

readr

Fundamentos lenguajes: R

Alberto Torres Barrán y Irene Rodríguez Luján

2019-12-14

readr

Introducción

- Paquete para importar y exportar ficheros de texto
- Importar datos:
 - `read_csv()` , para ficheros CSV
 - `read_csv2()` , para ficheros CSV separados por ";"
 - `read_delim()` , para ficheros ASCII delimitados por otros caracteres distintos de "," y ";"
 - `read_tsv()` , para ficheros ASCII delimitados por tabuladores
 - `read_table()` , para ficheros ASCII delimitados por espacios
- Exportar datos: `write_csv()` . `write_csv2()` , etc.

Ejemplo

```
write_csv(mpg, "mpg.csv")
mpg1 <- read_csv("mpg.csv")
head(mpg1)
## # A tibble: 6 x 11
##   manufacturer model displ  year   cyl trans  drv    cty   hwy fl      class
##   <chr>          <chr> <dbl> <int> <int> <chr> <chr> <int> <int> <chr> <chr>
## 1 audi          a4      1.8  1999     4 auto(...) f      18    29 p      comp...
## 2 audi          a4      1.8  1999     4 manua... f      21    29 p      comp...
## 3 audi          a4      2    2008     4 manua... f      20    31 p      comp...
## 4 audi          a4      2    2008     4 auto(...) f      21    30 p      comp...
## 5 audi          a4      2.8  1999     6 auto(...) f      16    26 p      comp...
## 6 audi          a4      2.8  1999     6 manua... f      18    26 p      comp...
```

Directorio de trabajo

- Directorio donde apunta RStudio

```
getwd()  
## [1] "/home/irenerodriguez/git_repos/curso-uam-R/src"
```

- Se puede cambiar con `setwd()` o en la pestaña `Files` de RStudio
- Directorio por defecto donde se buscan los ficheros a importar
- Alternativamente, podemos especificar el path completo o usar la herramienta gráfica de RStudio

Missing values en R

- `NA` es una constante que representa valores que faltan (*missing values*)
- Puede estar contenida dentro de vectores (columnas) de cualquier tipo
- `is.na()` devuelve `TRUE` si el valor es `NA` y `FALSE` en caso contrario
- Muchas funciones de R tienen un parámetro opcional `na.rm` que ignora `NA`s

```
dia <-  
  diamonds %>%  
    mutate(y = ifelse(!between(y, 3, 20), NA, y))  
  
dia %>%  
  summarize(y_na = sum(is.na(y)))  
## # A tibble: 1 x 1  
##   y_na  
##   <int>  
## 1     9
```

```
dia %>%  
  summarize(avg_y = mean(y))  
## # A tibble: 1 x 1  
##   avg_y  
##   <dbl>  
## 1    NA
```

```
dia %>%  
  summarize(avg_y = mean(y, na.rm = TRUE))  
## # A tibble: 1 x 1  
##   avg_y  
##   <dbl>  
## 1  5.73
```

Parámetros opcionales

- `col_names` , si TRUE, la primera fila es el nombre de las variables. También se le puede pasar un vector de cadenas de caracteres con los nombres.
- `delim` , carácter que separa las columnas (solo en `read_delim()`)
- `na` , vector con cadenas que se interpretan como missing values. Por defecto `NA` y la cadena vacía.
- `col_types` , vector de clases para las columnas (ver documentación de `col()`). Por defecto se intenta adivinar el tipo de cada columna a partir de las 1000 primeras líneas.
- `n_max` , número máximo de líneas a leer del fichero
- `skip` , número de líneas a ignorar al principio del fichero.
- `locale` , parámetro que nos permite cambiar el encoding, separador decimal y formato de fechas (ver documentación de `locale()`)
- `comment` , una cadena de caracteres que identifica líneas de texto a ignorar (comentarios)
- `trim_ws` , si vale TRUE, se eliminan los espacios en blanco al principio y al final de cada campo

Libreria readxl

- Podemos listar las hojas de un fichero Excel:

```
library(readxl)
excel_ex <- readxl_example("datasets.xlsx")
excel_sheets(excel_ex)
## [1] "iris"      "mtcars"    "chickwts"  "quakes"
```

- Leer como tibble/dataframe:

```
read_excel(excel_ex, sheet = "mtcars")
## # A tibble: 32 x 11
##   mpg   cyl  disp    hp  drat    wt  qsec    vs  am  gear  carb
##   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1  21     6  160   110   3.9   2.62  16.5     0     1     4     4
## 2  21     6  160   110   3.9   2.88  17.0     0     1     4     4
## 3  22.8    4  108    93   3.85   2.32  18.6     1     1     4     1
## 4  21.4    6  258   110   3.08   3.22  19.4     1     0     3     1
## 5  18.7    8  360   175   3.15   3.44  17.0     0     0     3     2
## 6  18.1    6  225   105   2.76   3.46  20.2     1     0     3     1
## 7  14.3    8  360   245   3.21   3.57  15.8     0     0     3     4
## 8  24.4    4  147.    62   3.69   3.19   20      1     0     4     2
## 9  22.8    4  141.    95   3.92   3.15  22.9     1     0     4     2
## 10 19.2    6  168.   123   3.92   3.44  18.3     1     0     4     4
## # ... with 22 more rows
```


Parámetros útiles

- `range` : rango de celdas a importar, en lugar de la hoja completa (por ejemplo: "C3:F14")
- `sheet` : número o nombre de la hoja a leer. Por defecto la primera
- `col_names` : `TRUE` si la primera fila contiene los nombres de las columnas
- `na` : vector con cadenas que se interpretan como missing values. Por defecto celdas vacías
- `col_types` : tipo de cada columna. Por defecto se intenta inferir de los datos. Posibles valores: "skip", "guess", "logical", "numeric", "date", "text" or "list"

Otros formatos

- `readr` solo tiene funciones para importar ficheros de texto
- Para otros formatos, existen librerías específicas:
 - `haven`, para ficheros de SPSS, Stata y SaS
 - `DBI` junto con otro paquete específico dependiendo de la BD (`RMySQL`, `RSQLite`, etc.) nos permite hacer *querys* contra una BD
 - `jsonlite`, para ficheros JSON
 - `xml2`, para ficheros XML