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.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

```
# 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   0.6847 0.2215    0.05819  0.1427
##          2   0.7167 0.1883    0.06720  0.2195
##          3   0.7324 0.2895    0.08760  0.2363
##          4   0.7285 0.3283    0.08879  0.2333
##          5   0.7416 0.3601    0.07273  0.1815
##          6   0.7534 0.3735    0.07571  0.2030
##          7   0.7551 0.3774    0.08722  0.2348
##          8   0.7618 0.3914    0.07957  0.2123
##          9   0.7635 0.3969    0.08173  0.2167
##         10   0.7716 0.4110    0.06909  0.1981        *
##         20   0.7584 0.3539    0.07375  0.2135
##
## The top 5 variables (out of 10):
##    account.balance, previous.credit.payment.status,
credit.duration.months, credit.amount, savings

varImp((rfe.results))

##                                  Overall
## account.balance                18.222013
## previous.credit.payment.status 11.930982
## credit.duration.months         10.022683
## credit.amount                   7.760819
## savings                         5.490977
## current.assets                  5.436902
## credit.purpose                  5.310719
## other.credits                   3.935792
## age                             3.572105
## apartment.type                  3.554416
## employment.duration             3.155205
## bank.credits                    2.943141
## marital.status                  2.921242
```

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]
```

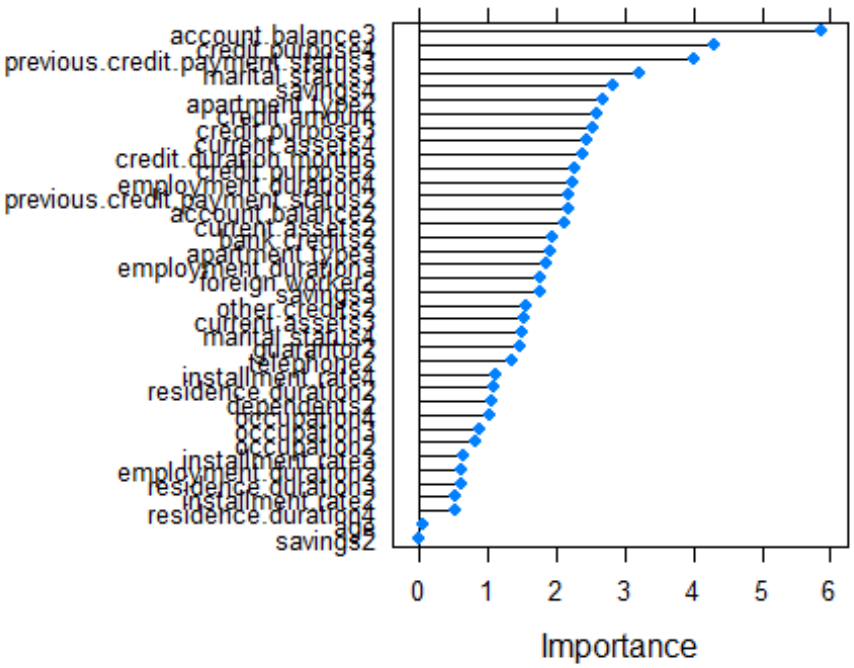
```r
# Building a logistic regression model
formula.init <- "credit.rating ~ ."
formula.init <- as.formula(formula.init)
lr.model <- glm(formula = formula.init, data = train.data, family =
"binomial")

# viewing the model
summary(lr.model)

##
## Call:
## glm(formula = formula.init, family = "binomial", data = train.data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6887  -0.6380   0.3482   0.6874   2.5185
##
## Coefficients:
##                                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)                     0.121763   1.195105   0.102  0.91885
## account.balance2                0.624048   0.286989   2.174  0.02967
*
## account.balance3                1.670212   0.285144   5.857 4.70e-09
***
## credit.duration.months         -0.343159   0.144754  -2.371  0.01776
*
## previous.credit.payment.status2  0.911850   0.416496   2.189  0.02857
*
## previous.credit.payment.status3  1.778319   0.442217   4.021 5.79e-05
***
## credit.purpose2                -1.195601   0.525391  -2.276  0.02287
*
## credit.purpose3                -1.277818   0.505239  -2.529  0.01143
*
## credit.purpose4                -2.083654   0.485218  -4.294 1.75e-05
***
## credit.amount                  -0.434797   0.168260  -2.584  0.00976
**
## savings2                       -0.001062   0.367872  -0.003  0.99770
## savings3                        0.740577   0.420287   1.762  0.07806
.
## savings4                        0.989707   0.349688   2.830  0.00465
**
## employment.duration2            0.190444   0.313712   0.607  0.54381
## employment.duration3            0.678519   0.365358   1.857  0.06329
.
## employment.duration4            0.827402   0.368668   2.244  0.02481
*
## installment.rate2               0.210157   0.394341   0.533  0.59408
```

```
## installment.rate3               -0.275079   0.429822  -0.640  0.52218
## installment.rate4               -0.414887   0.377487  -1.099  0.27173
## marital.status3                  0.870208   0.271980   3.200  0.00138
**
## marital.status4                  0.611462   0.408731   1.496  0.13465
## guarantor2                       0.581510   0.397459   1.463  0.14345
## residence.duration2             -0.427651   0.394686  -1.084  0.27858
## residence.duration3             -0.265394   0.438478  -0.605  0.54501
## residence.duration4             -0.206489   0.401642  -0.514  0.60717
## current.assets2                 -0.700530   0.330500  -2.120  0.03404
*
## current.assets3                 -0.482351   0.313871  -1.537  0.12435
## current.assets4                 -1.289950   0.528632  -2.440  0.01468
*
## age                             -0.005375   0.134483  -0.040  0.96812
## other.credits2                   0.445426   0.284114   1.568  0.11693
## apartment.type2                  0.816433   0.306158   2.667  0.00766
**
## apartment.type3                  1.170584   0.609177   1.922  0.05466
.
## bank.credits2                   -0.575229   0.297673  -1.932  0.05331
.
## occupation2                     -0.800721   0.990706  -0.808  0.41896
## occupation3                     -0.861040   0.973074  -0.885  0.37623
## occupation4                     -1.012575   0.997102  -1.016  0.30986
## dependents2                     -0.357981   0.339390  -1.055  0.29153
## telephone2                       0.361865   0.267633   1.352  0.17635
## foreign.worker2                  1.555164   0.880255   1.767  0.07728
.
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 742.92  on 599  degrees of freedom
## Residual deviance: 513.85  on 561  degrees of freedom
## AIC: 591.85
##
## 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  50  48
##    1  64 238
##
##                Accuracy : 0.72
##                  95% CI : (0.6732, 0.7635)
##     No Information Rate : 0.715
##     P-Value [Acc > NIR] : 0.4371
##
##                   Kappa : 0.2827
##
##  Mcnemar's Test P-Value : 0.1564
##
##             Sensitivity : 0.8322
##             Specificity : 0.4386
##          Pos Pred Value : 0.7881
##          Neg Pred Value : 0.5102
##              Prevalence : 0.7150
##          Detection Rate : 0.5950
##    Detection Prevalence : 0.7550
##       Balanced Accuracy : 0.6354
##
##        '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 =
control)
importance <- varImp(model, scale = FALSE)
plot(importance)
```




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

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

##
## Call:
## glm(formula = formula.new, family = "binomial", data = train.data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6458  -0.7972   0.4553   0.7394   2.2638
##
## Coefficients:
##                                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)                     -0.45198    0.50119  -0.902 0.367156
## account.balance2                 0.54383    0.25609   2.124 0.033706 *
## account.balance3                 1.62366    0.25992   6.247 4.19e-10
***
## credit.purpose2                 -0.82771    0.44394  -1.864 0.062255 .
## credit.purpose3                 -0.58878    0.41427  -1.421 0.155242
## credit.purpose4                 -1.44974    0.40885  -3.546 0.000391
```

```
***
## previous.credit.payment.status2  1.13542    0.35752   3.176 0.001494
**
## previous.credit.payment.status3  1.74343    0.37748   4.619 3.86e-06
***
## savings2                         0.02335    0.33260   0.070 0.944022
## savings3                         0.70701    0.38472   1.838 0.066104 .
## savings4                         0.89315    0.31033   2.878 0.004001
**
## credit.duration.months          -0.51066    0.10299  -4.959 7.10e-07
***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 742.92  on 599  degrees of freedom
## Residual deviance: 577.08  on 588  degrees of freedom
## AIC: 601.08
##
## Number of Fisher Scoring iterations: 5
```

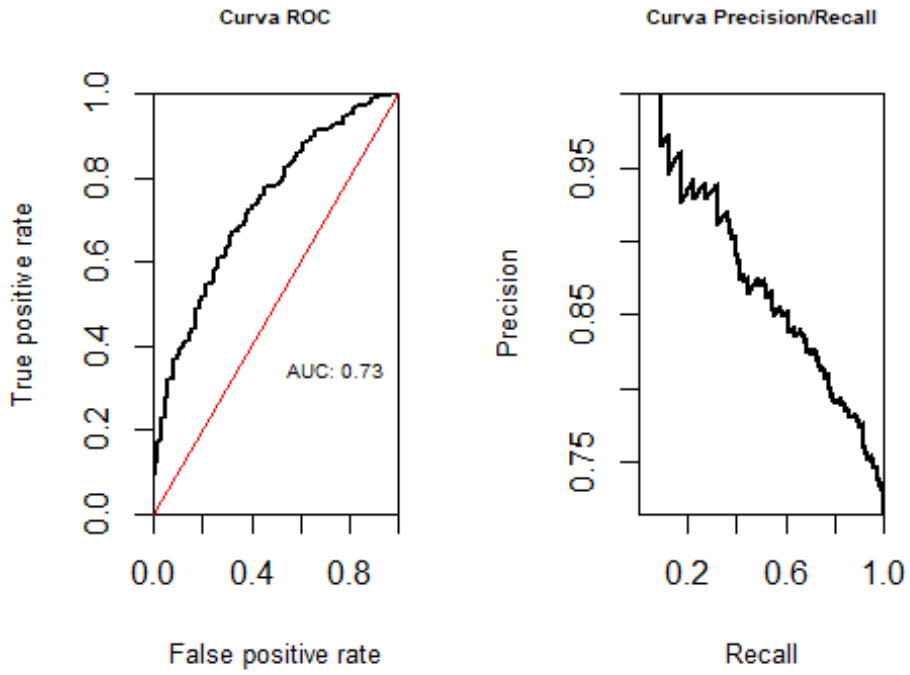
```
# Testing the model on test data
lr.predictions.new <- predict(lr.model.new, test.data, type="response")
lr.predictions.new <- round(lr.predictions.new)

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

```
## Confusion Matrix and Statistics
##
##     reference
## data   0   1
##    0  45  35
##    1  69 251
##
##                Accuracy : 0.74
##                  95% CI : (0.6941, 0.7823)
##     No Information Rate : 0.715
##     P-Value [Acc > NIR] : 0.146116
##
##                   Kappa : 0.2992
##
##  Mcnemar's Test P-Value : 0.001213
##
##             Sensitivity : 0.8776
##             Specificity : 0.3947
##          Pos Pred Value : 0.7844
##          Neg Pred Value : 0.5625
```

```
##              Prevalence : 0.7150
##          Detection Rate : 0.6275
##    Detection Prevalence : 0.8000
##       Balanced Accuracy : 0.6362
##
##        '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