Credit Risk Assessment

Edinor Junior

Feb 08, 2021

Project - Credit Risk Assessment

For this analysis, we will use a set of German Credit Data, already clean and organized for the creation of the predictive model.

The entire project will be described according to its stages.

This project is a initial study about credit risk assessment, build with the intent to create a base to more elaborate works.

Please, feel free to contribute and share your knowledge

Step 1 - Collecting the Data

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

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

Step 2 - Normalizing the Data

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

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

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

# Variables of type factor
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)
```

Step 3 - Splitting the data into training and test data

```
# Dividing the 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)
```

```
## 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
```

```
# Function to select 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)
}

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


# Visualize 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   0.6803 0.1759    0.05789  0.1516
##          2   0.7003 0.1736    0.05688  0.1614
##          3   0.7605 0.3681    0.05432  0.1339
##          4   0.7335 0.3209    0.05707  0.1376
##          5   0.7387 0.3448    0.07133  0.1696
##          6   0.7538 0.3802    0.06189  0.1612
##          7   0.7568 0.3845    0.05099  0.1310
##          8   0.7618 0.3995    0.06070  0.1477
##          9   0.7685 0.4110    0.05122  0.1316        *
##         10   0.7685 0.4130    0.04569  0.1176
##         20   0.7586 0.3661    0.04489  0.1278
##
## The top 5 variables (out of 9):
##    account.balance, previous.credit.payment.status, credit.duration.months, savings, credit.amount
```

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

```
##                                 Overall
## account.balance                18.898750
## previous.credit.payment.status 11.412913
## credit.duration.months         10.073348
## savings                         6.094415
## credit.amount                   4.611833
## age                             4.111389
## credit.purpose                  3.968846
## current.assets                  3.928788
## guarantor                       3.821616
## employment.duration             3.429810
## telephone                       3.402840
## bank.credits                    3.338885
## marital.status                  3.247341
## apartment.type                  3.216897
## dependents                      3.104294
```

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

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

# Utility 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]
```

```r
# 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 = "binomial")

# Visualize model
summary(lr.model)
```

```
##
## Call:
## glm(formula = formula.init, family = "binomial", data = train.data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.5977  -0.6665   0.3928   0.7191   1.9538
##
## Coefficients:
##                                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)                      0.38691    1.08336   0.357  0.72099
## account.balance2                 0.40900    0.28149   1.453  0.14623
## account.balance3                 1.69833    0.28659   5.926 3.10e-09 ***
## credit.duration.months          -0.35799    0.14843  -2.412  0.01587 *
## previous.credit.payment.status2  1.13911    0.40724   2.797  0.00516 **
## previous.credit.payment.status3  1.81893    0.42428   4.287 1.81e-05 ***
## credit.purpose2                 -1.48901    0.53601  -2.778  0.00547 **
## credit.purpose3                 -1.66309    0.50955  -3.264  0.00110 **
## credit.purpose4                 -1.95352    0.49859  -3.918 8.93e-05 ***
## credit.amount                   -0.17946    0.18563  -0.967  0.33368
## savings2                         0.37287    0.36211   1.030  0.30315
## savings3                         0.64053    0.40674   1.575  0.11530
## savings4                         0.79931    0.32733   2.442  0.01461 *
## employment.duration2            -0.04887    0.30548  -0.160  0.87289
## employment.duration3             0.87377    0.38842   2.250  0.02448 *
## employment.duration4             0.05872    0.35302   0.166  0.86789
## installment.rate2                0.11389    0.40565   0.281  0.77890
## installment.rate3                0.08778    0.46224   0.190  0.84939
## installment.rate4               -0.31761    0.39497  -0.804  0.42133
## marital.status3                  0.51074    0.25080   2.036  0.04170 *
## marital.status4                 -0.02983    0.39727  -0.075  0.94015
## guarantor2                       0.32225    0.36966   0.872  0.38334
## residence.duration2             -0.64294    0.37587  -1.711  0.08717 .
## residence.duration3             -0.30515    0.42849  -0.712  0.47637
## residence.duration4             -0.25773    0.38354  -0.672  0.50159
## current.assets2                 -0.59982    0.32886  -1.824  0.06816 .
## current.assets3                 -0.52524    0.30962  -1.696  0.08981 .
## current.assets4                 -1.35473    0.52830  -2.564  0.01034 *
## age                              0.06200    0.13149   0.472  0.63728
## other.credits2                   0.09590    0.29801   0.322  0.74760
## apartment.type2                  0.82303    0.30809   2.671  0.00755 **
## apartment.type3                  1.13402    0.60587   1.872  0.06124 .
## bank.credits2                   -0.18648    0.28581  -0.652  0.51411
## occupation2                     -0.40857    0.83716  -0.488  0.62552
## occupation3                     -0.28031    0.81025  -0.346  0.72937
## occupation4                     -0.43171    0.85318  -0.506  0.61285
```

```
## dependents2                    -0.10122    0.31643  -0.320  0.74906
## telephone2                     -0.02847    0.25618  -0.111  0.91151
## foreign.worker2                 1.30620    0.83700   1.561  0.11862
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 741.31  on 599  degrees of freedom
## Residual deviance: 546.86  on 561  degrees of freedom
## AIC: 624.86
##
## Number of Fisher Scoring iterations: 5
```

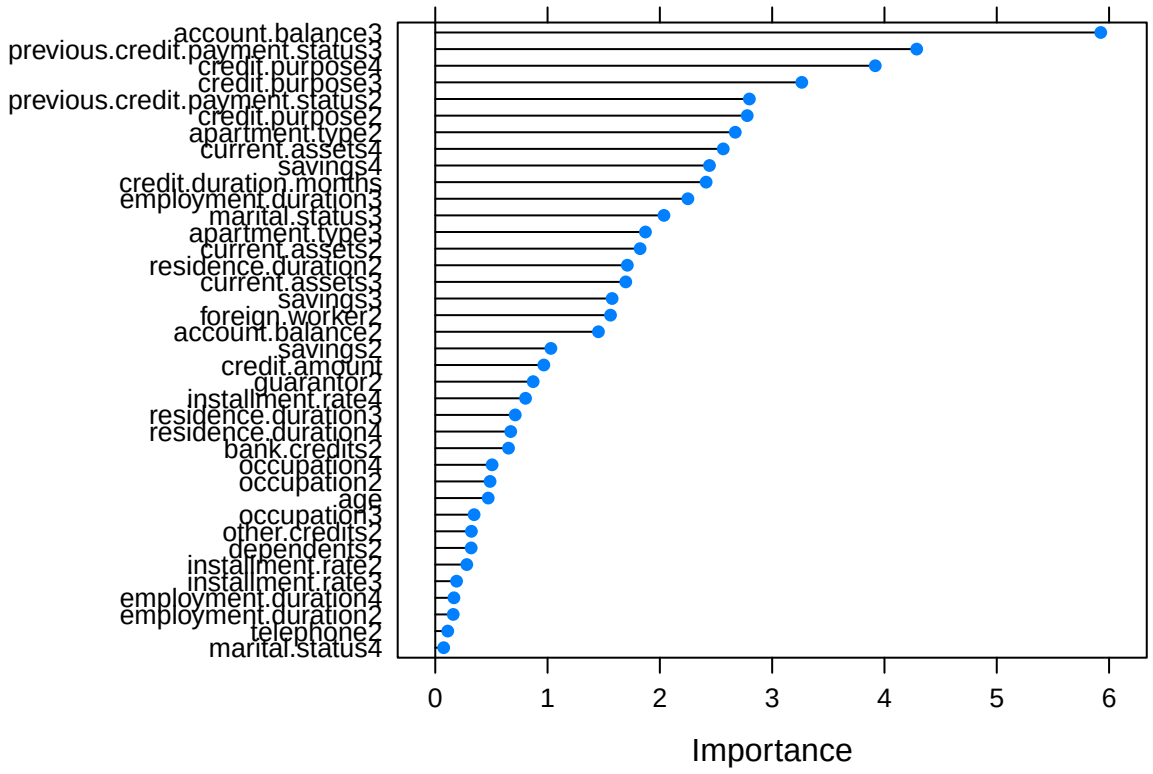
```
# Testing the model using training data
lr.predictions <- predict(lr.model, test.data, type="response")
lr.predictions <- round(lr.predictions)

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

```
## Confusion Matrix and Statistics
##
##     reference
## data   0   1
##    0  61  40
##    1  54 245
##
##                Accuracy : 0.765
##                  95% CI : (0.7203, 0.8057)
##     No Information Rate : 0.7125
##     P-Value [Acc > NIR] : 0.0107
##
##                   Kappa : 0.4048
##
##  Mcnemar's Test P-Value : 0.1800
##
##             Sensitivity : 0.8596
##             Specificity : 0.5304
##          Pos Pred Value : 0.8194
##          Neg Pred Value : 0.6040
##              Prevalence : 0.7125
##          Detection Rate : 0.6125
##    Detection Prevalence : 0.7475
##       Balanced Accuracy : 0.6950
##
##        'Positive' Class : 1
##
```

Step 6 - Optimizing the model

```
## Feature selection
formula <- "credit.rating ~ ."
formula <- as.formula(formula)
```

```r
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)
```




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

# Visualizing the model
summary(lr.model.new)
```

```
##
## Call:
## glm(formula = formula.new, family = "binomial", data = train.data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.5109  -0.8304   0.4689   0.7740   1.9701
##
## Coefficients:
##                                Estimate Std. Error z value Pr(>|z|)
## (Intercept)                    -0.06247    0.52566  -0.119 0.905405
## account.balance2                0.35502    0.25342   1.401 0.161235
## account.balance3                1.58030    0.26314   6.006 1.91e-09 ***
```

```
## credit.purpose2                -1.21998    0.48859  -2.497 0.012528 *
## credit.purpose3                -1.23501    0.45301  -2.726 0.006406 **
## credit.purpose4                -1.60081    0.45527  -3.516 0.000438 ***
## previous.credit.payment.status2  1.16634    0.35824   3.256 0.001131 **
## previous.credit.payment.status3  1.74197    0.37840   4.604 4.15e-06 ***
## savings2                         0.18933    0.33064   0.573 0.566916
## savings3                         0.63402    0.37985   1.669 0.095091 .
## savings4                         0.71854    0.30538   2.353 0.018625 *
## credit.duration.months          -0.49461    0.10221  -4.839 1.30e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 741.31  on 599  degrees of freedom
## Residual deviance: 592.01  on 588  degrees of freedom
## AIC: 616.01
##
## Number of Fisher Scoring iterations: 5
```

```
# Testing the model with the 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  51  33
##    1  64 252
##
##                Accuracy : 0.7575
##                  95% CI : (0.7124, 0.7987)
##     No Information Rate : 0.7125
##     P-Value [Acc > NIR] : 0.025175
##
##                   Kappa : 0.3563
##
##  Mcnemar's Test P-Value : 0.002319
##
##             Sensitivity : 0.8842
##             Specificity : 0.4435
##          Pos Pred Value : 0.7975
##          Neg Pred Value : 0.6071
##              Prevalence : 0.7125
##          Detection Rate : 0.6300
##    Detection Prevalence : 0.7900
##       Balanced Accuracy : 0.6638
##
##        'Positive' Class : 1
##
```

Step 7 - ROC Curve and Final Evaluating the model

```
# Assessing model performance

# 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 = "ROC Curve")
plot.pr.curve(predictions, title.text = "Precision/Recall Curve")
```