

Exploração de dados - Banco Czech

Bruno

Lucas de Jesus Matias
Luiz Cesar Costa Raymundo

Turma 1

Contents	
Objetivo	3
Carrega as bibliotecas	3
Carga dos dados	3
Tratamento dos dados	4
Cliente	4
Conta	5
Distrito	5
Cartão	6
Transação	7
Disposição	8
Empréstimo	8
Ordem de Pagamento	9
União dos dados	10
Preparar união	10
Preparar NAs	11
Preparar factors	11
Preparar dados de empréstimo	11
Estudo dos dados	12
Empréstimos	12
Estudo de cluster	16

Objetivo

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

Carrega as bibliotecas

#Funções utilitárias Criamos a função prepararNA para simplificar o tratamento dos dados indisponíveis.

```
prepararNA <- function (tabela, nomeColuna, valorSeNA) {  
  indiceDados <- which(colnames(tabela)==nomeColuna)  
  tabela[is.na(tabela[,indiceDados]),indiceDados] <- valorSeNA  
  return(tabela)  
}  
  
prepararData <- function (vetorDataYMMDD) {  
  as.Date(paste0("19", vetorDataYMMDD), "%Y%m%d")  
}
```

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) ->  
read.csv2("./dados/client.asc", stringsAsFactors = FALSE) ->  
read.csv2("./dados/card.asc", stringsAsFactors = FALSE) -> ca  
read.csv2("./dados/disp.asc", stringsAsFactors = FALSE) -> di  
read.csv2("./dados/district.asc", stringsAsFactors = FALSE) -  
read.csv2("./dados/trans.asc", stringsAsFactors = FALSE) -> t
```

```
read.csv2("./dados/loan.asc", stringsAsFactors = FALSE) -> lo
read.csv2("./dados/order.asc", stringsAsFactors = FALSE) -> o
```

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 numérico, obtemos 4 dígitos da 3^a à 4^a 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 intuito 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. tomamos 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, 6)))
  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 inclui 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$frequency <- gsub("POPLATEK PO OBRATU", "acadatransac", account$frequency)
account$frequency <- as.factor(account$frequency)
account$account_date <- prepararData(account$date)

account <- account %>%
  mutate(account_age = year(currentdate) - year(account_date))
select(account_id, frequency, account_date, account_age)
```

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'
colnames(district)[14] <- 'numb_enter'

#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)
district$numb_enter = as.numeric(district$numb_enter)

#Limpeza de NA
district = prepararNA(district, "unemp_95", 1)

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

```

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$date <- prepararData(trans$date)
```

```
trans <- trans %>%
  dplyr::filter(type != "VYBER") %>%
  dplyr::filter(year(date) > 97) %>%
  mutate(amount = as.numeric(amount)) %>%
  mutate(balance = as.numeric(balance)) %>%
  select(account_id, type, amount, balance) %>%
  group_by(account_id) %>%
  summarise(
    credito = sum(ifelse(type == "credito", amount, 0))
    debito = sum(ifelse(type == "debito", amount, 0)),
    trans_amount = sum(amount),
    saldo_medio_em_conta = mean(balance),
    mediana_saldo = median(balance),
    min_saldo = summary(balance)[1],
```

```

    fq_saldo = summary(balance)[2],
    quant_trans = n()) %>%
mutate(withdraw_rate = debito / credito) %>%
select(account_id, withdraw_rate, quant_trans, saldo_

```

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(dispatch$type)
```

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

```

disp <- disp %>%
  # dplyr::filter(type == "OWNER") %>%
  group_by(account_id) %>%
  summarise(
    client_id = first(client_id[type == "OWNER"]),
    disp_id = first(disp_id[type == "OWNER"]),
    no_account_users = n()
  )

```

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$status <- gsub("B", "nao pago", loan$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$loan_date = prepararData(loan$date)
```

```
loan <- loan %>%
```

```
  mutate(loan_year = year(loan_date)) %>%
```

```
  mutate(loan_end_year = loan_year + (duration / 12)) %>%
```

```
  dplyr::filter(loan_end_year > 97) %>%
```

```
  mutate(loan_age = year(currentdate) - loan_year) %>%
```

```
  mutate(payments_rate = payments / amount) %>%
```

```
  select(loan_id, account_id, amount, duration, status,
```

```
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(),
    paga_divida = any(k_symbol == "divida"),
    paga_leasing = any(k_symbol == "leasing"),
    ja_pagou_seguro = any(k_symbol == "seguro"),
    media_transf = sum(amount) / n()
  )
```

União dos dados

Preparar união

```
dados <- client %>%
  inner_join(dispatch, 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") %>%
inner_join(trans, by="account_id") %>%
left_join(loan, by="account_id")

```

Preparar NAs

```

dados <- dados %>%
  prepararNA("card_type", "nenhum") %>%
  prepararNA("total_ordem", 0) %>%
  prepararNA("quant_ordem", 0) %>%
  prepararNA("ja_pagou_seguro", FALSE) %>%
  prepararNA("media_transf", FALSE) %>%
  prepararNA("loan_amount", 0) %>%
  prepararNA("loan_duration", 0) %>%
  prepararNA("loan_status", "nenhum")

```

Preparar factors

```

dados$card_type = as.factor(dados$card_type)
dados$loan_status = as.factor(dados$loan_status)

```

Preparar dados de empréstimo

```

dadosLoan <- dados %>%
  dplyr::filter(loan_status != "nenhum") %>%
  mutate(problemas_loan = if_else(loan_status == "
dadosLoan$problemas_loan = as.factor(dadosLoan$problemas_loan)

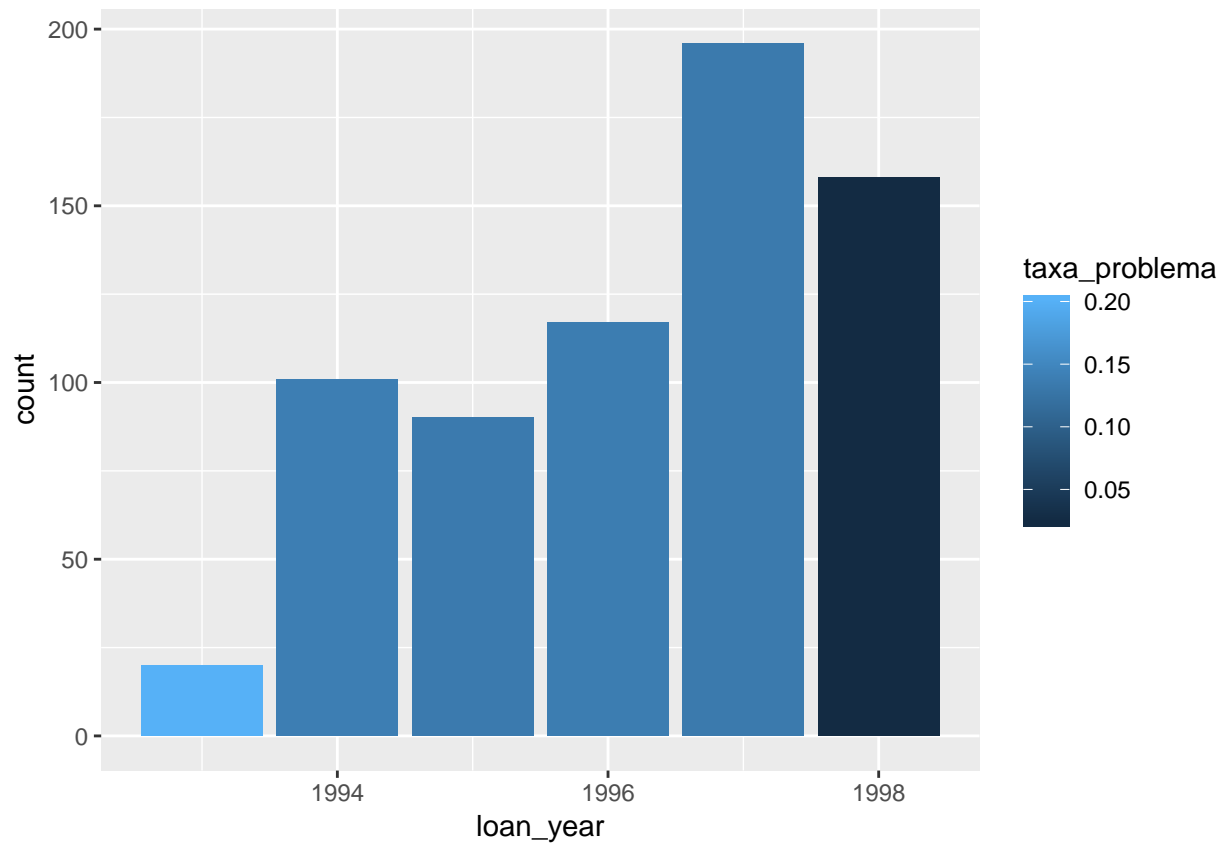
```

Estudo dos dados

Empréstimos

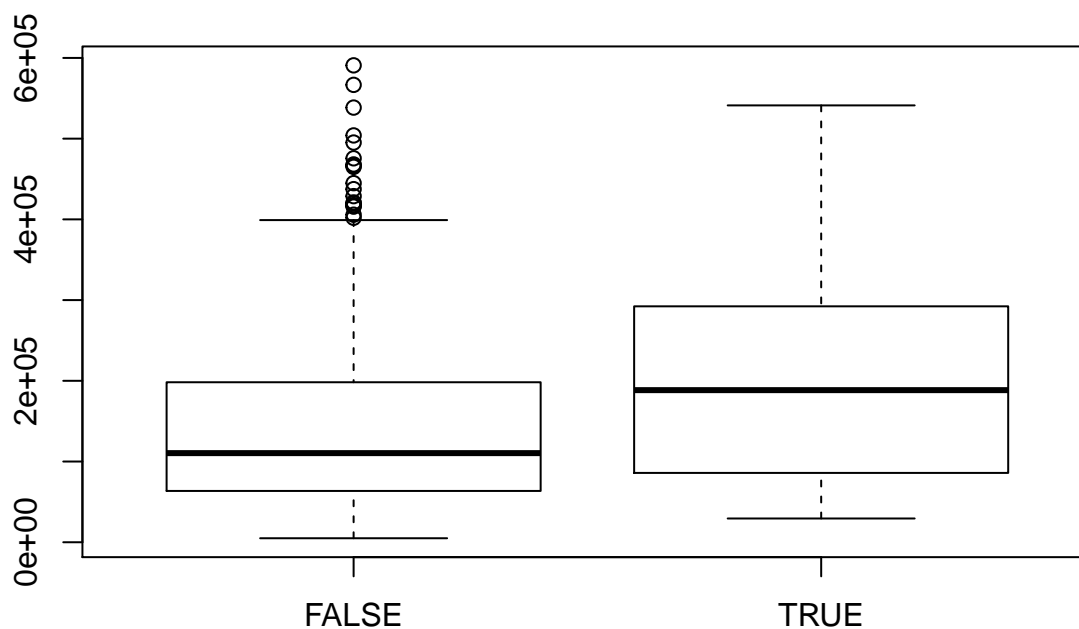
Observamos que contratos mais novos apresentam menos problemas com pagamentos

```
anoPorProblemas = dadosLoan %>%  
  mutate(loan_year = year(loan_date)) %>%  
  group_by(loan_year) %>%  
  summarise(  
    count = n(),  
    problemas = sum(ifelse(problemas_loan ==  
  ) %>%  
  mutate(taxa_problema = problemas / count)  
  
ggplot(anoPorProblemas, aes(x = loan_year, y = count)) +  
  geom_bar(stat = "identity", position=position_dodge(), aes(
```



E também observamos maiores problemas nos empréstimos quando a medida que o valor é maior

```
boxplot(loan_amount~problemas_loan, data = dadosLoan)
```



```
lm(dadosLoan, formula = loan_amount ~ problemas_loan) -> modelo
summary(modelo)
```

```
##
```

```
## Call:
```

```
## lm(formula = loan_amount ~ problemas_loan, data = dadosLoa
```

```
##
```

```
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -175554  -84425  -31595   59296  446131
```

```
##
```

```
## Coefficients:
```

```
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    144689      4544   31.844  < 2e-16 **
```

```
## problemas_loanTRUE      60313      13611      4.431 1.09e-05 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 111900 on 680 degrees of freedom
## Multiple R-squared:  0.02806,    Adjusted R-squared:  0.02
## F-statistic: 19.64 on 1 and 680 DF,  p-value: 1.092e-05
```

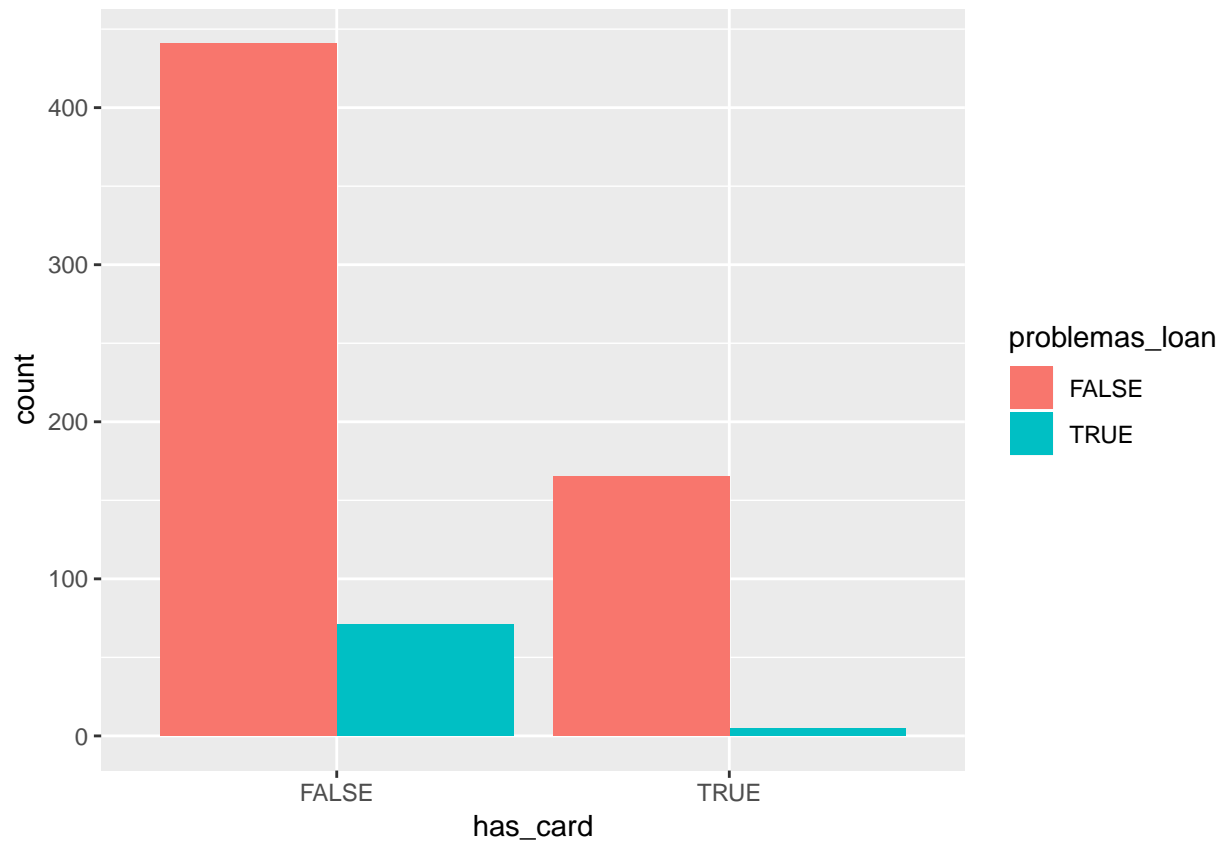
Identificamos também que pessoas que possuem cartões tem menores chances de ter problemas com os empréstimos

```
dadosLoan <- dadosLoan %>%
  mutate(has_card = ifelse(card_type=="nenhum", FALSE, TRUE))

chisq.test(table(dadosLoan$has_card, dadosLoan$problemas_loan))

##
## Pearson's Chi-squared test with Yates' continuity correction
##
## data:  table(dadosLoan$has_card, dadosLoan$problemas_loan)
## X-squared = 14.303, df = 1, p-value = 0.0001556

ggplot(dadosLoan, aes(has_card)) +
  geom_bar(position=position_dodge(), aes(fill=problemas_loan))
```



Estudo de cluster

```
dadosLoanCluster <- dadosLoan %>%
  select( problemas_loan, ja_pagou_seguro,
          fq_saldo, media_transf, quant_o)

dadosLoanCluster$problemas_loan = as.double(dadosLoan$problemas_loan)
dadosLoanCluster$ja_pagou_seguro = as.double(dadosLoan$ja_pagou_seguro)

lm(formula = problemas_loan ~ ., data = dadosLoanCluster) ->
summary(modelo)
```

##

Call:


```
## lm(formula = problemas_loan ~ ., data = dadosLoanCluster)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.46200 -0.12456 -0.05250  0.02771  0.87919
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    1.615e+00  4.977e-02  32.448  < 2e-16 **
## ja_pagou_seguro  1.779e-01  3.732e-02   4.767 2.29e-06 **
## no_account_users -8.648e-02  2.124e-02  -4.071 5.24e-05 **
## fq_saldo        -7.496e-06  8.226e-07  -9.112  < 2e-16 **
## media_transf     1.281e-05  4.305e-06   2.976  0.00303 **
## quant_ordem     -1.323e-01  1.978e-02  -6.690 4.69e-11 **
## min_saldo       -5.817e-05  3.629e-06 -16.031  < 2e-16 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2234 on 675 degrees of freedom
## Multiple R-squared:  0.5012, Adjusted R-squared:  0.4967
## F-statistic: 113 on 6 and 675 DF, p-value: < 2.2e-16
```

#achar a quantidade ideal de clusters

```
dadosLoanCluster_std <- scale(dadosLoanCluster[-1])
scaled_loan_data = as.matrix(dadosLoanCluster_std)
```

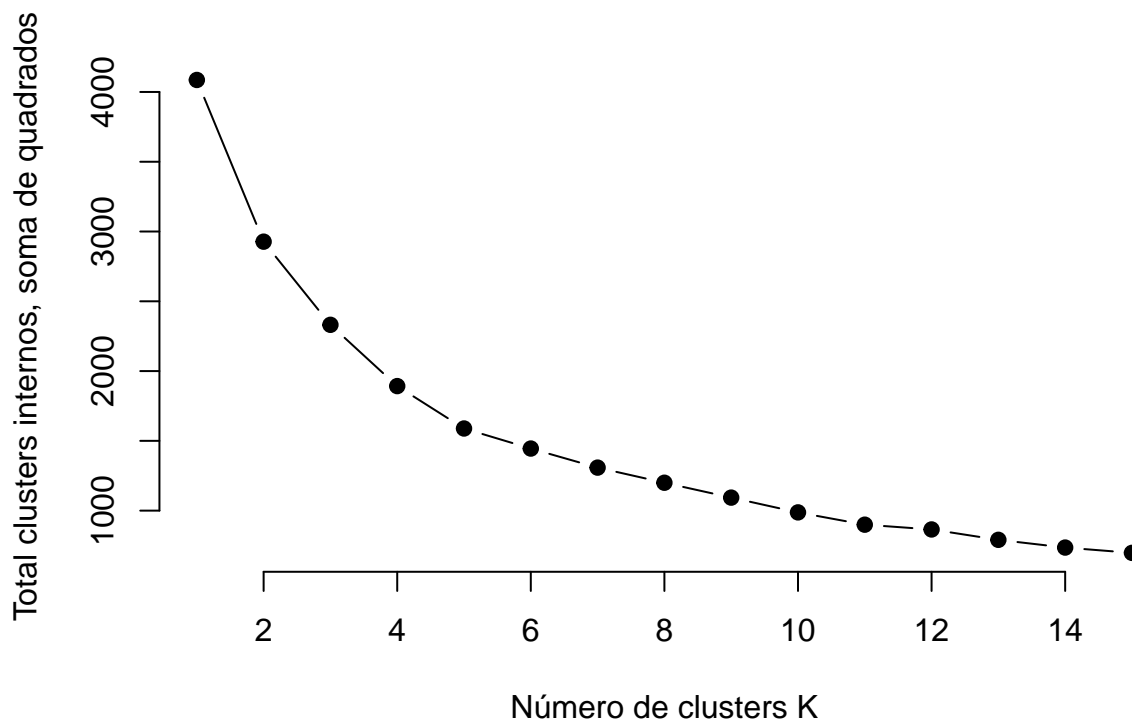
```
set.seed(123)
```

```
k.max <- 15
```

```
data <- scaled_loan_data
```

```
wss <- sapply(1:k.max,
              function(k){kmeans(data, k, nstart=50, iter.max

plot(1:k.max, wss,
     type="b", pch = 19, frame = FALSE,
     xlab="Número de clusters K",
     ylab="Total clusters internos, soma de quadrados")
```



#achar padrões

```
kmeans <- kmeans(dadosLoanCluster,3)
```

#sumario

kmeans

```
## K-means clustering with 3 clusters of sizes 166, 316, 200
##
## Cluster means:
##   problemas_loan ja_pagou_seguro no_account_users fq_saldo
## 1      1.319277      0.2650602      1.186747 16857.08
## 2      1.066456      0.1424051      1.218354 30090.45
## 3      1.010000      0.1250000      1.225000 44946.04
##   quant_ordem  min_saldo
## 1      1.915663 -1283.8988
## 2      1.829114   715.9772
## 3      1.705000   683.9030
##
## Clustering vector:
##   [1] 2 1 2 1 2 3 2 2 1 2 2 2 2 3 3 2 3 1 2 3 2 3 2 2 1 3
##  [36] 2 2 1 2 1 2 3 1 2 3 1 1 2 2 3 3 2 3 3 2 1 2 1 2 1 2
##  [71] 1 3 3 2 3 2 2 2 1 2 2 1 2 2 2 3 3 1 1 3 2 1 1 3 2 2
## [106] 3 1 3 1 3 2 1 2 1 2 3 1 2 2 2 2 1 1 3 1 3 3 3 2 1 3
## [141] 2 2 3 1 2 3 2 2 2 1 2 1 3 3 2 2 3 2 3 3 2 2 2 3 2 1
## [176] 2 1 1 3 2 1 3 2 2 2 2 2 1 2 2 2 1 3 2 1 1 3 3 2 1 2
## [211] 1 2 1 1 3 2 2 1 1 3 3 3 1 1 2 3 2 2 2 2 1 2 2 1 2 2
## [246] 2 2 3 1 1 2 3 3 1 2 3 1 2 3 1 2 2 2 3 1 1 3 1 2 2 3
## [281] 1 1 3 3 1 2 2 3 2 2 3 2 2 2 1 2 3 2 2 1 2 1 2 2 3 2
## [316] 2 1 3 1 2 2 2 1 1 2 3 3 2 2 3 3 3 2 3 2 2 2 3 1 1 2
## [351] 2 3 1 2 1 2 3 1 1 1 2 1 3 3 2 3 2 2 3 3 2 3 2 1 1 2
## [386] 3 1 2 3 2 2 2 3 3 1 2 2 1 1 2 2 3 2 3 3 3 1 1 2 2 1
## [421] 1 2 1 2 2 1 1 2 2 2 2 2 3 1 2 2 2 3 2 2 3 2 2 3 2
## [456] 3 3 1 3 2 2 2 2 2 3 2 3 3 3 2 3 2 2 1 1 3 2 3 2 3 2
## [491] 2 2 1 2 2 2 2 2 2 2 2 2 3 3 2 2 3 1 3 1 1 1 3 2 1 1 3
```

```
## [526] 1 3 1 2 2 1 3 3 1 1 2 1 2 2 2 2 1 3 1 2 1 2 2 2 2 1
## [561] 3 2 3 3 1 2 2 1 3 2 3 1 3 1 2 2 2 1 3 2 2 1 1 2 2 2
## [596] 3 1 2 3 3 2 2 3 1 2 3 2 3 2 3 2 1 3 2 2 1 1 2 2 2 2
## [631] 2 1 3 3 3 2 2 3 3 2 2 2 2 1 3 2 3 1 1 2 3 2 1 3 3 2
## [666] 3 2 2 2 3 3 3 2 2 1 1 3 3 2 2 2 1
##
## Within cluster sum of squares by cluster:
## [1] 8275067127 7301365869 7741187312
## (between_SS / total_SS = 75.8 %)
##
## Available components:
##
## [1] "cluster"          "centers"          "totss"            "withinss"
## [5] "tot.withinss"     "betweenss"        "size"             "iter"
## [9] "ifault"
```

```
#Vetor da soma dos quadrados, um componente por cluster
kmeans$withinss
```

```
## [1] 8275067127 7301365869 7741187312
```

```
#Distancia - Soma dos quadrados entre os clusters
kmeans$betweenss
```

```
## [1] 72916452403
```

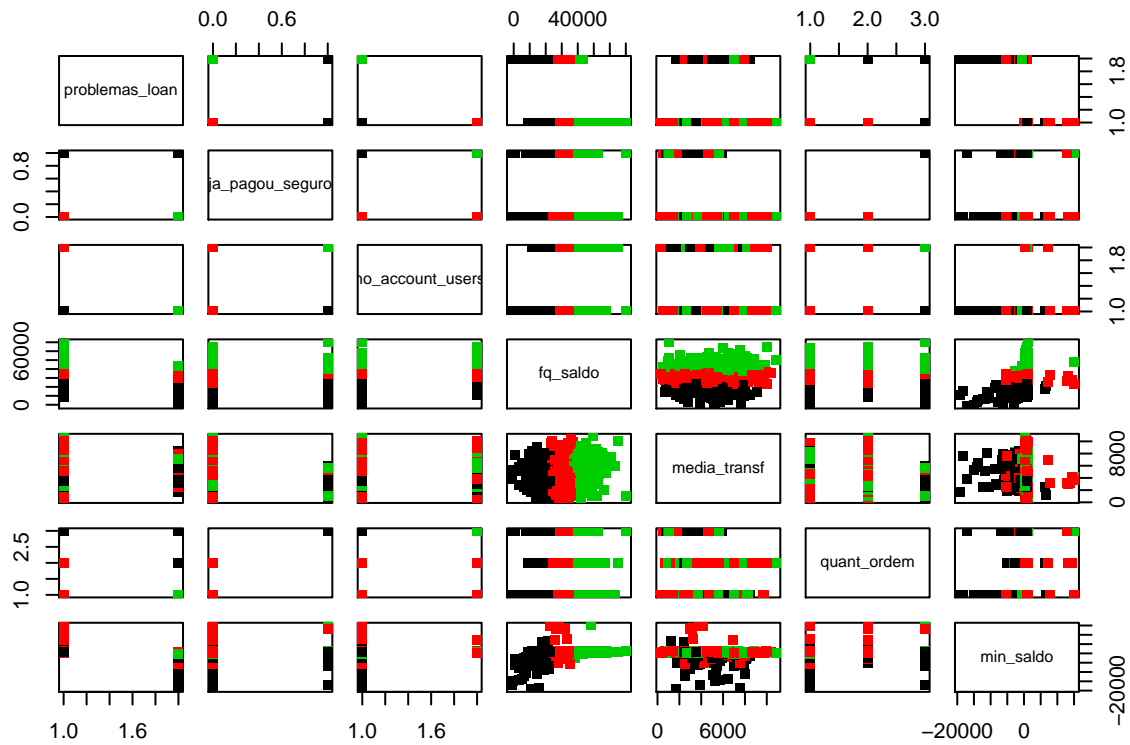
```
#Numero de pontos para cada cluster
kmeans$size
```

```
## [1] 166 316 200
```

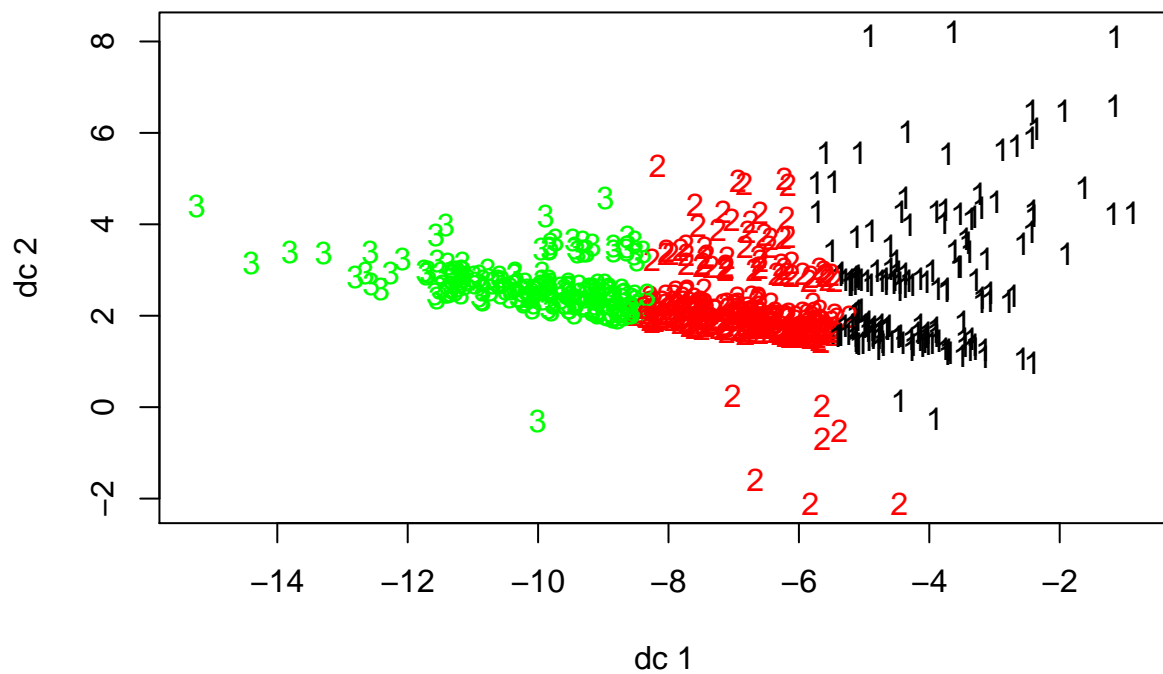
```
#Verificar Padrões
```

```
plot(dadosLoanCluster,col=kmeans$cluster,pch=15)
```

```
points(kmeans$centers,col=1:8,pch=3)
```



```
plotcluster(dadosLoanCluster,kmeans$cluster)
points(kmeans$centers,col=1:8,pch=16)
```



```
clusplot(dadosLoanCluster, kmeans$cluster, color=TRUE, shade=
```

A PCA plot showing the distribution of data points in a two-dimensional space defined by Component 1 (X-axis) and Component 2 (Y-axis). The plot displays three distinct clusters of data points, each represented by a different color and enclosed by a corresponding confidence ellipse. The green cluster is centered around Component 1 = -1 and Component 2 = 1. The red cluster is centered around Component 1 = 3 and Component 2 = 2. The blue cluster is centered around Component 1 = 0 and Component 2 = 0. The ellipses represent the confidence intervals for the mean of each cluster. The data points are labeled with numbers, indicating individual observations.

23