Enap

Aula 3 - Lendo dados no R

Marcelo Prudente e Rafael Giacomin

13 de março de 2018

Enap

Introdução

Leituras

Enap

Para aprofundar os tópicos desta aula:

- **Tutorial de importação de dados do Datacamp**

Objetivos da Aula

Enap

- Nesta aula, apresentaremos:
  - Como ler dados no **R**
  - Como exportar dados no R
  - Os fundamentos da manipulação de dados no **R**
    - ★ Foco na sintaxe
  - Como mesclar dados no R

Importando Dados no R

Qual o diretório (pasta) de trabalho

Enap

- Duas funções importantes para a organização do trabalho são a **getwd()** e a **setwd()**.
  - *digite*: **getwd()** em seu console. O que você obtêm?
- Elas informam e definem as pastas onde os arquivos estão.

Escolhendo o diretório de trabalho

Enap

- É possível criar uma pasta para guardar seus dados e definí-la como a seu diretório de trabalho.

```
# Estabelecer diretorio de trabalho
setwd("C:/curso_r_enap")
```

- Esse comando permite baixar arquivos com maior facilidade

Os tipos de dados

Enap

- Em geral, as bases governamentais estão disponibilizadas em:
  - formatos de texto (*.csv* ou *.txt*) ou
  - em excel (*.xls* ou *.xlsx*).
- Há ainda outros formatos relacionados aos diversos programas estatésticos do mercado:
  - *.dta*, do STATA
  - *.sav*, do SPSS
  - *.sas7bdat*, do SAS
  - *.fwf*, formato fixo
  - entre outros

Lendo os dados no R

Enap

- Apesar dos diversos tipos de dados, a leitura deles no programa é bastante simples.
- Há inúmeros pacotes desenhados para essa função:
  - o pacote de base do sistema, com *read.csv()*
  - o pacote *tidyverse*, com o *read_csv()* e outros
  - o pacote *data.table*, com o *fread()*
  - o pacote *foreign*, com *read.dta()*, *read.spss()*, entre outros

Lendo os dados no R

Enap

- Na prática, a leitura de dados em todos os pacotes tem uma fórmula comum para a leitura dos dados:
  1. Determinar o local do arquivo (ex: "C:/minha_pasta/meu_arquivo")
  2. Determinar o separador do arquivo (**","**, **";"**)
  3. Determinar os separadores dos números decimais (**"."**, **","**)
  4. Determinar se a primeira linha contém o nome das variáveis (**header == TRUE**)

Arquivos de texto: .csv e .txt

Lendo .csv

Enap

- Antes de executar os comandos, é importante verificar a pasta em que está o arquivo.
- Depois, fazer uma inspeção visual nos dados do arquivo *exemplo1.csv*.
  - identificar o separador do arquivo
  - identificar o separador dos decimais
- **Note**: o separador das colunas do arquivo *exemplo1.csv* é uma **,**
  - Esse formato é muito comum nos dados americanos

Lendo .csv separado por vírgulas

Enap

```
# Base do sistema
ex1_base <- read.csv("C:/curso_r_enap/exemplo1.csv")

# Tidyverse - readr
ex1_tdv  <- read_csv("C:/curso_r_enap/dados/exemplo1.csv")

# Ou ainda
ex1_tdv <- read_csv(file.choose())
```

- Os comandos acima assumem que:
  - o **sep** = “,”
  - o **dec** = “.”
  - o **header** == **TRUE**

Lendo .csv separado por ponto e vírgula

Enap

- Agora, vamos inspecionar o arquivo **exemplo2.csv**

```
# Base do sistema
ex2_base <- read.csv2("C:/curso_r_enap/dados/exemplo2.csv")

# Tidyverse
ex2_tdv  <- read_csv2("C:/curso_r_enap/dados/exemplo2.csv")
```

- Os comandos acima assumem que:
  - o **sep** = “;”
  - o **dec** = “,”
  - o **header** == **TRUE**

Lendo txt

Enap

- o **R** fornece o **read.csv** como programa de base para ler arquivos de texto (*.txt*).
- Todavia, o **fread()** traz as mesmas funções com maior velocidade.

```
# Base
ex3_base <- read_csv("C:/curso_r_enap/dados/exemplo1a.txt")
# readr
ex3_tdv <- read_csv("C:/curso_r_enap/dados/exemplo1a.txt")
# fread
ex3_fread <- fread("C:/curso_r_enap/dados/exemplo1a.txt")
```

Lendo com readr()

Enap

- A abordagem do readr permite importar dados no formato tbl_df (tiblle), cuja estrutura é mais moderna para leitura de dados.
- É mais rápido que a base do sistema e bastante intuitivo.

função	o que processa
read_csv()	arquivos csv separados por ,
read_tsv()	arquivos separados por tab
read_delim()	arquivos separados por delimitador geral (espaço ou tab)
read_fwf()	arquivos com largura fixa
read_table()	arquivos separados por um espaço em branco

Lendo com readr()

Enap

- É possível importar apenas algumas colunas dos dados - no caso de arquivos grandes, por exemplo, 1 milhão de linhas - com a extensão **n_max**

```
ex1_tdv  <- read_csv("exemplo1.csv",
                     n_max =  2)
```

- Também é possível pular algumas linhas. Porém, com o readr os nomes das colunas não serão importados:

```
ex1_tdv  <- read_csv("exemplo1.csv", skip = 3,
                     n_max =  2)
```

fread(): velocidade

Enap

- As funções do readr são versáteis e produzem tbl_df, mas o fread(), do pacote **data.table** é mais rápido entre todas as soluções.
  - Melhor solução para arquivos com **1gb ou +**
- Além disso, o *fread()* deduz qual o separador do arquivo.
- Ainda, permite:
  - ler apenas algumas linhas (**nrow = 10**, por exemplo) do arquivo para uma rápida inspeção
  - também, ler algumas colunas desejadas (**select(c("col1", "col5"))**, por exemplo))
- No entanto, às vezes apresenta erros para a leitura dos decimais.

O problema do encoding

Enap

- Ao baixar dados no R no Windows, muitas vezes a leitura de dados de texto pode ser problemática.
  - especialmente com as soluções do **readr**

```
# leitura com problema
problema <- read_csv("C:/curso_r_enap/dados/exemplo1a.txt")
head(problema, 10)
```

- Observe que a leitura dos acentos é problemática, pois o programa é focado em dados produzidos nos EUA.

Solução do problema

Enap

- Perceba que o problema de leitura ocorrerá com texto. Se o banco tem apenas números, não há problema.
- Esse problema da leitura de dados no windows pode ser resolvido com a especificação do encoding.

```
resolvido  <- read_csv("exemplo1a.txt",
                       locale = locale(encoding = "Latin1"))
head(resolvido, 15)
```

xls e xlsx

Lendo Excel

Enap

- Há duas formas práticas de ler dados do *Excel* no **R**.
- Vamos observar o arquivo **exemplo4.xlsx** e aplicar a leitura por comando.

```
library(readxl)
# Lendo os dados do excel
ex4  <- read_excel("C:/curso_r_enap/dados/exemplo4.xlsx")

# Importar a segunda planilha
ex4_s2  <- read_excel("C:/curso_r_enap/dados/exemplo4.xlsx",
```

- Há também a possibilidade de ler os dados com o cursor.

Lendo outros tipos de arquivo

Enap

- É possível ler outros arquivos, como os de SPSS, SAS ou STATA. Também, dados *.jason*, *.xml*, entre outros.
- Todos os comandos acima podem ser executados facilmente com o uso do mouse.
- Siga as instruções:
  - *File*
  - *Import Dataset*
  - *From . . .*
- Apesar disso, recomendamos aprender os comandos. É mais rápido!!

Exportando dados

Exportando dados

Enap

- É possível utilizar as funções **write** para exportar dados do **R** para a sua pasta.
- Como os dados de texto podem ser lidos em qualquer programa (EXCEL,SPSS, SAS, STATA, entre outros), vamos aprender a exportar arquivos nessa forma.

```
# Você pode determinar a pasta na qual o arquivo ser? salvo
setwd("C:/curso_r_enap/meus_dados/")

# Exportando arquivos de texto
write.table(exemplo1_base, "meu_exemplo.csv")
```

Exportando dados

Enap

- Essa função permite determinar:

```
# Separador
write.table(exemplo1_base, "meu_exemplo2.csv",
            sep = ";")

# Decimal
write.table(exemplo1_base, "meu_exemplo2.csv",
            sep = ";", dec = ",")

# Omitir o número das linhas
write.table(exemplo1_base, "meu_exemplo2.csv",
            sep = ";", dec = ",", row.names = FALSE)

# mais?
?write.table; help(write.table)
```

Exercício

Enap

- Resolva o exercício 3 do arquivo **exercício_importacao_de_dados.pdf**.

Manipulando dados

Manipulando dados: rudimentos

Enap

- A manipulação de dados demanda um bom tempo de qualquer analista de dados.
- O objetivo deste tópico é apresentar algumas estratégias importantes para a manipulação dos dados.
- Os comandos aprendidos nesta aula trazem "massa crítica" para as aulas seguintes.

Manipulando dados: rudimentos

Enap

- Na realidade da análise de dados, alguns problemas simples podem surgir. A partir do banco **dados_sociais.csv**, podem surgir algumas questões iniciais.
  - Quantos anos são especificados na base de dados? Quais são eles?
  - Como extrair um subconjunto dos dados?
  - Como renomear variáveis?
- Nas aulas seguintes, o tema será aprofundado.

Extraindo valores únicos

Enap

- Em geral, ao lidar com dados, é necessário identificar os valores únicos de um *vetor* ou de uma coluna de um *data.frame*. Para tanto, utiliza-se o comando **unique()**.
-

```
# Valores únicos de um vetor de nomes
x <- c("João", "Pedro", "Carlos", "Maria", "Josefa",
       "Mariana", "João", "José", "Mariana",
       "Carlos", "José", "João")
unique(x)
length(unique(x))

# Valores únicos
unique(dados_sociais$uf)
```

Verificar valores únicos: tabela

Enap

- Para contar quantas vezes cada observação aparece utiliza-e a função **table()**.

```
# Quantas vezes cada nome aparece
table(x)

# Quantas vezes cada UF aparece
table(dados_sociais$uf)

# Quantas vezes cada UF aparece por ano
table(dados_sociais$uf, dados_sociais$ano)
```

Subconjunto dos dados: a lógica

Enap

- Criar subconjunto dos dados é um tipo de operação cotidiana para analista de dados.
- É possível criar diversos subconjuntos de bancos e manipulá-los separadamente.
- Para tanto, é necessário o uso dos operadores lógicos.

```
# Argumento básico (pkg base do sistema)
subset(x, subset, select)

# Exemplo: selecionar ano 1991
b_rj <- subset(dados_sociais, uf == 33,
               select = c("ano", "municipio", "pop"))
```

Exercício

Enap

- A base *dados sociais* compreende quantos períodos?
- Crie novas bases a partir da base original, uma para cada período.
- Extraia a média da expectativa de vida de cada período
- Crie novas bases em que a expecativa de vida seja maior que a média de cada período?
  - Você consegue fazer isso diretamente na base original? Lembre dos seus conhecimentos dos operadores lógicos.

Renomear variáveis

Enap

- Comumente, há necessidade de renomear variáveis. Pelos conhecimentos básicos dos vetores, dá para extrair os nomes das variáveis e manipulá-los.

```
nomes<- colnames(dados_sociais)
nomes

# Atribuindo novo nome
nomes[2] <- "UF"
nomes

# De forma direta
colnames(dados_sociais)[2] <- "UF"
```

Renomear variáveis: maiúsculas e minúsculas

Enap

- O pacote base do sistema traz duas funções muito úteis: **toupper()** e **tolower()**

```
# Primeiro, salvar nomes originais do banco
nome <- colnames(dados_sociais)

# Transformar e maiúsculas
NOME <- toupper(nome)

# Transformar em minúsculas
nome <- tolower(NOME)
```