Más funciones de dplyr

Sesión 2

Natalie Julian - www.nataliejulian.com

Estadística UC y Data Scientist en Zippedi Inc.

Datos `mtcars`

Trabajaremos en esta sesión complementaria con los datos `mtcars`. Para acceder a estos basta con utilizar:

```
data(mtcars)
```

Y se guardará un objeto de tipo dataframe llamado mtcars:

```
class(mtcars)
[1] "data.frame"
```

count y n(): Contando casos

Contar casos

La variable `vs` de los datos `mtcars` toma los valores 1 (si el motor del auto es recto) y 0 (si el motor del auto es en forma de V). ¿Cómo podríamos contar cuántos casos por tipo de motor hay?

Usualmente podemos utilizar la función `table()` que entrega una tabla de frecuencias de la variable vs:

```
table(mtcars$vs)
```

```
 0  1
18 14
```

Contar casos

Con `dplyr` podemos utilizar las funciones `n()` y `count()`:

```
mtcars%>%
  group_by(vs)%>%
  count()
# A tibble: 2 x 2
     vs     n
  <dbl> <int>
1     0    18
2     1    14

mtcars%>%
  group_by(vs)%>%
  summarise(n=n())
# A tibble: 2 x 2
     vs     n
  <dbl> <int>
1     0    18
2     1    14
```

UNGROUP: DESAGRUPAR

Group by

Cuando agrupamos, las estadísticas se calculan en base a esta agrupación, por ejemplo:

```
mtcars%>%
  group_by(vs)%>%
  mutate(prom=mean(disp)) #Calcula promedio de disp para vs=0 y vs=1
# A tibble: 32 x 12
# Groups:   vs [2]
     mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb  prom
   <dbl> <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  307.
 2  21       6  160    110  3.9   2.88  17.0     0     1     4     4  307.
 3  22.8     4  108     93  3.85  2.32  18.6     1     1     4     1  132.
 4  21.4     6  258    110  3.08  3.22  19.4     1     0     3     1  132.
```

Pero, ¿qué pasa si quisiera ahora añadir una columna con el promedio de wt de todos los registros?

Luego de group by, R piensa en ese nivel de agregación

Si sólo añadimos `mutate(promwt=mean(wt))` no obtendremos lo que queremos, pues se está respetando la agrupación anterior:

```
mtcars%>%
  group_by(vs)%>%
  mutate(prom=mean(disp)) %>% #Calcula promedio de disp para vs=0 y vs=1
  mutate(promwt=mean(wt))  #Calcula promedio de wt para vs=0 y vs=1
# A tibble: 32 x 13
# Groups:   vs [2]
     mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb  prom promwt
   <dbl> <dbl> <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  307.   3.69
 2  21       6  160    110  3.9   2.88  17.0     0     1     4     4  307.   3.69
 3  22.8     4  108     93  3.85  2.32  18.6     1     1     4     1  132.   2.61
 4  21.4     6  258    110  3.08  3.22  19.4     1     0     3     1  132.   2.61
```

Efecto de ungroup

Si utilizamos ungroup ya no tendremos este problema:

```
mtcars%>%
  group_by(vs)%>%
  mutate(prom=mean(disp)) %>% #Calcula promedio de disp para vs=0 y vs=1
  ungroup()%>%
  mutate(promwt=mean(wt))  #Calcula promedio de wt para todas las observaciones
# A tibble: 32 x 13
     mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb  prom promwt
   <dbl> <dbl> <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  307.   3.22
 2  21       6  160    110  3.9   2.88  17.0     0     1     4     4  307.   3.22
 3  22.8     4  108     93  3.85  2.32  18.6     1     1     4     1  132.   3.22
 4  21.4     6  258    110  3.08  3.22  19.4     1     0     3     1  132.   3.22
 5  18.7     8  360    175  3.15  3.44  17.0     0     0     3     2  307.   3.22
```

¿Qué es lo que hacemos con ungroup? De cierta forma, se divide la tubería en dos caminos o subtuberías, la primera rama respeta la agrupación y obtiene resultados en base a la agrupación, y la segunda rama considera los datos no agrupados.

sample_n y sample_frac: Muestrear

sample_n

Supongamos que nos interesa muestrear 10 observaciones de los datos, esto es bastante sencillo:

```
mtcars%>%
sample_n(10)
```

También podemos muestrear 10 observaciones luego de realizar una agrupación:

```
mtcars%>%
  group_by(vs)%>%
  sample_n(10)
```

Incluso, podemos asignar prioridad (o pesos) a cada observación dependiendo de una variable en particular:

```
mtcars%>%
  group_by(vs)%>%
  sample_n(10, weight = 1/gear)
```

sample_frac

Pero...siendo bastante realistas, no siempre tenemos la misma cantidad de observaciones por grupo, por lo cual, dejar un número fijo para muestrear quizás no sea siempre la mejor opción.

Supongamos que en realidad, lo que necesitamos es muestrear el 30% de las observaciones por grupo (es decir, la cantidad de observaciones muestreadas por grupo será proporcional a la cantidad de observaciones totales en cada grupo). Esto lo podemos realizar fácilmente, indicando la fracción que queremos por grupo:

```
mtcars%>%
  group_by(vs)%>%
  sample_frac(0.3, weight = 1/gear)
# A tibble: 9 x 11
# Groups:   vs [2]
    mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
1  10.4     8 460     215  3     5.42  17.8     0     0     3     4
2  18.7     8 360     175  3.15  3.44  17.0     0     0     3     2
3  19.7     6 145     175  3.62  2.77  15.5     0     1     5     6
4  21       6 160     110  3.9   2.62  16.5     0     1     4     4
5  21       6 160     110  3.9   2.88  17.0     0     1     4     4
6  22.8     4 141.     95  3.92  3.15  22.9     1     0     4     2
7  21.5     4 120.     97  3.7   2.46  20.0     1     0     3     1
8  27.3     4  79      66  4.08  1.94  18.9     1     1     4     1
9  32.4     4  78.7    66  4.08  2.2   19.5     1     1     4     1
```

ROWNAMES_TO_COLUMN

Filas con nombres

Seguramente ya lo notaste, los datos `mtcars` corresponden a autos y el modelo de cada auto está como nombre de fila (no explícitamente como columna). Podría ser útil tener esta información como variable. ¿Cómo lograrlo? Con la función de tidyverse `rownames_to_column`:

```
(mtcars<-mtcars%>%
  rownames_to_column())
               rowname  mpg cyl  disp  hp drat    wt  qsec vs am gear
1            Mazda RX4 21.0   6 160.0 110 3.90 2.620 16.46  0  1    4
2        Mazda RX4 Wag 21.0   6 160.0 110 3.90 2.875 17.02  0  1    4
3           Datsun 710 22.8   4 108.0  93 3.85 2.320 18.61  1  1    4
4       Hornet 4 Drive 21.4   6 258.0 110 3.08 3.215 19.44  1  0    3
5    Hornet Sportabout 18.7   8 360.0 175 3.15 3.440 17.02  0  0    3
6              Valiant 18.1   6 225.0 105 2.76 3.460 20.22  1  0    3
7           Duster 360 14.3   8 360.0 245 3.21 3.570 15.84  0  0    3
8            Merc 240D 24.4   4 146.7  62 3.69 3.190 20.00  1  0    4
9             Merc 230 22.8   4 140.8  95 3.92 3.150 22.90  1  0    4
```

¿De qué me podría servir tener el modelo del auto?

Por ejemplo, quizás nos interesa filtrar por ciertas marcas. Por ejemplo, supongamos nos interesan los autos de la marca Toyota o Mazda. Extraemos los registros respectivos a los autos de la siguiente forma:

```
mtcars%>%
  filter(grepl("Mazda|Toyota", rowname))
 #grepl indica si se encuentra el texto Mazda o Toyota
 # en la variable rowname
```

```
         rowname  mpg cyl  disp  hp drat    wt  qsec vs am gear carb
1      Mazda RX4 21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
2  Mazda RX4 Wag 21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
3 Toyota Corolla 33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
4  Toyota Corona 21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
```

`recode`: Recodificar variables

recode

Recordemos que la variable vs toma los valores 1 si el motor del auto es recto y 0 si el motoro del auto es en forma de V. Podríamos recodificar directamente la variable de 1 y 0 a Motor recto y Motor forma V respectivamente. Podríamos hacerlo con `ifelse` (¿Recuerdas que lo vimos en el curso R basics?):

```
mtcars%>%
  mutate(vs=ifelse(vs==1, "Motor Recto", "Motor Forma V"))
```

O también con `recode`:

```
mtcars%>%
  mutate(vs=recode(vs, "0"="Motor Forma V", "1"="Motor Recto"))
              rowname  mpg cyl  disp  hp drat    wt  qsec            vs am gear carb
1           Mazda RX4 21.0   6 160.0 110 3.90 2.620 16.46 Motor Forma V  1    4    4
2       Mazda RX4 Wag 21.0   6 160.0 110 3.90 2.875 17.02 Motor Forma V  1    4    4
3          Datsun 710 22.8   4 108.0  93 3.85 2.320 18.61   Motor Recto  1    4    1
4      Hornet 4 Drive 21.4   6 258.0 110 3.08 3.215 19.44   Motor Recto  0    3    1
5   Hornet Sportabout 18.7   8 360.0 175 3.15 3.440 17.02 Motor Forma V  0    3    2
6             Valiant 18.1   6 225.0 105 2.76 3.460 20.22   Motor Recto  0    3    1
7          Duster 360 14.3   8 360.0 245 3.21 3.570 15.84 Motor Forma V  0    3    4
```

case_when

Y también podríamos utilizar `case_when`:

```
mtcars%>%
  mutate(vs=case_when(vs==1 ~ "Motor Recto", TRUE~ "Motor Forma V"))
                rowname  mpg cyl  disp  hp drat    wt  qsec           vs am gear carb
1           Mazda RX4 21.0   6 160.0 110 3.90 2.620 16.46 Motor Forma V  1    4    4
2       Mazda RX4 Wag 21.0   6 160.0 110 3.90 2.875 17.02 Motor Forma V  1    4    4
3          Datsun 710 22.8   4 108.0  93 3.85 2.320 18.61   Motor Recto  1    4    1
4      Hornet 4 Drive 21.4   6 258.0 110 3.08 3.215 19.44   Motor Recto  0    3    1
5   Hornet Sportabout 18.7   8 360.0 175 3.15 3.440 17.02 Motor Forma V  0    3    2
6             Valiant 18.1   6 225.0 105 2.76 3.460 20.22   Motor Recto  0    3    1
7          Duster 360 14.3   8 360.0 245 3.21 3.570 15.84 Motor Forma V  0    3    4
```

Entregando los mismos resultados! ¿Fácil no?

cut: Agrupar variables numéricas

cut

Supongamos que queremos categorizar la variable `hp` como sigue:

- Si hp $\in (50, 122]$ indique *bajo*
- si hp $\in (122, 180]$ indique *alto*
- si hp > 180 indique *potente*

Esto se puede hacer facilmente con la función `cut`:

```
mtcars%>%
  mutate(categoriahp = cut(hp,
                           breaks = c(50, 122, 180, Inf),
                           labels = c("Bajo", "Alto", "Potente"),
                           right = TRUE))
              rowname  mpg cyl  disp  hp drat    wt  qsec vs am gear carb categoriahp
1           Mazda RX4 21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4        Bajo
2       Mazda RX4 Wag 21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4        Bajo
3          Datsun 710 22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1        Bajo
4      Hornet 4 Drive 21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1        Bajo
5   Hornet Sportabout 18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2        Alto
6             Valiant 18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1        Bajo
7          Duster 360 14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4     Potente
8           Merc 240D 24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2        Bajo
```

¿Y si no tengo las categorías? ¿Puedo obtenerlas?

Así es! Puedes utilizar la función `cut_width` y definir cuál es el ancho del intervalo deseado:

```
(mtcars<-mtcars%>%
  mutate(categoria2hp=cut_width(hp, width=60)))
               rowname  mpg cyl  disp  hp drat    wt  qsec vs am gear carb categoria2hp
1            Mazda RX4 21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4     (90,150]
2        Mazda RX4 Wag 21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4     (90,150]
3           Datsun 710 22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1     (90,150]
4       Hornet 4 Drive 21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1     (90,150]
5    Hornet Sportabout 18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2    (150,210]
6              Valiant 18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1     (90,150]
7           Duster 360 14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4    (210,270]
8            Merc 240D 24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2      [30,90]
9            Merc 230 22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2     (90,150]
10           Merc 280 19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4     (90,150]
11           Merc 280C 17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4     (90,150]
```

Y podemos filtrar por esta nueva categoría:

```
mtcars%>%
  filter(categoria2hp %in% c("[30,90]", "(150,210]"))
              rowname  mpg cyl  disp  hp drat    wt  qsec vs am gear carb categoria2hp
1   Hornet Sportabout 18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2    (150,210]
2           Merc 240D 24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2      [30,90]
3          Merc 450SE 16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3    (150,210]
4          Merc 450SL 17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3    (150,210]
5         Merc 450SLC 15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3    (150,210]
6  Cadillac Fleetwood 10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4    (150,210]
7            Fiat 128 32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1      [30,90]
8         Honda Civic 30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2      [30,90]
9      Toyota Corolla 33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1      [30,90]
10   Pontiac Firebird 19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2    (150,210]
11          Fiat X1-9 27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1      [30,90]
12       Ferrari Dino 19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6    (150,210]
```