
  variable.sizes <- 1:10
  control <- rfeControl(functions = rfFuncs, method = "cv",
                        verbose = FALSE, returnResamp = "all",
                        number = num.iters)
  results.rfe <- rfe(x = feature.vars, y = class.var,
                     sizes = variable.sizes,
                     rfeControl = control)
  return(results.rfe)
}

# Executando a função
rfe.results <- run.feature.selection(feature.vars = train.data[,-1],
                                     class.var = train.data[,1])


# Visualizando os resultados
rfe.results
```

```
##
## Recursive feature selection
##
## Outer resampling method: Cross-Validated (20 fold)
##
## Resampling performance over subset size:
##
```

```
##  Variables Accuracy  Kappa AccuracySD KappaSD Selected
##          1   0.7084 0.0000    0.01123  0.0000
##          2   0.7122 0.1448    0.05088  0.1557
##          3   0.7219 0.1737    0.06723  0.1935
##          4   0.7354 0.2981    0.05283  0.1515
##          5   0.7400 0.3064    0.06121  0.1539
##          6   0.7315 0.2788    0.07091  0.1858
##          7   0.7367 0.2866    0.05637  0.1545
##          8   0.7450 0.3085    0.05959  0.1497
##          9   0.7517 0.3384    0.06083  0.1289
##         10   0.7519 0.3268    0.07723  0.2035
##         20   0.7569 0.3145    0.06955  0.1886        *
##
## The top 5 variables (out of 20):
##    account.balance, credit.duration.months, savings, previous.credit.payment.status, credit.amount
```

```
varImp((rfe.results))
```

```
##                                     Overall
## account.balance                 16.00728097
## credit.duration.months           8.99705143
## savings                          6.84046896
## previous.credit.payment.status   6.49983486
## credit.amount                    5.66019351
## current.assets                   4.15215355
## credit.purpose                   3.62016764
## guarantor                        3.21205102
## age                              3.07676541
## occupation                       2.10634980
## residence.duration               2.01883636
## bank.credits                     1.38677930
## employment.duration              1.17958790
## marital.status                   1.17537926
## telephone                        1.15876517
## other.credits                    0.88461665
## apartment.type                   0.60053602
## foreign.worker                   0.45767643
## installment.rate                -0.04331768
## dependents                      -0.06021166
```

Etapa 5 - Criando e Avaliando a Primeira Versão do Modelo

```
# Criando e Avaliando o Modelo
library(caret)
library(ROCR)
```

```
## Loading required package: gplots
```

```
##
## Attaching package: 'gplots'
```

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

```r
# Biblioteca de utilitários para construção de gráficos
source("plot_utils.R")

## separate feature and class variables
test.feature.vars <- test.data[,-1]
test.class.var <- test.data[,1]

# Construindo um modelo de regressão logística
formula.init <- "credit.rating ~ ."
formula.init <- as.formula(formula.init)
lr.model <- glm(formula = formula.init, data = train.data, family = "binomial")

# Visualizando o modelo
summary(lr.model)
```

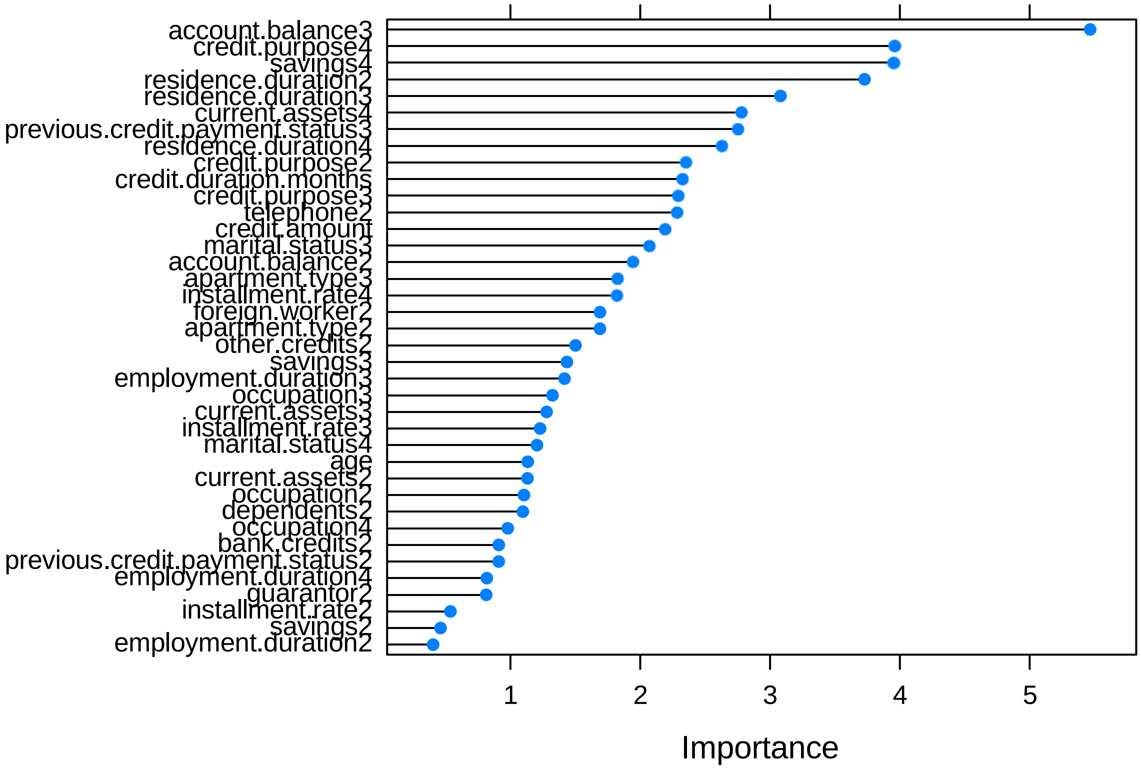
```
## 
## Call:
## glm(formula = formula.init, family = "binomial", data = train.data)
## 
## Deviance Residuals: 
##     Min       1Q   Median       3Q      Max  
## -2.5198  -0.7107   0.3672   0.7170   2.0816  
## 
## Coefficients:
##                                  Estimate Std. Error z value Pr(>|z|)    
## (Intercept)                        0.4512     1.0293   0.438 0.661108    
## account.balance2                   0.5516     0.2839   1.943 0.052059 .  
## account.balance3                   1.5246     0.2789   5.466 4.60e-08 ***
## credit.duration.months            -0.3501     0.1506  -2.324 0.020127 *  
## previous.credit.payment.status2    0.3764     0.4137   0.910 0.362816    
## previous.credit.payment.status3    1.1709     0.4252   2.754 0.005896 ** 
## credit.purpose2                   -1.2081     0.5141  -2.350 0.018777 *  
## credit.purpose3                   -1.1228     0.4899  -2.292 0.021908 *  
## credit.purpose4                   -1.8748     0.4734  -3.960 7.48e-05 ***
## credit.amount                     -0.3483     0.1589  -2.192 0.028404 *  
## savings2                           0.1848     0.4000   0.462 0.644037    
## savings3                           0.5381     0.3749   1.435 0.151200    
## savings4                           1.4534     0.3677   3.953 7.72e-05 ***
## employment.duration2               0.1274     0.3149   0.404 0.685904    
## employment.duration3               0.5427     0.3831   1.417 0.156547    
## employment.duration4               0.3008     0.3676   0.818 0.413139    
## installment.rate2                 -0.2313     0.4295  -0.538 0.590266    
## installment.rate3                 -0.5768     0.4695  -1.229 0.219196    
## installment.rate4                 -0.7767     0.4266  -1.820 0.068687 .  
## marital.status3                    0.5461     0.2635   2.072 0.038253 *  
## marital.status4                    0.4800     0.3985   1.204 0.228472    
## guarantor2                         0.3143     0.3863   0.814 0.415809    
## residence.duration2               -1.6768     0.4498  -3.728 0.000193 ***
## residence.duration3               -1.4765     0.4793  -3.080 0.002067 ** 
## residence.duration4               -1.1884     0.4519  -2.630 0.008539 ** 
## current.assets2                   -0.3744     0.3307  -1.132 0.257576    
## current.assets3                   -0.3907     0.3055  -1.279 0.200870    
## current.assets4                   -1.5252     0.5487  -2.780 0.005443 ** 
## age                                0.1510     0.1331   1.134 0.256823    
## other.credits2                   0.4215     0.2807   1.502  0.133206
## apartment.type2                  0.5295     0.3134   1.690  0.091069 .
## apartment.type3                  1.1317     0.6199   1.826  0.067888 .
## bank.credits2                   -0.2730     0.2999  -0.910  0.362636
## occupation2                      0.8544     0.7734   1.105  0.269281
## occupation3                      0.9920     0.7490   1.324  0.185373
## occupation4                      0.7644     0.7803   0.980  0.327268
## dependents2                     -0.3670     0.3349  -1.096  0.273068
## telephone2                       0.6370     0.2791   2.283  0.022456 *
## foreign.worker2                  1.3968     0.8264   1.690  0.090998 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 724.36  on 599  degrees of freedom
## Residual deviance: 534.53  on 561  degrees of freedom
## AIC: 612.53
##
## Number of Fisher Scoring iterations: 5
```

```
# Testando o modelo nos dados de teste
lr.predictions <- predict(lr.model, test.data, type="response")
lr.predictions <- round(lr.predictions)

# Avaliando o modelo
confusionMatrix(data = lr.predictions, reference = test.class.var, positive = '1')
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0  65  35
##          1  60 240
##
##                Accuracy : 0.7625
##                  95% CI : (0.7177, 0.8034)
##     No Information Rate : 0.6875
##     P-Value [Acc > NIR] : 0.0005677
##
##                   Kappa : 0.4154
##  Mcnemar's Test P-Value : 0.0138031
##
##             Sensitivity : 0.8727
##             Specificity : 0.5200
##          Pos Pred Value : 0.8000
##          Neg Pred Value : 0.6500
##              Prevalence : 0.6875
##          Detection Rate : 0.6000
##    Detection Prevalence : 0.7500
##       Balanced Accuracy : 0.6964
##
##        'Positive' Class : 1
##
```

Etapa 6 - Otimizando o Modelo

```
## Feature selection
formula <- "credit.rating ~ ."
formula <- as.formula(formula)
control <- trainControl(method = "repeatedcv", number = 10, repeats = 2)
model <- train(formula, data = train.data, method = "glm", trControl = control)
importance <- varImp(model, scale = FALSE)
plot(importance)
```



```
# Construindo o modelo com as variáveis selecionadas
formula.new <- "credit.rating ~ account.balance + credit.purpose + previous.credit.payment.status + sav
formula.new <- as.formula(formula.new)
lr.model.new <- glm(formula = formula.new, data = train.data, family = "binomial")

# Visualizando o modelo
summary(lr.model.new)
```

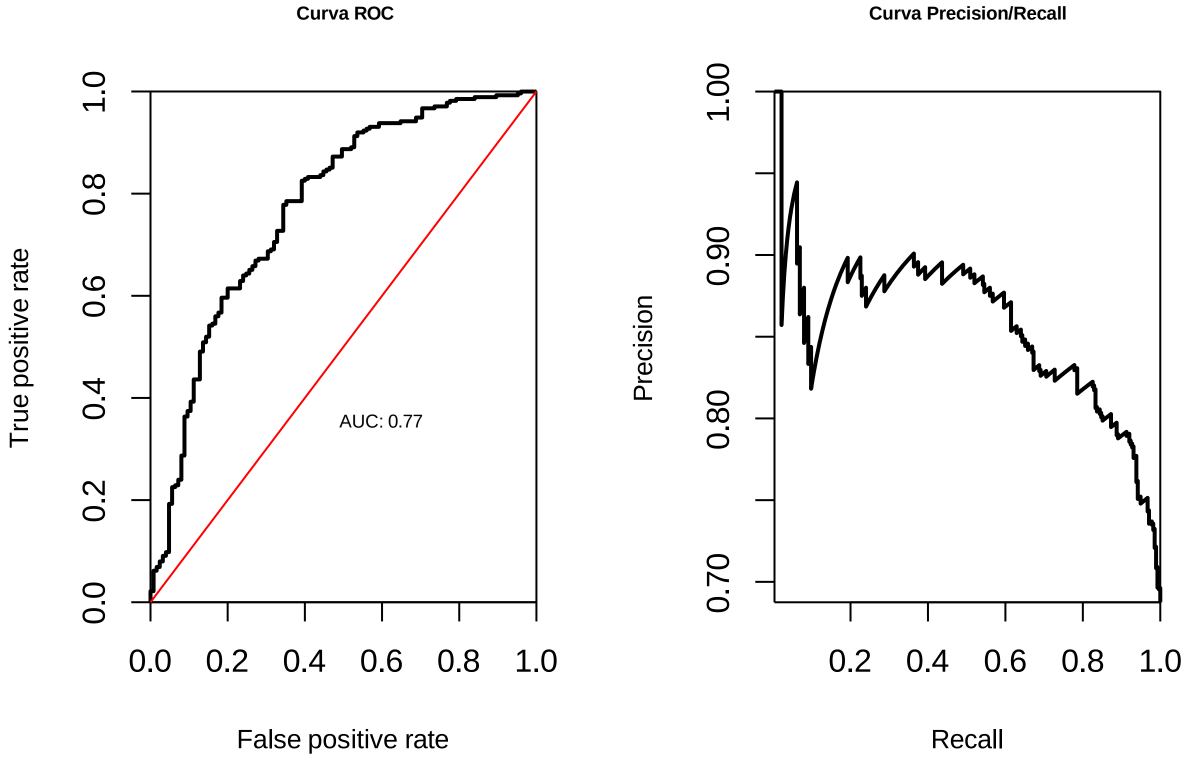
```
##
## Call:
## glm(formula = formula.new, family = "binomial", data = train.data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7632  -0.8394   0.4919   0.7565   1.8926
##
## Coefficients:
##                                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)                        0.1861     0.5123   0.363 0.716438
## account.balance2                  0.4906     0.2564   1.914 0.055676 .  
## account.balance3                  1.4456     0.2525   5.724 1.04e-08 ***
## credit.purpose2                  -0.9837     0.4530  -2.172 0.029880 *  
## credit.purpose3                  -0.8004     0.4262  -1.878 0.060394 .  
## credit.purpose4                  -1.4636     0.4189  -3.494 0.000476 ***
## previous.credit.payment.status2   0.6305     0.3594   1.754 0.079397 .  
## previous.credit.payment.status3   1.3134     0.3745   3.507 0.000453 ***
## savings2                          0.0856     0.3588   0.239 0.811448    
## savings3                          0.4412     0.3487   1.265 0.205760    
## savings4                          1.2358     0.3381   3.655 0.000257 ***
## credit.duration.months           -0.5619     0.1075  -5.229 1.70e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
##     Null deviance: 724.36  on 599  degrees of freedom
## Residual deviance: 590.90  on 588  degrees of freedom
## AIC: 614.9
## 
## Number of Fisher Scoring iterations: 5
```

```
# Testando o modelo nos dados de teste
lr.predictions.new <- predict(lr.model.new, test.data, type="response")
lr.predictions.new <- round(lr.predictions.new)

# Avaliando o modelo
confusionMatrix(data=lr.predictions.new, reference=test.class.var, positive='1')
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction   0   1
##          0  59  35
##          1  66 240
## 
##                Accuracy : 0.7475          
##                  95% CI : (0.7019, 0.7894)
##     No Information Rate : 0.6875          
##     P-Value [Acc > NIR] : 0.004988        
## 
##                   Kappa : 0.3697          
##  Mcnemar's Test P-Value : 0.002835        
## 
##             Sensitivity : 0.8727          
##             Specificity : 0.4720          
##          Pos Pred Value : 0.7843          
##          Neg Pred Value : 0.6277          
##              Prevalence : 0.6875          
##          Detection Rate : 0.6000          
##    Detection Prevalence : 0.7650          
##       Balanced Accuracy : 0.6724          
## 
##        'Positive' Class : 1               
## 
```

Etapa 7 - Curva ROC e Avaliação Final do Modelo

```
# Avaliando a performance do modelo

# Criando curvas ROC
lr.model.best <- lr.model
lr.prediction.values <- predict(lr.model.best, test.feature.vars, type = "response")
predictions <- prediction(lr.prediction.values, test.class.var)
par(mfrow = c(1,2))
plot.roc.curve(predictions, title.text = "Curva ROC")
plot.pr.curve(predictions, title.text = "Curva Precision/Recall")
```


Curva ROC
True positive rate
AUC: 0.77
False positive rate
Curva Precision/Recall
Precision
Recall