Aula 2 - Criando e manipulando dados

Vitor Rios

16 de outubro de 2017

Tipos de dados

Já vimos 3 tipos de valores de dados, mas existem outros 4 especiais

```
class(1)
```

```
## [1] "numeric"
```

```
class("1")
```

```
## [1] "character"
```

```
class(1 == "1")
```

```
## [1] "logical"
```

```r
10/0
```

```
## [1] Inf
```

```r
sqrt(-3)
```

```
## Warning in sqrt(-3): NaNs produzidos
```

```
## [1] NaN
```

```r
c(10, NA, 30)
```

```
## [1] 10 NA 30
```

```r
fator = factor(1:3, levels = c("chuchu", "goiaba", "iogurte"))
fator
```

```
## [1] <NA> <NA> <NA>
## Levels: chuchu goiaba iogurte
```

```r
class(fator)
```

```
## [1] "factor"
```

`Inf` representa valores infinitos, pode ser `+Inf` ou `-Inf`

`NaN` Not a Number, resultado de operações matemáticas impossíveis

`NA` Not Available, representa dados ausentes, útil para preencher tabelas e matrizes. A maioria das funções estatísticas do R possui um argumento `na.rm` que lida com valores ausentes

`factor` especifica que um conjunto de valores representa os níveis de um fator, usado para análises de dados categóricos

Ao ler dados de arquivos .csv, colunas representadas por caracteres são automaticamente convertidas em fatores. Para evitar isto, use o argumento `as.is = TRUE`, quando disponível (consulte o help)

Criando um conjunto de dados simulado

Iremos investigar o crescimento dos dragões usando quatro rações diferentes. Após mandar nossos estagiários fazerem os experimentos, os que sobreviveram trouxeram os seguintes dados, em centenas de quilos

```
vacas = c(121.72355, 103.79754, 130.15442, 98.29305, 103.43365, 102.44998,
    111.07215, 113.74047, 103.16081, 80.87149, 98.66692, 65.09549, 155.749
    88.30168, 147.4361, 114.60806, 109.87433, 149.54772, 83.54137)
fazendeiros = c(77.91352, 78.07251, 81.95604, 75.64862, 78.45213, 79.11058
    79.98952, 79.18127, 840.1635, 74.8686, 82.01886, 78.26936, 77.94691, 78
    77.64901, 77.64097, 77.19803, 72.48175, 83.45336, 78.99681)
virgens = c(127.9366, 201.7158, 136.1366, 136.588, 131.7213, 118.1486, 125
    139.6544, 163.589, 139.7455, NA, 141.445, 110.7311, 157.5921, 176.8437
    102.8659, 121.8286, 134.7097, 157.1392, 166.7133)
aventureiros = c(191.3721, 216.1671, 165.438, 196.273, 172.6565, 178.2955,
    189.7674, 160.2968, 208.44, 204.0934, 208.1798, 186.638, 193.9446, 197
    198.6853, 213.8838, 210.1881, 209.9109, 210.9228)
```

Podemos juntar tudo num dataframe para facilitar

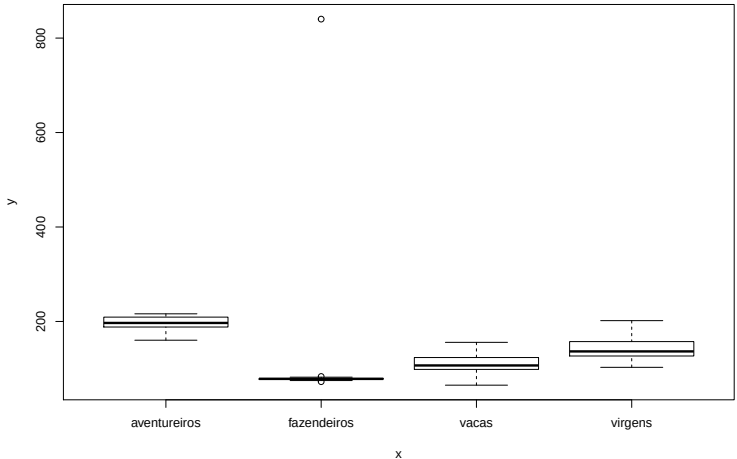
```
alimento = c(rep("vacas", 20), rep("fazendeiros", 20), rep("virgens", 20),
    20))
dragoes = data.frame(x = alimento, y = c(vacas, fazendeiros, virgens, aven
head(dragoes)
```

```
##       x         y
## 1 vacas 121.72355
## 2 vacas 103.79754
## 3 vacas 130.15442
## 4 vacas  98.29305
## 5 vacas 103.43365
## 6 vacas 102.44998
```

```
summary(dragoes)
```

```
##             x            y        
##  aventureiros:20   Min.   : 65.10  
##  fazendeiros :20   1st Qu.: 82.74  
##  vacas       :20   Median :125.75  
##  virgens     :20   Mean   :140.93  
##                    3rd Qu.:174.75  
##                    Max.   :840.16  
##                    NA's   :1
```

```
plot(dragoes)
```



Para facilitar suas análises

Formate seus dados sempre que possível da seguinte forma:
- Cada linha uma observação ou indivíduo
- Cada coluna uma variável

```
head(dragoes)
```

```
##       x         y
## 1 vacas 121.72355
## 2 vacas 103.79754
## 3 vacas 130.15442
## 4 vacas  98.29305
## 5 vacas 103.43365
## 6 vacas 102.44998
```

```
summary(lm(dragoes$y ~ dragoes$x))

##
## Call:
## lm(formula = dragoes$y ~ dragoes$x)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -45.26 -34.50  -6.92  10.08 723.68
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)             195.31      19.62   9.957 2.31e-15 ***
## dragoes$xfazendeiros    -78.82      27.74  -2.841  0.00578 **
## dragoes$xvacas          -84.96      27.74  -3.063  0.00305 **
## dragoes$xvirgens        -53.69      28.10  -1.910  0.05993 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Temos problemas

Erros de digitação de dados nos objetos fazendeiros e virgens

```
max(dragoes[, 2])
```

```
## [1] NA
```

```
max(dragoes[, 2], na.rm = T)
```

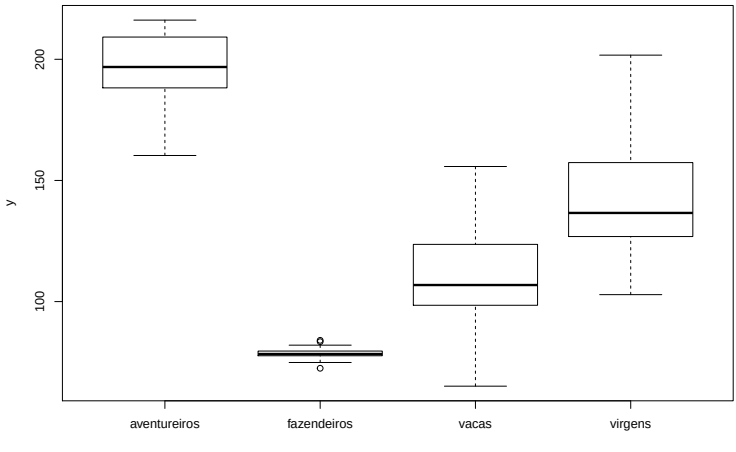
```
## [1] 840.1635
```

Erro na posição do decimal

```
dragoes[dragoes == 840.1635] = 84.01635
max(dragoes[, 2], na.rm = TRUE)
```

```
## [1] 216.1671
```

```
plot(dragoes)
```



Para NA, não há remédio

Só verificando o dado original ou removendo

```
mean(dragoes[, 2])  ##qualquer cálculo com NA é igual a NA
```

```
## [1] NA
```

```
mean(dragoes[, 2], na.rm = TRUE)
```

```
## [1] 131.363
```

```
## na.rm = TRUE remove o NA para o calculo da média
```

```
aggregate(dragoes$y ~ dragoes$x, FUN = "mean")
```

```
##      dragoes$x dragoes$y
## 1 aventureiros  195.3093
## 2  fazendeiros   78.6809
```

Para verificar erros de digitação ao trabalharmos com texto, podemos usar a função `unique()`

```
### dados dispníveis em
### http://ecologia.ib.usp.br/bie5782/doku.php?id=dados:dados-caixeta
nomesComErro = read.table("arquivos/caixetaComErro.csv", header = T, sep =
head(nomesComErro)
```

```
##    local parcela arvore fuste cap   h                especie
## 1 chauas       1      1     1 210  80      Myrcia sulfiflora
## 2 chauas       1      3     1 170  80      Myrcia sulfiflora
## 3 chauas       1      4     1 720  70 Syagrus romanzoffianus
## 4 chauas       1      5     1 200  80   Tabebuia cassinoides
## 5 chauas       1      6     1 750 170                indet.1
## 6 chauas       1      7     1 320  80      Myrcia sulfiflora
```

```
unique(nomesComErro$especie)
```

```
##  [1] Myrcia sulfiflora         Syagrus romanzoffianus
##  [3] Tabebuia cassinoides      indet.1
##  [5] myrtaceae4                myrtaceae1
##  [7] indet.2                   Tabebuia casinoides
##  [9] Tabebuia casssinoides     Psidium sp
## [11] Coussapoa micropoda       Tibouchina nutticeps
## [13] myrtaceae2                Callophyllum brasiliensis
## [15] indet.3                   Cabralea canjerana
## [17] jussara                   Calophyllum brasiliensis
## [19] Pisonia sp                Matayba sp
## [21] Pera glabrata             Persea sp
## [23] Ficus sp                  bombacaceae
## [25] Coussapoa macrocarpa      Alchornea triplinervia
## [27] Andira fraxinifolia       Cryptocaria moschata
## [29] Gomidesia sp              eugenia3
## [31] Ilex durosa               Inga sp
```

```r
# podemos também colocar os nomes em ordem alfabética
sort(unique(nomesComErro$especie))
```

```
##  [1] Alchornea triplinervia    Andira fraxinifolia
##  [3] bombacaceae               Cabralea canjerana
##  [5] Callophyllum brasiliensis Calophyllum brasiliensis
##  [7] Cecropia sp               Coussapoa macrocarpa
##  [9] Coussapoa micropoda       Cryptocaria moschata
## [11] Cyathea sp                Eugenia oblongata
## [13] eugenia3                  fabaceae1
## [15] Ficus sp                  Gomidesia sp
## [17] Ilex durosa               Ilex sp
## [19] indet.1                   indet.2
## [21] indet.3                   Inga sp
## [23] Jacaranda puberula        jussara
## [25] Matayba sp                Mela 1
## [27] Mela 2                    Myrcia sulfiflora
## [29] Myrtaceae 3               myrtaceae1
```

```
nomesComErro$especie[nomesComErro$especie == "Tabebuia casssinoides"] = "Ta
sort(unique(nomesComErro$especie))
```

```
##  [1] Alchornea triplinervia    Andira fraxinifolia
##  [3] bombacaceae               Cabralea canjerana
##  [5] Callophyllum brasiliensis Calophyllum brasiliensis
##  [7] Cecropia sp               Coussapoa macrocarpa
##  [9] Coussapoa micropoda       Cryptocaria moschata
## [11] Cyathea sp                Eugenia oblongata
## [13] eugenia3                  fabaceae1
## [15] Ficus sp                  Gomidesia sp
## [17] Ilex durosa               Ilex sp
## [19] indet.1                   indet.2
## [21] indet.3                   Inga sp
## [23] Jacaranda puberula        jussara
## [25] Matayba sp                Mela 1
## [27] Mela 2                    Myrcia sulfiflora
## [29] Myrtaceae 3               myrtaceae1
```

Entendendo o comando passo a passo:

`nomesComErro\$especie[nomesComErro\$especie=="Tabebuia casinoides"] = "Tabebuia cassinoides"`

Precisamos saber a posição dos elementos com erro: O comando abaixo nos retorna um vetor lógico com as posições de todos elementos "Tabebuia casinoides" dentro do objeto `nomesComErro`, na coluna `especie`

```
nomesComErro$especie == "Tabebuia casinoides"
```

```
##    [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##   [12] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##   [23] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE
##   [34]  TRUE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##   [45] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##   [56] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##   [67] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##   [78] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##   [89] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

Selecionar o que será substituído:

Usaremos o vetor lógico do comando anterior para acessar os elementos dentro do objeto, usando colchetes []

```
nomesComErro$especie[nomesComErro$especie == "Tabebuia casinoides"]
```

```
## [1] Tabebuia casinoides Tabebuia casinoides Tabebuia casinoides
## [4] Tabebuia casinoides Tabebuia casinoides Tabebuia casinoides
## 45 Levels: Alchornea triplinervia Andira fraxinifolia ... Tibouchina nu
```

Este comando retorna apenas os elementos que cumprem a condição *(==)* dentro do colchete

Substituir

Utilizamos o operador de atribuição = para trocar os valores indexados nos passos acima

```
nomesComErro$especie[nomesComErro$especie == "Tabebuia casinoides"] = "Tabe
sort(unique(nomesComErro$especie))
```

```
##  [1] Alchornea triplinervia    Andira fraxinifolia
##  [3] bombacaceae               Cabralea canjerana
##  [5] Callophyllum brasiliensis Calophyllum brasiliensis
##  [7] Cecropia sp               Coussapoa macrocarpa
##  [9] Coussapoa micropoda       Cryptocaria moschata
## [11] Cyathea sp                Eugenia oblongata
## [13] eugenia3                  fabaceae1
## [15] Ficus sp                  Gomidesia sp
## [17] Ilex durosa               Ilex sp
## [19] indet.1                   indet.2
## [21] indet.3                   Inga sp
## [23] Jacaranda puberula        jussara
```

Operações lógicas e comparações

O R permite fazer todo tipo de comparação e teste lógico, abrindo possibilidade de selecionar itens para indexação baseados em diversas condições. os objetos testados podem ser de qualquer tipo

Testes lógicos retornam valores `TRUE` ou `FALSE`
`==` é a comparação de igualdade. Leia como é `igual a`?
`!=` é a comparação de desigualdade. Leia como é `diferente de`?

```
"sorvete" == "feijão"  #sorvete é igual a feijão?
```

```
## [1] FALSE
```

```
(1 + 1) != 2  # um mais um é diferente de dois?
```

```
## [1] FALSE
```

```
42 != "A pergunta da vida, do universo e tudo mais"
```

```
pi
```

```
## [1] 3.141593
```

```
3.141593
```

```
## [1] 3.141593
```

```
pi == 3.141593  ### Algo de errado não está certo
```

```
## [1] FALSE
```

```
i = 0.1
i = i + 0.05
i == 0.15
```

```
## [1] FALSE
```

Comparações de igualdade com ponto flutuante

Números decimais são representados internamente como `floating point`, e a precisão varia de máquina para máquina. Veja mais detalhes em http://floating-point-gui.de/

```
i = 0.1
i = i + 0.05
i == 0.15
```

```
## [1] FALSE
```

Resumidamente, se você quiser comparar com precisão a igualdade de operações com decimais, use a funçãoo `all.equal()`. Para comparações do tipo maior/menor, não é preciso se preocupar (muito)

```
i = 0.1
i = i + 0.05
all.equal(i, 0.15)
```

```
## [1] TRUE
```

Outras comparações

`a <= b` a é menor ou igual a b?

`a >= b` a é maior que ou igual a b?

`!x` não x

Operadores lógicos e if()

O R também pode testar duas condições ao mesmo tempo, com os operadores *E* e *OU*

`a & b` retorna TRUE se tanto `a` quanto `b` forem verdade

`a | b` retorna TRUE se *pelo menos um*, `a` ou `b`, for verdade

```
teste = c(1, 2, 2, 3, 4, 5, 6, 7)
(teste[1] == 1) & (teste[2]/2 == 1)
```

```
## [1] TRUE
```

```
(teste[1] == 1) & (teste[2]/3 == 1)
```

```
## [1] FALSE
```

```
(teste[2] == 1) | (teste[2]/2 == 1)
```

```
## [1] TRUE
```

Por exemplo, se quisermos todos os dragões que pesam entre 100 e 150 centenas de kilos

```
dragoes[dragoes$y >= 100 & dragoes$y <= 150, ]
```

```
##          x        y
## 1    vacas 121.7236
## 2    vacas 103.7975
## 3    vacas 130.1544
## 5    vacas 103.4337
## 6    vacas 102.4500
## 7    vacas 125.4999
## 8    vacas 111.0721
## 9    vacas 113.7405
## 10   vacas 103.1608
## 16   vacas 147.4361
## 17   vacas 114.6081
## 18   vacas 109.8743
## 19   vacas 149.5477
## 41 virgens 127.9366
```

if()

`if()` é uma função que permite executar um trecho de código se, e somente se, uma dada condição for verdadeira

`if(condição) {código a ser executado}`

Sempre coloque o código a ser executado dentro de chaves

```
teste.de.normalidade = shapiro.test(dragoes$y)
if (teste.de.normalidade$p.value >= 0.05) {
    fit = lm(dragoes$y ~ dragoes$x)
    summary(fit)
}
```

Fazendo a mesma coisa várias vezes

se precisamos repetir uma tarefa várias vezes, usamos o *laço (loop)*

```
for (i in 1:10) {
    a = i^2
    message(a)
}
```

`i` é chamado de `índice`. Pode ser qualquer letra ou expressão, é utilizado para contar quantas vezes o laço foi executado
`1:10` 1 é o valor inicial de `i`, 10 é o valor final.

`(i in 1:10)` significa `comece com i = 1, e vá aumentando de um em um até 1 = 10`
O bloco entre `{}` será executado enquanto a condição for verdadeira

Para ler vários arquivos:

```
arquivos = list.files()  #lê os nomes dos arquivos no diretório de trabalho
for (i in 1:length(arquivos)) {
    # resultados[i] = read.csv(arquivos[i])
    message(arquivos[i])
}
```

Para plotar vários objetos separadamente

```
niveis = levels(dragoes$x)
i = 1
for (i in 1:length(niveis)) {
    nome = paste("figuras/", niveis[i], ".png")  #prepara o nome do arquiv
    png(filename = nome)  #prepara para gravar arquivo tipo png
    plot(dragoes[dragoes$x == niveis[i], ]$y)  #escreve o arquivo
    dev.off()  #salva o arquivo
}
```