

Manipulación de data.frame con dplyr

Luciano Selzer

28 June, 2018

La manipulación de datos es una de las operaciones más comunes:

- seleccionar observaciones
- seleccionar variables
- agrupar por características
- calcular estadísticas

Podemos hacerlo en R base:

```
mean(gapminder[gapminder$continent == "Africa", "gdpPercap"])
[1] 2193.755
mean(gapminder[gapminder$continent == "Americas", "gdpPercap"])
[1] 7136.11
mean(gapminder[gapminder$continent == "Asia", "gdpPercap"])
[1] 7902.15
```

Pero lleva mucha repetición

El paquete dplyr

El paquete **dplyr** tiene varias funciones útiles que:

- Reducen la repetición
- Reducen la probabilidad de cometer errores
- Reducen la cantidad de tipeo
- Es más fácil de leer

Vamos a usar las 6 más comunes y también el *pipe* (tubo) (`%>%`) para combinarlas:

1. `select()`
2. `filter()`
3. `group_by()`
4. `summarize()`
5. `mutate()`

Si no lo han instalado antes, háganlo con:

```
install.packages('dplyr')
```

Y carguen el paquete:

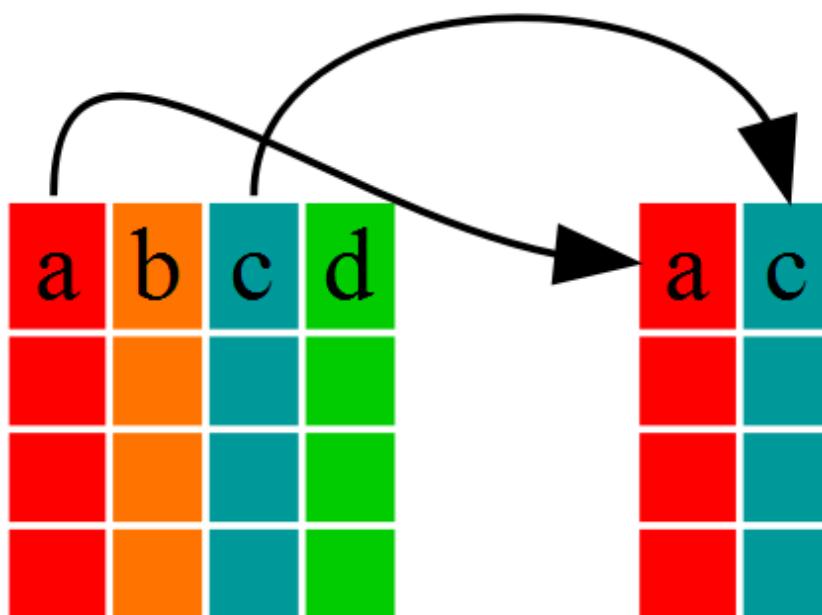
```
library(dplyr)
```

Usando select()

Seleccionar variables

```
year_country_gdp <- select(gapminder, year, country, gdpPercap)
```

select(data.frame,a,c)



El verdadero poder aparece usando los pipes

```
year_country_gdp <- gapminder %>% select(year, country, gdpPercap)
```

¿Cómo funcionan los pipes?

Usando filter()

Podemos filtrar usando filter()

```
year_country_gdp_euro <- gapminder %>%
  filter(continent == "Europe") %>%
  select(year, country, gdpPercap)
```

Ejercicio 1

Escribe en una sola operación (que puede tener varias líneas e incluir pipes) que produzca un data.frame que tenga los valores de África para lifeExp, country y year, pero no otros continentes.

¿Cuántas filas tiene tu data.frame? ¿Por qué?

Usando group_by() y summarize()

Podemos definir variables de agrupación usando group_by()

```
str(gapminder)
```

```
'data.frame': 1704 obs. of 6 variables:  
 $ country : Factor w/ 142 levels "Afghanistan",...: 1 1 1 1 1 1 1 1 1 1 ...  
 $ year    : int 1952 1957 1962 1967 1972 1977 1982 1987 1992 1997 ...  
 $ pop     : num 8425333 9240934 10267083 11537966 13079460 ...  
 $ continent: Factor w/ 5 levels "Africa","Americas",...: 3 3 3 3 3 3 3 3 3 3 ...  
 $ lifeExp : num 28.8 30.3 32 34 36.1 ...  
 $ gdpPercap: num 779 821 853 836 740 ...
```

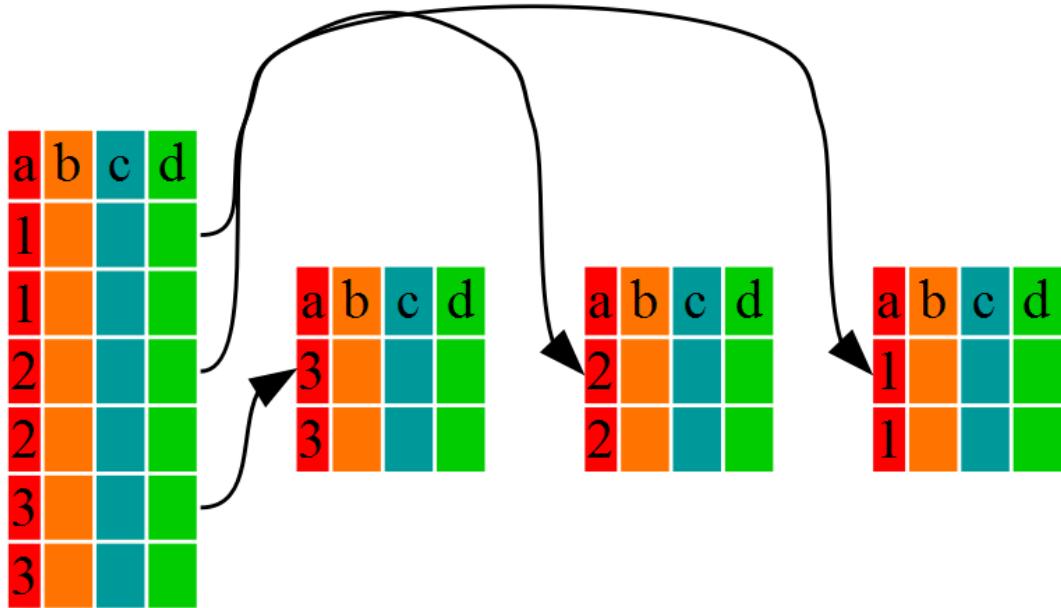
```
str(gapminder %>% group_by(continent))
```

```
Classes 'grouped_df', 'tbl_df', 'tbl' and 'data.frame': 1704 obs. of 6 variables:  
 $ country : Factor w/ 142 levels "Afghanistan",...: 1 1 1 1 1 1 1 1 1 1 ...  
 $ year    : int 1952 1957 1962 1967 1972 1977 1982 1987 1992 1997 ...  
 $ pop     : num 8425333 9240934 10267083 11537966 13079460 ...  
 $ continent: Factor w/ 5 levels "Africa","Americas",...: 3 3 3 3 3 3 3 3 3 3 ...  
 $ lifeExp : num 28.8 30.3 32 34 36.1 ...  
 $ gdpPercap: num 779 821 853 836 740 ...  
 - attr(*, "vars")= chr "continent"  
 - attr(*, "drop")= logi TRUE  
 - attr(*, "indices")=List of 5  
   ..$ : int 24 25 26 27 28 29 30 31 32 33 ...  
   ..$ : int 48 49 50 51 52 53 54 55 56 57 ...  
   ..$ : int 0 1 2 3 4 5 6 7 8 9 ...  
   ..$ : int 12 13 14 15 16 17 18 19 20 21 ...  
   ..$ : int 60 61 62 63 64 65 66 67 68 69 ...  
 - attr(*, "group_sizes")= int 624 300 396 360 24  
 - attr(*, "biggest_group_size")= int 624  
 - attr(*, "labels")='data.frame': 5 obs. of 1 variable:  
   ..$ continent: Factor w/ 5 levels "Africa","Americas",...: 1 2 3 4 5  
   ..- attr(*, "vars")= chr "continent"  
   ..- attr(*, "drop")= logi TRUE
```

La estructura no es igual que en original.

Una lista que contiene las filas que corresponden a un continente en particular.

gapminder %>% group_by(a)

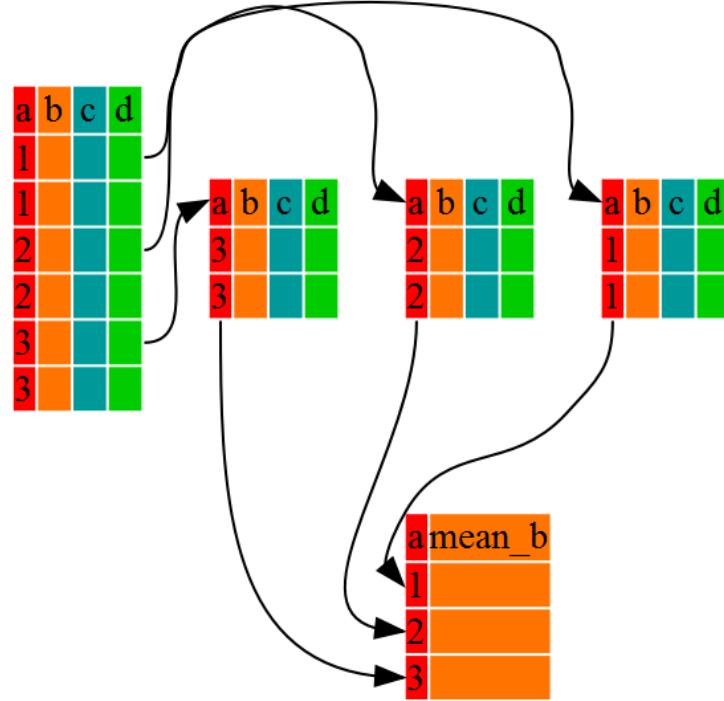


Usando summarize()

`group_by()` no es muy interesante por si solo. Es más interesante de usar con `summarize()`. Esto nos permite usar otras funciones como `mean` o `sd` de forma similar a `plyr`

```
gdp_bycontinents <- gapminder %>%
  group_by(continent) %>%
  summarize(mean_gdpPerCap = mean(gdpPerCap))
```

```
gapminder %>% group_by(a) %>% summarize(mean_b=mean(b))
```



Ejercicio 2

Calcula la expectativa de vida promedio por país ¿Cuál tiene la mayor? Y cuál la menor?

Nos permite calcular la media de PBI por continente pero se pueden hacer más cosas. También podemos agrupar por varias variables. Por ejemplo `year` y `continent`.

```
gdp_bycontinents_byyear <- gapminder %>%
  group_by(continent, year) %>%
  summarize(mean_gdpPercap = mean(gdpPercap))
```

Eso de por si es bastante útil, pero además podemos calcular ¡varias variables por vez!

```
gdp_pop_bycontinents_byyear <- gapminder %>%
  group_by(continent, year) %>%
  summarize(mean_gdpPercap = mean(gdpPercap),
           sd_gdpPercap = sd(gdpPercap),
           mean_pop = mean(pop),
           sd_pop = sd(pop))
```

Usando `mutate()`

Podemos crear nuevas antes de (o incluso después) de resumir la información usando `mutate()`.

```
gdp_pop_bycontinents_byyear <- gapminder %>%
  mutate(gdp_billion = gdpPercap*pop/10^9) %>%
  group_by(continent,year) %>%
  summarize(mean_gdpPercap = mean(gdpPercap),
            sd_gdpPercap = sd(gdpPercap),
            mean_pop = mean(pop),
            sd_pop = sd(pop),
            mean_gdp_billion = mean(gdp_billion),
            sd_gdp_billion = sd(gdp_billion))
```

Ejercicio Avanzdo

Calcula la expectativa de vida promedio en 2002 para dos países seleccionados al azar de cada continente. Luego ordena los nombres de los continentes en el orden inverso a la expectativa de vida.

Pista: usa las funciones de `dplyr`, `arrange()` y `sample_n()`, tienen una sintaxis similar a otras funciones del paquete.

Otros recursos muy buenos

- Data Wrangling Cheat sheet
- Introduction to dplyr