Facultatea de Ştiinţe Economice Şi Gestiunea Afacerilor

Universitatea Babeș-Bolyai,Cluj-Napoca

Specializare:Informatică Economică

Tehnologii No-SQL

Detectarea și prezicerea riscului de credit

Prof.univ.dr. **Gheorghe Cosmin SILAGHI**

Studenti:

Lorincz Marius-Laurențiu

Pitiriciu Iulian

• Introducere

În teorie un risc de credit este riscul de neplată al unei datorii care poate apărea de la un împrumutat care nu efectuează plățile necesare. În primul rând, riscul este cel al creditorului și include capitalul pierdut și dobânda, întreruperea fluxurilor de numerar și costuri de colectare crescute. Pierderea poate fi completă sau parțială. Pe o piață eficientă, niveluri mai ridicate de risc de credit vor fi asociate cu costuri mai mari de împrumut. Din această cauză, măsurile de costuri ale împrumutului, cum ar fi diferențele de randament, pot fi utilizate pentru a deduce nivelurile de risc de credit bazate pe evaluările participanților pe piață. Totodată, problema riscului de credit este asociată domeniului financiar.

În cazul nostru o să avem de aface cu o bancă care se comportă ca o organizație financiară, acordând credite pentru clienții care aplică pentru ele. Clienții care primesc un credit pot deveni un risc de credit. Prin analiza datelor clienților și prin aplicarea algoritmilor machine-learning, banca va fi capabilă să prezică clienții care vor devein un risc de credit. Acest lucru va duce la atenuarea riscurilor și la minimizarea pierderilor prin neacordarea de credite clienților care pot să fie un risc de credit pentru bancă.

Cele 2 întrebări sunt :

1. Putem optimiza datele dintr-un set prin analiza descriptivă a datelor și prin aplicarea algoritmilor machine-learning?
2.

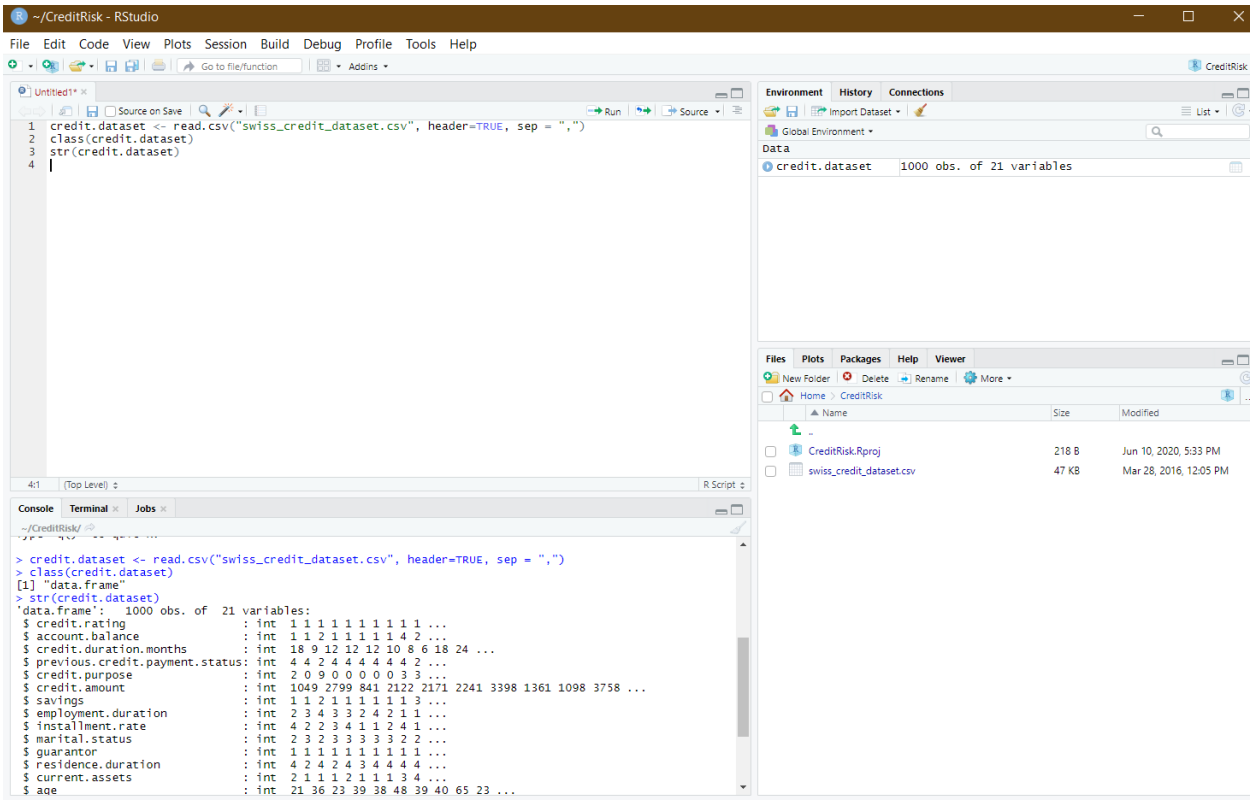
- Setul de date

Setul de date cuprinde datele financiare ale unei bănci din Elveția.

Load-ul datelor a fost realizat prin comanda credit.dataset <-read.csv("swiss_credit_dataset.csv", header=TRUE, sep = ",")

Prin metoda class(credit.dataset) ne putem da seama că tipul clasei este data.frame, acest lucru indică faptul că obiectul nostru credit.dataset este o colecție de vectori cu lungimi identice. Fiecare vector reprezintă o coloană și fiecare vector poate să aibe un tip de date diferit(de exemplu character, integer, etc.

Pentru a inspecta datele am folosit metoda str(credit.dataset). Această funcție ne va ajuta să aflăm mai multe informații despre datele pe care urmează să le analizăm / manipulăm. Ne sunt oferite următoarele informații : numărul de înregistrări, în cazul nostru avem 1000 de înregistrări în tabelă, numărul de atribute(coloane), în cazul nostru avem 21 de atribute, informații despre atributele noastre, precum numele acestora, tipul acestora și câteva exemple de înregistrări pentru fiecare atribut.



În imaginea de mai sus putem observa faptul că atributul credit.rating este de tipul integer și are ca și exemple de înregistrări cifra 1.

Folosind aceste metode ne putem forma o idee despre atribute și tipul de date ale acestora, astfel putem să ne dăm seam ace fel de transformări putem să facem asupra lor și ce metode statistice putem să folosim în timpul descrierii analitice.

- Preprocesarea datelor(curățarea datelor, transformarea datelor, normalizarea datelor)

Primul lucru pe care o să-l facem este să verificăm dacă data frame-ul nostrum conține date NA(date care lipsesc). Acest lucru o să-l facem prin metoda sum(is.na(credit.dataset)), is.na va verifica dacă data frame-ul conține date NA.

```
> sum(is.na(credit.dataset))
[1] 0
>
```

Output-ul ne arată că nu avem date NA în data frame-ul nostru.

În urma executării metodei str(credit.dataset) am putut să observăm că toate datele sunt declarate ca și integer by default. În statistică avem de aface cu două tipuri de variabile:

- variabile numerice(variabilele numerice sunt variabile ale căror valori au o semnificație matematică, adică pot să fie adunate, înmulțite ș.a.m.d, de exemplu vârsta unei personae, greutatea etc.)
- variabile categorice(variabilele categorice sunt variabile ale căror valori nu au o semnificație matematică, adică nu putem să efectuăm nicio operație matematică asupra acestora)

În setul nostru de date putem observa că avem trei attribute care au tipul de date int însă se pot încadra ca și variabilă numerică. Aceste trei atribute sunt:

- credit.duration.months
- credit.amount
- age

Restul de optsprezece aparțin tipul categoric, deci va trebui să ne folosim de o metodă ca să le convertim din tipul int în tipul categoric. Metoda pe care o să o folosim este următoarea:

```
# metodă pentru transformarea datelor
to.factors <- function(dataset, variables){
  for (variable in variables){
    dataset[[variable]] <- as.factor(dataset[[variable]])
  }
  return(dataset)
}
```

După vom selecta atributele pe care vrem să le transformăm

```
# selectarea datelor pe care vrem să le transformăm
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')
```

După vom transforma datele, folosind metoda pe care am definit-o acum doi pași

```
# transformarea datelor
credit.dataset <- to.factors(dataset=credit.dataset,variables = categorical.vars)
```

Putem observa că cele 18 atribute și-au schimbat tipul

```
> str(credit.dataset)
'data.frame':   1000 obs. of  21 variables:
 $ credit.rating                : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
 $ account.balance              : Factor w/ 4 levels "1","2","3","4": 1 1 2 1 1 1 1 1 4 2 ...
 $ credit.duration.months       : int  18 9 12 12 12 10 8 6 18 24 ...
 $ previous.credit.payment.status: Factor w/ 5 levels "0","1","2","3",..: 5 5 3 5 5 5 5 5 5 3 ...
 $ credit.purpose               : Factor w/ 10 levels "0","1","2","3",..: 3 1 9 1 1 1 1 1 4 4 ...
 $ credit.amount                : int  1049 2799 841 2122 2171 2241 3398 1361 1098 3758 ...
 $ savings                      : Factor w/ 5 levels "1","2","3","4",..: 1 1 2 1 1 1 1 1 1 3 ...
 $ employment.duration          : Factor w/ 5 levels "1","2","3","4",..: 2 3 4 3 3 2 4 2 1 1 ...
 $ installment.rate             : Factor w/ 4 levels "1","2","3","4": 4 2 2 3 4 1 1 2 4 1 ...
 $ marital.status               : Factor w/ 4 levels "1","2","3","4": 2 3 2 3 3 3 3 3 2 2 ...
 $ guarantor                    : Factor w/ 3 levels "1","2","3": 1 1 1 1 1 1 1 1 1 1 ...
 $ residence.duration           : Factor w/ 4 levels "1","2","3","4": 4 2 4 2 4 3 4 4 4 4 ...
 $ current.assets               : Factor w/ 4 levels "1","2","3","4": 2 1 1 1 2 1 1 1 3 4 ...
 $ age                          : int  21 36 23 39 38 48 39 40 65 23 ...
 $ other.credits                : Factor w/ 3 levels "1","2","3": 3 3 3 3 1 3 3 3 3 3 ...
```

Înainte de a începe analiza datelor vom crea niște metode care vor fi folosite pentru a putea analiza datele. Prima data o să implementăm o metodă care ne va ajuta să luam niste statistice sumare despre datele numerice

```
# analizarea variabilelor numerice
get.numeric.variable.stats <- function(indep.var, detailed=FALSE){
  options(scipen=100)
  options(digits=2)
  if (detailed){
    var.stats <- stat.desc(indep.var)
  }else{
    var.stats <- summary(indep.var)
  }

  dataset <- data.frame(round(as.numeric(var.stats),2))
  colnames(dataset) <- deparse(substitute(indep.var))
  rownames(dataset) <- names(var.stats)

  if (names(dev.cur()) != "null device"){
    dev.off()
  }
  grid.table(t(dataset))
}
```

După vom crea niste funcții pentru a vizualiza datele numerice

```r
# vizualizare
# histograms\density
visualize.distribution <- function(indep.var){
  pl1 <- qplot(indep.var, geom="histogram",
               fill=I('gray'), binwidth=5,
               col=I('black'))+ theme_bw()
  pl2 <- qplot(indep.var, geom="density",
               fill=I('gray'), binwidth=5,
               col=I('black'))+ theme_bw()

  grid.arrange(pl1,pl2, ncol=2)
}


# box plots
visualize.boxplot <- function(indep.var, dep.var){
  pl1 <- qplot(factor(0),indep.var, geom="boxplot",
               xlab = deparse(substitute(indep.var)),
               ylab="values") + theme_bw()
  pl2 <- qplot(dep.var,indep.var,geom="boxplot",
               xlab = deparse(substitute(dep.var)),
               ylab = deparse(substitute(indep.var))) + theme_bw()

  grid.arrange(pl1,pl2, ncol=2)
}
```

Acum o să ne focusăm asupra variabilelor categorice. Prima data o să implementăm o metodă care ne va ajuta să luam niste statistice sumare despre datele categorice

```r
# analizarea variabilelor categorice
get.categorical.variable.stats <- function(indep.var){

  feature.name = deparse(substitute(indep.var))
  df1 <- data.frame(table(indep.var))
  colnames(df1) <- c(feature.name, "Frequency")
  df2 <- data.frame(prop.table(table(indep.var)))
  colnames(df2) <- c(feature.name, "Proportion")

  df <- merge(
    df1, df2, by = feature.name
  )
  ndf <- df[order(-df$Frequency),]
  if (names(dev.cur()) != "null device"){
    dev.off()
  }
  grid.table(ndf)
}
```

După vom genera un tabel contingent

```
# generare tabel contingent
get.contingency.table <- function(dep.var, indep.var,
                                  stat.tests=F){
  if(stat.tests == F){
    CrossTable(dep.var, indep.var, digits=1,
               prop.r=F, prop.t=F, prop.chisq=F)
  }else{
    CrossTable(dep.var, indep.var, digits=1,
               prop.r=F, prop.t=F, prop.chisq=F,
               chisq=T, fisher=T)
  }
}
```

După vom crea niște funcții pentru vizualizare

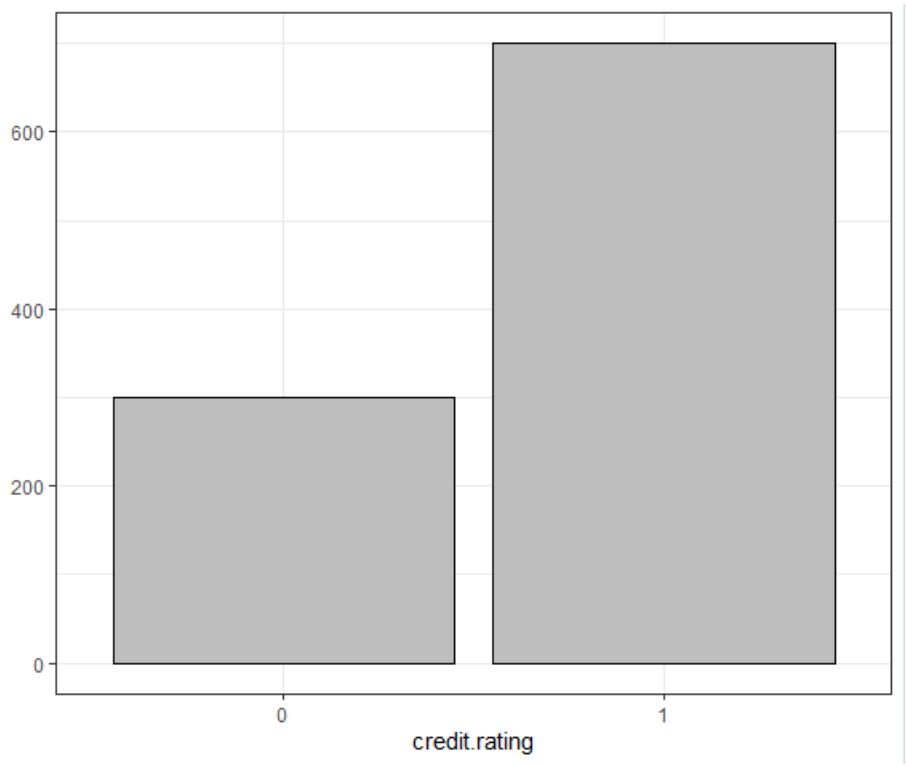
```
# vizualizare
# barcharts
visualize.barchart <- function(indep.var){
  qplot(indep.var, geom="bar",
        fill=I('gray'), col=I('black'),
        xlab = deparse(substitute(indep.var))) + theme_bw()
}


# mosaic plots
visualize.contingency.table <- function(dep.var, indep.var){
  if (names(dev.cur()) != "null device"){
    dev.off()
  }
  mosaicplot(dep.var ~ indep.var, color=T,
             main = "Contingency table plot")
}
```

• Rezultate și discuții

Urmează o analiză descriptivă a datelor. Prima dată vom atașa date frame-ul

Um exemplu de analiză a datelor:

Prin apelarea metodei visualize.barchart pe credit.rating se va genera un grafic de tip barchart.Putem observa că credit.rating are două clase: 1 respectiv 0. Clasa 1 reprezintă plătitorii de credite care nu prezintă un risc de credit pentru bancă. Clasa 0 reprezintă plătitorii de credite care prezintă un risc de credit pentru bancă. Din acest grafic putem observa că banca are un număr mai mare de plătitori în care are încredere să le acorde credite. Numărul plătitorilor în care banca are încredere este 700, numărul plătitorilor în care banca nu are încredere este 300.



În codul sursă(analizaDescriptivă.R) am analizat datele folosind mai multe grafice, am adus transformări și rescrieri claselor care apartin de anumite atribute, am realizat teste statistice asupra anumitor atribute.Toate explicațiile o să le oferim în timpul examinării, deoareceam rămas în pană de timp și ne-am focusat pe codul sursă. În final setul de date a suferit modificări, valorile atributelor modifcandu-se în anumite locuri.Totodată o să prezentăm raportul complet cu analiza tututor elementelor în sesiunea de restanțe. Toate aceste modificări au fost scrise în alt fișier pe care îl vom folosi la analiza predictivă.

Urmează analiza predictivă a datelor. Întâi o să facem load la noul set de date pe care l-am generat în urma analizei descriptive, iar apoi o să ne focusăm pe transformarea datelor și pe normalizarea lor folosind anumite metode

```
# transformarea - factoring a datelor
to.factors <- function(dataset, variables){
  for (variable in variables){
    dataset[[variable]] <- as.factor(dataset[[variable]])
  }
  return(dataset)
}

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

În următorul pas o să convertim cele optsprezece atribute din int în categoric, iar pe cele trei din int în numeric

```
numeric.vars<- c("credit.duration.months","age","credit.amount")

credit.dataset<- scale.features(credit.dataset,numeric.vars)

#selectarea datelor pe care vrem să le transformăm
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.dataset<- to.factors(dataset=credit.dataset,variables = categorical.vars)
```

Din lipsa timpului nu am reusit să facem o analiză mai detaliată, în sesiunea de restanțe o să revenim cu o modelare folosind logistic regression, decision trees, random forests.

• Concluzia

Prin analiza descriptivă și prin aplicarea algorimilor machine-learning am reusit să răspundem la prima întrebare care este “Putem optimiza datele dintr-un set prin analiza descriptivă a datelor și prin aplicarea algoritmilor machine-learning?”. Acest lucru a fost posibil prin analiza datelor clienților și prin aplicarea algoritmilor machine-learning care au dus la generarea unui nou set de date.

• Bibliografie

R: Unleash Machine Learning Techniques
De Raghav Bali, Dipanjan Sarkar, Brett Lantz, Cory Lesmeister