Introducción al Tidyverse

intro-R

Febrero 2018

Secciones

Secciones

Data frames en *base R*

El objeto por excelencia para manipular datos es el data frame. Pero... ¿qué es exactamente?

Data frames en *base R*

El objeto por excelencia para manipular datos es el data frame. Pero... ¿qué es exactamente?

Data frames

Un data frame es un conjunto de observaciones de diferentes variables. Vamos, lo que viene siendo una tabla.

Data frames en *base R*

El objeto por excelencia para manipular datos es el data frame. Pero... ¿qué es exactamente?

Data frames

Un data frame es un conjunto de observaciones de diferentes variables. Vamos, lo que viene siendo una tabla.

R está hecho para trabajar con datos, fundamentalmente **vectores**. Aunque es posible hacer cosas increíbles con data frames en base R, existen **una serie de paquetes** que permiten hacer lo mismo más fácil y más rápido.

Secciones

¿Qué es el *Tidyverse*?

El tidyverse es un conjunto de paquetes desarrollados, casi en su totalidad, por Hadley Wickham. Se basa en un concepto fundamental:

Tidy data

Una estructura única y lógica para ordenar datos en una tabla.

Tidy data

Existen muchas maneras de colocar nuestros datos en una tabla. Los dos formatos más comunes:

- *Long* (también llamado **tidy**): Cada fila representa una observación y cada columna, una variable de esa observación.
- *Wide*: Podemos poner varias observaciones en una sola fila. Nos centramos en el sujeto de la observación, en lugar de en la observación en sí.

Tidy data

Existen muchas maneras de colocar nuestros datos en una tabla. Los dos formatos más comunes:

- *Long* (también llamado **tidy**): Cada fila representa una observación y cada columna, una variable de esa observación.
- *Wide*: Podemos poner varias observaciones en una sola fila. Nos centramos en el sujeto de la observación, en lugar de en la observación en sí.

Nombre	Ojos	1999	2000	2001
Mauricio	Marrones	1.23	1.32	1.38
Genoveva	Azules	1.31	1.35	1.41
Félix	Marrones	1.52	1.55	1.59
Isabel	Verdes	1.12	1.16	1.25

Table: Alturas de niños/as por año. Formato *wide*

Wide vs. long

Nombre	Ojos	Año	Altura
Mauricio	Marrones	1999	1.23
Mauricio	Marrones	2000	1.32
Mauricio	Marrones	2001	1.38
Genoveva	Azules	1999	1.31
Genoveva	Azules	2000	1.35
Genoveva	Azules	2001	1.41
Félix	Marrones	1999	1.52
Félix	Marrones	2000	1.55
Félix	Marrones	2001	1.59
Isabel	Verdes	1999	1.12
Isabel	Verdes	2000	1.16
Isabel	Verdes	2001	1.25

Table: Alturas de niños/as por año. Formato *long*

Ventajas

El formato tidy es menos intuitivo para humanos y más largo de escribir a mano, pero es inequívoco: cada observación es una fila, cada variable es una columna.

Messy data

"Tidy datasets are all alike but every messy dataset is messy in its own way." – Hadley Wickham

Ventajas

El formato tidy es menos intuitivo para humanos y más largo de escribir a mano, pero es inequívoco: cada observación es una fila, cada variable es una columna.

Messy data

"Tidy datasets are all alike but every messy dataset is messy in its own way." – Hadley Wickham

A la hora de manipular data frames, el tidyverse es:

- más rápido que base R
- proporciona soluciones sencillas a problemas complejos
- más fácil de entender

¿Cómo instalarlo?

- install.packages("tidyverse")
- library(tidyverse)

Secciones

magrittr

La única librería no diseñado por Hadley Wickham, magrittr tiene una única función: permitir el uso de **pipes**

magrittr

La única librería no diseñado por Hadley Wickham, magrittr tiene una única función: permitir el uso de **pipes**

Pipes

Una pipe es un operador especial que pasa el objeto a su izquierda como primer argumento de la función a su derecha.

magrittr

La única librería no diseñado por Hadley Wickham, magrittr tiene una única función: permitir el uso de **pipes**

Pipes

Una pipe es un operador especial que pasa el objeto a su izquierda como primer argumento de la función a su derecha.

Las pipes permiten escribir código de manera más natural:

```
## Sin pipe
x <- c(3, 6, 1)
round(exp(diff(log(x))), 1)

## Con pipe
x <- c(3, 6, 1)
x %>% log() %>% diff() %>% exp() %>% round(1)
```

readr

Lectura de tablas en formato rectangular: CSV, TSV...

- read_csv("tabla.csv")

Otro paquete similar, no incluido en el *tidyverse* es **readxl**.

tidyr

Conversión entre formatos *long* y *wide*. Dos funciones principales:

- **gather**: agrupar datos de varias columnas en una sola. Para pasar de wide a long
- **spread**: "extender" los datos de una columna en varias. Para pasar de long a wide.

tidyr

Conversión entre formatos *long* y *wide*. Dos funciones principales:

- **gather**: agrupar datos de varias columnas en una sola. Para pasar de wide a long
- **spread**: "extender" los datos de una columna en varias. Para pasar de long a wide.

```
alturas_long <- alturas_wide %>%
gather(key = "year", value = "altura", -nombre)
```

```
alturas_wide <- alturas_long %>%
spread(key = "year", value = "altura")
```

dplyr

La esencia del *tidyverse*. Proporciona una serie de funciones (verbos) que permiten manipular fácilmente nuestros data frames.

- **select**: selecciona las columnas que queramos.

```
alturas_long %>% select(year, altura) # Opción 1.
alturas_long %>% select(-nombre)      # Opción 2.
```

dplyr

La esencia del *tidyverse*. Proporciona una serie de funciones (verbos) que permiten manipular fácilmente nuestros data frames.

- **select**: selecciona las columnas que queramos.

```
alturas_long %>% select(year, altura) # Opción 1.
alturas_long %>% select(-nombre)      # Opción 2.
```

- **rename**: cambia los nombres de una o más columnas

```
alturas_long %>% rename(name = nombre, height = altura)
```

dplyr

La esencia del *tidyverse*. Proporciona una serie de funciones (verbos) que permiten manipular fácilmente nuestros data frames.

- **select**: selecciona las columnas que queramos.

```
alturas_long %>% select(year, altura) # Opción 1.
alturas_long %>% select(-nombre)      # Opción 2.
```

- **rename**: cambia los nombres de una o más columnas

```
alturas_long %>% rename(name = nombre, height = altura)
```

- **filter**: filtra las observaciones (filas) por una o más condiciones

```
alturas_long %>% filter(altura >= 1.40, year >= 2000)
```

dplyr

Más funciones:

- **mutate**: añade columnas sin eliminar las ya existentes.

  `alturas_long %>% mutate(altura_cuadrado = altura^2)2.`

dplyr

Más funciones:

- **mutate**: añade columnas sin eliminar las ya existentes.

```
alturas_long %>% mutate(altura_cuadrado = altura^2)2.
```

- **transmute**: añade columnas y elimina las ya existentes

```
alturas_long %>% transmute(altura_cuadrado = altura^2)
```

- **group_by** y **summarize**: calcula nuevas variables sobre los datos *agrupados*

```
alturas_long %>% group_by(ojos)
  %>% summarize(media_alturas = mean(altura))
```

Flujo de trabajo

La importancia de los datos

"In God we trust. All others must bring data." - W. Edwards Demming

Una vez que se tienen los datos necesarios, conviene seguir un flujo de trabajo organizado:

- Importar o leer datos {readr}
- Formatear (hacer tidy, en caso de que no lo estén) {tidyr}
- Análisis exploratorio (ver histogramas, contar NAs...) {base R}
- Limpiar datos (tratar outliers, NAs...) {dplyr}
- Manipular (crear/modificar variables) {dplyr}
- Modelizar, si fuese necesario {base R} o {purrr}
- Crear gráficos, si fuesen necesarios {ggplot2}
- Extraer conclusiones