RIsk Analysis

Oracy Martos

November 25, 2018

Risk Analysis

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 projeto será descrito de acordo com suas etapas.

Step 1 - Collecting Data

Set Directory

setwd("/home/oracy/Documents/DSA_Projetos/DSA_Projetos/Big Data Analytics com R e Microsoft Azure Machine Learning/4.Analise de Risco de Credito") getwd()

```r
# Data gathering
# Loading the dataset into a dataframe
df <- read.csv('credit_dataset.csv', header = TRUE, sep = ',')
head(df)
```

```
##   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
```

```r
str(df)
```

```
## '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 ...
```

Step 2 - Normalizando os Dados

```r
# Loading my library
source('utils.R')

# Normalizing the variables
numeric.vars <- c("credit.duration.months", "age", "credit.amount")
df <- scale.features(df, numeric.vars)

# Factor type variables
factor.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')

df <- to.factors(df, factor.vars)
```

Step 3 - Splitting data in training and testing - 60:40 ratio

Font: https://cran.r-project.org/web/packages/dataPreparation/vignettes/train_test_prep.html

```
nrow(df)
```

```
## [1] 1000
```

```
index <- sample(1:nrow(df), 0.6 * nrow(df))
df_train <- df[index,]
df_test <- df[-index,]
length(df_train)
```

```
## [1] 21
```

```
length(df_test)
```

```
## [1] 21
```

Step 4 - Feature Selection

Font: https://machinelearningmastery.com/feature-selection-with-the-caret-r

Font: https://machinelearningmastery.com/feature-selection-with-the-caret-r

```
#install.packages('caret')
#install.packages('e1071', dependencies = TRUE)
#install.packages('randomForest')
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
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
```

```r
# Running the function
rfe.results <- feature.selection(feature.vars = df_train[,-1],
                                  class.var = df_train[,1])

# Viewing Results
rfe.results
```

```
##
## Recursive feature selection
##
## Outer resampling method: Cross-Validated (20 fold)
##
## Resampling performance over subset size:
##
##  Variables Accuracy  Kappa AccuracySD KappaSD Selected
##          1   0.7167 0.0000    0.01565  0.0000
##          2   0.7566 0.2980    0.06029  0.2039
##          3   0.7585 0.2949    0.06140  0.1916
##          4   0.7621 0.3714    0.08562  0.2223
##          5   0.7737 0.4056    0.07506  0.2041
##          6   0.7870 0.4316    0.07471  0.2096
##          7   0.8019 0.4652    0.07167  0.1915        *
##          8   0.7851 0.4073    0.06226  0.1844
##          9   0.7750 0.3892    0.06814  0.2054
##         10   0.7717 0.3883    0.07182  0.2038
##         20   0.7784 0.3627    0.06502  0.2026
##
## The top 5 variables (out of 7):
##    account.balance, credit.duration.months, savings, previous.credit.payment.status, credit.amount
```

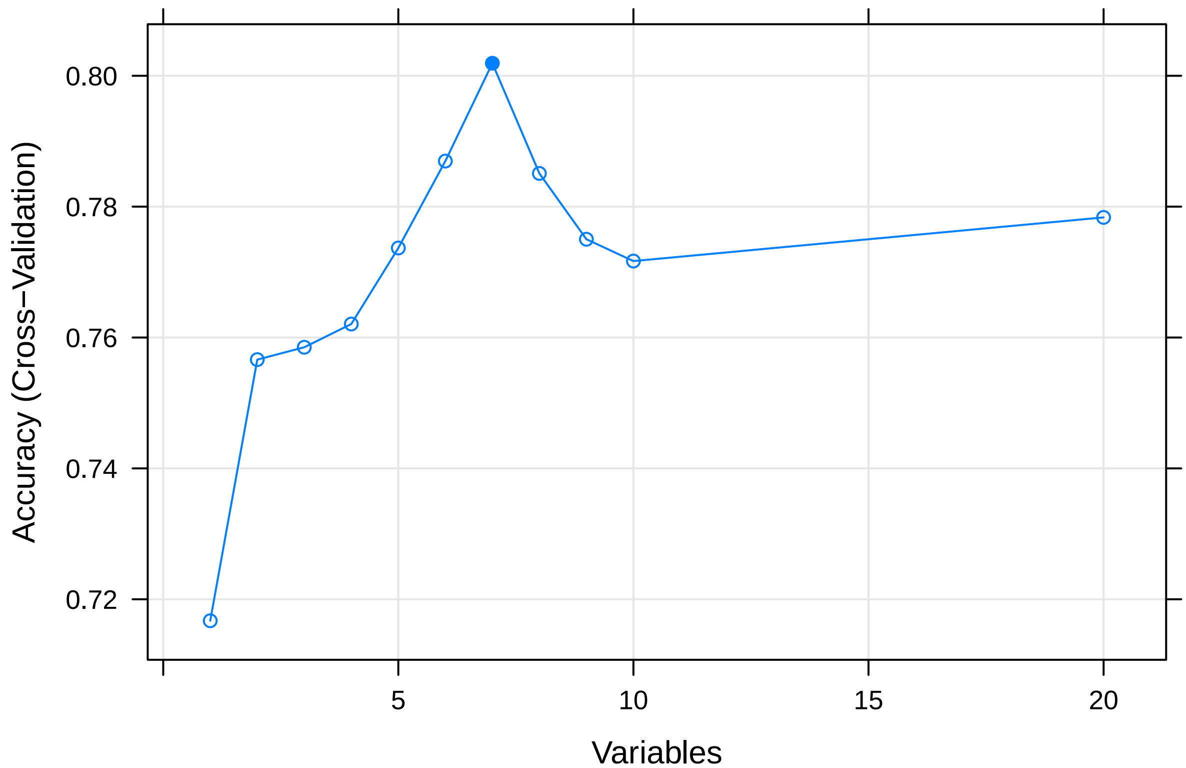
```r
varImp(rfe.results)
```

```
##                                  Overall
## account.balance                20.160172
## credit.duration.months         13.237115
## savings                         7.615321
## previous.credit.payment.status  6.745180
## credit.amount                   5.675587
## current.assets                  5.075549
## apartment.type                  4.812198
## other.credits                   4.602777
## age                             4.564267
```

```r
predictors(rfe.results)
```

```
## [1] "account.balance"                "credit.duration.months"
## [3] "savings"                        "previous.credit.payment.status"
## [5] "credit.amount"                  "current.assets"
## [7] "apartment.type"
```

```r
plot(rfe.results, type = c("g","o"))
```



Step 5 - Creating and Evaluating the Model

```r
#install.packages('ROCR')
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.var <- df_test[,-1]
test.class.var <- df_test[,1]
```

```r
# Building a Logistic Regression Model
lm.init <- 'credit.rating ~ .'
lm.init <- as.formula(lm.init)
lr.model <- glm(formula = lm.init, data = df_train, family = "binomial")

# Viewing the template
summary(lr.model)
```

```
##
## Call:
## glm(formula = lm.init, family = "binomial", data = df_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.5710  -0.5553   0.3298   0.6342   1.8583
##
## Coefficients:
##                                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)                       0.87067    1.10488   0.788 0.430681
## account.balance2                  0.33642    0.29275   1.149 0.250479
## account.balance3                  1.68502    0.29668   5.680 1.35e-08 ***
## credit.duration.months           -0.51362    0.16191  -3.172 0.001513 **
## previous.credit.payment.status2   0.48080    0.41970   1.146 0.251968
## previous.credit.payment.status3   1.24952    0.45801   2.728 0.006369 **
## credit.purpose2                  -1.19390    0.53825  -2.218 0.026549 *
## credit.purpose3                  -1.56486    0.49801  -3.142 0.001677 **
## credit.purpose4                  -1.66292    0.48946  -3.397 0.000680 ***
## credit.amount                    -0.20804    0.18192  -1.144 0.252793
## savings2                          0.38695    0.38717   0.999 0.317583
## savings3                          1.16367    0.42529   2.736 0.006216 **
## savings4                          1.52042    0.40927   3.715 0.000203 ***
## employment.duration2              0.46089    0.32015   1.440 0.149984
## employment.duration3              0.78712    0.40065   1.965 0.049458 *
## employment.duration4             -0.13814    0.38284  -0.361 0.718217
## installment.rate2                -0.37538    0.43098  -0.871 0.383756
## installment.rate3                -0.82307    0.46961  -1.753 0.079661 .
## installment.rate4                -1.02120    0.41380  -2.468 0.013593 *
## marital.status3                   0.74691    0.28336   2.636 0.008390 **
## marital.status4                   0.34154    0.41634   0.820 0.412027
## guarantor2                        0.53480    0.38295   1.397 0.162563
## residence.duration2              -1.05137    0.40265  -2.611 0.009024 **
## residence.duration3              -0.54960    0.43489  -1.264 0.206310
## residence.duration4              -0.38488    0.40303  -0.955 0.339593
## current.assets2                  -0.25434    0.34680  -0.733 0.463309
## current.assets3                  -0.16482    0.32998  -0.499 0.617443
## current.assets4                  -1.18559    0.57234  -2.071 0.038314 *
## age                               0.34891    0.15182   2.298 0.021549 *
## other.credits2                    0.30067    0.29861   1.007 0.313989
## apartment.type2                   0.50156    0.32576   1.540 0.123641
## apartment.type3                   0.21613    0.64628   0.334 0.738058
## bank.credits2                    -0.46282    0.31218  -1.483 0.138194
## occupation2                       0.26013    0.93982   0.277 0.781940
## occupation3                       0.09446    0.91128   0.104 0.917441
```

```
## occupation4                        0.13534    0.96571   0.140 0.888543
## dependents2                       -0.57593    0.32421  -1.776 0.075668 .
## telephone2                         0.33770    0.26650   1.267 0.205097
## foreign.worker2                    0.99270    0.85558   1.160 0.245941
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 715.29  on 599  degrees of freedom
## Residual deviance: 495.06  on 561  degrees of freedom
## AIC: 573.06
##
## Number of Fisher Scoring iterations: 5
```

```
any(is.na(df))
```

```
## [1] FALSE
```

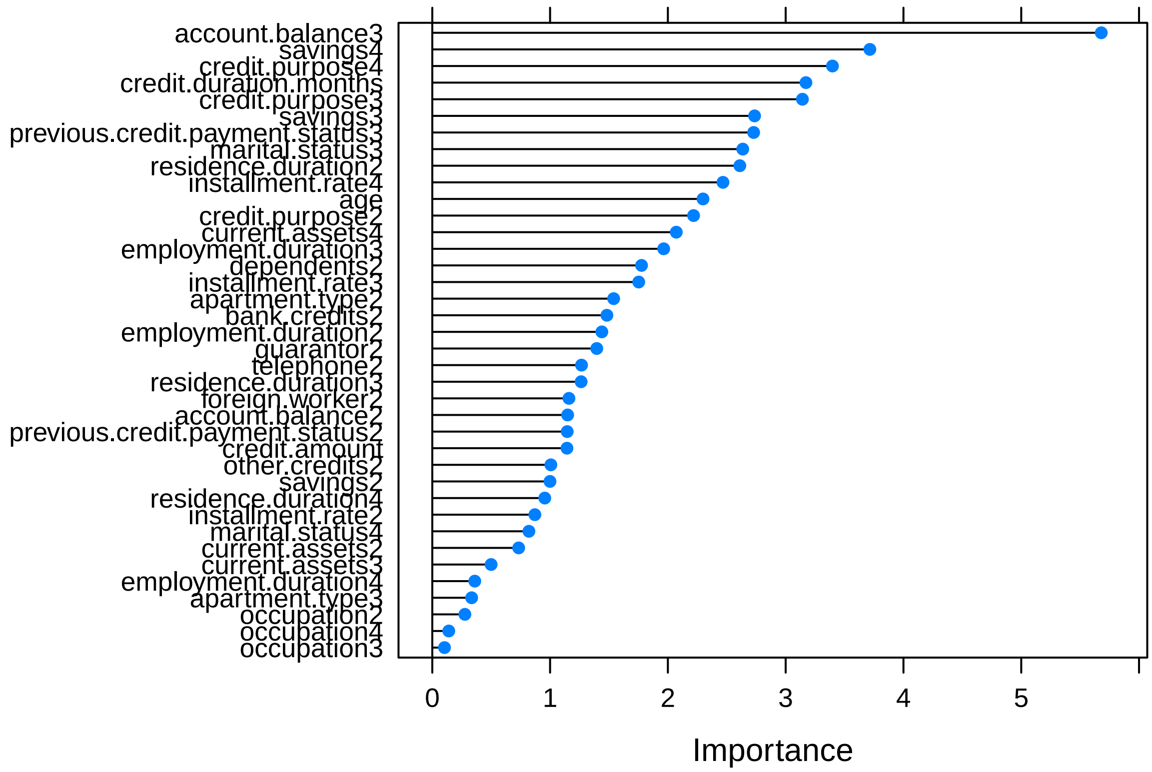
```
# Testing the Model in Test Data
lr.predictions <- predict(lr.model, df_test, type = "response")
lr.predictions <- round(lr.predictions)

# Evaluating the model
confusionMatrix(table(data = lr.predictions, reference = test.class.var), positive = '1')
```

```
## Confusion Matrix and Statistics
##
##     reference
## data   0   1
##    0  57  36
##    1  73 234
##
##                Accuracy : 0.7275
##                  95% CI : (0.681, 0.7706)
##     No Information Rate : 0.675
##     P-Value [Acc > NIR] : 0.0133836
##
##                   Kappa : 0.3294
##  Mcnemar's Test P-Value : 0.0005644
##
##             Sensitivity : 0.8667
##             Specificity : 0.4385
##          Pos Pred Value : 0.7622
##          Neg Pred Value : 0.6129
##              Prevalence : 0.6750
##          Detection Rate : 0.5850
##    Detection Prevalence : 0.7675
##       Balanced Accuracy : 0.6526
##
##        'Positive' Class : 1
##
```

Step 6 - Optimizing Model

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




```r
# Building the model with the selected variables
newFormula <- "credit.rating ~ account.balance + credit.purpose + previous.credit.payment.status + savi
newFormula <- as.formula(newFormula)
lrNewModel <- glm(formula = newFormula, data = df_train, family = "binomial")

# Viewing the template
summary(lrNewModel)
```

```
##
## Call:
## glm(formula = newFormula, family = "binomial", data = df_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
```

```
## -2.5657  -0.7436   0.4378   0.7457   1.9897
##
## Coefficients:
##                                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)                         0.1553     0.4844   0.321 0.748582
## account.balance2                    0.3158     0.2596   1.216 0.223810
## account.balance3                    1.5544     0.2685   5.788 7.11e-09 ***
## credit.purpose2                    -0.9148     0.4565  -2.004 0.045062 *
## credit.purpose3                    -1.0956     0.4162  -2.632 0.008479 **
## credit.purpose4                    -1.1549     0.4143  -2.788 0.005312 **
## previous.credit.payment.status2     0.7503     0.3503   2.142 0.032212 *
## previous.credit.payment.status3     1.2022     0.3664   3.281 0.001033 **
## savings2                            0.3161     0.3406   0.928 0.353377
## savings3                            0.9564     0.3861   2.477 0.013241 *
## savings4                            1.3025     0.3739   3.483 0.000495 ***
## credit.duration.months             -0.6511     0.1083  -6.009 1.87e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 715.29  on 599  degrees of freedom
## Residual deviance: 559.15  on 588  degrees of freedom
## AIC: 583.15
##
## Number of Fisher Scoring iterations: 5
```

```r
# Testing the Model in Test Data
lrNewPrediction <- predict(lrNewModel, df_test, type = "response")
lrNewPrediction <- round(lrNewPrediction)

# Evaluating the model
confusionMatrix(table(data = lrNewPrediction, reference = test.class.var), positive = '1')
```

```
## Confusion Matrix and Statistics
##
##     reference
## data   0   1
##    0  45  31
##    1  85 239
##
##                Accuracy : 0.71
##                  95% CI : (0.6628, 0.754)
##     No Information Rate : 0.675
##     P-Value [Acc > NIR] : 0.07375
##
##                   Kappa : 0.2593
##  Mcnemar's Test P-Value : 8.614e-07
##
##             Sensitivity : 0.8852
##             Specificity : 0.3462
##          Pos Pred Value : 0.7377
##          Neg Pred Value : 0.5921
##              Prevalence : 0.6750
```

```
##          Detection Rate : 0.5975
##    Detection Prevalence : 0.8100
##       Balanced Accuracy : 0.6157
##
##        'Positive' Class : 1
##
```

Step 7 - ROC Curve e Final Model Evaluation

```r
# Avaliando a performance do modelo
# Creating ROC Curves
lrModelBest <- lr.model
lrPredictionValue <- predict(lrModelBest, test.feature.var, type = "response")
predictions <- prediction(lrPredictionValue, test.class.var)
par(mfrow = c(1, 2))
plot.roc.curve(predictions, title.text = "ROC Curve")
plot.pr.curve(predictions, title.text = "Precision/Recall Curve")
```