

dplyr

Entornos de Análisis de Datos: R

Alberto Torres Barrán

2020-02-02

dplyr

Introducción

- Implementa una gramática para realizar operaciones básicas con data frames
- Muy eficiente
- Operaciones principales: `slice`, `filter`, `select`, `arrange`, `mutate` y `summarize`
- Estas operaciones se pueden componer para realizar otras más complejas

slice

Selecciona filas por su posición

```
slice(mpg, 1:5)
## # A tibble: 5 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(l5)  f      18    29 p    compa
## 2 audi         a4      1.8  1999     4 manual(m5) f      21    29 p    compa
## 3 audi         a4      2    2008     4 manual(m6) f      20    31 p    compa
## 4 audi         a4      2    2008     4 auto(av)  f      21    30 p    compa
## 5 audi         a4      2.8  1999     6 auto(l5)  f      16    26 p    compa
```

filter

Selecciona filas por condición

```
filter(mpg, model == "a4")  
## # A tibble: 7 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(l5) f      18    29 p    compa  
## 2 audi          a4      1.8  1999     4 manual(m5) f      21    29 p    compa  
## 3 audi          a4      2    2008     4 manual(m6) f      20    31 p    compa  
## 4 audi          a4      2    2008     4 auto(av) f      21    30 p    compa  
## 5 audi          a4      2.8  1999     6 auto(l5) f      16    26 p    compa  
## 6 audi          a4      2.8  1999     6 manual(m5) f      18    26 p    compa  
## 7 audi          a4      3.1  2008     6 auto(av) f      18    27 p    compa
```

Operadores lógicos

- R implementa todos los operadores relacionales habituales `>`, `<`, `>=`, `<=`, `==`, `!=`
- Los operadores lógicos son la negación `!`, and `&` y or `|`
- El resultado de todas estas operaciones son valores lógicos `TRUE` (`T`) o `FALSE` (`F`)

filter (cont.)

- Se pueden combinar multiples condiciones separadas por `,` (and lógico)

```
filter(mpg, model == "a4", cyl >= 5)
```

- También se puede usar explicitamente el operador

```
filter(mpg, model == "a4" & cyl >= 5)
```

- En el caso del or lógico es obligatorio el uso del operador

```
filter(mpg, model == "a4" | model == "mustang")
```

select

Seleccionar variables (columnas) de un data frame

```
select(mpg, model, displ, cyl)
## # A tibble: 234 x 3
##   model      displ    cyl
##   <chr>      <dbl> <int>
## 1 a4          1.8     4
## 2 a4          1.8     4
## 3 a4          2      4
## 4 a4          2      4
## 5 a4          2.8     6
## 6 a4          2.8     6
## 7 a4          3.1     6
## 8 a4 quattro  1.8     4
## 9 a4 quattro  1.8     4
## 10 a4 quattro  2      4
## # ... with 224 more rows
```

Ignorar variables

Con un `-` se ignoran variables

```
select(mpg, -manufacturer)
## # A tibble: 234 x 10
##   model      displ  year   cyl trans      drv      cty   hwy fl      class
##   <chr>      <dbl> <int> <int> <chr>      <chr> <int> <int> <chr>  <chr>
## 1 a4          1.8  1999     4 auto(l5)    f       18    29 p      compact
## 2 a4          1.8  1999     4 manual(m5)  f       21    29 p      compact
## 3 a4          2    2008     4 manual(m6)  f       20    31 p      compact
## 4 a4          2    2008     4 auto(av)    f       21    30 p      compact
## 5 a4          2.8  1999     6 auto(l5)    f       16    26 p      compact
## 6 a4          2.8  1999     6 manual(m5)  f       18    26 p      compact
## 7 a4          3.1  2008     6 auto(av)    f       18    27 p      compact
## 8 a4 quattro  1.8  1999     4 manual(m5)  4       18    26 p      compact
## 9 a4 quattro  1.8  1999     4 auto(l5)    4       16    25 p      compact
## 10 a4 quattro  2    2008     4 manual(m6)  4       20    28 p      compact
## # ... with 224 more rows
```


Rangos

Puesto que las variables están ordenadas, se puede seleccionar un rango con :

```
select(mpg, model:trans)
## # A tibble: 234 x 5
##   model      displ  year   cyl trans
##   <chr>      <dbl> <int> <int> <chr>
## 1 a4          1.8  1999     4 auto(15)
## 2 a4          1.8  1999     4 manual(m5)
## 3 a4          2    2008     4 manual(m6)
## 4 a4          2    2008     4 auto(av)
## 5 a4          2.8  1999     6 auto(15)
## 6 a4          2.8  1999     6 manual(m5)
## 7 a4          3.1  2008     6 auto(av)
## 8 a4 quattro  1.8  1999     4 manual(m5)
## 9 a4 quattro  1.8  1999     4 auto(15)
## 10 a4 quattro  2    2008     4 manual(m6)
## # ... with 224 more rows
```

Funciones auxiliares

Las siguientes funciones se pueden usar dentro de `select()`

- `starts_with()` : empiezan con un prefijo
- `ends_with()` : terminan con un sufijo
- `contains()` : contienen una string
- `matches()` : concuerdan con una expresión regular
- `num_range()` : rango numérico como "X01", "X02", "X03"

arrange

Ordena las filas de un data frame

```
arrange(mpg, desc(year), cyl)
## # A tibble: 234 x 11
##   manufacturer model      displ  year   cyl trans      drv      cty   hwy fl
##   <chr>          <chr>    <dbl> <int> <int> <chr>    <chr> <int> <int> <chr>
## 1 audi          a4              2    2008     4 manual(m6) f      20    31 p
## 2 audi          a4              2    2008     4 auto(av)   f      21    30 p
## 3 audi          a4 quattro      2    2008     4 manual(m6) 4      20    28 p
## 4 audi          a4 quattro      2    2008     4 auto(s6)   4      19    27 p
## 5 chevrolet     malibu         2.4    2008     4 auto(l4)   f      22    30 r
## 6 honda         civic          1.8    2008     4 manual(m5) f      26    34 r
## 7 honda         civic          1.8    2008     4 auto(l5)   f      25    36 r
## 8 honda         civic          1.8    2008     4 auto(l5)   f      24    36 c
## 9 honda         civic              2    2008     4 manual(m6) f      21    29 p
## 10 hyundai      sonata         2.4    2008     4 auto(l4)   f      21    30 r
## # ... with 224 more rows
```

mutate

Añade nuevas variables (columnas) al data frame como combinación de las ya existentes

```
mutate(mpg, avg_mpg = (cty+hwy)/2)
## # A tibble: 234 x 12
##   manufacturer model    displ  year   cyl trans  drv    cty   hwy fl  clas
##   <chr>          <chr>    <dbl> <int> <int> <chr> <chr> <int> <int> <chr> <chr>
## 1 audi          a4        1.8  1999     4 auto(l~ f    18    29 p    comp
## 2 audi          a4        1.8  1999     4 manual~ f    21    29 p    comp
## 3 audi          a4        2    2008     4 manual~ f    20    31 p    comp
## 4 audi          a4        2    2008     4 auto(a~ f    21    30 p    comp
## 5 audi          a4        2.8  1999     6 auto(l~ f    16    26 p    comp
## 6 audi          a4        2.8  1999     6 manual~ f    18    26 p    comp
## 7 audi          a4        3.1  2008     6 auto(a~ f    18    27 p    comp
## 8 audi          a4 quat~  1.8  1999     4 manual~ 4    18    26 p    comp
## 9 audi          a4 quat~  1.8  1999     4 auto(l~ 4    16    25 p    comp
## 10 audi         a4 quat~  2    2008     4 manual~ 4    20    28 p    comp
## # ... with 224 more rows
```

Operadores y funciones aritméticas

- R implementa los operadores aritméticos habituales
 - suma `+`
 - resta `-`
 - multiplicación `*`
 - división `/`
 - exponenciación `^`
 - división entera `%/%`
 - módulo (resto) `%%`
- También las funciones aritméticas comunes: `log()`, `exp()`, `sin()`, `cos()`, `tan()`, `cumsum()`, `cumprod()`, `abs()`, `sqrt()`, `round()`, `ceiling()`, `floor()`, `trunc()`, ...
- Operan sobre vectores (columnas de un data frame) elemento a elemento

summarize

Colapsa el data frame a una única fila

```
summarize(mpg, max_cyl = max(cyl), avg_cty = mean(cty), min_year = min(year))  
## # A tibble: 1 x 3  
##   max_cyl avg_cty min_year  
##   <int>   <dbl>   <int>  
## 1      8    16.9    1999
```

Funciones de agregación

- Las funciones más comunes para usar dentro de `summarize()` son:
 - Aritméticas: `prod()`, `sum()`
 - Centralidad: `mean()`, `median()`
 - Dispersión: `sd()`, `var()`, `mad()`
 - Rango: `max()`, `min()`, `quantile()`
 - Posición: `first()`, `last()`, `nth()`
 - Lógicas: `any()`, `all()`
 - *Conteo*: `n()`, `n_distinct()` (solo se pueden usar dentro de `summarize()`)
- Todas reducen un vector de números a un único resultado

Concatenación de funciones

- Todas las funciones de `dplyr` toman como primer argumento un data frame y devuelven otro data frame
- Se pueden aplicar de manera consecutiva:

```
arrange(select(filter(mpg, model == "a4"), model, year), year)

arrange(
  select(
    filter(mpg, model == "a4"),
    model, year
  ),
  year
)
```


Concatenación de funciones (cont.)

-Otra opción:

```
df1 <- filter(mpg, model == "a4")  
df2 <- select(df1, model, year)  
df3 <- arrange(df2, year)
```

- Habitualmente no nos interesan los valores intermedios, solo el resultado final

Operador "tubería" (*pipe*)

- La sintaxis es `%>%` y permite reescribir el código anterior como

```
mpg %>%  
  filter(model == "a4") %>%  
  select(model, year) %>%  
  arrange(year)
```

- En general el código `df %>% foo()` es equivalente a `foo(df)`
- Esto permite concatenar funciones sin almacenar resultados intermedios y siguiendo el orden lógico