

Trabalho Final - Analise de Dados em R - CZECH BANK

Author “Edna Nakano - Matricula - 076819/2016”

Author “Cesar Marcondes - Matricula 079149/2016”

Author “Cecilia Kimura - Matricula 076551/2016”

31 October, 2016

Contents

1 Relatorio Final	1
1.1 Limpeza e Exploração dos Dados	2
1.1.1 Base de dados Account	2
1.1.2 Base de dados Client	3
1.1.3 Base de dados Disposition	4
1.1.4 Base de dados Credit Card	5
1.1.5 Base de dados Loan	6
1.1.6 Base de dados Demographic	7
1.1.7 Base de dados Order	9
1.1.8 Base de dados Transactions	10
1.2 Estudo de Grupos de Dados e Testes de Hipótese	12
1.2.1 Comparando os saldos médios das contas de clientes de BONS e MAL pagadores . . .	12
1.2.2 Verificar os cartões de crédito de BONS pagadores.	14
1.2.3 Análise BONS pagadores e a relação com ORDERS entre bancos	15
1.2.4 Comparação Movimentação Financeira de BONS e MAUS pagadores em ORDERS .	18
1.2.5 Análise de correlação entre fatores com uso de Clusters k-means	19
1.3 Conclusões	22

1 Relatorio Final

A análise de dados sendo realizada nesse trabalho tenta melhorar o entendimento de como é o perfil dos clientes do banco. Estamos particularmente interessados, em encontrar características dos clientes que possuem cartão de crédito, e encontrar indicadores de empréstimos para bons e ruins pagadores

O projeto está estruturado nas seguintes fases:

- Limpeza e Exploração dos Dados
- Estudo de Grupos de Dados e Testes de Hipótese

1.1 Limpeza e Exploração dos Dados

Esta primeira seção é focada em tratamento dos dados e análise exploratória. Para tanto, iniciamos o trabalho carregando as bibliotecas dplyr e ggplot2. Além disso, também carregando as tabelas do dataset que iremos trabalhar. Os arquivos contendo as tabelas, que compõem o dataset são: account, card, client, disp, district, loan, order e trans.asc. Todos os arquivos estão em formato ascii e possuem como separador de campos o simbolo “;”. O maior arquivo é o trans.asc que possui 1.056.320 objetos e 66 MB de tamanho. Usaremos o mesmo nome do arquivo para criar cada DataFrame em R.

```
library(dplyr)
library(ggplot2)
library(gridExtra)
library(lubridate)
library(scales)
library(ggmap)
account <- tbl_df(read.csv("account.asc",
    sep = ";", stringsAsFactors = FALSE))
card <- tbl_df(read.csv("card.asc",
    sep = ";", stringsAsFactors = FALSE))
client <- tbl_df(read.csv("client.asc",
    sep = ";", stringsAsFactors = FALSE))
disp <- tbl_df(read.csv("disp.asc",
    sep = ";", stringsAsFactors = FALSE))
district <- tbl_df(read.csv("district.asc",
    sep = ";", stringsAsFactors = FALSE))
loan <- tbl_df(read.csv("loan.asc",
    sep = ";", stringsAsFactors = FALSE))
order <- tbl_df(read.csv("order.asc",
    sep = ";", stringsAsFactors = FALSE))
trans <- tbl_df(read.csv("trans.asc",
    sep = ";", stringsAsFactors = FALSE))
```

1.1.1 Base de dados Account

```
str(account)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame': 4500 obs. of 4 variables:
## $ account_id : int 576 3818 704 2378 2632 1972 1539 793 2484 1695 ...
## $ district_id: int 55 74 55 16 24 77 1 47 74 76 ...
## $ frequency : chr "POPLATEK MESICNE" "POPLATEK MESICNE" "POPLATEK MESICNE" ...
## $ date       : int 930101 930101 930101 930101 930102 930102 930103 930103 930103 ...
```

Iniciamos a limpeza de dados por Account. Podemos observar que existe um campo chamado Frequency, que representa a frequencia com que a conta cria extratos para o correntista, pode ser mensal, semanal ou a cada transacao. Alteramos os nomes em Tcheco para ingles para simplicar a visualizacao e o entendimento. E também convertemos esse campo para fator, pois só existem 3 fatores.

```
names(account)[3] <- paste("statement")
account$statement <- gsub("POPLATEK MESICNE", "monthly", account$statement)
account$statement <- gsub("POPLATEK TYDNE", "weekly", account$statement)
account$statement <- gsub("POPLATEK PO OBRATU", "pertransaction", account$statement)
account$statement <- as.factor(account$statement)
```

Tambem convertemos as datas para o formato de datas no R. E fizemos o agrupamento dos dados das contas por ano e mes, de modo a mostrar a evolução do crescimento do numero de contas ao longo dos 5 anos do dataset. Notamos que nos anos de 1993 e 1996 houve um crescimento maior de numero de contas.

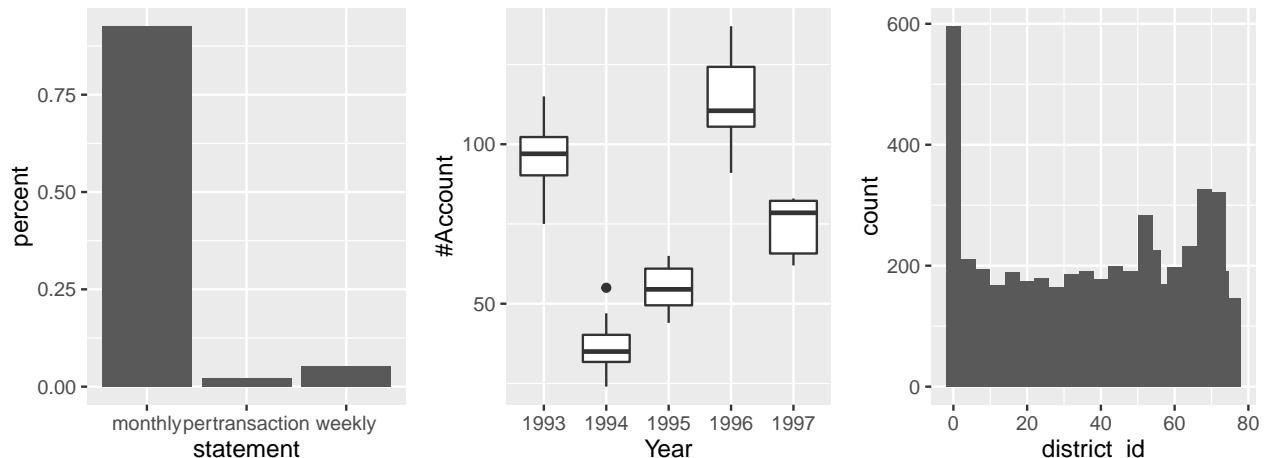
```
account$date <- as.Date(paste(account$date), "%y%m%d")
acc_time <- account %>%
  mutate(month = format(date, "%m"), year = format(date, "%Y")) %>%
  group_by(year, month) %>%
  summarise(total = n()) %>%
  mutate(yearmonth = paste(year, month))
```

A seguir podemos ver essas características das contas, visualmente. As contas também estao distribuídas (quase uniformemente) nos diferentes distritos.

```
p1 <- ggplot(account, aes(x = statement))
p1 <- p1 + geom_bar(aes(y = (.count...)/sum(..count..)))
p1 <- p1 + scale_y_continuous("percent")

p2 <- ggplot(acc_time, aes(factor(year), total))
p2 <- p2 + geom_boxplot()
p2 <- p2 + xlab("Year") + ylab("#Account")

p3 <- ggplot(account, aes(x = district_id))
p3 <- p3 + stat_bin(bins = 20)
p3 <- p3 + geom_histogram()
grid.arrange(p1, p2, p3, ncol=3)
```



1.1.2 Base de dados Client

```
str(client)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame': 5369 obs. of 3 variables:
## $ client_id : int 1 2 3 4 5 6 7 8 9 10 ...
## $ birth_number: int 706213 450204 406009 561201 605703 190922 290125 385221 351016 430501 ...
## $ district_id : int 18 1 1 5 5 12 15 51 60 57 ...
```

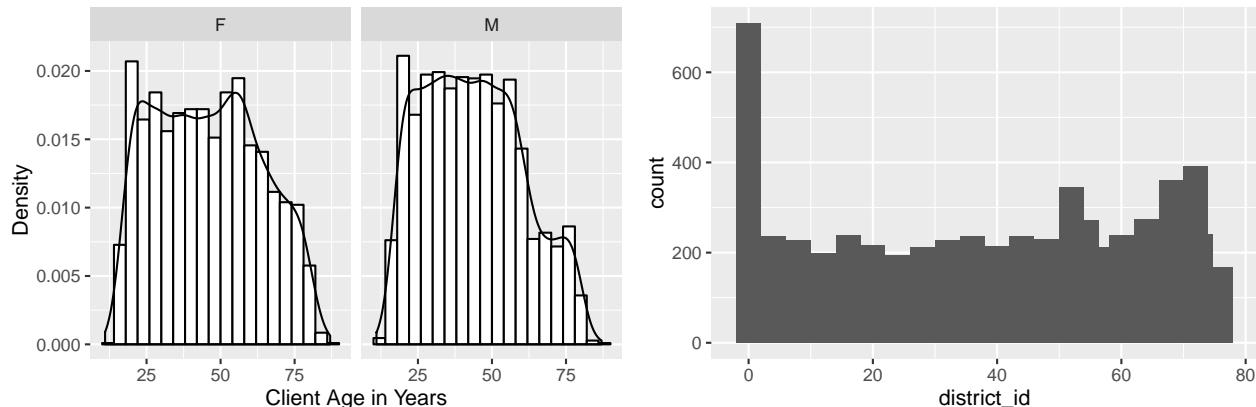
A limpeza de dados por Client consiste basicamente em criar uma nova coluna contendo o genero dos clientes, masculino ou feminino, e também converter as datas de nascimento para um formato de datas no R.

```
client <- client %>%
  mutate(mesajustado = as.numeric(stringr::str_sub(birth_number,3,4))) %>%
  mutate(sex = ifelse(mesajustado > 50, "F", "M")) %>%
  mutate(birth_number = ifelse(sex=="F", birth_number - 5000, birth_number))
client$birth_number <- paste0("19", client$birth_number)
client$birth_number <- as.Date(client$birth_number, "%Y%m%d")
```

Podemos plotar a variação de idade dos grupos feminino e masculino dos correntistas por banco. Os dados mostram uma população média normal, de homens e mulheres, sendo que a fração dos homens maiores que 70 tende a ser menor que as mulheres. A distribuição dos distritos é quase a mesma da descoberta nos dados das contas, e portanto, continua uniformemente distribuída e sem nenhuma característica especial.

```
currentdate <- as.Date("1998/01/01", format="%Y/%m/%d")
p1 <- ggplot(client, aes(x = year(currentdate)-year(client$birth_number)))
p1 <- p1 + geom_histogram(aes(y=..density..), binwidth = 4, colour="black", fill="white")
p1 <- p1 + geom_density(alpha=0.8)
p1 <- p1 + facet_grid(. ~ sex)
p1 <- p1 + ylab("Density") + xlab("Client Age in Years")

p2 <- ggplot(client, aes(x = district_id))
p2 <- p2 + stat_bin(bins = 20)
p2 <- p2 + geom_histogram()
grid.arrange(p1, p2, ncol=2)
```



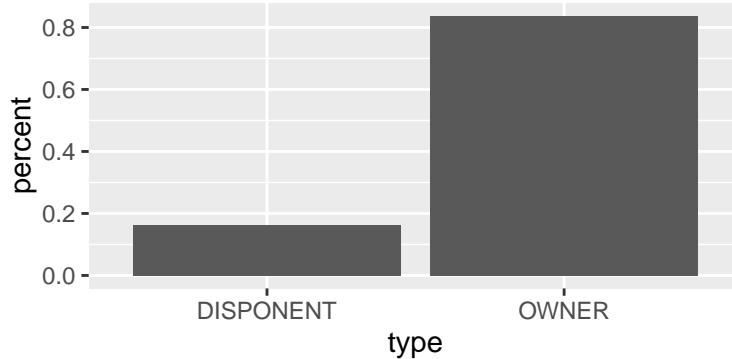
1.1.3 Base de dados Disposition

```
str(disp)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame': 5369 obs. of 4 variables:
## $ disp_id    : int 1 2 3 4 5 6 7 8 9 10 ...
## $ client_id  : int 1 2 3 4 5 6 7 8 9 10 ...
## $ account_id: int 1 2 2 3 3 4 5 6 7 8 ...
## $ type       : chr "OWNER" "OWNER" "DISPONENT" "OWNER" ...
```

Disposition é basicamente um relacionamento entre correntistas e suas contas bancárias. A aspecto mais interessante é que alguns desses relacionamentos indicam disposition como “dono” que pode emitir transferências bancárias e pedir empréstimos. Vamos transformar esse campo em uma variável categórica (factor) e plotar a diferença de frequências entre owners e disponent. Como podemos perceber na figura, a quantidade de “dono” chega acima de 80%.

```
disp$type <- as.factor(disp$type)
p1 <- ggplot(disp, aes(x = type))
p1 <- p1 + geom_bar(aes(y =(..count..)/sum(..count..)))
p1 + scale_y_continuous("percent")
```



1.1.4 Base de dados Credit Card

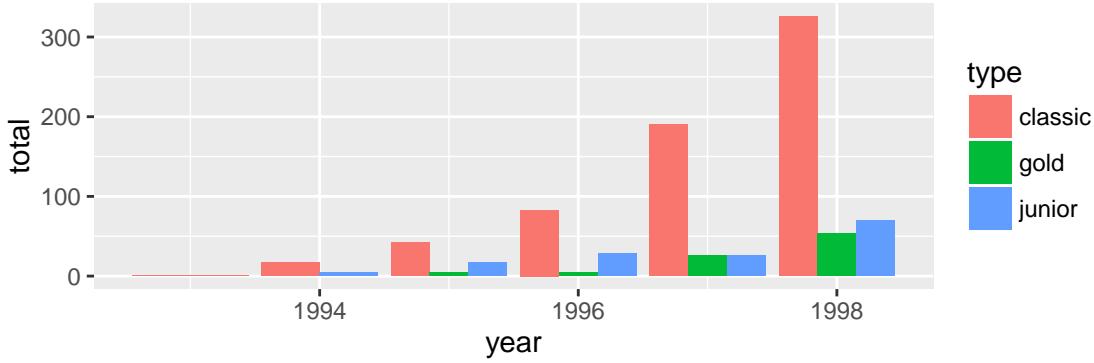
Os cartões de crédito emitidos pelos bancos vinculados a contas bancárias. Esses podem ser de três tipos, junior, classic e gold. Uma variável categórica e portanto precisamos convertê-la para factor, e também precisamos limpar o formato data, para entender por quanto tempo de existência de um determinado cartão.

```
card$type <- as.factor(card$type)
card$issued <- paste0("19", card$issued)
card$issued <- as.Date(card$issued, "%Y%m%d")
old.digits <- options('digits')
options(digits=2)
tmp <- table(card$type)
prop.table(tmp)

##
## classic      gold    junior
##   0.739     0.099    0.163

card_time <- card %>%
  group_by(year = year(issued), type) %>%
  summarise(total = n())

ggplot(card_time, aes(x=year, y=total, fill=type)) +
  geom_bar(stat="identity", position=position_dodge())
```



1.1.5 Base de dados Loan

```
str(loan)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame': 682 obs. of 7 variables:
## $ loan_id    : int 5314 5316 6863 5325 7240 6687 7284 6111 7235 5997 ...
## $ account_id: int 1787 1801 9188 1843 11013 8261 11265 5428 10973 4894 ...
## $ date       : int 930705 930711 930728 930803 930906 930913 930915 930924 931013 931104 ...
## $ amount     : int 96396 165960 127080 105804 274740 87840 52788 174744 154416 117024 ...
## $ duration   : int 12 36 60 36 60 24 12 24 48 24 ...
## $ payments   : num 8033 4610 2118 2939 4579 ...
## $ status     : chr "B" "A" "A" "A" ...
```

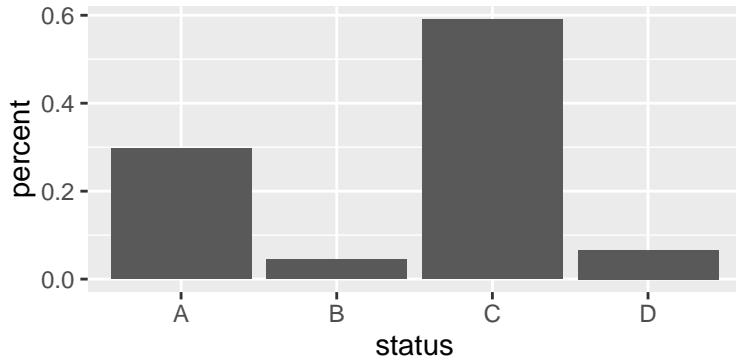
Os empréstimos tem informações muito úteis para ajudar a classificação dos correntistas para oferecimento de cartões de crédito e/ou outros empréstimos. É usando essa informação que podemos detectar bons pagadores e maus pagadores. Bem como ver o volume e o tempo médio dos empréstimos. O data frame contém alguns campos para adequação, como por exemplo, o campo status é do tipo categórico e precisamos defini-lo como factor. Também precisamos fixar as datas no formato correto.

```
loan$status <- as.factor(loan$status)
loan$date <- paste0("19", loan$date)
loan$date <- as.Date(loan$date, "%Y%m%d")
head(loan)
```

```
## # A tibble: 6 × 7
##   loan_id account_id      date amount duration payments status
##   <int>     <int> <date>   <int>   <int>    <dbl> <fctr>
## 1     5314      1787 1993-07-05    96396      12     8033     B
## 2     5316      1801 1993-07-11   165960      36     4610     A
## 3     6863      9188 1993-07-28   127080      60     2118     A
## 4     5325      1843 1993-08-03   105804      36     2939     A
## 5     7240     11013 1993-09-06   274740      60     4579     A
## 6     6687      8261 1993-09-13    87840      24     3660     A
```

Podemos observar algumas variáveis e suas distribuições. Começando pelas proporções entre A, C e B e D.

```
p1 <- ggplot(loan, aes(x=status))
p1 <- p1 + geom_bar(aes(y = ..count../sum(..count..)))
p1 + scale_y_continuous("percent")
```

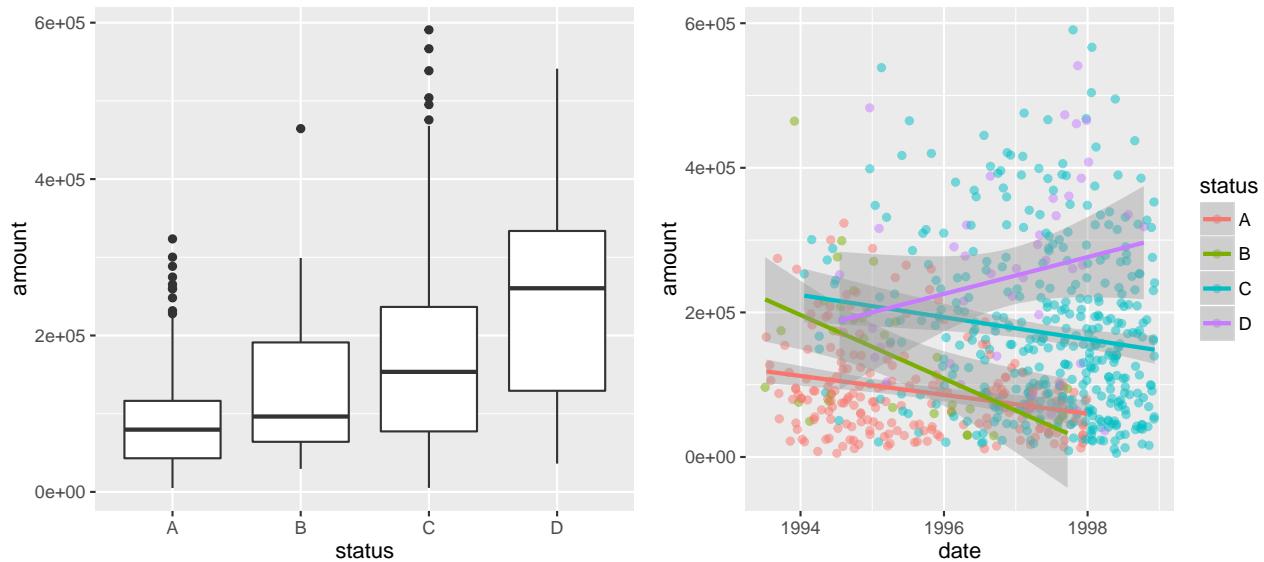


Vamos estudar a distribuição dos empréstimos e dos pagamentos por status. Os pontos mostrando as datas de contratação dos empréstimos e seus respectivos valores.

```
p1 <- ggplot(loan, aes(x=status, y=amount))
p1 <- p1 + geom_boxplot()

p2 <- ggplot(loan, aes(x=date, y=amount, col=status))
p2 <- p2 + geom_point(alpha=0.5)
p2 <- p2 + stat_smooth(method = "lm")

grid.arrange(p1, p2, ncol=2)
```



1.1.6 Base de dados Demographic

```
str(district)

## Classes 'tbl_df', 'tbl' and 'data.frame':    77 obs. of  16 variables:
## $ A1 : int  1 2 3 4 5 6 7 8 9 10 ...
## $ A2 : chr  "Hl.m. Praha" "Benesov" "Beroun" "Kladno" ...
```

```

## $ A3 : chr "Prague" "central Bohemia" "central Bohemia" "central Bohemia" ...
## $ A4 : int 1204953 88884 75232 149893 95616 77963 94725 112065 81344 92084 ...
## $ A5 : int 0 80 55 63 65 60 38 95 61 55 ...
## $ A6 : int 0 26 26 29 30 23 28 19 23 29 ...
## $ A7 : int 0 6 4 6 4 4 1 7 4 4 ...
## $ A8 : int 1 2 1 2 1 2 3 1 2 3 ...
## $ A9 : int 1 5 5 6 6 4 6 8 6 5 ...
## $ A10: num 100 46.7 41.7 67.4 51.4 51.5 63.4 69.4 55.3 46.7 ...
## $ A11: int 12541 8507 8980 9753 9307 8546 9920 11277 8899 10124 ...
## $ A12: chr "0.29" "1.67" "1.95" "4.64" ...
## $ A13: num 0.43 1.85 2.21 5.05 4.43 4.02 2.87 1.44 3.97 0.54 ...
## $ A14: int 167 132 111 109 118 126 130 127 149 141 ...
## $ A15: chr "85677" "2159" "2824" "5244" ...
## $ A16: int 99107 2674 2813 5892 3040 3120 4846 4987 2487 4316 ...

```

Algumas limpezas necessárias são com relação a region, tornando-a uma variável categórica, que deve indicar determinadas regiões geográficas da Republica Tcheca.

```

district$A3 <- as.factor(district$A3)
head(district)

```

```

## # A tibble: 6 × 16
##   A1     A2          A3     A4     A5     A6     A7     A8     A9
##   <int> <chr>      <fctr> <int> <int> <int> <int> <int>
## 1 1 Hl.m. Praha Prague 1204953    0     0     0     1     1
## 2 2 Benesov central Bohemia 88884    80    26     6     2     5
## 3 3 Beroun central Bohemia 75232    55    26     4     1     5
## 4 4 Kladno central Bohemia 149893   63    29     6     2     6
## 5 5 Kolín central Bohemia 95616    65    30     4     1     6
## 6 6 Kutna Hora central Bohemia 77963   60    23     4     2     4
## # ... with 7 more variables: A10 <dbl>, A11 <int>, A12 <chr>, A13 <dbl>,
## #   A14 <int>, A15 <chr>, A16 <int>

```

Nós imprimimos o mapa apenas para fins estéticos. E podemos realizar a análise das varias variáveis de interesse geográfico como numero de habitantes, salário médio, razão de urbanismo, taxa de desemprego em 96, numero de municipios com mais de 100 mil habitantes.

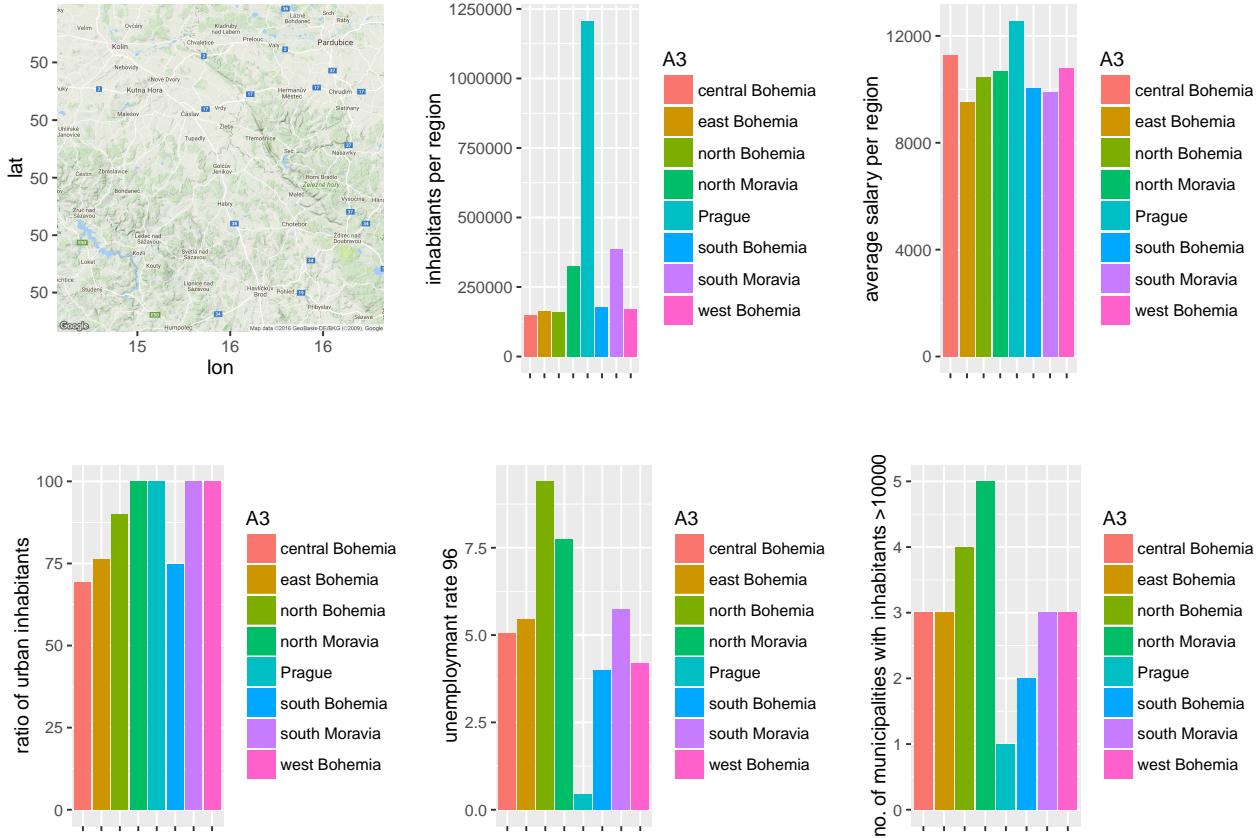
```

map <- get_map("czech republic", zoom=10)
p1 <- ggmap(map)

p2 <- ggplot(district, aes(x = A3, y = A4, fill = A3))
p2 <- p2 + geom_bar(stat="identity", position=position_dodge())
p2 <- p2 + ylab("inhabitants per region")
p2 <- p2 + theme(axis.title.x=element_blank(), axis.text.x=element_blank())

p3 <- ggplot(district, aes(x = A3, y = A11, fill = A3))
p3 <- p3 + geom_bar(stat="identity", position=position_dodge())
p3 <- p3 + ylab("average salary per region")
p3 <- p3 + theme(axis.title.x=element_blank(), axis.text.x=element_blank())
grid.arrange(p1, p2, p3, ncol=3)

```



1.1.7 Base de dados Order

```
str(order)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame': 6471 obs. of 6 variables:
## $ order_id : int 29401 29402 29403 29404 29405 29406 29407 29408 29409 29410 ...
## $ account_id: int 1 2 2 3 3 3 4 4 5 6 ...
## $ bank_to   : chr "YZ" "ST" "QR" "WX" ...
## $ account_to: int 87144583 89597016 13943797 83084338 24485939 59972357 26693541 5848086 37390208 ...
## $ amount    : num 2452 3373 7266 1135 327 ...
## $ k_symbol   : chr "SIP0" "UVER" "SIP0" "SIP0" ...
```

As transferencias bancarias são feitas para diversos bancos de destino. Portanto, é possível verificar quantos bancos, quais os valores de transferencia entre bancos e caracterizar as finalidades de “DOC” ou “TED”. Faremos algumas limpezas, colocando no simbolo K, uma variavel categoria e trocaremos as legendas para ingles de modo a obter melhor entendimento.

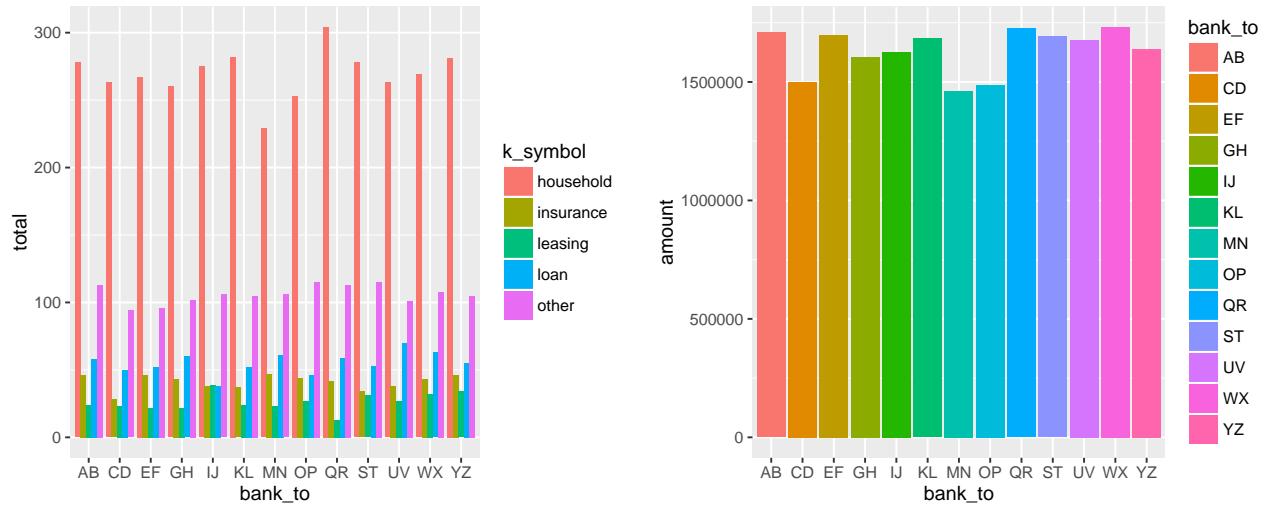
```
order$bank_to <- as.factor(order$bank_to)
order$k_symbol <- gsub("SIP0", "household", order$k_symbol)
order$k_symbol <- gsub("POJISTNE", "insurance", order$k_symbol)
order$k_symbol <- gsub("LEASING", "leasing", order$k_symbol)
order$k_symbol <- gsub("UVER", "loan", order$k_symbol)
order$k_symbol <- gsub(" ", "other", order$k_symbol)
order$k_symbol <- as.factor(order$k_symbol)
```

Vamos processar os graficos dos tipos de transferencias por banco. E também as quantidades totais de transferencia por banco.

```
orderbanks <- order %>%
  group_by(bank_to, k_symbol) %>%
  summarise(total = n())

p1 <- ggplot(orderbanks, aes(x=bank_to, y=total, fill=k_symbol))
p1 <- p1 + geom_bar(stat="identity", position=position_dodge())

p2 <- ggplot(order, aes(x=bank_to, y=amount, fill=bank_to))
p2 <- p2 + geom_bar(stat="identity")
grid.arrange(p1, p2, ncol=2)
```



Os resultados indicam que a maioria das transferencias é pagamento de algum tipo de financiamento habitacional ou aluguel, e que os volumes transferidos para outros bancos esta muito proximo um do outro.

1.1.8 Base de dados Transactions

```
str(trans)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame': 1056320 obs. of 10 variables:
## $ trans_id : int 695247 171812 207264 1117247 579373 771035 452728 725751 497211 232960 ...
## $ account_id: int 2378 576 704 3818 1972 2632 1539 2484 1695 793 ...
## $ date      : int 930101 930101 930101 930101 930102 930102 930103 930103 930103 930103 ...
## $ type      : chr "PRIJEM" "PRIJEM" "PRIJEM" "PRIJEM" ...
## $ operation : chr "VKLAD" "VKLAD" "VKLAD" "VKLAD" ...
## $ amount    : num 700 900 1000 600 400 1100 600 1100 200 800 ...
## $ balance   : num 700 900 1000 600 400 1100 600 1100 200 800 ...
## $ k_symbol  : chr "" "" "" ...
## $ bank      : chr "" "" ...
## $ account   : int NA NA NA NA NA NA NA NA NA ...
```

O ultimo dataset isolado a ser limpado e analisado é o que mostra as transações bancárias. Precisamos limpar as datas, também melhorar os textos das variáveis categóricas como tipo de saque, finalidade da transação bancária.

```

trans$operation <- gsub("^$", "other", trans$operation)
trans$k_symbol <- gsub("^$", "other", trans$k_symbol)
trans$k_symbol <- gsub(" ", "other", trans$k_symbol)

trans$type <- gsub("PRIJEM", "credit", trans$type)
trans$type <- gsub("VYDAJ", "withdraw", trans$type)
trans$type <- gsub("VYBER", "withdraw", trans$type)
trans$type <- as.factor(trans$type)

trans$operation <- gsub("VYBER KARTOU", "credit card withdraw", trans$operation)
trans$operation <- gsub("VYBER", "cash withdraw", trans$operation)
trans$operation <- gsub("VKLAD", "credit cash", trans$operation)
trans$operation <- gsub("PREVOD Z UCTU", "collection bank", trans$operation)
trans$operation <- gsub("PREVOD NA UCET", "remittance bank", trans$operation)
trans$operation <- as.factor(trans$operation)

trans$k_symbol <- gsub("POJISTNE", "insurance", trans$k_symbol)
trans$k_symbol <- gsub("SLUZBY", "statement payment", trans$k_symbol)
trans$k_symbol <- gsub("UROK", "interest", trans$k_symbol)
trans$k_symbol <- gsub("SANKC.otherinterest", "interest", trans$k_symbol)
trans$k_symbol <- gsub("SANKC. UROK", "negative balance", trans$k_symbol)
trans$k_symbol <- gsub("SIPÓ", "household", trans$k_symbol)
trans$k_symbol <- gsub("DUCHOD", "old-age pension", trans$k_symbol)
trans$k_symbol <- gsub("UVER", "loan", trans$k_symbol)
trans$k_symbol <- as.factor(trans$k_symbol)

trans$date <- paste0("19", trans$date)
trans$date <- as.Date(trans$date, "%Y%m%d")

```

Para fins ilustrativos, plotaremos o desempenho do saldo das contas de algumas contas bancárias aleatórias ao longo do tempo. Lembrando que os passos de subida são créditos na conta, e os passos de descida são saques (withdraws).

```

rndacct <- sample(trans$account_id, 2)
rndacct

```

```
## [1] 2302 9242
```

```

trans_sample <- trans %>%
  group_by(account_id, date) %>%
  filter(account_id == rndacct[1])
#head(trans_sample, 20)

```

Agora, temos o gráfico da evolução das 2 contas aleatórias.

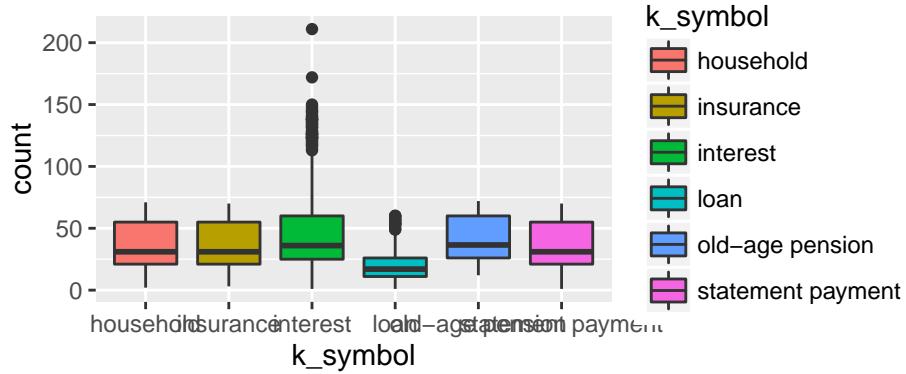


Explorando os dados, agrupando por conta e calculando por conta, as distribuições médias de cada variável categórica por conta: insurance, statement, interest, negative, household, pension e loan.

```
trans_symbols <- trans %>%
  group_by(k_symbol, account_id) %>%
  summarize(count = n()) %>%
  filter(k_symbol != "other") %>%
  select(k_symbol, account_id, count)
```

Plotando em gráfico esses dados por k_symbol.

```
ggplot(trans_symbols, aes(x = k_symbol, y = count, fill = k_symbol)) +
  geom_boxplot()
```



1.2 Estudo de Grupos de Dados e Testes de Hipótese

Esta segunda seção é focada nos estudos comparativos de grupos de dados, por exemplo, bons pagadores de empréstimos versus mal pagadores. Além disso, fazemos uso de testes de hipótese, regressão e clusterização para analisar aspectos desse dataset.

Essa parte de estudos dos grupos de dados e testes de hipótese darão confiança estatística que os grupos são diferentes e classificáveis. O restante dessa seção está organizado em termos de questões elaboradas para estudar as relações das variáveis.

1.2.1 Comparando os saldos médios das contas de clientes de BONS e MAL pagadores

A estratégia realizada foi fazer o join da base de dados LOAN com a base de dados ACCOUNT. E em seguida, separar BONS pagadores (Classes A e C) de MAUS pagadores (Classes B e D).

```

acc_loan <- inner_join(account, loan, by = c("account_id"))
acc_ok <- filter (acc_loan, status %in% c("A","C"))
acc_nok <- filter (acc_loan, status %in% c("B","D"))

```

Após separar os grupos BONS e MAUS pagadores, iniciamos essa parte, usando a base de dados TRANSACTIONS apenas as operações de BONS e MAUS pagadores separadamente.

```

trans_acc_ok <- inner_join(acc_ok, trans, by = c("account_id"))
trans_acc_nok <- inner_join(acc_nok, trans, by = c("account_id"))
nrow(trans_acc_ok)

```

```
## [1] 168796
```

```
nrow(trans_acc_nok)
```

```
## [1] 22760
```

E finalmente, agrupar os dados das novas bases por account_id e summarizar saldos médios. Usaremos os saldos médios em seguida.

```

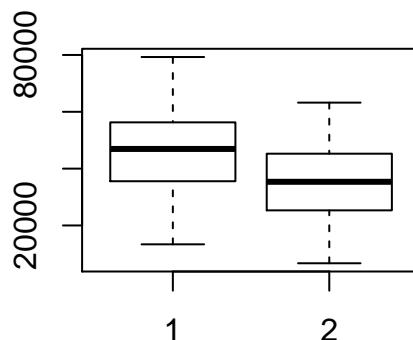
trans_acc_nok_group <- trans_acc_nok %>%
  group_by(account_id) %>%
  summarise (average = mean(balance))

trans_acc_ok_group <- trans_acc_ok %>%
  group_by(account_id) %>%
  summarise (average = mean(balance))

```

Após o cálculo dos saldos médios, realizamos uma comparação entre boxplots dos saldos médios de BONS pagadores versus boxplot dos MAUS pagadores. Essa comparação permite visualizar rapidamente uma diferença entre os dois conjuntos. Adicionalmente, aplicamos um teste de hipótese entre os dois conjuntos de dados.

```
boxplot(trans_acc_ok_group$average,
        trans_acc_nok_group$average)
```



```
t.test(trans_acc_ok_group$average, trans_acc_nok_group$average)
```

```

## Welch Two Sample t-test
##
## data: trans_acc_ok_group$average and trans_acc_nok_group$average
## t = 7, df = 100, p-value = 6e-10
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## 7823 14165
## sample estimates:
## mean of x mean of y
## 46160 35166

```

Conforme o t.test apresenta, o resultado foi p-value = 6e-10. O que permite afirmar que existem evidências estatísticas de que os saldos médios de BONS pagadores são maiores que os de MAUS pagadores.

1.2.2 Verificar os cartões de crédito de BONS pagadores.

Para esta tarefa, primeiro realizamos o join da base de dados de clientes BONS pagadores com a base de dados DISP para obter a chave disp_id. Essa chave é necessária para a relação com a base cartões de crédito.

E logo em seguida, precisamos realizar o join de clientes BONS pagadores com a base de dados CARD para identificar informações de cartões. Nesse caso utilizamos LEFT join, porque NEM todos os clientes possuem cartões. Essa informação pode ser bastante importante para nossas decisões estratégicas.

```

ok_group_disp <- inner_join(trans_acc_ok_group,
                             disp, by = c("account_id"))
ok_group_card <- left_join(ok_group_disp,
                            card, by = c("disp_id"))

```

Com base no nosso estudo anterior do saldo médio, criamos alguns criterios para recomendação de cartão.

- Somente clientes com saldo médio igual ou acima da média participarão desta ação.
- Cartão Classic - clientes BONS pagadores com saldo medio igual ou até saldo médio + 1 desvio padrão.
- Cartão Gold - clientes BONS pagadores com saldo medio maior que saldo médio + 1 desvio padrão.

Observação: No cálculo de média e desvio padrão, foi necessário utilizar a base de transações porque a base de cartões possui entradas adicionais devido a base DISP, o que pode gerar distorções na media e desvio padrão.

```

mean_ok <- mean(trans_acc_ok_group$average)
gold_ok <- (mean_ok + sd(trans_acc_ok_group$average))

ok_card_recom <- ok_group_card %>%
  filter(average > mean_ok) %>%
  mutate(rec_type = ifelse(average < gold_ok, "classic", "gold"))

```

Ao finalizar essa ação, temos as contas candidatas e a opção de cartão recomendada. Por último, vamos filtrar apenas as contas onde o cartão recomendado é diferente do cartão atual. Isso inclui os clientes que não possuem cartão. Com isso, teremos uma lista de 310 clientes selecionados, que podem ser usados para propor Cartões de Crédito, onde o tipo de cartão proposto, poderia variar de acordo com os saldos médios.

```

target_group_card <- filter(ok_card_recom,
                           (is.na(type.y) | type.y != rec_type))
str(target_group_card)

## Classes 'tbl_df', 'tbl' and 'data.frame': 310 obs. of 9 variables:
## $ account_id: int 25 67 132 176 290 319 339 339 378 501 ...
## $ average   : num 56279 56893 52603 49256 53457 ...
## $ disp_id   : int 31 78 159 215 352 389 414 415 458 603 ...
## $ client_id : int 31 78 159 215 352 389 414 415 458 603 ...
## $ type.x    : Factor w/ 2 levels "DISPONENT","OWNER": 2 2 1 2 2 2 2 1 2 2 ...
## $ card_id   : int NA NA NA NA NA NA 65 NA NA NA ...
## $ type.y    : Factor w/ 3 levels "classic","gold",...: NA NA NA NA NA NA 1 NA NA NA ...
## $ issued    : Date, format: NA NA ...
## $ rec_type   : chr "classic" "classic" "classic" "classic" ...

```

1.2.3 Análise BONS pagadores e a relação com ORDERS entre bancos

Uma outra pergunta que pode ser feita, e que pode ser explorada, é a análise dos volumes de recursos enviados entre outros bancos, que estão registrados no banco de dados ORDER. Uma possível estratégia seria manter esses recursos no próprio banco. Mas, vamos entender como isso funciona.

Iniciamos a análise para BONS pagadores e suas ordens de débito emitidas periodicamente. O primeiro passo é fazer o join da base de dados acc-ok com a base de dados ORDER. Ao agrupar a nova base de BONS e MAUS pagadores relacionado com suas ORDERS por account-id podemos visualizar as características de pagamento (symbol) e sumarizar por valores totais. E plotar os boxplots correspondentes dessas operações de débito e compará-los.

Primeiro para BONS pagadores.

```

acc_ok_order <- inner_join (acc_ok, order, by = c("account_id"))

acc_ok_order_group <- acc_ok_order %>%
  group_by(account_id,k_symbol) %>%
  summarise (total = sum(amount.y))

```

Depois as mesmas operações para MAUS pagadores.

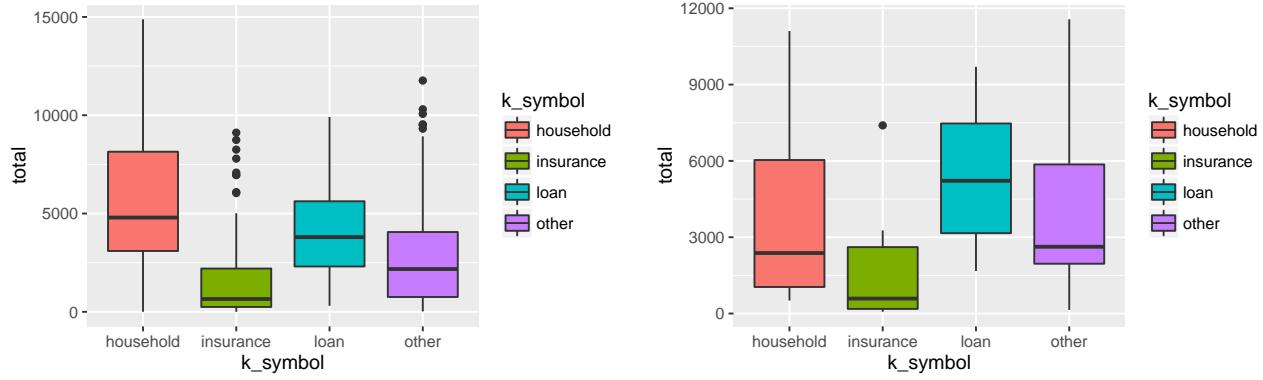
```

acc_nok_order <- inner_join (acc_nok, order, by = c("account_id"))

acc_nok_order_group <- acc_nok_order %>%
  group_by(account_id,k_symbol) %>%
  summarise (total = sum(amount.y))

```

E visualizando o resultado lado a lado.



Também podemos, considerar qual o volume total que esses BONS e MAUS pagadores, nessa ordem, movimentaram para outros bancos nos últimos 5 anos (dados contidos na base ORDER). MAUS pagadores tiveram um volume menor.

```
sum(acc_ok_order$amount.y)
```

```
## [1] 5559521
```

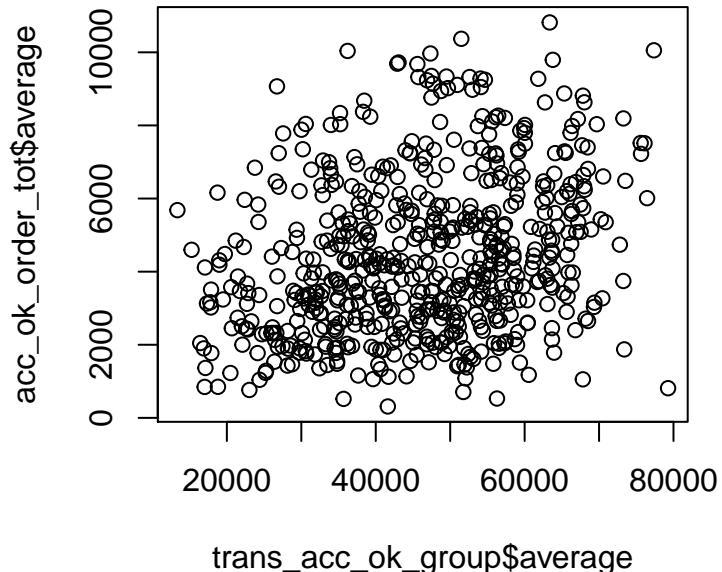
```
sum(acc_nok_order$amount.y)
```

```
## [1] 580741
```

Finalmente executamos uma análise de regressão para verificar se existe correlação entre os saldos médios e os volumes transferidos para outros bancos. Vamos iniciar com BONS pagadores.

```
acc_ok_order_tot <- acc_ok_order %>%
  group_by(account_id) %>%
  summarise (total = sum(amount.y), average = mean(amount.y))
```

```
plot (trans_acc_ok_group$average, acc_ok_order_tot$average)
abline(lm(trans_acc_ok_group$average ~ acc_ok_order_tot$average))
```



```

lm.result = lm(trans_acc_ok_group$average ~ acc_ok_order_tot$average)
summary(lm.result)

##
## Call:
## lm(formula = trans_acc_ok_group$average ~ acc_ok_order_tot$average)
##
## Residuals:
##    Min     1Q Median     3Q    Max 
## -34963 -10003    378   9976  39400 
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 3.85e+04  1.24e+03 31.11 < 2e-16 ***
## acc_ok_order_tot$average 1.73e+00  2.51e-01   6.91  1.2e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 
##
## Residual standard error: 13200 on 604 degrees of freedom
## Multiple R-squared:  0.0733, Adjusted R-squared:  0.0717 
## F-statistic: 47.8 on 1 and 604 DF,  p-value: 1.23e-11

```

Neste caso, o p-value é menor que 0,05 (p-value: 1.23e-11) indicando que existem evidências de relação entre as duas variáveis (saldo médio em conta e valores transferidos para outros bancos). Entretanto, verificou-se que o r-quadrado é igual a 0.0733, o que pode indicar que a relação entre as duas variáveis é fraca.

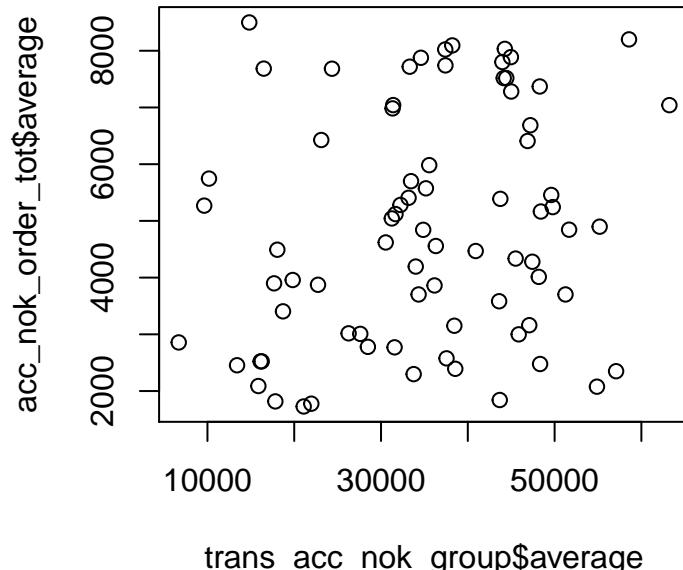
Fazendo a mesma análise para MAUS pagadores.

```

acc_nok_order_tot <- acc_nok_order %>%
  group_by(account_id) %>%
  summarise (total = sum(amount.y), average = mean(amount.y))

plot (trans_acc_nok_group$average, acc_nok_order_tot$average)
abline(lm(trans_acc_nok_group$average ~ acc_nok_order_tot$average))

```



```

lm.result = lm (trans_acc_nok_group$average ~ acc_nok_order_tot$average)
summary(lm.result)

## 
## Call:
## lm(formula = trans_acc_nok_group$average ~ acc_nok_order_tot$average)
## 
## Residuals:
##    Min     1Q Median     3Q    Max 
## -26172  -7280   -521   9671  25321 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 2.86e+04  3.85e+03   7.42  1.6e-10 ***
## acc_nok_order_tot$average 1.35e+00  7.31e-01   1.85   0.069 .  
## --- 
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 
## 
## Residual standard error: 12800 on 74 degrees of freedom
## Multiple R-squared:  0.044, Adjusted R-squared:  0.0311 
## F-statistic:  3.4 on 1 and 74 DF, p-value: 0.069

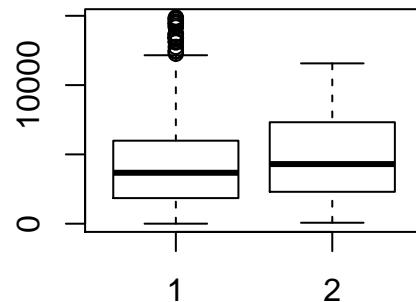
```

No caso dos MAUS pagadores, o p-value é igual a 0.069 e o r-quadrado é igual a 0.044. Nesse caso, podemos concluir que não existem evidências estatísticas de relação entre as variáveis.

1.2.4 Comparação Movimentação Financeira de BONS e MAUS pagadores em ORDERS

Analizando os dois grupos de operações, verificamos que em média, os MAUS pagadores têm um volume maior de transferências para outros bancos do que os BONS pagadores. Note que para BONS pagadores, existem outliers. A seguir, vamos aplicar um teste de hipótese, comparando BONS e MAUS pagadores para verificar se existem evidências que os volumes movimentados para outros bancos pelos dois grupos seriam muito diferentes.

```
boxplot(acc_ok_order_group$total,acc_nok_order_group$total)
```



```
t.test (acc_ok_order_group$total,acc_nok_order_group$total)
```

```
## 
## Welch Two Sample t-test
## 
```

```

## data: acc_ok_order_group$total and acc_nok_order_group$total
## t = -2, df = 200, p-value = 0.08
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## -997 52
## sample estimates:
## mean of x mean of y
## 4137 4609

```

Note que neste teste, o valor_p não é tão significante (p-value = 0.08). Isso indica que não existem fortes evidências estatísticas de que os volumes movimentados para outros bancos pelos dois grupos sejam muito diferentes. Como verificou-se que OS valores transferidos para outros bancos nos últimos 5 anos são consideráveis , recomendamos ao banco propor aos correntistas, principalmente para os BONS pagadores, ações que evitem a transferência de valores para outros bancos, aumentando a permanência dos montantes no próprio banco.

1.2.5 Análise de correlação entre fatores com uso de Clusters k-means

Nessa seção, realizamos a escolha de outros fatores para verificar se existe correlação entre eles, permitindo determinar alguns critérios na aprovação de empréstimos e cartão de crédito.

Para tanto, realizamos algumas operações para utilizarmos somente informações completas. Além disso, como o uso de Clusters pode ser muito intensivo, realizamos uma amostragem das transações, para diminuir o volume de dados a ser mostrado.

```

reducedtrans <- trans[sample(nrow(trans), 10000), ]
trans.c <- reducedtrans[complete.cases(reducedtrans), ]
disp.c <- disp[complete.cases(disp), ]

```

Manipulamos o dataframe para mudar o nome da coluna A1 do district para district_id.

```

district <- rename(district, district_id = A1)
district$A12 <- as.double(district$A12)

```

```
## Warning: NAs introduced by coercion
```

```
head(district)
```

```

## # A tibble: 6 × 16
##   district_id      A2          A3     A4     A5     A6     A7     A8
##       <int> <chr>      <fctr> <int> <int> <int> <int>
## 1           1 Hl.m. Praha Prague 1204953     0     0     0     1
## 2           2 Benesov central Bohemia 88884    80    26     6     2
## 3           3 Beroun central Bohemia 75232    55    26     4     1
## 4           4 Kladno central Bohemia 149893   63    29     6     2
## 5           5 Kolin central Bohemia 95616    65    30     4     1
## 6           6 Kutna Hora central Bohemia 77963   60    23     4     2
## # ... with 8 more variables: A9 <int>, A10 <dbl>, A11 <int>, A12 <dbl>,
## #   A13 <dbl>, A14 <int>, A15 <chr>, A16 <int>

```

Outras manipulações são necessárias, adicionar a idade dos clientes, e limpar as informações incompletas.

```

client <- mutate(client, age = year(currentdate) - year(birth_number))
client.c <- client[complete.cases(client), ]
district.c <- district[complete.cases(district), ]

```

Finalmente, passaremos para a fase de joins de tabelas, unindo as tabelas de trans, disp, client e district.

```

trans.c_disp.c <- inner_join(trans.c, disp.c, by = c("account_id"))
trans.c_disp.c_client.c <- inner_join(trans.c_disp.c, client.c, by = c("client_id"))
trans.c_disp.c_client.c_district.c <- inner_join(trans.c_disp.c_client.c, district.c, by = c("district_id"))

```

Posterior limpeza de dados, para eliminar tuplas com informações incompletas que não são necessárias para a clusterização.

```
trans.c_disp.c_client.c_district.c_complete <- trans.c_disp.c_client.c_district.c[complete.cases(trans.c_disp.c_client.c_district.c)]
```

Finalizar o processo, montando um data frame BALANÇO com todas as informações pertinentes para a análise dos clusters.

```

balanco <- select(trans.c_disp.c_client.c_district.c_complete,
                    balance, age, district_id, A11, A12, A13)
head(balanco)

```

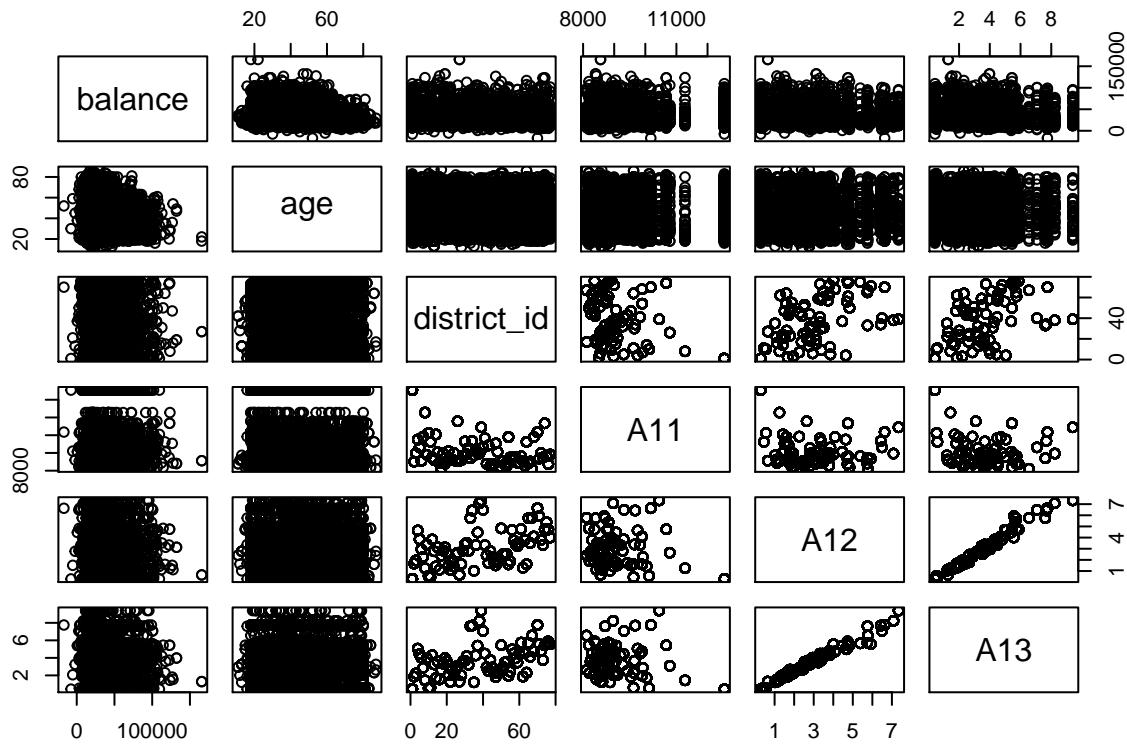
```

## # A tibble: 6 × 6
##   balance   age district_id   A11   A12   A13
##   <dbl> <dbl>      <int> <int> <dbl> <dbl>
## 1    73814     53        55  8743   1.9   2.4
## 2    73814     55        55  8743   1.9   2.4
## 3    46137     59        39 10446   7.3   9.4
## 4    12943     44        30  9650   3.4   3.7
## 5    31911     36        70 10177   6.6   7.8
## 6    64602     55        55  8743   1.9   2.4

```

Em seguida podemos fazer as análises de correlação em grupo das variáveis no dataframe balanço.

```
pairs(balanco)
```



```
cor(balanco)
```

```
##          balance      age district_id      A11      A12      A13
## balance    1.0000 -0.1977     -0.016  0.0191  0.0049  0.0078
## age        -0.1977  1.0000      0.028 -0.0018  0.0182  0.0166
## district_id -0.0165  0.0281      1.000 -0.4577  0.5867  0.5871
## A11         0.0191 -0.0018     -0.458  1.0000 -0.3998 -0.4125
## A12         0.0049  0.0182      0.587 -0.3998  1.0000  0.9897
## A13         0.0078  0.0166      0.587 -0.4125  0.9897  1.0000
```

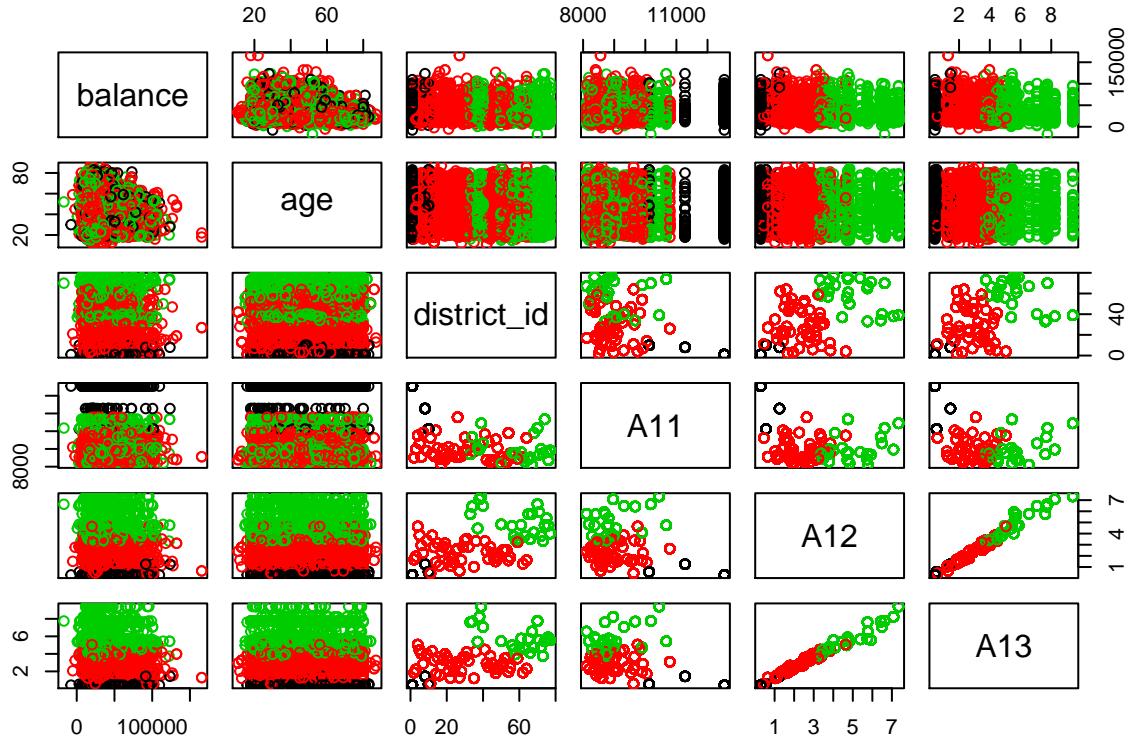
Analisando os gráficos e os valores de correlação não podemos concluir que exista relação entre esses fatores para a tomada de uma decisão na aprovação de empréstimos e cartões de crédito.

Aplicaremos o algorítimo de k-means para realizar uma análise de clusters entre esses fatores.

```
balanco_std <- scale(balanco[-1])
fit_km <- kmeans(balanco_std, 3)
fit_km$centers

##          age district_id      A11      A12      A13
## 1  0.0015     -1.38   1.99 -1.33 -1.37
## 2 -0.0130     -0.21  -0.43 -0.33 -0.31
## 3  0.0193      0.96 -0.27  1.13  1.11

plot(balanco, col = fit_km$cluster)
```



Em resumo, ao analisar esse último gráfico, parece haver formação de cluster apenas entre district_id e taxas de desemprego (A12 e A13), indicando que os distritos com identificação mais elevada apresentam maiores taxas de desemprego que os de identificação mais baixa. Essa informação poderia ser usada para a formação de uma escala de risco para classificação do cliente juntamente com outras informações como faixas de balanço, bom/mal pagador etc.

1.3 Conclusões

O trabalho explorou em profundidade o banco de dados das transações financeiras do banco CZECH. Foram feitos alguns uma limpeza de dados inicial, e plotagem de gráficos iniciais para ter alguma noção dos dados à mão. Em seguida, foram feitas análises estatísticas para separar clientes potenciais para novos produtos, como empréstimos ou cartão de crédito. Nas análises, focamos em comparações entre saldos médios, tipos de cartão de créditos vs saldo, volume de ordens bancárias de bons e maus pagadores. Concluímos o trabalho, com uma análise rápida comparando vários fatores ao mesmo tempo, usando correlação e análise de clusters através do uso de k-médias. Todos esses resultados podem ser utilizados pelo time do banco para escolha dos clientes potenciais para novos produtos.

Analise Exploratória de Dados com R

Trab. 02 - Estudo de Caso Czech Bank

Bruna Osti - A57026726

Euclides Ayala - A57277755

Luiz Fernando Pereira - A57188588

Prof. Gustavo Mirapalheta

MBA FGV - TBABD - T4 - Faria Lima

15/Out/2018

Estudo de Caso Czech Bank

1. Introdução

O **Czech Bank**, instituição financeira privada da República Tcheca, pretende executar um projeto de melhoria de seus serviços. Para guiar este projeto, ele deseja fazer a exploração e análise de sua base de dados de clientes, entendendo melhor o perfil de seus correntistas e planejando ações de acordo com estes perfis.

O conhecimento deste perfil de clientes envolve explorar as transações efetuadas por eles e verificar informações do volume de movimentações financeiras, quem são os bons e maus pagadores, demandas por empréstimos e cartões de crédito, distribuição geográfica/demográfica de clientes com diferentes perfis, etc. As ações a serem planejadas podem ser, por exemplo, oferta de produtos, direcionamento de negociação de dívidas, planos para oferta de cartões de crédito, distribuição de caixas eletrônicos, etc.

Neste estudo temos o objetivo de fazer a Análise Exploratória da base de dados do Czech Bank, lendo a base de dados fornecida em arquivos formato *CSV* e usando recursos da linguagem R, especialmente do *tidyverse*:

1.1 Organização do estudo de caso

- Importação de dados usando *readr* e *tibble*
- Organização e tratamento dos dados com *dplyr*, *lubridate*, *stringr* e *tidyverse*
- Análise da informação e inferência estatística
- Visualização de dados usando *ggplot2* e *ggmap*

Este documento foi feito usando *R Markdown*.

2. Carga e Preparação Inicial dos Dados

As tabelas da base de dados do Czech Bank foram enviadas para análise em arquivos *CSV* e importadas como data frames R com o suporte de *tibbles*. A importação inicial foi feita “*as is*”, para posterior tratamento e organização dos dados.

Obs.: os tipos de dados deste modelo foram inferidos a partir das informações disponíveis no descriptivo da base de dados (PDF) e na própria base Access. A nomenclatura dos tipos de dados segue o padrão Oracle pois usamos a ferramenta *Oracle Data Modeler* para desenhar o diagrama, e servem como referência apenas.

Foi usada a função *read_delim* do pacote *readr* para a leitura dos dados com mais desempenho e flexibilidade na definição do layout dos arquivos.

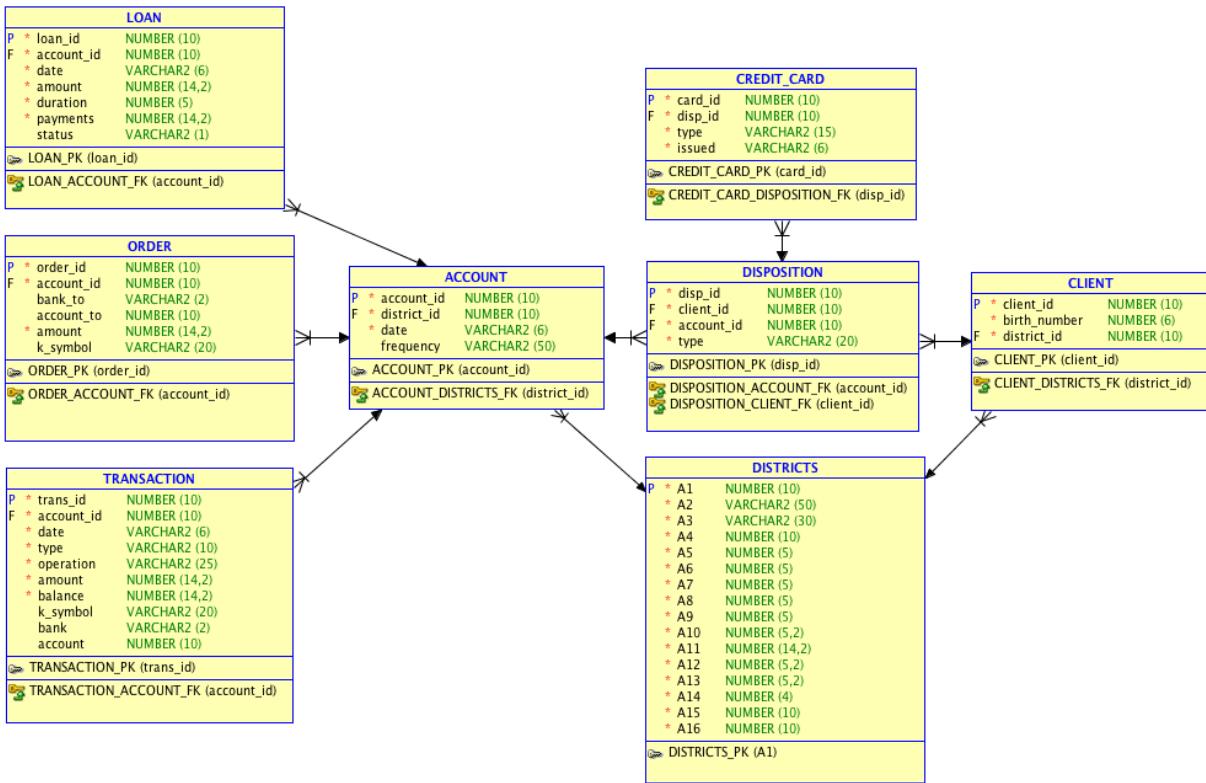


Figure 1: Modelo original da base de dados Czech Bank

```

# carregando datasets originais "as is"
# estes dataframes (ou melhor, tibbles) serão tratados e armazenados em
# outras variáveis
dados_dir <- "./dados"

faccounts      <- read_delim(paste(dados_dir, "account.asc", sep = "/"), delim = ";")
fcards         <- read_delim(paste(dados_dir, "card.asc", sep = "/"), delim = ";")
fcustomers     <- read_delim(paste(dados_dir, "client.asc", sep = "/"), delim = ";")
fdispositions  <- read_delim(paste(dados_dir, "disp.asc", sep = "/"), delim = ";")
fdistricts    <- read_delim(paste(dados_dir, "district.asc", sep = "/"), delim = ";")
floans         <- read_delim(paste(dados_dir, "loan.asc", sep = "/"), delim = ";")
forders        <- read_delim(paste(dados_dir, "order.asc", sep = "/"), delim = ";")
ftransactions  <- read_delim(paste(dados_dir, "trans.asc", sep = "/"), delim = ";")

# Importando dados de coordenadas e mapeando pontos de interesse inicial
# (principais cidades)
czech_coord <- read_delim(paste(dados_dir, "districts_geo.csv", sep = "/"), delim = ";")
head(czech_coord, 10)

## # A tibble: 10 x 5
##   district_id district_name   region       lat   long
##       <int>      <chr>      <chr>      <dbl>  <dbl>
## 1           1 Hl.m. Praha Prague      50.1  14.4
## 2           2 Benesov central Bohemia 49.8  14.7
## 3           3 Beroun central Bohemia 50.0  14.1
## 4           4 Kladno central Bohemia 50.1  14.1
## 5           5 Kolin  central Bohemia 50.0  15.2
## 6           6 Kutna Hora central Bohemia 50.0  15.3
## 7           7 Melnik central Bohemia 50.4  14.5
## 8           8 Mlada Boleslav central Bohemia 50.4  14.9
## 9           9 Nymburk central Bohemia 50.2  15.0
## 10          10 Praha - vychod central Bohemia 49.9  14.8

# dataframe para as coordenadas da capital
czech_coord_capital <- czech_coord %>%
  filter(district_name %in% c("Hl.m. Praha"))
# dataframe para as coordenadas das principais cidades fora a capital
czech_coord_principal <- czech_coord %>%
  filter(district_name %in% c("Plzen - mesto", "Brno - mesto", "Ostrava - mesto"))
# dataframe para as coordenadas das demais cidades
czech_coord_outras <- czech_coord %>%
  anti_join(czech_coord_capital, by = c("district_name", "region")) %>%
  anti_join(czech_coord_principal, by = c("district_name", "region"))

```

3. Preparação dos Data Frames

3.1 Organização da informação (*tidy* e *joins*)

Como esta é uma base de dados vinda de um sistema legado da antiga *URSS*, encontramos alguns problemas nas informações importadas dos arquivos CSV. Algumas destas informações estão **hardcoded** de acordo com padrões definidos nesta base e algumas colunas possuem dados *nulos* (**NA**) gravados de modo inconsistente. E também temos algumas colunas com nomes pouco descritivos ou repetidos.

Além disso, no *parsing* de alguns valores, o *readr* não conseguiu deduzir corretamente o tipo de dados de algumas variáveis (colunas). Aqui destacamos valores do tipo *dbl* (*double*, ou *decimal*) e *date*, que serão importantes na análise e que precisam ser convertidos para possibilitar seu uso em cálculos e agregações.

Tudo isso dificulta a leitura e análise da informação (dados “*machine-friendly*” e não “*human-friendly*”), então vamos limpar e organizar estes dados com *dplyr* para poder avançar na análise. Também vamos traduzir alguns valores das variáveis qualitativas que estão no idioma tcheco.

```
# funções de tradução de valores categóricos escritos em "tcheco"
translate_account_freq <- function(freq) {
  translation <- case_when(
    freq == "POPLATEK MESICNE" ~ "Mensal",
    freq == "POPLATEK TYDNE" ~ "Semanal",
    freq == "POPLATEK PO OBRATU" ~ "Imediato",
    TRUE ~ as.character(freq)
  )
  return(translation)
}

translate_order_k_symbol <- function(k_symbol) {
  translation <- case_when(
    k_symbol == "POJISTNE" ~ "Seguro",
    k_symbol == "SIPÓ" ~ "Financ. Imob.",
    k_symbol == "LEASING" ~ "Leasing",
    k_symbol == "UVER" ~ "Pag. Empréstimo",
    TRUE ~ as.character(k_symbol)
  )
  return(translation)
}

translate_trans_type <- function(trans_type) {
  translation <- case_when(
    trans_type == "PRIJEM" ~ "Crédito",
    trans_type == "VYDAJ" ~ "Débito",
    trans_type == "VYBER" ~ "Débito 2",
    TRUE ~ as.character(trans_type)
  )
  return(translation)
}

translate_trans_oper <- function(trans_oper) {
  translation <- case_when(
    trans_oper == "VYBER KARTOU" ~ "Saque no cartão",
    trans_oper == "VKLAD" ~ "Depósito em dinheiro",
    trans_oper == "PREVOD Z UCTU" ~ "Recebimento de banco",
    trans_oper == "VYBER" ~ "Saque em dinheiro",
    trans_oper == "PREVOD NA UCET" ~ "Transf. para banco",
    TRUE ~ as.character(trans_oper)
  )
  return(translation)
}

translate_trans_k_symbol <- function(k_symbol) {
  translation <- case_when(
    k_symbol == "POJISTNE" ~ "Pagamento Seguro",
```

```

    k_symbol == "SLUZBY"      ~ "Pagamento Boleto",
    k_symbol == "UROK"        ~ "Crédito Juros",
    k_symbol == "SANKC. UROK" ~ "Juros Cheque Especial",
    k_symbol == "SIPO"        ~ "Financ. Imob.",
    k_symbol == "DUCHOD"     ~ "Aposentadoria",
    k_symbol == "UVER"        ~ "Pag. Empréstimo",
    TRUE ~ as.character(k_symbol)
)
return(translation)
}

linhas_amostra <- 3

# Utilizando o transmute para ao mesmo tempo aplicar mutate (ajuste de valores) e rename
# das colunas, e retirando as colunas antigas sem processamento, "limpando" assim o dataframe.
#
# Também convertemos colunas de data para o tipo "date" (lubridate::ymd)
# dataframe importado
head(faccounts, linhas_amostra)

## # A tibble: 3 x 4
##   account_id district_id frequency      date
##       <int>      <int> <chr>      <int>
## 1         576          55 POPLATEK MESICNE 930101
## 2         3818          74 POPLATEK MESICNE 930101
## 3         704          55 POPLATEK MESICNE 930101

# organização do tibble
accounts <- faccounts %>%
  transmute(account_id,
            district_id,
            creation_date = ymd(date),
            frequency = translate_account_freq(frequency))

head(accounts, linhas_amostra)

## # A tibble: 3 x 4
##   account_id district_id creation_date frequency
##       <int>      <int> <date>      <chr>
## 1         576          55 1993-01-01 Mensal
## 2         3818          74 1993-01-01 Mensal
## 3         704          55 1993-01-01 Mensal

# Cartões de crédito, com problemas na coluna de data de emissão
# formato como horário "inútil" e tipo de dados incorreto
head(fcards, linhas_amostra)

## # A tibble: 3 x 4
##   card_id disp_id type issued
##       <int>  <int> <chr> <chr>
## 1     1005    9285 classic 931107 00:00:00
## 2      104     588 classic 940119 00:00:00
## 3      747    4915 classic 940205 00:00:00

# ajustando tipo de dados do campo data, organizando as substrings da data
# e convertendo o tipo de dados para date

```

```

credit_cards <- fcards %>%
  transmute(card_id,
            disp_id,
            card_type = type,
            issue_date = ymd(paste(str_sub(issued, 1, 2),
                                    str_sub(issued, 3, 4),
                                    str_sub(issued, 5, 6), sep = "")))

head(credit_cards, linhas_amostra)

## # A tibble: 3 x 4
##   card_id disp_id card_type issue_date
##       <int>    <int>   <chr>     <date>
## 1     1005      9285 classic  1993-11-07
## 2      104       588 classic  1994-01-19
## 3      747      4915 classic  1994-02-05

# Clientes
head(fcustomers, linhas_amostra)

## # A tibble: 3 x 3
##   client_id birth_number district_id
##       <int>        <int>        <int>
## 1         1          706213         18
## 2         2          450204         1
## 3         3          406009         1

# para clientes, vamos tratar a data do nascimento e extrair o gênero (sexo) do cliente
# de acordo com o documento do sistema legado:
# -- the number is in the form YYMMDD for men, the number is in the form YYMM+50DD for women,
# -- where YYMMDD is the date of birth
# Usamos mutate para cálculos intermediários e transmute para finalizar a limpeza
customers <- fcustomers %>%
  mutate(birth_year = as.integer(str_sub(birth_number, 1, 2)),
         birth_month = as.integer(str_sub(birth_number, 3, 4)),
         birth_day = str_sub(birth_number, 5, 6)) %>%
  mutate(gender = ifelse((birth_month >= 51 & birth_month <= 62),
                        "F",
                        "M")) %>%
  mutate(birth_month = str_pad(ifelse((birth_month >= 51 & birth_month <= 62),
                                       birth_month - 50,
                                       birth_month), 2, "left", "0")) %>%
  mutate(birth_year = as.character(birth_year + 1900)) %>%
  transmute(customer_id = client_id,
            district_id,
            gender,
            birth_date = ymd(paste(birth_year, birth_month, birth_day, sep = "")))

head(customers, linhas_amostra)

## # A tibble: 3 x 4
##   customer_id district_id gender birth_date
##       <int>        <int>   <chr>     <date>
## 1           1          18 F  1970-12-13
## 2           2          1 M  1945-02-04

```

```

## 3           3       1 F      1940-10-09
# "Dispositions"
head(fdispositions, linhas_amostra)

## # A tibble: 3 x 4
##   disp_id client_id account_id type
##       <int>     <int>      <int> <chr>
## 1         1         1          1 OWNER
## 2         2         2          2 OWNER
## 3         3         3          2 DISPONENT

# o dataframe que faz o papel de "tabela de-para" está ok, precisa de
# ajuste de nomes de colunas apenas
dispositions <- fdispositions %>%
  transmute(disp_id,
            customer_id = client_id,
            account_id,
            disp_type = type)

head(dispositions, linhas_amostra)

## # A tibble: 3 x 4
##   disp_id customer_id account_id disp_type
##       <int>     <int>      <int> <chr>
## 1         1         1          1 OWNER
## 2         2         2          2 OWNER
## 3         3         3          2 DISPONENT

# Distritos, ou dados Demográficos
head(fdistricts, linhas_amostra)

## # A tibble: 3 x 16
##   A1 A2 A3      A4 A5 A6 A7 A8 A9 A10 A11 A12
##   <int> <chr> <chr> <int> <int> <int> <int> <dbl> <int> <chr>
## 1     1 Hl.m~ Prag~ 1.20e6    0    0    0    1    1 100 12541 0.29
## 2     2 Bene~ cent~ 8.89e4   80   26    6    2    5 46.7 8507 1.67
## 3     3 Bero~ cent~ 7.52e4   55   26    4    1    5 41.7 8980 1.95
## # ... with 4 more variables: A13 <dbl>, A14 <int>, A15 <chr>, A16 <int>

# Vamos renomear as colunas com códigos para nomes significativos,
# nomeando as colunas de informações demográficas e ajustando alguns tipos de dados
# Também encontramos valores desconhecidos gravados por algum operador como "?"
# Vamos substituir valores desconhecidos por NA (null)
districts <- fdistricts %>%
  inner_join(czech_coord, by = c("A1" = "district_id")) %>%
  transmute(district_id = A1,
            district_name = A2,
            region = A3,
            num_inhab = A4,
            num_mun_inhab_0_499 = A5,
            num_mun_inhab_500_1999 = A6,
            num_mun_inhab_2000_9999 = A7,
            num_mun_inhab_10000 = A8,
            num_cities = A9,
            urban_ratio = A10,
            avg_salary = as.double(A11),

```

```

    unemployment_1995 = as.double(ifelse(A12 == "?", NA, A12)),
    unemployment_1996 = as.double(A13),
    num_enterpr_1000 = A14,
    num_crimes_1995 = as.integer(ifelse(A15 == "?", NA, A15)),
    num_crimes_1996 = as.integer(A16),
    lat,
    long
  )

head(districts, linhas amostra)

## # A tibble: 3 x 18
##   district_id district_name region num_inhab num_mun_inhab_0~
##       <int>      <chr>     <chr>      <int>          <int>
## 1           1 Hl.m. Praha Prague    1204953            0
## 2           2 Benesov    centr~    88884             80
## 3           3 Beroun    centr~    75232             55
## # ... with 13 more variables: num_mun_inhab_500_1999 <int>,
## #   num_mun_inhab_2000_9999 <int>, num_mun_inhab_10000 <int>,
## #   num_cities <int>, urban_ratio <dbl>, avg_salary <dbl>,
## #   unemployment_1995 <dbl>, unemployment_1996 <dbl>,
## #   num_enterpr_1000 <int>, num_crimes_1995 <int>, num_crimes_1996 <int>,
## #   lat <dbl>, long <dbl>

# Empréstimos
head(floans, linhas amostra)

## # A tibble: 3 x 7
##   loan_id account_id   date amount duration payments status
##       <int>      <int> <int>   <int>    <dbl> <chr>
## 1     5314      1787 930705  96396      12    8033 B
## 2     5316      1801 930711 165960      36    4610 A
## 3     6863      9188 930728 127080      60    2118 A

# ajuste de tipo de dados *data* e *double*
loans <- floans %>%
  transmute(loan_id,
            account_id,
            loan_date = ymd(date),
            loan_amount = as.double(amount),
            duration,
            payments,
            status)

head(loans, linhas amostra)

## # A tibble: 3 x 7
##   loan_id account_id loan_date loan_amount duration payments status
##       <int>      <int>   <date>      <dbl>    <int>    <dbl> <chr>
## 1     5314      1787 1993-07-05     96396      12    8033 B
## 2     5316      1801 1993-07-11    165960      36    4610 A
## 3     6863      9188 1993-07-28    127080      60    2118 A

# Permanent Orders
head(forders, linhas amostra)

```

```

## # A tibble: 3 x 6
##   order_id account_id bank_to account_to amount k_symbol
##       <int>      <int> <chr>     <int>   <dbl> <chr>
## 1    29401          1 YZ        87144583  2452   SIPO
## 2    29402          2 ST        89597016  3373.  UVER
## 3    29403          2 QR       13943797  7266   SIPO
# tratando k_symbol com espaço em branco, trocando por NA (null)
orders <- forders %>%
  transmute(order_id,
            account_id,
            bank_to,
            account_to,
            order_amount = amount,
            order_k_symbol = translate_order_k_symbol(ifelse(k_symbol == " ", NA, k_symbol)))

head(orders, linhas_amostra)

## # A tibble: 3 x 6
##   order_id account_id bank_to account_to order_amount order_k_symbol
##       <int>      <int> <chr>     <int>       <dbl> <chr>
## 1    29401          1 YZ        87144583    2452 Financ. Imob.
## 2    29402          2 ST        89597016   3373. Pag. Empréstimo
## 3    29403          2 QR       13943797   7266 Financ. Imob.

# Transações
head(ftransactions, linhas_amostra)

## # A tibble: 3 x 10
##   trans_id account_id date type operation amount balance k_symbol bank
##       <int>      <int> <int> <chr> <chr>   <dbl>   <dbl> <chr>   <chr>
## 1    695247        2378 930101 PRIJ~ VKLAD      700     700 <NA>   <NA>
## 2    171812        576  930101 PRIJ~ VKLAD      900     900 <NA>   <NA>
## 3    207264        704  930101 PRIJ~ VKLAD     1000    1000 <NA>   <NA>
## # ... with 1 more variable: account <int>

# A maior tabela da base possui problemas na coluna k_symbol, com alta ocorrência
# de espaços em branco no lugar de NA (~53k)
ftransactions %>%
  group_by(k_symbol) %>%
  count()

## # A tibble: 9 x 2
## # Groups:   k_symbol [9]
##   k_symbol     n
##   <chr>     <int>
## 1 " "       53433
## 2 DUCHOD    30338
## 3 POJISTNE  18500
## 4 SANKC. UROK  1577
## 5 SIPO      118065
## 6 SLUZBY    155832
## 7 UROK      183114
## 8 UVER      13580
## 9 <NA>      481881

```

```

# Ajustando tipo de dados "data" e tratando k_symbol com espaço em branco,
# consolidando todos os valores ausentes como NA (null)
transactions <- ftransactions %>%
  transmute(trans_id,
            account_id,
            trans_date = ymd(date),
            trans_type = translate_trans_type(type),
            operation = translate_trans_oper(operation),
            trans_amount = amount,
            balance,
            trans_k_symbol = translate_trans_k_symbol(ifelse(k_symbol == " ", NA, k_symbol)),
            trans_bank = bank,
            trans_account = account)

head(transactions, linhas amostra)

## # A tibble: 3 x 10
##   trans_id account_id trans_date trans_type operation trans_amount balance
##       <int>      <int>    <date>     <chr>        <chr>        <dbl>      <dbl>
## 1     695247        2378 1993-01-01 Crédito Depósito~      700       700
## 2     171812        576  1993-01-01 Crédito Depósito~      900       900
## 3     207264        704  1993-01-01 Crédito Depósito~     1000      1000
## # ... with 3 more variables: trans_k_symbol <chr>, trans_bank <chr>,
## #   trans_account <int>

```

O dataframe com as informações demográficas possui colunas que podem ser melhoradas com o uso de *tidyverse*. Vamos fazer a *normalização* dos dados demográficos usando *gather*.

```

# "normalizando" dados de número de municipalidades por *ranges* de habitantes
district_inhab_mun <- districts %>%
  gather(num_mun_inhab_0_499,
         num_mun_inhab_500_1999,
         num_mun_inhab_2000_9999,
         num_mun_inhab_10000,
         key = "range_inhab_mun",
         value = "num_mun") %>%
  select(district_id, district_name, range_inhab_mun, num_mun) %>%
  mutate(range_inhab_mun = str_sub(range_inhab_mun, 15)) %>%
  mutate(range_inhab_mun = str_replace_all(range_inhab_mun, "_", "-")) %>%
  mutate(range_inhab_mun = str_replace_all(range_inhab_mun, "10000", "10000\\+")) %>%
  arrange(district_id, district_name)

head(district_inhab_mun, linhas amostra)

## # A tibble: 3 x 4
##   district_id district_name range_inhab_mun num_mun
##       <int>      <chr>        <chr>        <int>
## 1           1 Hl.m. Praha  0-499                 0
## 2           1 Hl.m. Praha  500-1999               0
## 3           1 Hl.m. Praha  2000-9999              0
# fazendo o gather de dados de desemprego por ano
district_unemployment <- districts %>%
  gather(unemployment_1995,

```

```

    unemployment_1996,
    key = "unemployment_year",
    value = "unemployment_rate") %>%
select(district_id, district_name, unemployment_year, unemployment_rate) %>%
mutate(unemployment_year = str_sub(unemployment_year, 14)) %>%
arrange(district_id, district_name)

head(district_unemployment, linhas amostra)

## # A tibble: 3 x 4
##   district_id district_name unemployment_year unemployment_rate
##       <int>     <chr>          <chr>              <dbl>
## 1           1 Hl.m. Praha    1995            0.290
## 2           1 Hl.m. Praha    1996            0.43
## 3           2 Benesov        1995            1.67

# gather de números de crimes por ano
district_crimes <- districts %>%
  gather(num_crimes_1995,
        num_crimes_1996,
        key = "crime_year",
        value = "num_crimes") %>%
select(district_id, district_name, crime_year, num_crimes) %>%
mutate(crime_year = str_sub(crime_year, 12)) %>%
arrange(district_id, district_name)

head(district_crimes, linhas amostra)

## # A tibble: 3 x 4
##   district_id district_name crime_year num_crimes
##       <int>     <chr>      <chr>        <int>
## 1           1 Hl.m. Praha  1995        85677
## 2           1 Hl.m. Praha  1996        99107
## 3           2 Benesov      1995        2159

```

Após a limpeza e organização dos dados temos os dataframes que servirão de base para a análise.

Obs.: os tipos de dados deste modelo seguem o padrão Oracle por conta da ferramenta usada para desenhá-lo e servem como referência apenas. Deste ponto em diante consideraremos os tipos de dados definidos para dataframes R, como *int*, *dbl*, *chr* e *date*.

4. Análise de Variáveis

4.1. Desempenho do banco

A base de dados fornece informações sobre os produtos, **contas**, **empréstimos** e **cartões de crédito**. Também temos os dados das **transações bancárias** efetuadas no período. Verificando o volume de transações, criação de contas, quantidade de empréstimos e quantidade de cartões de crédito ano a ano, podemos perceber que a partir do ano de 1995 houve o aumento significativo das atividades bancárias, conforme demonstrados nos gráficos abaixo.

```

lsize <- 1.0

plot_acc_ano <- accounts %>%
  group_by(ano = year(creation_date)) %>%

```

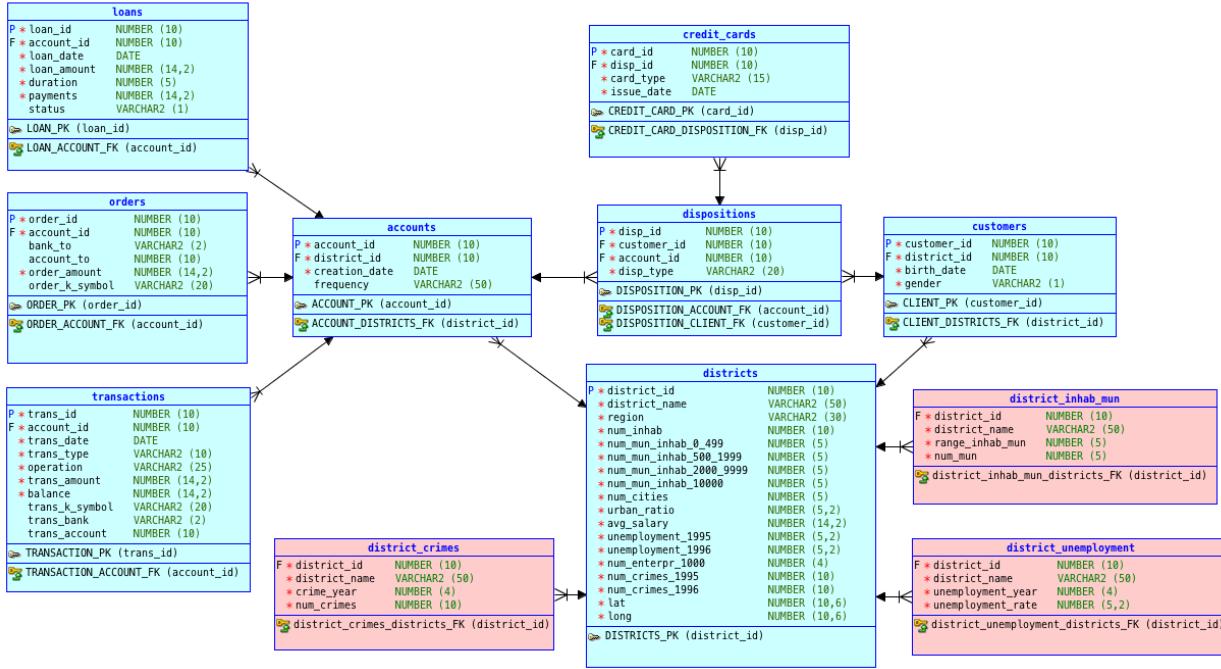


Figure 2: Dataframes tratados para análise do caso Czech Bank

```

summarize(contas_criadas = n()) %>%
ggplot(aes(x = ano, y = contas_criadas)) +
  geom_line(size = lsize, color = "blue") +
  labs(x = "Ano", y = "Contas abertas / ano") +
  scale_y_continuous(labels = function(x) format(x, scientific = FALSE),
                     limits = c(0, 1500)) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1),
        legend.position = "none")

plot_loans_ano <- loans %>%
  group_by(ano = year(loan_date)) %>%
  summarize(emprestimos_ano = n()) %>%
  ggplot(aes(x = ano, y = emprestimos_ano)) +
  geom_line(size = lsize, color = "red") +
  labs(x = "Ano", y = "Emprestimos / ano") +
  scale_y_continuous(labels = function(x) format(x, scientific = FALSE),
                     limits = c(0, 200)) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1),
        legend.position = "none")

plot_cartoes_ano <- credit_cards %>%
  group_by(ano = year(issue_date), card_type) %>%
  summarize(cartoes_ano = n()) %>%
  ggplot(aes(x = ano, y = cartoes_ano)) +
  geom_line(aes(color = card_type), size = lsize) +
  labs(x = "Ano", y = "Cartões / ano", color = "Tipo cartão") +
  scale_y_continuous(labels = function(x) format(x, scientific = FALSE),
                     limits = c(0, 350)) +

```

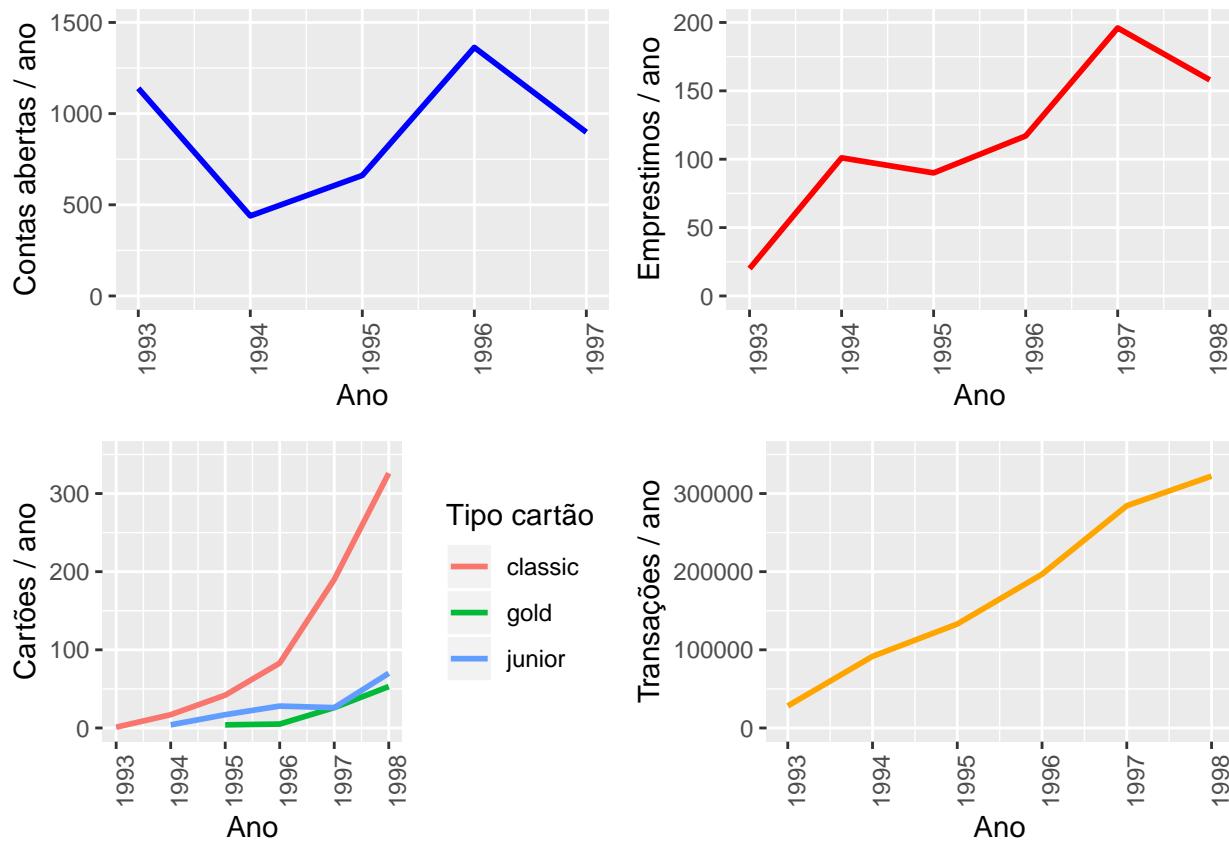
```

theme(axis.text.x = element_text(angle = 90, hjust = 1))

plot_transacoes_ano <- transactions %>%
  group_by(ano = year(trans_date)) %>%
  summarize(transacoes_ano = n()) %>%
  ggplot(aes(x = ano, y = transacoes_ano)) +
  geom_line(color = "orange", size = lsize) +
  labs(x = "Ano", y = "Transações / ano") +
  scale_y_continuous(labels = function(x) format(x, scientific = FALSE),
                     limits = c(0, 350000)) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1),
        legend.position = "none")

grid.arrange(plot_acc_ano, plot_loans_ano,
            plot_cartoes_ano, plot_transacoes_ano,
            nrow = 2, ncol = 2)

```



Distribuição dos produtos vendidos pelo banco.

```

cust_orders <- orders %>%
  inner_join(accounts, by = "account_id") %>%
  inner_join(dispositions, by = "account_id") %>%
  inner_join(customers, by = "customer_id") %>%
  filter(disp_type == "OWNER")

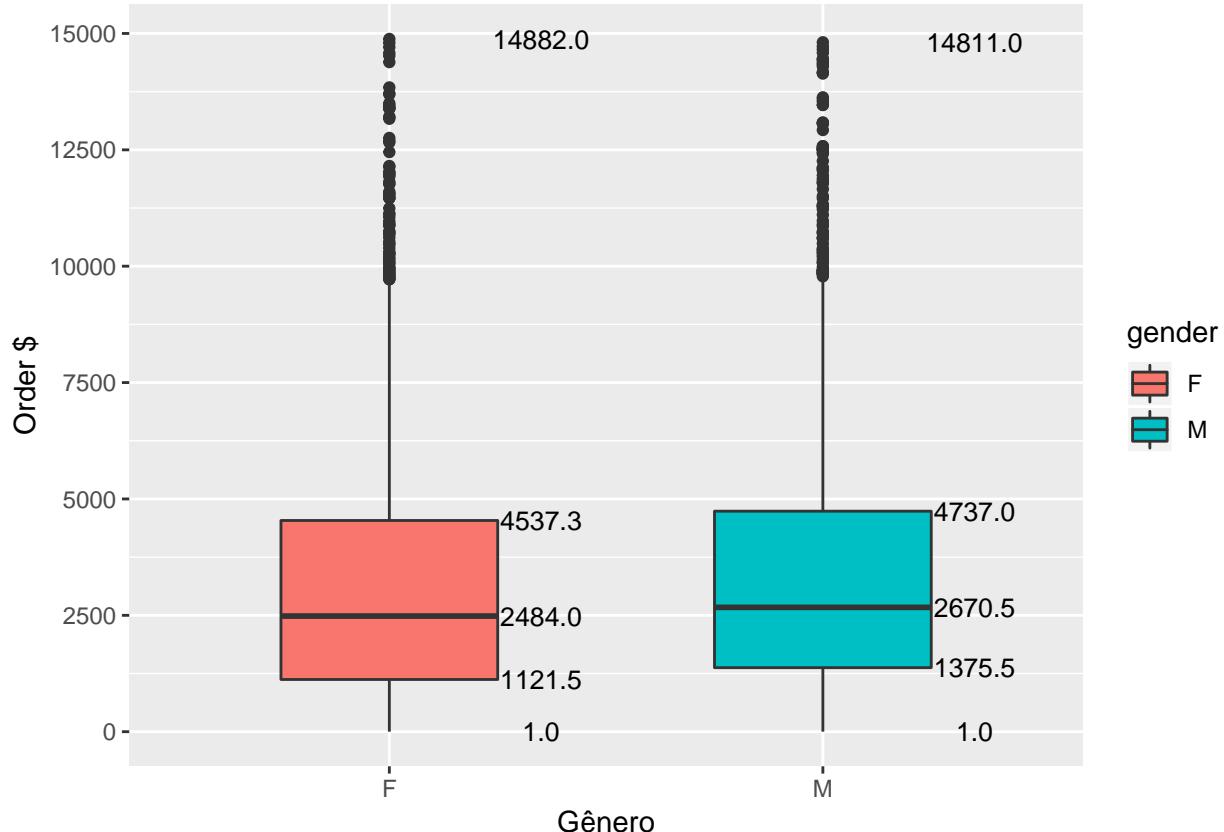
ggplot(cust_orders, aes(x = gender, y = order_amount, fill = gender)) +
  geom_boxplot(width = 0.5) +

```

```

stat_summary(geom="text", fun.y=quantile,
             aes(label=sprintf("%1.1f", ..y..)),
             position=position_nudge(x = 0.35), size = 3.5) +
labs(x = "Gênero", y = "Order $") +
scale_y_continuous(breaks = seq(0, 15000, 2500))

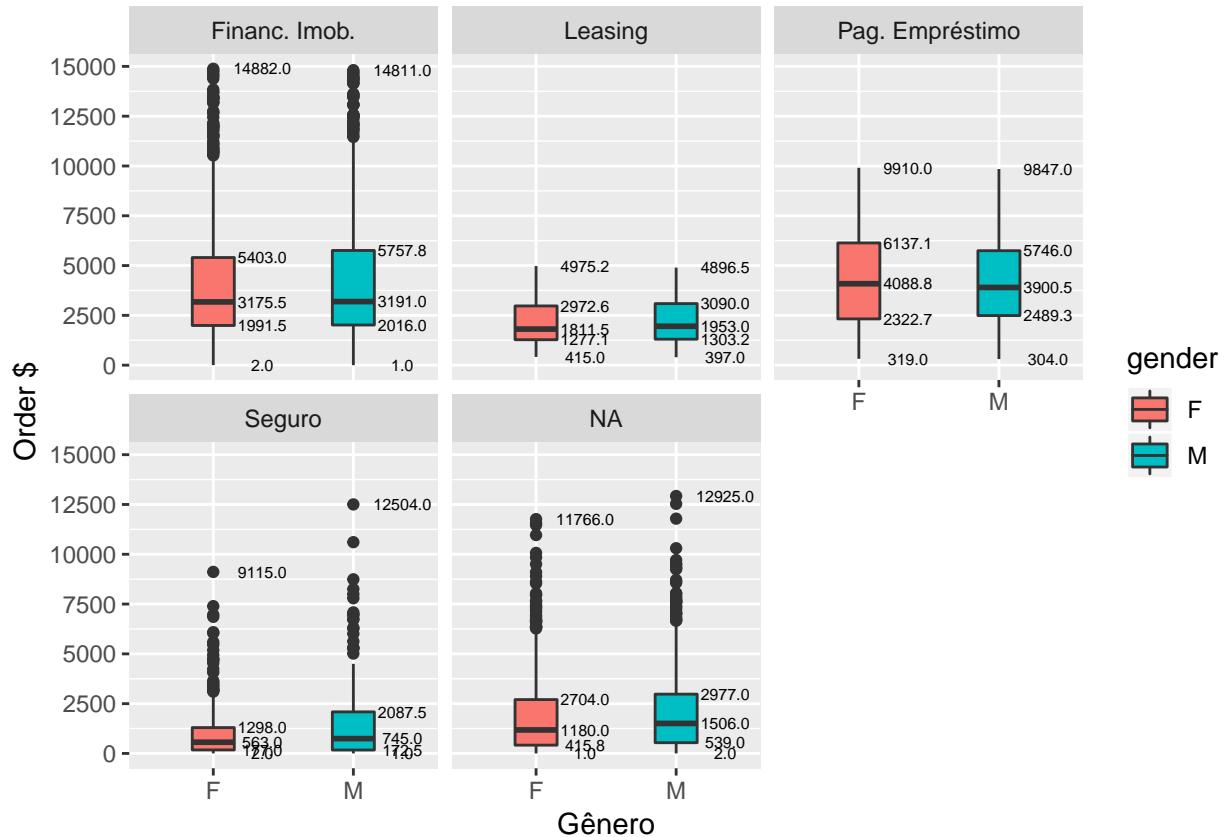
```



```

ggplot(cust_orders, aes(x = gender, y = order_amount, fill = gender)) +
  geom_boxplot(width = 0.3) +
  stat_summary(geom="text", fun.y=quantile,
               aes(label=sprintf("%1.1f", ..y..)),
               position=position_nudge(x = 0.35), size = 2) +
  facet_wrap(~ order_k_symbol) +
  labs(x = "Gênero", y = "Order $") +
  scale_y_continuous(breaks = seq(0, 15000, 2500))

```



Observando os dados disponíveis, esse estudo de caso irá focar em dois serviços principais: cartão de crédito e empréstimos.

4.2. Análise das Transações

Mapa da movimentação financeira do banco, no qual vemos a concentração de recursos na capital Praga e no leste da República Tcheca.

```
t <- transactions %>%
  inner_join(accounts, by = "account_id") %>%
  inner_join(dispositions, by = "account_id") %>%
  #filter(disp_type == "OWNER") %>%
  inner_join(districts, by = "district_id")
#%>%
#mutate(idade = 1996 - year(birth_date))

# top 10 por numero de transacoes
num_trans <- t %>%
  group_by(district_name, lat, long) %>%
  summarize(num_transacoes = n()) %>%
  arrange(desc(num_transacoes))

head(num_trans, 10)

## # A tibble: 10 x 4
## # Groups:   district_name, lat [10]
##   district_name     lat   long num_transacoes
##   <chr>           <dbl> <dbl>        <dbl>
## 1 Prague          50.1  14.4         10000
## 2 Prague          50.1  14.4         8000
## 3 Prague          50.1  14.4         7000
## 4 Prague          50.1  14.4         6000
## 5 Prague          50.1  14.4         5000
## 6 Prague          50.1  14.4         4000
## 7 Prague          50.1  14.4         3000
## 8 Prague          50.1  14.4         2000
## 9 Prague          50.1  14.4         1500
## 10 Prague          50.1  14.4         1000
```

```

##      <chr>          <dbl> <dbl>     <int>
## 1 Hl.m. Praha      50.1  14.4    160444
## 2 Karvina         49.9  18.5    42139
## 3 Ostrava - mesto 49.8  18.2    41659
## 4 Brno - mesto    49.2  16.6    34048
## 5 Zlin            49.2  17.7    27545
## 6 Olomouc         49.6  17.3    25485
## 7 Frydek - Mistek 49.7  18.4    24571
## 8 Zdar nad Sazavou 49.6  15.9    17354
## 9 Kolin           50.0  15.2    17343
## 10 Usti nad Orlici 50.0  16.4    16830

# top 10 por total de movimentacoes, em milhões de $
montante_trans <- t %>%
  group_by(district_name, lat, long) %>%
  summarize(montante_M = sum(trans_amount)/1000000) %>%
  arrange(desc(montante_M))

head(montante_trans, 10)

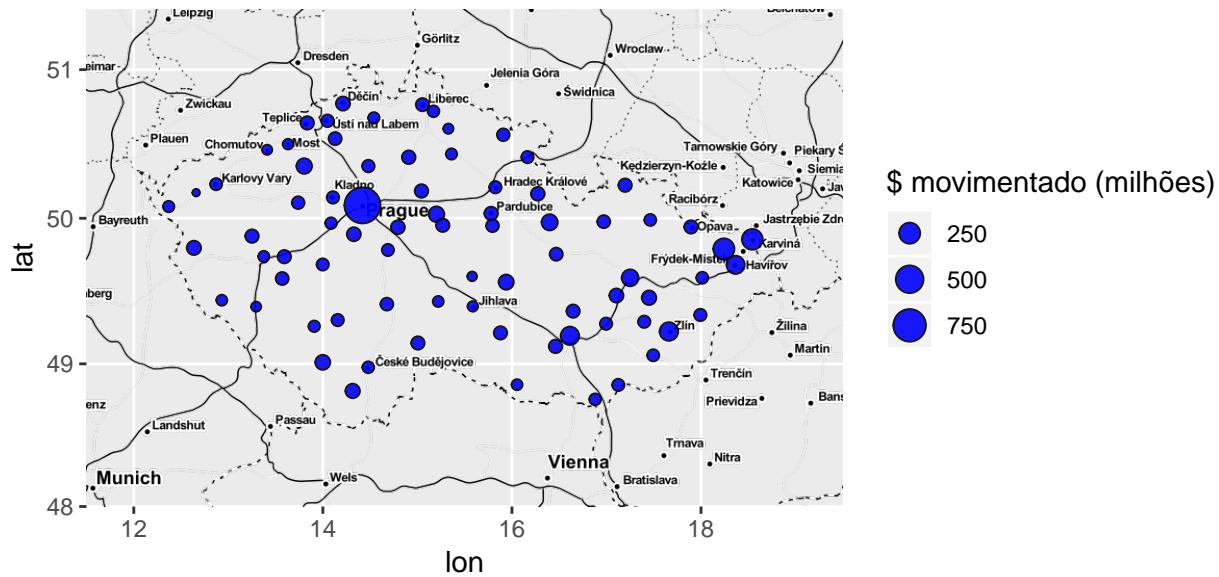
## # A tibble: 10 x 4
## # Groups:   district_name, lat [10]
##   district_name      lat   long montante_M
##   <chr>          <dbl> <dbl>     <dbl>
## 1 Hl.m. Praha      50.1  14.4     959.
## 2 Ostrava - mesto  49.8  18.2     251.
## 3 Karvina         49.9  18.5     239.
## 4 Zlin            49.2  17.7     181.
## 5 Brno - mesto    49.2  16.6     178.
## 6 Frydek - Mistek 49.7  18.4     171.
## 7 Olomouc         49.6  17.3     142.
## 8 Usti nad Orlici 50.0  16.4     128.
## 9 Kolin           50.0  15.2     119.
## 10 Louny          50.4  13.8     117.

# Mapa de distribuicao das transacoes
# Plotando mapa usando ggmap (https://github.com/dkahle/ggmap)
map <- get_stamenmap(bbox = c(left = 11.5, bottom = 48.0, right = 19.5, top = 51.4),
                      zoom = 7, maptype = "toner-hybrid")

plottrans <- ggmap(map) +
  geom_point(data = montante_trans,
             aes(x = long, y = lat, size = montante_M),
             shape = 21, alpha = 0.9, fill = "blue") +
  labs(size = "$ movimentado (milhões)")

plottrans

```



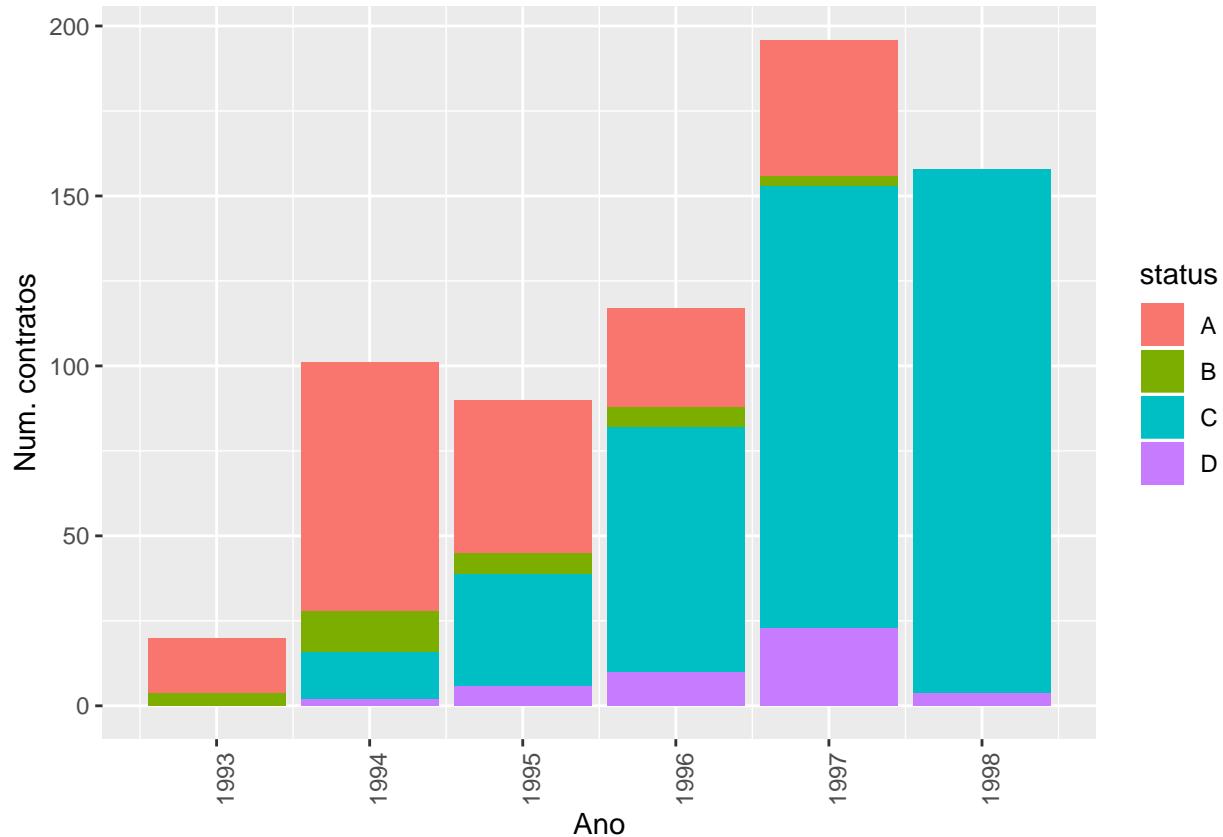
4.3. Análise de Empréstimos: distribuições dos *Loans*

Primeiramente, identificaremos a quantidade de contratos com empréstimo e quais categorias eles pertencem. O Objetivo é identificar se o banco possui empréstimos não pagos. Segundo a classificação do banco, temos 4 categorias de contratos:

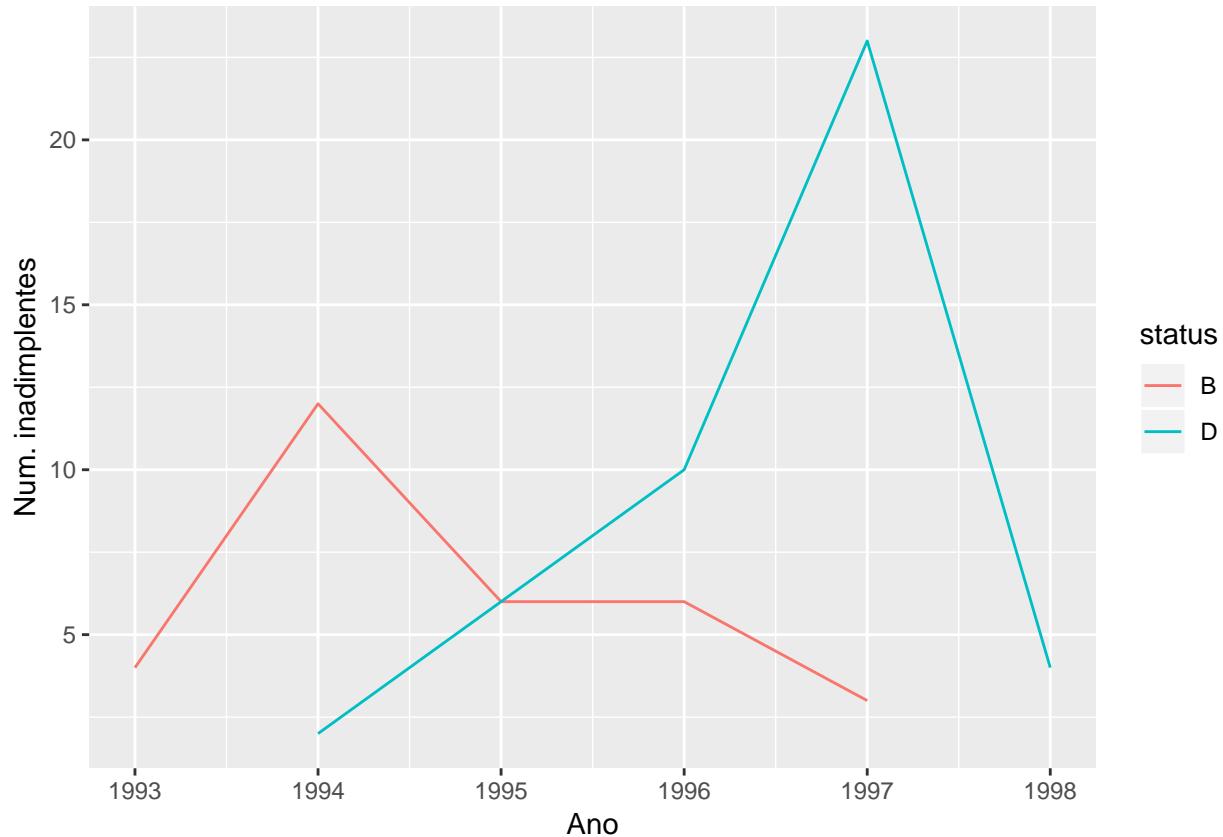
- **A** - Contratos finalizados e sem problemas
 - **B** - Contrato finalizado e sem pagamento
 - **C** - Contratos vigentes e com pagamento em dia
 - **D** - Contrato vigente e sem pagamento

Classificamos os clientes das categorias **B** e **D** como **inadimplentes**. Segundo o gráfico, a maior parte dos contratos estão na categoria C, ainda estão em vigência e sem pagamentos atrasados. Entre os inadimplentes temos um aumento durante os anos de 1996 e 1997, mas com redução dos contratos finalizados e sem pagamentos.

```
# contratos de loan por ano/status
ggplot(loans) +
  geom_bar(aes(x = year(loan_date), fill = status)) +
  labs(x = "Ano", y = "Num. contratos") +
  scale_x_continuous(breaks = seq(1993, 1998, 1)) +
  scale_y_continuous(labels = function(x) format(x, scientific = FALSE)) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



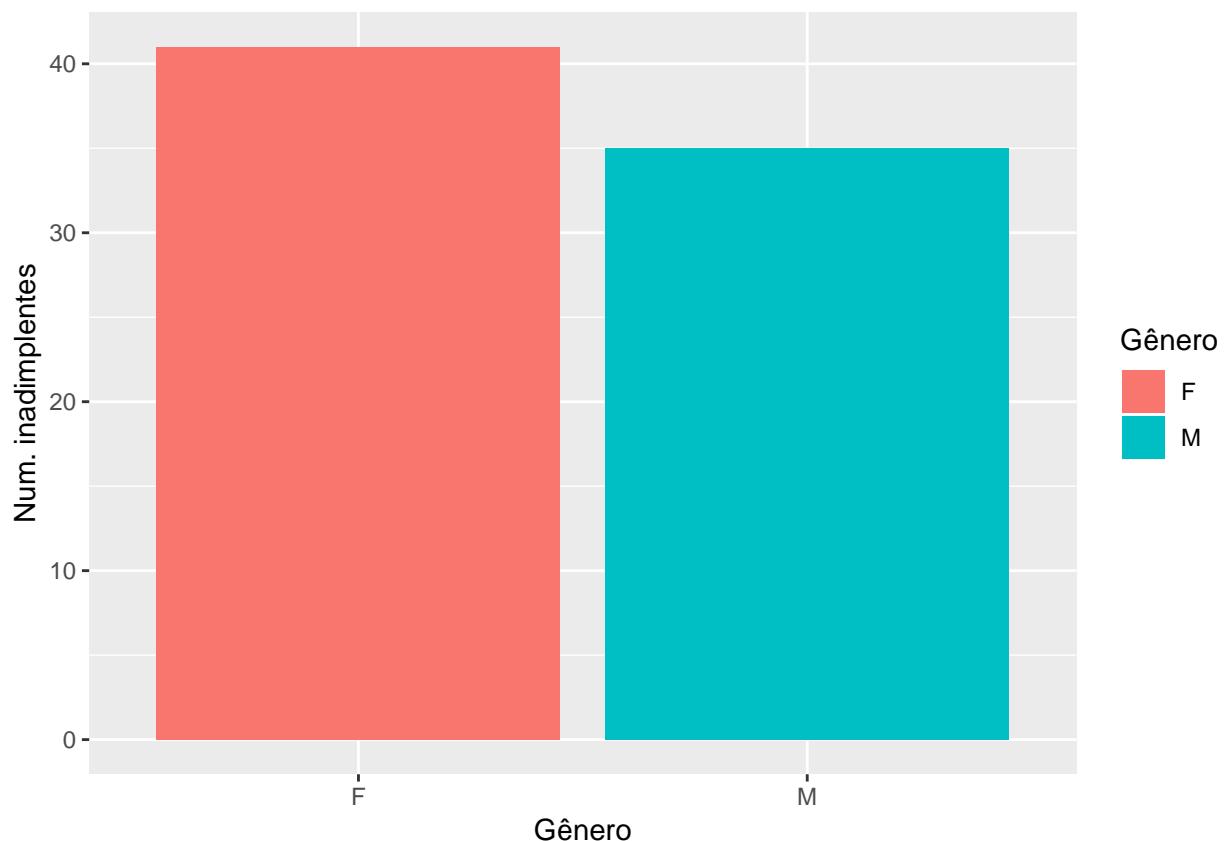
```
# inadimplentes
loans %>%
  filter(status %in% c("B", "D")) %>%
  group_by(ano = year(loan_date), status) %>%
  summarize(num_inadimplentes = n()) %>%
  ggplot() +
  geom_line(aes(x = ano, y = num_inadimplentes, color = status)) +
  labs(x = "Ano", y = "Num. inadimplentes")
```



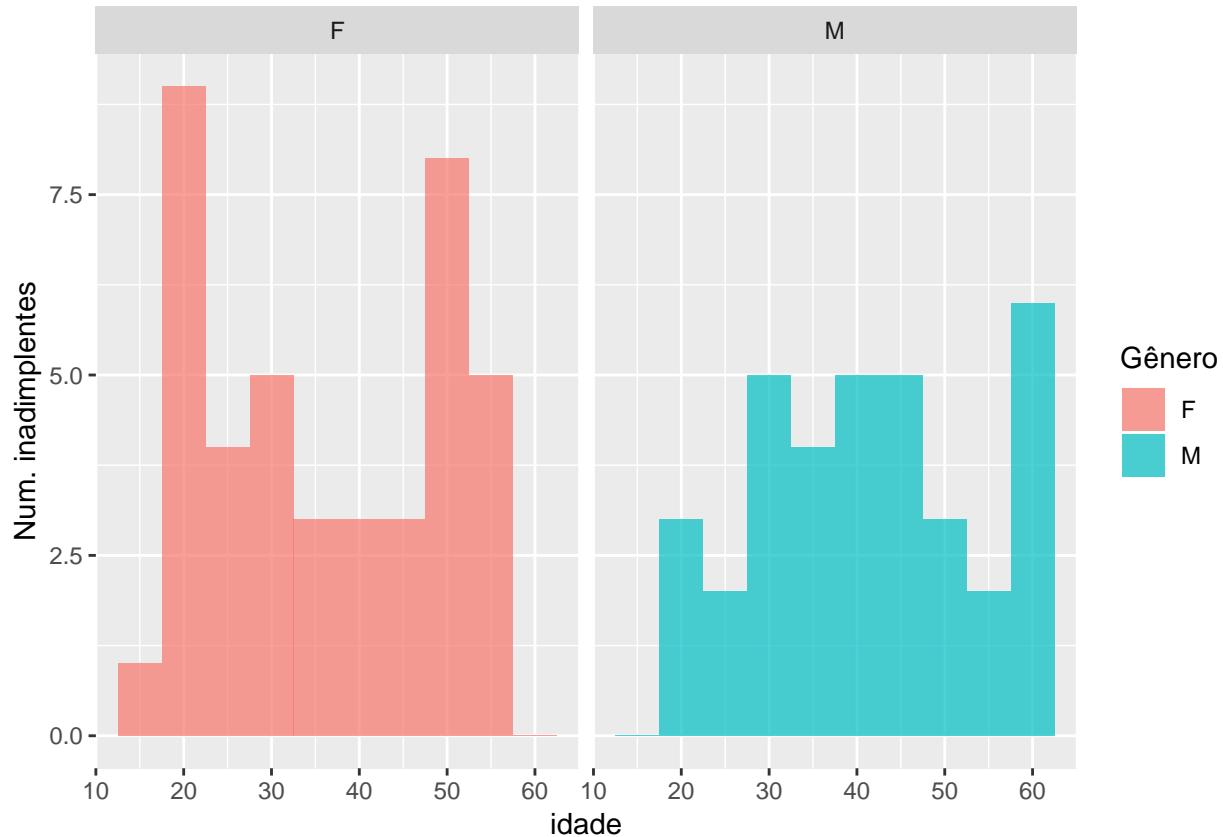
```
# inadimplentes por gênero
loans_cust <- loans %>%
  inner_join(accounts, by = "account_id") %>%
  inner_join(dispositions, by = "account_id") %>%
  inner_join(customers, by = "customer_id") %>%
  inner_join(districts, by = c("district_id.x" = "district_id")) %>%
  mutate(idade = 1996 - year(birth_date))

inad <- loans_cust %>%
  filter(status %in% c("B", "D"))

ggplot(inad) +
  geom_bar(aes(x = gender, fill = gender)) +
  scale_fill_discrete(name = "Gênero") +
  labs(x = "Gênero", y = "Num. inadimplentes")
```



```
ggplot(inad) +  
  geom_histogram(aes(x = idade, fill = gender), bins = 10, alpha = .7) +  
  facet_wrap(~ gender) +  
  scale_fill_discrete(name = "Gênero") +  
  labs(y = "Num. inadimplentes")
```



Descrição do perfil: Mulheres mais inadimplentes quando estão na faixa etária de 20 e de 50 anos. Entre os homens, os mais inadimplentes estão na faixa etária dos 60 anos.

Visualizando inadimplentes por cidade e cruzando esta informação com o salário médio, vemos que a capital Praga concentra o maior nível de inadimplência, mesmo tendo o melhor salário médio. Também verificamos que ao redor de Praga temos um salário médio maior que em outras regiões, mas com um nível de inadimplência baixo.

```
inad_map <- inad %>%
  group_by(district_name, lat, long, avg_salary) %>%
  summarize(num_inad = n())

# inadimplentes
inad_status <- inad %>%
  group_by(status) %>%
  summarize(num_contratos = n())

inad_status

## # A tibble: 2 x 2
##   status num_contratos
##   <chr>        <int>
## 1 B            31
## 2 D            45

loans_cust_total <- loans_cust %>%
  count()
```

```

inad_status_total <- loans_cust_total$n
inad_status_BD <- sum(inad_status$num_contratos)

# percentual de empréstimos inadimplentes
inad_status_BD / inad_status_total * 100

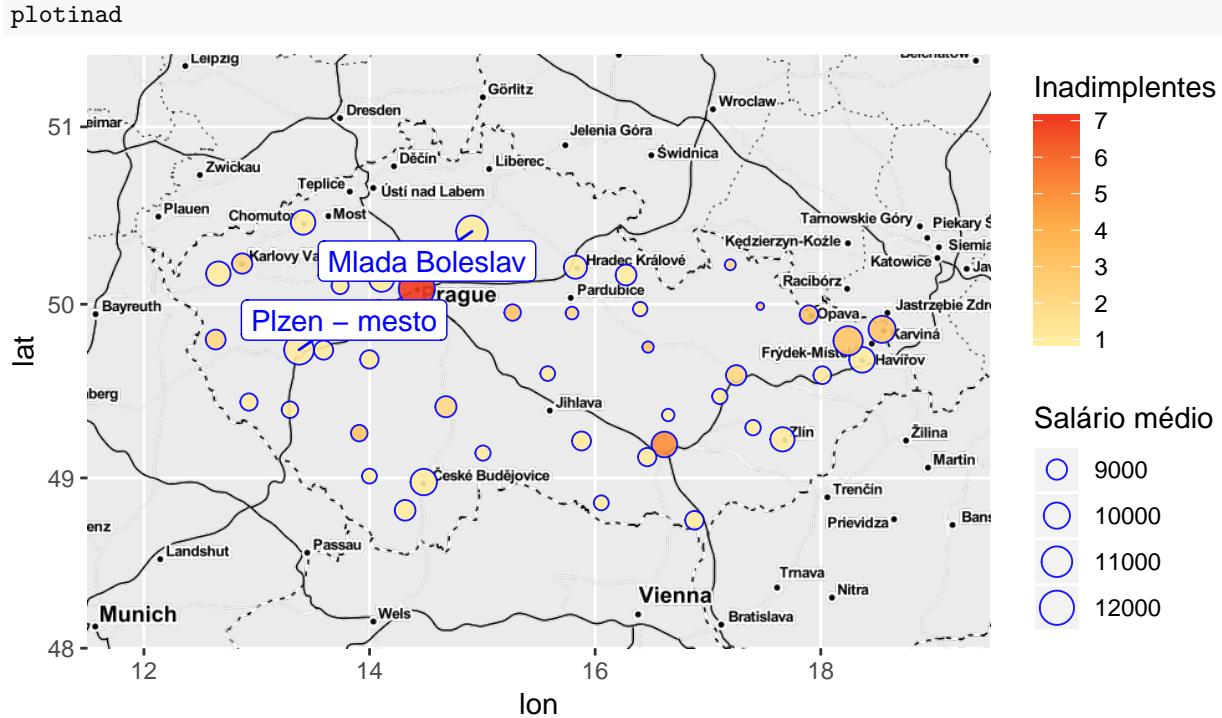
## [1] 9.189843

# Inadimplentes por Distrito
# cidades de interesse para análise de inadimplência
czech_coord_inad <- czech_coord %>%
  filter(district_name %in% c("Plzen - mesto", "Mlada Boleslav"))

plotinad <- ggmap(map) +
  geom_point(data = inad_map,
             aes(x = long, y = lat, size = avg_salary, fill = num_inad),
             shape = 21, alpha = 0.9, color = "blue") +
  labs(fill = "Inadimplentes", size = "Salário médio") +
  scale_fill_gradientn(colors = brewer.pal(3, "YlOrRd")) +
  geom_label_repel(data = czech_coord_inad,
                    aes(x = long, y = lat, label = district_name),
                    color = "blue",
                    min.segment.length = 0)

plotinad

```



Ao analisar os gráficos e o mapa, conclui-se que o número de inadimplentes é de 76 que representa aproximadamente 9% das atividades bancárias de empréstimos. Ou seja, as pessoas que moram em Praga, com salário médio de aproximadamente \$12.000,00, tendem a apresentar maior risco de inadimplência.

Para pessoas que possuem esse perfil identificado, o banco deverá ser mais rigoroso ao ceder empréstimos, especialmente na capital Praga.

Já em cidades como Pilsen e Mlada Boleslav, há um alto salário médio e um baixo índice de inadimplência.

Com base neste dado o banco pode melhorar seu serviço de empréstimo.

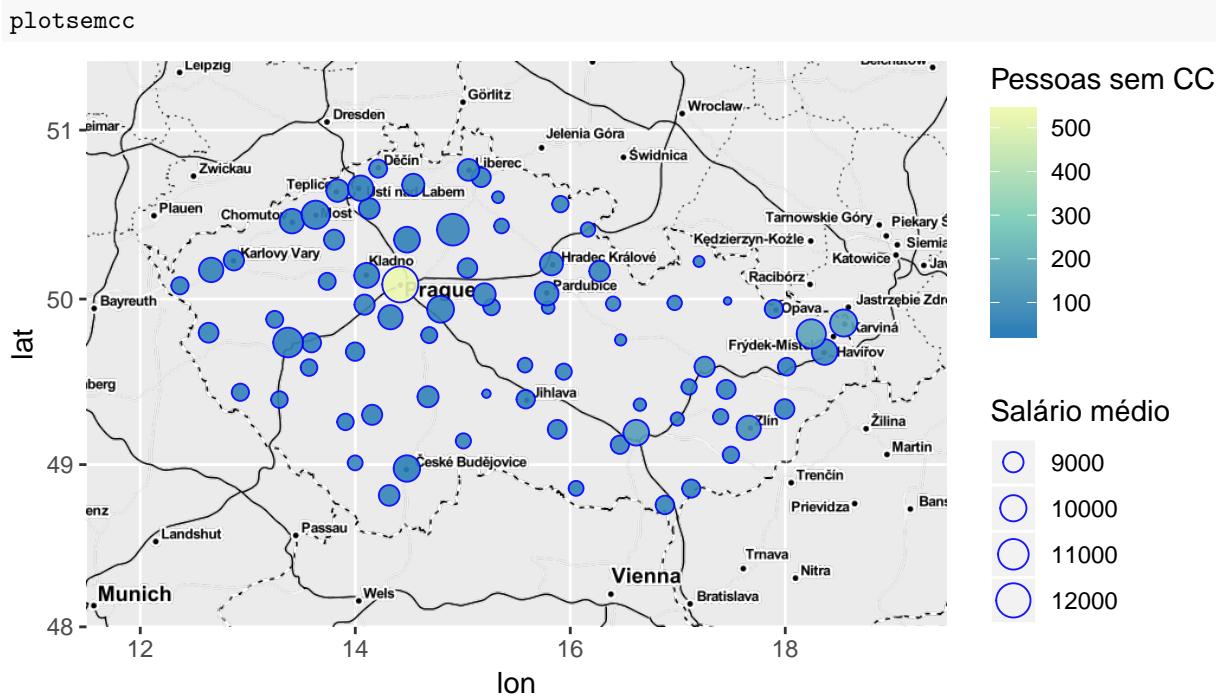
4.4. Análise dos cartões de crédito

Verificamos a classificação das pessoas por salário médio, região onde vivem e gênero.

```
# base de cartões
cc <- customers %>%
  inner_join(districts, by = "district_id") %>%
  inner_join(dispositions, by = "customer_id") %>%
  left_join(credit_cards, by = "disp_id")

# verificando pessoas sem cartão pelo país
sem_cc <- cc %>%
  filter(is.na(card_id)) %>%
  group_by(avg_salary, district_name, lat, long) %>%
  summarize(num_sem_cartao = n())

plotsemcc <- ggmap(map) +
  geom_point(data = sem_cc,
             aes(x = long, y = lat, size = avg_salary, fill = num_sem_cartao),
             shape = 21, alpha = 0.9, color = "blue") +
  labs(fill = "Pessoas sem CC", size = "Salário médio") +
  scale_fill_gradientn(colors = rev(brewer.pal(3, "YlGnBu")))
```



```
sem_cc %>%
  select(district_name, num_sem_cartao, avg_salary) %>%
  arrange(desc(num_sem_cartao)) %>%
  top_n(10, num_sem_cartao)
```

```
## # A tibble: 77 x 4
## # Groups:   avg salary, district name, lat [77]
```

```

##      lat district_name  num_sem_cartao avg_salary
##      <dbl> <chr>           <int>      <dbl>
## 1  50.1 Hl.m. Praha        533     12541
## 2  49.8 Ostrava - mesto    156     10673
## 3  49.9 Karvina          143     10177
## 4  49.2 Brno - mesto      138      9897
## 5  49.2 Zlin              93      9624
## 6  49.6 Olomouc          92      8994
## 7  49.7 Frydek - Mistek    73      9893
## 8  50.4 Nachod            65      8369
## 9  50.0 Usti nad Orlici    61      8363
## 10 49.1 Brno - venkov      57      8743
## # ... with 67 more rows

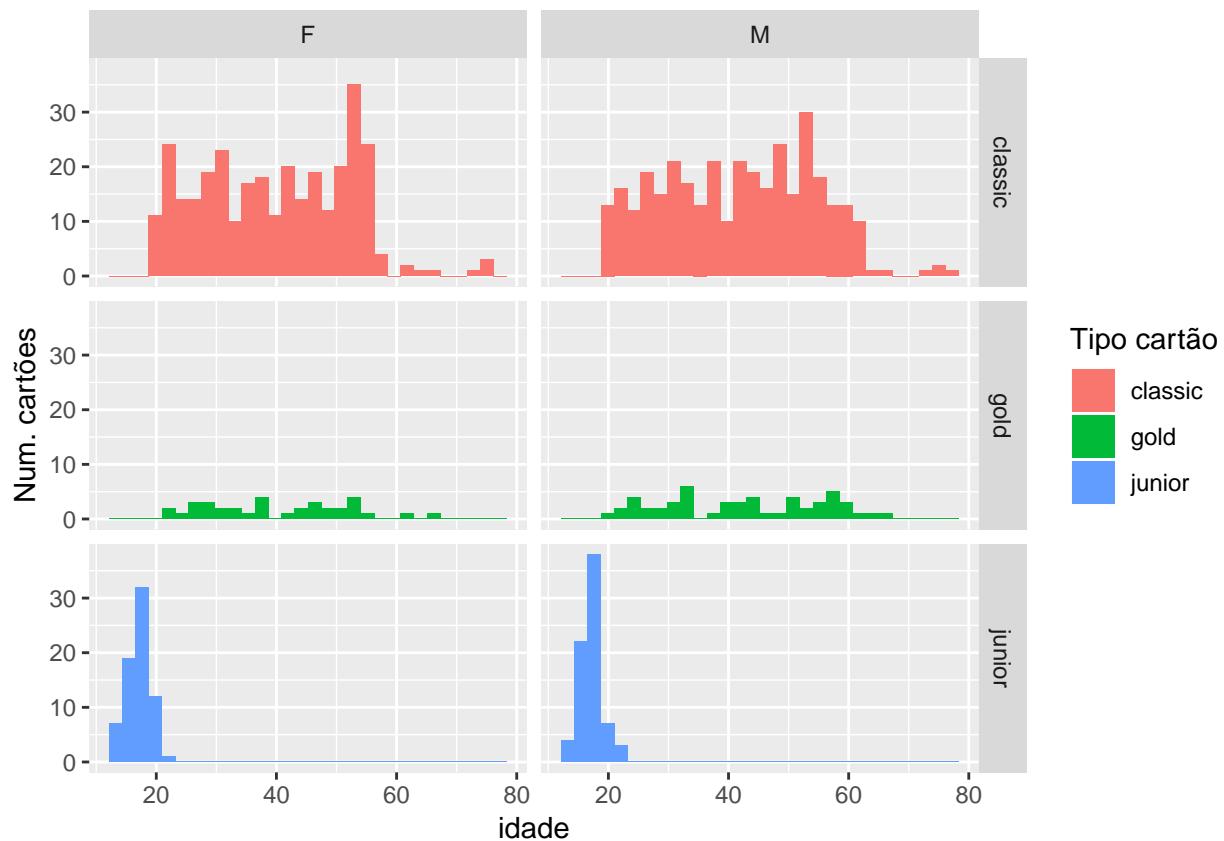
```

```

# tipos de cartões por gênero
cc2 <- credit_cards %>%
  inner_join(dispositions, by = "disp_id") %>%
  inner_join(customers, by = "customer_id") %>%
  mutate(idade = 1996 - year(birth_date))

cc2 %>%
  ggplot() +
  geom_histogram(aes(x = idade, fill = card_type)) +
  facet_grid(card_type ~ gender) +
  labs(y = "Num. cartões") +
  scale_fill_discrete(name = "Tipo cartão")

```



Com base nas cidades de Praga, Ostrava, Karvina e Brno, onde se concentram a maior quantidade de clientes

sem cartões e com um salário médio elevado, o banco pode priorizar a expansão do serviço de cartões de crédito.

Já para os cartões de crédito existentes, vê-se que a maior parte da população possui cartões classic, entre as idades de 20 a 50 anos, tanto no gênero masculino quanto no feminino.

Portanto, o banco poderá oferecer seu serviço de cartões para cada cidade de acordo com a faixa etária dos seus clientes, utilizando assim uma campanha mais segmentada para este produto.

5. Fontes e referências

- R Core Team (2017). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL <https://www.R-project.org/>.
- RStudio: <https://www.rstudio.com>
- R for DataScience - Hadley wickham, Garrett Grolemund: <http://r4ds.had.co.nz>
- Tidyverse, R packages for data science: <https://www.tidyverse.org>
- CRAN, The Comprehensive R Archive Network: <https://cran.r-project.org>
- ggmap, a package for plotting maps in R with ggplot2: <https://github.com/dkahle/ggmap>
- Oracle Data Modeler - <https://www.oracle.com/database/technologies/appdev/datamodeler.html>