Mini-Projeto 4

Mini-Project - Credit Risk Assessment

For this analysis, we will use a German Credit Data dataset, already properly cleaned and organized to create the predictive model.

The entire project will be described according to its stages.

Step 1 - Collecting the Data

Here is the data collection, in this case a csv file.

```
# collecting data
credit.df <- read.csv("credit_dataset.csv", header = TRUE, sep = ",")
```

Step 2 - Normalizing the Data

```
## Converting variables to factor type (categorical)
to.factors <- function(df, variables){
  for (variable in variables){
    df[[variable]] <- as.factor(df[[variable]])
  }
  return(df)
}

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

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

# Factor type variables
categorical.vars <- c('credit.rating', 'account.balance', 'previous.credi
t.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)
```

Step 3 - Splitting the data into training and test data

```
# Splitting data into training and testing - 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,]
```

Step 4 - Feature Selection

```
library(caret)

## Carregando pacotes exigidos: ggplot2

## Carregando pacotes exigidos: lattice

## Warning: package 'lattice' was built under R version 4.1.3

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

# Function for selecting variables
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)
}

# running the function
rfe.results <- run.feature.selection(feature.vars = train.data[,-1],
                                     class.var = train.data[,1])


# Viewing the 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   0.6984 0.0000   0.005442  0.0000
##          2   0.7351 0.2008   0.063992  0.2066
##          3   0.7485 0.2817   0.069221  0.2075
##          4   0.7383 0.3124   0.062454  0.1808
##          5   0.7550 0.3493   0.064307  0.1912
##          6   0.7485 0.3382   0.059665  0.1706
##          7   0.7318 0.2896   0.065640  0.1894
##          8   0.7435 0.3202   0.064377  0.1813
##          9   0.7434 0.3309   0.075491  0.2078
##         10   0.7451 0.3237   0.076808  0.2189
##         20   0.7601 0.3356   0.067445  0.1945        *
##
## The top 5 variables (out of 20):
##    account.balance, previous.credit.payment.status, credit.duration.mo
nths, credit.amount, current.assets

varImp((rfe.results))

##                                  Overall
## account.balance                18.0937553
## previous.credit.payment.status 10.3875749
## credit.duration.months          7.7379111
## credit.amount                   6.4655192
## current.assets                  5.1123341
## guarantor                       4.1569535
## age                             4.1248084
## apartment.type                  3.2220725
## savings                         2.9319061
## dependents                      2.0625061
## bank.credits                    1.9607506
## marital.status                  1.8168694
## employment.duration             1.7079459
## telephone                       1.5034421
## installment.rate                1.4751598
## credit.purpose                  1.2068991
## occupation                      1.1527123
## foreign.worker                  0.9225191
## other.credits                   0.8825651
## residence.duration              0.5905537
```

Step 5 - Creating and Evaluating the First Version of the Model

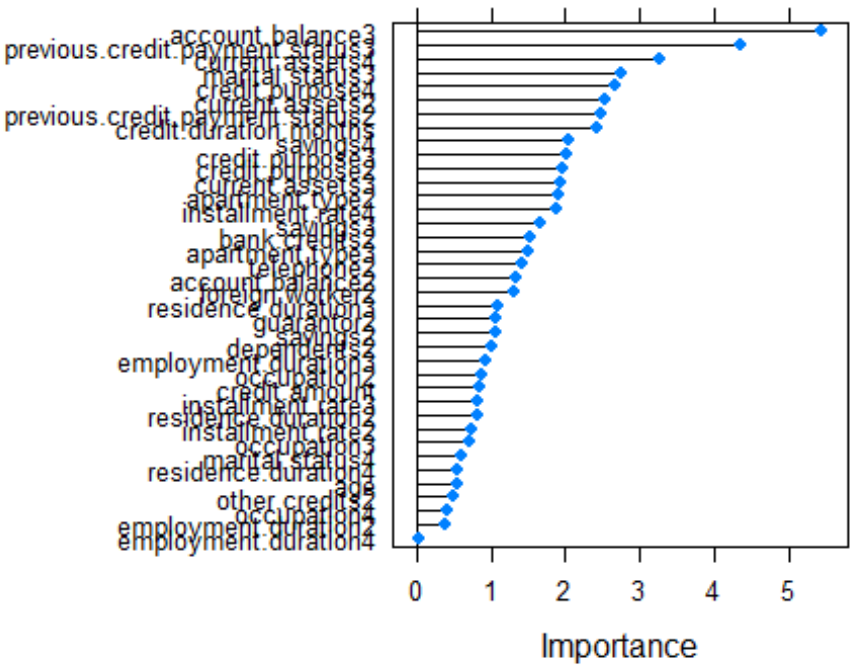
```
# Creating and Evaluating the Model
library(caret)
library(ROCR)

## Warning: package 'ROCR' was built under R version 4.1.3

# Utilities library for building graphics
source("plot_utils.R")

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

# Building a logistic regression model
formula.init <- "credit.rating ~ ."
formula.init <- as.formula(formula.init)
lr.model <- glm(formula = formula.init, data = train.data, family = "bino
mial")

# viewing the model
summary(lr.model)

##
## Call:
## glm(formula = formula.init, family = "binomial", data = train.data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4487  -0.6790   0.4081   0.7546   1.7636
##
## Coefficients:
##                                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)                      0.69964    1.06961   0.654  0.51304
## account.balance2                 0.36737    0.27421   1.340  0.18034
## account.balance3                 1.49674    0.27542   5.434 5.50e-08 *
**
## credit.duration.months          -0.36146    0.14932  -2.421  0.01549 *
## previous.credit.payment.status2  0.91038    0.36868   2.469  0.01354 *
## previous.credit.payment.status3  1.77895    0.40818   4.358 1.31e-05 *
**
## credit.purpose2                 -1.03709    0.52552  -1.973  0.04845 *
## credit.purpose3                 -1.03217    0.51120  -2.019  0.04347 *
## credit.purpose4                 -1.34301    0.50123  -2.679  0.00737 *
*
## credit.amount                   -0.14123    0.16648  -0.848  0.39625
## savings2                         0.40137    0.38009   1.056  0.29098
## savings3                         0.68605    0.41246   1.663  0.09625 .
## savings4                         0.67747    0.33171   2.042  0.04112 *
## employment.duration2            -0.12129    0.30991  -0.391  0.69553
```

```
## employment.duration3              0.33403    0.36271   0.921  0.35708
## employment.duration4             -0.01482    0.35753  -0.041  0.96694
## installment.rate2                -0.28567    0.38990  -0.733  0.46377
## installment.rate3                -0.35906    0.44165  -0.813  0.41622
## installment.rate4                -0.71899    0.38435  -1.871  0.06139 .
## marital.status3                   0.71510    0.25944   2.756  0.00585 *
*
## marital.status4                   0.23371    0.38897   0.601  0.54794
## guarantor2                        0.38385    0.36330   1.057  0.29071
## residence.duration2              -0.30198    0.37487  -0.806  0.42050
## residence.duration3               0.48078    0.44000   1.093  0.27453
## residence.duration4               0.20944    0.38062   0.550  0.58214
## current.assets2                  -0.82538    0.32562  -2.535  0.01125 *
## current.assets3                  -0.59569    0.30740  -1.938  0.05264 .
## current.assets4                  -1.68697    0.51415  -3.281  0.00103 *
*
## age                              -0.07572    0.13914  -0.544  0.58633
## other.credits2                    0.13140    0.27458   0.479  0.63226
## apartment.type2                   0.55804    0.29243   1.908  0.05635 .
## apartment.type3                   0.85607    0.56805   1.507  0.13181
## bank.credits2                    -0.44296    0.29116  -1.521  0.12817
## occupation2                      -0.74575    0.84402  -0.884  0.37693
## occupation3                      -0.58717    0.82008  -0.716  0.47399
## occupation4                      -0.36872    0.87277  -0.422  0.67268
## dependents2                      -0.31787    0.31280  -1.016  0.30952
## telephone2                        0.35780    0.25289   1.415  0.15711
## foreign.worker2                   0.88817    0.68356   1.299  0.19383
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 734.72  on 599  degrees of freedom
## Residual deviance: 558.31  on 561  degrees of freedom
## AIC: 636.31
##
## Number of Fisher Scoring iterations: 5

# Testing the model on test data
lr.predictions <- predict(lr.model, test.data, 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  46  20
##    1  73 261
##
##                Accuracy : 0.7675
##                  95% CI : (0.723, 0.808)
##     No Information Rate : 0.7025
##     P-Value [Acc > NIR] : 0.002209
##
##                   Kappa : 0.3618
##
##  Mcnemar's Test P-Value : 6.962e-08
##
##             Sensitivity : 0.9288
##             Specificity : 0.3866
##          Pos Pred Value : 0.7814
##          Neg Pred Value : 0.6970
##              Prevalence : 0.7025
##          Detection Rate : 0.6525
##    Detection Prevalence : 0.8350
##       Balanced Accuracy : 0.6577
##
##        'Positive' Class : 1
##
```

Step 6 - Optimizing the Model

```
## 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 = co
ntrol)
importance <- varImp(model, scale = FALSE)
plot(importance)
```



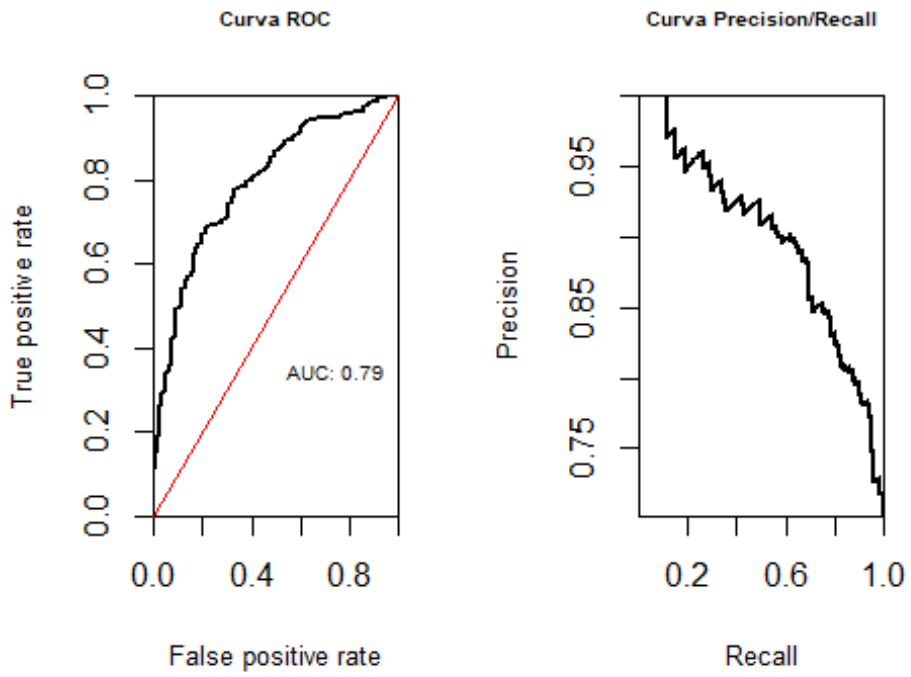

```
# Building the model with the selected variables
formula.new <- "credit.rating ~ account.balance + credit.purpose + previo
us.credit.payment.status + savings + credit.duration.months"
formula.new <- as.formula(formula.new)
lr.model.new <- glm(formula = formula.new, data = train.data, family = "b
inomial")

# viewing the modelo
summary(lr.model.new)

##
## Call:
## glm(formula = formula.new, family = "binomial", data = train.data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4893  -0.8283   0.4942   0.7945   1.8499
##
## Coefficients:
##                                Estimate Std. Error z value Pr(>|z|)
## (Intercept)                     0.01861    0.51764   0.036  0.97133
## account.balance2                0.42660    0.24986   1.707  0.08776 .
## account.balance3                1.48056    0.25505   5.805 6.44e-09 *
**
## credit.purpose2                -1.00400    0.48695  -2.062  0.03923 *
## credit.purpose3                -0.97157    0.46364  -2.096  0.03613 *
## credit.purpose4                -1.22462    0.46128  -2.655  0.00794 *
```

```
*
## previous.credit.payment.status2  0.96311    0.32276   2.984  0.00284 *
*
## previous.credit.payment.status3  1.62644    0.34994   4.648 3.36e-06 *
**
## savings2                          0.21862    0.34442   0.635  0.52560
## savings3                          0.58572    0.38501   1.521  0.12819
## savings4                          0.49425    0.30411   1.625  0.10412
## credit.duration.months           -0.42322    0.09855  -4.295 1.75e-05 *
**
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 734.72  on 599  degrees of freedom
## Residual deviance: 603.07  on 588  degrees of freedom
## AIC: 627.07
##
## Number of Fisher Scoring iterations: 4

# Testing the model on test data
lr.predictions.new <- predict(lr.model.new, test.data, type="response")
lr.predictions.new <- round(lr.predictions.new)

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

## Confusion Matrix and Statistics
##
##     reference
## data   0   1
##    0  28  13
##    1  91 268
##
##                Accuracy : 0.74
##                  95% CI : (0.6941, 0.7823)
##     No Information Rate : 0.7025
##     P-Value [Acc > NIR] : 0.05501
##
##                   Kappa : 0.2331
##
##  Mcnemar's Test P-Value : 4.337e-14
##
##             Sensitivity : 0.9537
##             Specificity : 0.2353
##          Pos Pred Value : 0.7465
##          Neg Pred Value : 0.6829
##              Prevalence : 0.7025
```

```
##          Detection Rate : 0.6700
##    Detection Prevalence : 0.8975
##       Balanced Accuracy : 0.5945
##
##        'Positive' Class : 1
##
```

Step 7 - ROC Curve and Final Model Assessment

```
# Evaluating the model

# Creating ROC curves
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")
```



The end