Exploração de dados - Banco Czech

Contents

Objetivo

O objetivo do nosso trabalho é identificar quais fatores que podem impactar no atraso e não pagamento das dívidas.

Carga dos dados

Utilizamos a função read.csv2 que nos permite carregar os dados disponíveis em texto, no formato CSV.

```
read.csv2("./dados/account.asc", stringsAsFactors = FALSE) -> account
read.csv2("./dados/client.asc", stringsAsFactors = FALSE) -> client
read.csv2("./dados/card.asc", stringsAsFactors = FALSE) -> card
read.csv2("./dados/disp.asc", stringsAsFactors = FALSE) -> disp
read.csv2("./dados/district.asc", stringsAsFactors = FALSE) -> district
read.csv2("./dados/trans.asc", stringsAsFactors = FALSE) -> trans
read.csv2("./dados/loan.asc", stringsAsFactors = FALSE) -> loan
read.csv2("./dados/order.asc", stringsAsFactors = FALSE) -> order
```

Tratamento dos dados

Cliente

O tratamento inicial da relação cliente envolve a separação do campo data de nascimento e gênero, que será tratado por M e F. E a transformação do campo birth_number em uma data válida para o R. Para isso transformamos o birth_number em um campo númerico, obtemos 4 dígitos da 3ª à 4ª posição, sendo este valor superior a 50 consideramos como feminino, pois o mês de nascimento das mulheres está com uma soma de 50 unidades. Subtraímos 5000 do mês das mulheres, pois como um único campo numérico, implica em diminuir 50 do campo mensal. Após isso concatenamos o número 19 ao começo do birth_number, no intuíto de deixar melhor preparado para a formatação da data, que ocorre logo em seguida. Logo após, calculamos a idade e selecionamos apenas os campos que nos serão úteis para o nosso estudo. samos a data de referência como 01/01/1998, devido a referência dos dados, para calcular a idade dos clientes.

```
currentdate <- as.Date("1998/01/01", format="%Y/%m/%d")
client <- client %>%
  mutate(mesajustado = as.numeric(stringr::str_sub(birth_number,3,4))) %>%
  mutate(gender = ifelse(mesajustado > 50, "F", "M")) %>%
  mutate(birth_number = ifelse(gender=="F", birth_number - 5000, birth_number)) %>%
  mutate(birth_number = paste0("19", birth_number)) %>%
  mutate(birth_number = as.Date(birth_number, "%Y%m%d")) %>%
  mutate(age = year(currentdate) - year(birth_number)) %>%
  select(client_id, age, district_id, gender)
```

Conta

O tratamento da conta incluí renomear as frequências da conta e seleção dos campos importantes ao estudo.

```
#Tradução
account$frequency <- gsub("POPLATEK MESICNE", "mensal", account$frequency)
account$frequency <- gsub("POPLATEK TYDNE", "semanal", account$frequency)
account$frequency <- gsub("POPLATEK PO OBRATU", "acadatransacao", account$frequency)
account$frequency <- as.factor(account$frequency)
account <- select(account, account_id, frequency)
```

Distrito

O tratamento inicial da relação distrito começa na renomeação dos campos para melhor entendimento. Conversão dos campos de unemp_95 e unemp_96 para numéricos. Limpeza dos valores NA. Cálculo da taxa de desemprego entre os anos 95 e 96. E seleção dos valores que serão usados neste estudo.

```
#Renomear campos para melhor entendimento
colnames(district)[1] <- 'district_id'
colnames(district)[2] <- 'district_name'
colnames(district)[11] <- 'avg_sal'
colnames(district)[12] <- 'unemp_95'
colnames(district)[13] <- 'unemp_96'

#Converter campos para numérico
district$unemp_95 = as.numeric(district$unemp_95)
```

```
## Warning: NAs introduzidos por coerção
```

```
district$unemp_96 = as.numeric(district$unemp_96)

#Limpeza de NA
district[is.na(district$unemp_95),12] <- 1

#Cálculo da taxa de desemprego e seleção de valores
  district %>%
  mutate(unemp_r = ifelse(unemp_95 == 0 | unemp_96 == 0, 1, unemp_96/unemp_95)) %>%
  select(district_id, district_name, avg_sal, unemp_r) -> district
```

Cartão

O tratamento inicial da relação cartão seleciona os campos que serão úteis para este estudo. Preparamos então o tipo do cartão como categórica.

```
colnames(card)[3] <- 'card_type'
card <- card %>%
        select(card_id, disp_id, card_type)
#%>%     mutate(card_type = as.factor(card_type))
```

Transação

Na relação de transação, identificamos alguns valores como "VYBER" que não estão descritos na documentação e não pareceu ser pertinente aos nossos estudos, por isso foi filtrado. O tipo de transação foi traduzido para facilitar o entendimento. E foi feito um agrupamento pela conta para facilitar o relacionamento de 1 para 1 com os valores que nos interessam.

```
unique(trans$type)
```

```
## [1] "PRIJEM" "VYDAJ"  "VYBER"
```

```
trans$type <- gsub("PRIJEM", "credito", trans$type)
trans$type <- gsub("VYDAJ", "debito", trans$type)


trans <- trans %>%
        dplyr::filter(type != "VYBER") %>%
        mutate(amount = as.numeric(amount)) %>%
        select(account_id, type, amount) %>%
```

```
        group_by(account_id) %>%
        summarise(
          credito = sum(ifelse(type == "credito", amount, 0)),
          debito = sum(ifelse(type == "debito", amount, 0)),
          quant_trans = n()) %>%
        mutate(withdraw_rate = debito / credito) %>%
        select(account_id, withdraw_rate, quant_trans)
```

Disposição

A relação de disposição é útil para auxiliar nos relacionamentos. Como nosso estudo centraliza no pagamento ou não de dívidas, sendo que somente os donos da conta conseguem pegar empréstimos, filtramos as disposições para os donos das contas que serão o alvo do estudo.

```
unique(disp$type)
```

```
## [1] "OWNER"     "DISPONENT"
```

```
disp <- disp %>%
        dplyr::filter(type == "OWNER") %>%
        select(disp_id, client_id, account_id)
```

Empréstimo

No preparo da relação de empréstimo, traduzimos os status para os significados reais, também preparamos a taxa de pagamento e selecionamos os campos úteis ao estudo.

```
unique(loan$status)
```

```
## [1] "B" "A" "C" "D"
```

```
loan$status <- gsub("A", "finalizado", loan$status)
loan$status <- gsub("B", "finalizado nao pago", loan$status)
loan$status <- gsub("C", "vigente", loan$status)
loan$status <- gsub("D", "em debito", loan$status)
loan$status <- as.factor(loan$status)
loan$payments = as.numeric(loan$payments)
loan$amount = as.numeric(loan$amount)

loan <- loan %>%
        mutate(payments_rate = payments / amount) %>%
        select(loan_id, account_id, amount, duration, status, payments_rate)

colnames(loan)[3] <- 'loan_amount'
colnames(loan)[4] <- 'loan_duration'
colnames(loan)[5] <- 'loan_status'
colnames(loan)[6] <- 'loan_payment_rate'
```

Ordem de Pagamento

No preparo da ordem de pagamento, o texto do tipo de ordem é traduzido, ignoramos os que não possuem nenhum tipo especificado e selecionamos os dados importantes a este estudo através do agrupamento pela conta.

```
unique(order$k_symbol)
```

```
## [1] "SIPO"     "UVER"     " "        "POJISTNE" "LEASING"
```

```
order$k_symbol = gsub("POJISTNE", "seguro", order$k_symbol)
order$k_symbol = gsub("SIPO", "domestico", order$k_symbol)
order$k_symbol = gsub("LEASING", "leasing", order$k_symbol)
order$k_symbol = gsub("UVER", "divida", order$k_symbol)
order$amount = as.numeric(order$amount)

order <- order %>%
         dplyr::filter(k_symbol != " ") %>%
         group_by(account_id) %>%
         summarise(
            total_ordem = sum(amount),
            quant_ordem = n(),
            ja_pagou_seguro = any(k_symbol == "seguro"),
            media_transf = sum(amount) / n()
         )
```

União dos dados

```
dados <- client %>%
         inner_join(disp, by = "client_id") %>%
         left_join(card, by = "disp_id") %>%
         left_join(district, by = "district_id") %>%
         inner_join(account, by = "account_id") %>%
         left_join(order, by= "account_id") %>%
         left_join(trans, by="account_id") %>%
         inner_join(loan, by="account_id")
```