Projeto - Analise de Risco de Credito

Lauro

09/11/2017

Projeto 4 - Avaliacao de Risco de Credito

Para esta analise, vamos usar um conjunto de dados German Credit Data, ja devidamente limpo e organizado para a criacao do modelo preditivo.

Todo o projeto sera descrito de acordo com suas etapas. Os acentos foram ignorados para evitar erros de interpretacao de caracteres por diferentes sistemas operacionais.

Etapa 1 - Coletando os Dados

Aqui esta a coleta de dados. Neste caso, um arquivo csv.

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

Etapa 2 - Normalizando os Dados

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

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

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

# Variaveis 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)
```

```
## Warning: package 'caret' was built under R version 3.4.2
```

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

```
## Warning: package 'lattice' was built under R version 3.4.2
```

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

```
## Warning: package 'ggplot2' was built under R version 3.4.2
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.4.2
```

```
## 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
```

```
# Funcao para selecao de variaveis
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 funcao
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   0.7049 0.2822    0.05863  0.1483
##          2   0.7163 0.2290    0.04780  0.1509
##          3   0.7435 0.3416    0.05975  0.1553
##          4   0.7632 0.4215    0.06227  0.1510
##          5   0.7380 0.3547    0.07486  0.1826
##          6   0.7513 0.3914    0.07820  0.1992
##          7   0.7432 0.3573    0.05848  0.1501
##          8   0.7617 0.4064    0.05340  0.1356
##          9   0.7703 0.4320    0.05679  0.1266
##         10   0.7723 0.4379    0.05676  0.1347
##         20   0.7800 0.4402    0.06732  0.1679        *
##
## The top 5 variables (out of 20):
##    account.balance, previous.credit.payment.status, credit.duration.months, savings, current.assets
```

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

```
##                                  Overall
## account.balance                22.5549721
## previous.credit.payment.status 11.7499934
## credit.duration.months         11.1216456
## savings                         8.8183078
## current.assets                  4.5878659
## age                             4.3589983
## credit.amount                   3.5524291
## dependents                      3.1710775
## occupation                      2.8503281
## credit.purpose                  2.8339707
## residence.duration              2.8179951
## employment.duration             2.7318366
## apartment.type                  2.5377835
## guarantor                       2.3326970
## other.credits                   1.9306104
## bank.credits                    1.8347217
## telephone                       1.5729445
## marital.status                  1.0425255
## foreign.worker                  1.0221089
## installment.rate                0.6717579
```

Etapa 5 - Criando e Avaliando a Primeira Versao do Modelo

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

```
## Warning: package 'ROCR' was built under R version 3.4.2
```

```
## Loading required package: gplots

## Warning: package 'gplots' was built under R version 3.4.2

##
## Attaching package: 'gplots'

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

```
# Biblioteca de utilitarios para construcao de graficos
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 regressao logistica
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
## -3.0100  -0.5627   0.2973   0.6327   2.3446
##
## Coefficients:
##                                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)                     -0.22297    1.04265  -0.214 0.830667
## account.balance2                 0.74184    0.30223   2.455 0.014108 *
## account.balance3                 1.87157    0.30354   6.166 7.01e-10 ***
## credit.duration.months          -0.41344    0.15141  -2.731 0.006324 **
## previous.credit.payment.status2  0.97509    0.39731   2.454 0.014119 *
## previous.credit.payment.status3  1.80071    0.43136   4.174 2.99e-05 ***
## credit.purpose2                 -0.99589    0.55332  -1.800 0.071887 .
## credit.purpose3                 -1.11012    0.52796  -2.103 0.035497 *
## credit.purpose4                 -1.87622    0.51758  -3.625 0.000289 ***
## credit.amount                   -0.27472    0.17120  -1.605 0.108560
## savings2                         0.44917    0.39112   1.148 0.250793
## savings3                         1.70683    0.52680   3.240 0.001195 **
## savings4                         1.22704    0.35662   3.441 0.000580 ***
## employment.duration2             0.83402    0.31862   2.618 0.008855 **
## employment.duration3             1.45232    0.39593   3.668 0.000244 ***
## employment.duration4             0.29971    0.38479   0.779 0.436056
## installment.rate2                0.23406    0.39600   0.591 0.554478
## installment.rate3               -0.07992    0.45992  -0.174 0.862046
## installment.rate4               -0.58353    0.38795  -1.504 0.132546
## marital.status3                  0.26731    0.27581   0.969 0.332454
```

```
## marital.status4                -0.71499    0.43638  -1.638 0.101324
## guarantor2                     -0.11877    0.39501  -0.301 0.763670
## residence.duration2            -1.32156    0.39866  -3.315 0.000916 ***
## residence.duration3            -0.89472    0.42986  -2.081 0.037396 *
## residence.duration4            -1.09997    0.40209  -2.736 0.006226 **
## current.assets2                -0.45878    0.34814  -1.318 0.187573
## current.assets3                -0.79813    0.33473  -2.384 0.017107 *
## current.assets4                -1.40589    0.55578  -2.530 0.011419 *
## age                             0.31725    0.15631   2.030 0.042392 *
## other.credits2                  0.67903    0.28162   2.411 0.015903 *
## apartment.type2                 0.33128    0.32165   1.030 0.303040
## apartment.type3                 0.58330    0.62693   0.930 0.352165
## bank.credits2                  -0.18196    0.31125  -0.585 0.558818
## occupation2                    -0.11917    0.79496  -0.150 0.880837
## occupation3                     0.32332    0.76696   0.422 0.673343
## occupation4                     0.54298    0.82151   0.661 0.508640
## dependents2                    -0.25040    0.33112  -0.756 0.449529
## telephone2                      0.21884    0.27131   0.807 0.419885
## foreign.worker2                 1.64299    0.92248   1.781 0.074903 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 749.20  on 599  degrees of freedom
## Residual deviance: 494.21  on 561  degrees of freedom
## AIC: 572.21
##
## Number of Fisher Scoring iterations: 5
```

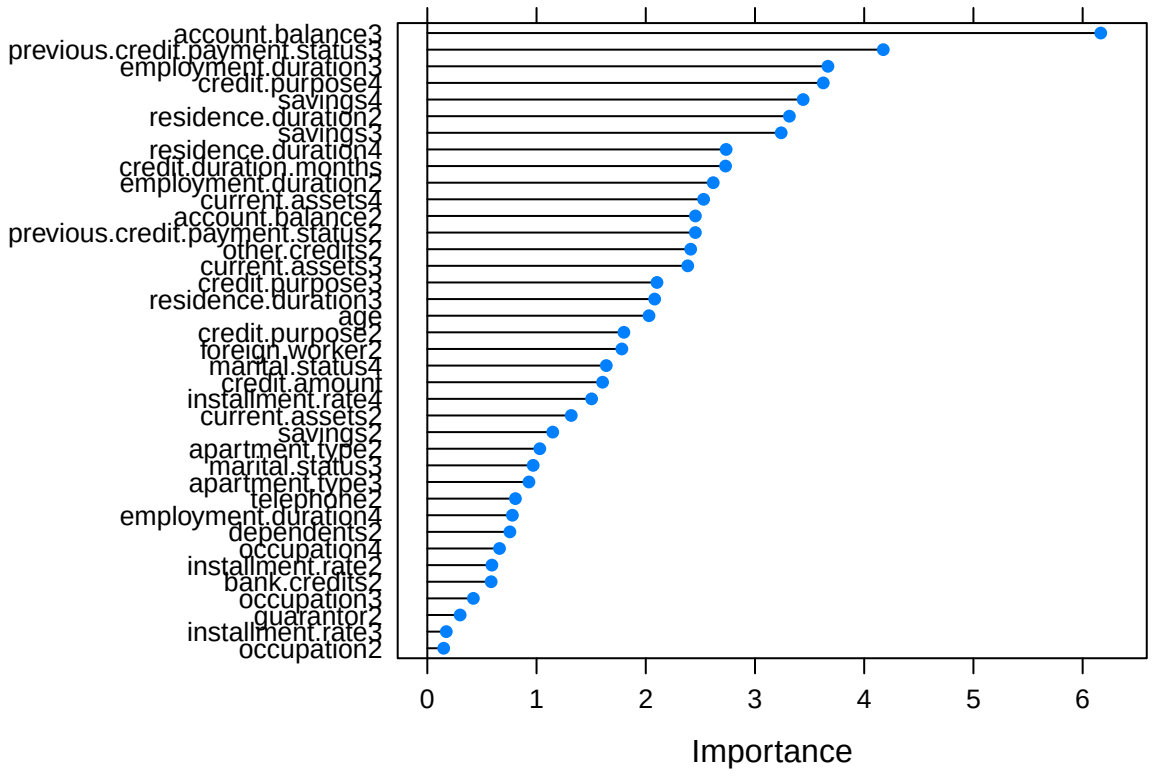
```r
# Testando o modelo nos dados de teste
lr.predictions <- predict(lr.model, test.data, type="response")
lr.predictions <- round(lr.predictions)

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

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0  51  62
##          1  59 228
##
##                Accuracy : 0.6975
##                  95% CI : (0.6499, 0.7422)
##     No Information Rate : 0.725
##     P-Value [Acc > NIR] : 0.9002
##
##                   Kappa : 0.2477
##  Mcnemar's Test P-Value : 0.8557
##
##             Sensitivity : 0.7862
##             Specificity : 0.4636
##          Pos Pred Value : 0.7944
```

```
##          Neg Pred Value : 0.4513
##              Prevalence : 0.7250
##          Detection Rate : 0.5700
##    Detection Prevalence : 0.7175
##       Balanced Accuracy : 0.6249
##
##        '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 variaveis selecionadas
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")

# Visualizando o modelo
```

```r
summary(lr.model.new)
```

```
##
## Call:
## glm(formula = formula.new, family = "binomial", data = train.data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6620  -0.7554   0.4231   0.7308   2.2768
##
## Coefficients:
##                                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)                      -0.4190     0.5584  -0.750  0.45309
## account.balance2                  0.4753     0.2592   1.834  0.06667 .
## account.balance3                  1.7137     0.2663   6.434 1.24e-10 ***
## credit.purpose2                  -0.9066     0.4989  -1.817  0.06915 .
## credit.purpose3                  -0.9463     0.4716  -2.006  0.04481 *
## credit.purpose4                  -1.4751     0.4708  -3.133  0.00173 **
## previous.credit.payment.status2   1.1178     0.3537   3.160  0.00158 **
## previous.credit.payment.status3   1.7869     0.3755   4.759 1.95e-06 ***
## savings2                          0.1977     0.3419   0.578  0.56315
## savings3                          1.3100     0.4570   2.866  0.00415 **
## savings4                          0.9536     0.3140   3.038  0.00239 **
## credit.duration.months           -0.5754     0.1086  -5.298 1.17e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 749.20  on 599  degrees of freedom
## Residual deviance: 565.43  on 588  degrees of freedom
## AIC: 589.43
##
## Number of Fisher Scoring iterations: 5
```

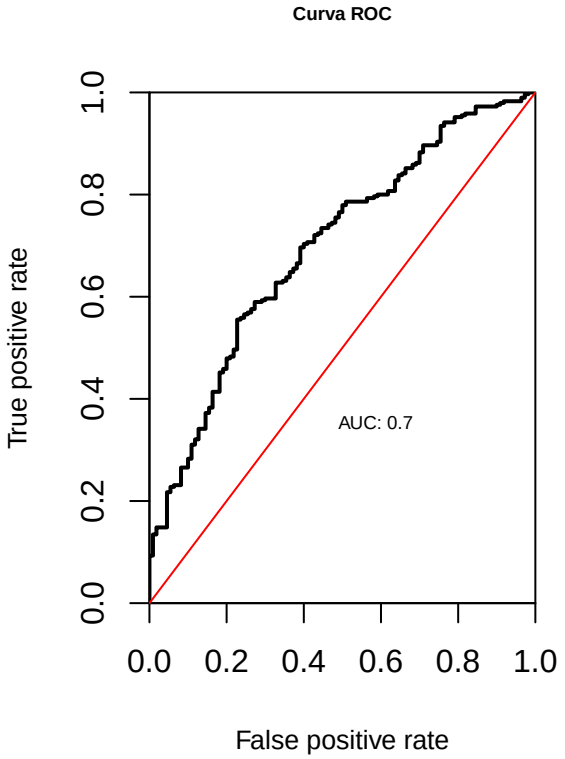
```r
# Testando o modelo nos dados de teste
lr.predictions.new <- predict(lr.model.new, test.data, type="response")
lr.predictions.new <- round(lr.predictions.new)

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

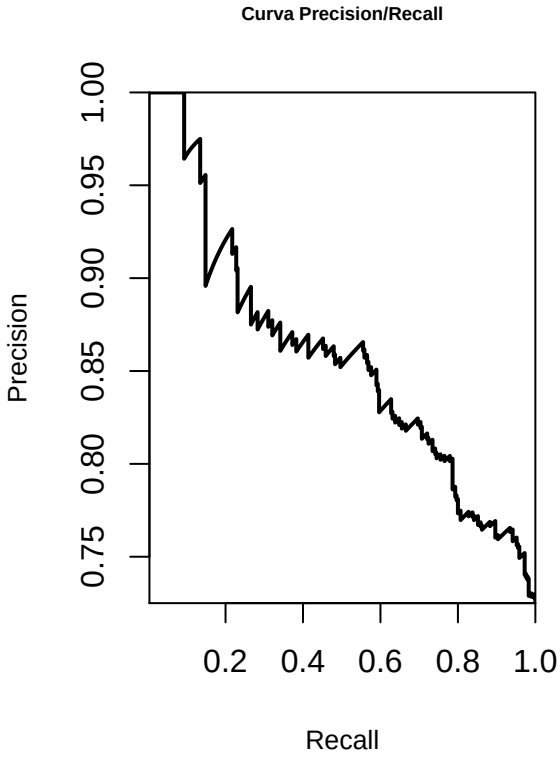
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0  44  49
##          1  66 241
##
##                Accuracy : 0.7125
##                  95% CI : (0.6654, 0.7564)
##     No Information Rate : 0.725
##     P-Value [Acc > NIR] : 0.7327
##
```

```
##                  Kappa : 0.2427
##  Mcnemar's Test P-Value : 0.1357
##
##             Sensitivity : 0.8310
##             Specificity : 0.4000
##          Pos Pred Value : 0.7850
##          Neg Pred Value : 0.4731
##              Prevalence : 0.7250
##          Detection Rate : 0.6025
##    Detection Prevalence : 0.7675
##       Balanced Accuracy : 0.6155
##
##        'Positive' Class : 1
##
```

Etapa 7 - Curva ROC e Avaliacao 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
False positive rate
AUC: 0.7
0.0
0.2
0.4
0.6
0.8
1.0


Curva Precision/Recall
Precision
Recall
0.75
0.80
0.85
0.90
0.95
1.00
0.2
0.4
0.6
0.8
1.0