tidyverse. dplyr

Oleksii Yehorchenkov

2023-11-20

This tutorial is based on https://genomicsclass.github.io/book/pages/dplyr_tutorial.html (https://genomicsclass.github.io/book/pages/dplyr_tutorial.html)

Data: mammals sleep

```
library(dplyr)
library(readr)

msleep <- read_csv(url("https://raw.githubusercontent.com/genomicsclass/dagdata/master/inst/extdata/msleep_ggplot2.csv"))
```

Data: The msleep (mammals sleep) data set contains the sleeptimes and weights for a set of mammals and is available in the dagdata repository on github. This data set contains 83 rows and 11 variables.

Let's take a look to the dataset

```
glimpse(msleep)
```

```
## Rows: 83
## Columns: 11
## $ name         <chr> "Cheetah", "Owl monkey", "Mountain beaver", "Greater shor…
## $ genus        <chr> "Acinonyx", "Aotus", "Aplodontia", "Blarina", "Bos", "Bra…
## $ vore         <chr> "carni", "omni", "herbi", "omni", "herbi", "herbi", "carn…
## $ order        <chr> "Carnivora", "Primates", "Rodentia", "Soricomorpha", "Art…
## $ conservation <chr> "lc", NA, "nt", "lc", "domesticated", NA, "vu", NA, "dome…
## $ sleep_total  <dbl> 12.1, 17.0, 14.4, 14.9, 4.0, 14.4, 8.7, 7.0, 10.1, 3.0, 5…
## $ sleep_rem    <dbl> NA, 1.8, 2.4, 2.3, 0.7, 2.2, 1.4, NA, 2.9, NA, 0.6, 0.8, …
## $ sleep_cycle  <dbl> NA, NA, NA, 0.1333333, 0.6666667, 0.7666667, 0.3833333, N…
## $ awake        <dbl> 11.9, 7.0, 9.6, 9.1, 20.0, 9.6, 15.3, 17.0, 13.9, 21.0, 1…
## $ brainwt      <dbl> NA, 0.01550, NA, 0.00029, 0.42300, NA, NA, NA, 0.07000, 0…
## $ bodywt       <dbl> 50.000, 0.480, 1.350, 0.019, 600.000, 3.850, 20.490, 0.04…
```

The columns (in order) correspond to the following:

column name	Description
name	common name
genus	taxonomic rank
vore	carnivore, omnivore or herbivore?
order	taxonomic rank
conservation	the conservation status of the mammal

column name	Description
sleep_total	total amount of sleep, in hours
sleep_rem	rem sleep, in hours
sleep_cycle	length of sleep cycle, in hours
awake	amount of time spent awake, in hours
brainwt	brain weight in kilograms
bodywt	body weight in kilograms

Important dplyr verbs to remember

dplyr verbs	Description
select()	select columns
filter()	filter rows
arrange()	re-order or arrange rows
mutate()	create new columns
summarise()	summarise values
group_by()	allows for group operations in the “split-apply-combine” concept

dplyr verbs in action

The two most basic functions are `select()` and `filter()` which selects columns and filters rows, respectively.

Selecting columns using select()

Select a set of columns: the name and the sleep_total columns.

```
sleepData <- select(msleep, name, sleep_total)
head(sleepData)
```

```
## # A tibble: 6 × 2
##   name                          sleep_total
##   <chr>                               <dbl>
## 1 Cheetah                              12.1
## 2 Owl monkey                           17
## 3 Mountain beaver                      14.4
## 4 Greater short-tailed shrew           14.9
## 5 Cow                                   4
## 6 Three-toed sloth                     14.4
```

To select all the columns except a specific column, use the “-” (subtraction) operator (also known as negative indexing)

```
head(select(msleep, -name))
```

```
## # A tibble: 6 × 10
##   genus      vore  order    conservation sleep_total sleep_rem sleep_cycle awake
##   <chr>      <chr> <chr>    <chr>              <dbl>     <dbl>       <dbl> <dbl>
## 1 Acinonyx   carni Carnivo… lc                  12.1      NA        NA      11.9
## 2 Aotus      omni  Primates <NA>                17         1.8      NA       7
## 3 Aplodontia herbi Rodentia nt                  14.4       2.4      NA       9.6
## 4 Blarina    omni  Soricom… lc                  14.9       2.3       0.133   9.1
## 5 Bos        herbi Artioda… domesticated         4         0.7       0.667  20
## 6 Bradypus   herbi Pilosa   <NA>                14.4       2.2       0.767   9.6
## # ℹ 2 more variables: brainwt <dbl>, bodywt <dbl>
```

```
head(select(msleep, -c("name", "sleep_total")))
```

```
## # A tibble: 6 × 9
##   genus    vore  order conservation sleep_rem sleep_cycle awake  brainwt  bodywt
##   <chr>    <chr> <chr> <chr>            <dbl>       <dbl> <dbl>    <dbl>   <dbl>
## 1 Acinonyx carni Carn… lc                NA        NA      11.9 NA        50
## 2 Aotus    omni  Prim… <NA>               1.8      NA       7    0.0155    0.48
## 3 Aplodon… herbi Rode… nt                 2.4      NA       9.6 NA         1.35
## 4 Blarina  omni  Sori… lc                 2.3       0.133   9.1  0.00029   0.019
## 5 Bos      herbi Arti… domesticated       0.7       0.667  20    0.423   600
## 6 Bradypus herbi Pilo… <NA>               2.2       0.767   9.6 NA         3.85
```

To select a range of columns by name, use the ":" (colon) operator

```
head(select(msleep, name:order))
```

```
## # A tibble: 6 × 4
##   name                        genus      vore  order
##   <chr>                       <chr>      <chr> <chr>
## 1 Cheetah                     Acinonyx   carni Carnivora
## 2 Owl monkey                  Aotus      omni  Primates
## 3 Mountain beaver             Aplodontia herbi Rodentia
## 4 Greater short-tailed shrew Blarina    omni  Soricomorpha
## 5 Cow                         Bos        herbi Artiodactyla
## 6 Three-toed sloth            Bradypus   herbi Pilosa
```

To select all columns that start with the character string "sl", use the function `starts_with()`

```
head(select(msleep, starts_with("sl")))
```

```
## # A tibble: 6 × 3
##   sleep_total sleep_rem sleep_cycle
##         <dbl>     <dbl>       <dbl>
## 1        12.1      NA        NA
## 2        17         1.8      NA
## 3        14.4       2.4      NA
## 4        14.9       2.3       0.133
## 5         4         0.7       0.667
## 6        14.4       2.2       0.767
```

Some additional options to select columns based on a specific criteria include

`ends_with()` = Select columns that end with a character string `contains()` = Select columns that contain a character string `matches()` = Select columns that match a regular expression `one_of()` = Select columns names that are from a group of names

Selecting rows using filter()

Filter the rows for mammals that sleep a total of more than 16 hours.

```
filter(msleep, sleep_total >= 16)
```

```
## # A tibble: 8 × 11
##   name    genus vore  order conservation sleep_total sleep_rem sleep_cycle awake
##   <chr>   <chr> <chr> <chr> <chr>              <dbl>     <dbl>       <dbl> <dbl>
## 1 Owl mo… Aotus omni  Prim… <NA>                17         1.8      NA       7
## 2 Long-n… Dasy… carni Cing… lc                  17.4       3.1       0.383   6.6
## 3 North … Dide… omni  Dide… lc                  18         4.9       0.333   6
## 4 Big br… Epte… inse… Chir… lc                  19.7       3.9       0.117   4.3
## 5 Thick-… Lutr… carni Dide… lc                  19.4       6.6      NA       4.6
## 6 Little… Myot… inse… Chir… <NA>                19.9       2         0.2     4.1
## 7 Giant … Prio… inse… Cing… en                  18.1       6.1      NA       5.9
## 8 Arctic… Sper… herbi Rode… lc                  16.6      NA        NA       7.4
## # ℹ 2 more variables: brainwt <dbl>, bodywt <dbl>
```

Filter the rows for mammals that sleep a total of more than 16 hours and have a body weight of greater than 1 kilogram.

```
filter(msleep, sleep_total >= 16, bodywt >= 1)
```

```
## # A tibble: 3 × 11
##   name    genus vore  order conservation sleep_total sleep_rem sleep_cycle awake
##   <chr>   <chr> <chr> <chr> <chr>              <dbl>     <dbl>       <dbl> <dbl>
## 1 Long-n… Dasy… carni Cing… lc                  17.4       3.1       0.383   6.6
## 2 North … Dide… omni  Dide… lc                  18         4.9       0.333   6
## 3 Giant … Prio… inse… Cing… en                  18.1       6.1      NA       5.9
## # ℹ 2 more variables: brainwt <dbl>, bodywt <dbl>
```

Filter the rows for mammals in the Perissodactyla and Primates taxonomic order

```
filter(msleep, order %in% c("Perissodactyla", "Primates"))
```

```
## # A tibble: 15 × 11
##    name   genus vore  order conservation sleep_total sleep_rem sleep_cycle awake
##    <chr>  <chr> <chr> <chr> <chr>              <dbl>     <dbl>       <dbl> <dbl>
##  1 Owl m… Aotus omni  Prim… <NA>                17         1.8      NA       7
##  2 Grivet Cerc… omni  Prim… lc                  10         0.7      NA      14
##  3 Horse  Equus herbi Peri… domesticated         2.9       0.6       1      21.1
##  4 Donkey Equus herbi Peri… domesticated         3.1       0.4      NA      20.9
##  5 Patas… Eryt… omni  Prim… lc                  10.9       1.1      NA      13.1
##  6 Galago Gala… omni  Prim… <NA>                 9.8       1.1       0.55   14.2
##  7 Human  Homo  omni  Prim… <NA>                 8         1.9       1.5    16
##  8 Mongo… Lemur herbi Prim… vu                   9.5       0.9      NA      14.5
##  9 Macaq… Maca… omni  Prim… <NA>                10.1       1.2       0.75   13.9
## 10 Slow … Nyct… carni Prim… <NA>                11        NA        NA      13
## 11 Chimp… Pan   omni  Prim… <NA>                 9.7       1.4       1.42   14.3
## 12 Baboon Papio omni  Prim… <NA>                 9.4       1         0.667  14.6
## 13 Potto  Pero… omni  Prim… lc                  11        NA        NA      13
## 14 Squir… Saim… omni  Prim… <NA>                 9.6       1.4      NA      14.4
## 15 Brazi… Tapi… herbi Peri… vu                   4.4       1         0.9    19.6
## # ℹ 2 more variables: brainwt <dbl>, bodywt <dbl>
```

You can use the boolean operators (e.g. >, <, >=, <=, !=, %in%) to create the logical tests.

Pipe operator: |>

Before we go any futher, let's introduce the pipe operator: |> . This operator allows you to pipe the output from one function to the input of another function. Instead of nesting functions (reading from the inside to the outside), the idea of of piping is to read the functions from left to right.

Note. There is another pipe operator %>% from magrittr package with the same functionality. The base pipe operator was introduced in R 4.1.0 in 2021 and it is recommended for use in your code.

Here's an example you have seen:

```
head(select(msleep, name, sleep_total))
```

```
## # A tibble: 6 × 2
##   name                        sleep_total
##   <chr>                             <dbl>
## 1 Cheetah                            12.1
## 2 Owl monkey                         17
## 3 Mountain beaver                    14.4
## 4 Greater short-tailed shrew         14.9
## 5 Cow                                 4
## 6 Three-toed sloth                   14.4
```

Now in this case, we will pipe the msleep data frame to the function that will select two columns (name and sleep_total) and then pipe the new data frame to the function `head()` which will return the head of the new data frame.

```
msleep |>
    select(name, sleep_total) |>
    head(10)
```

```
## # A tibble: 10 × 2
##    name                        sleep_total
##    <chr>                             <dbl>
##  1 Cheetah                            12.1
##  2 Owl monkey                         17
##  3 Mountain beaver                    14.4
##  4 Greater short-tailed shrew         14.9
##  5 Cow                                 4
##  6 Three-toed sloth                   14.4
##  7 Northern fur seal                   8.7
##  8 Vesper mouse                        7
##  9 Dog                                10.1
## 10 Roe deer                           3
```

You will soon see how useful the pipe operator is when we start to combine many functions.

Back to dplyr verbs in action

Now that you know about the pipe operator (|>), we will use it throughout the rest of this tutorial.

Arrange or re-order rows using `arrange()`

To arrange (or re-order) rows by a particular column such as the taxonomic order, list the name of the column you want to arrange the rows by

```
msleep |> arrange(order) |> head()
```

```
## # A tibble: 6 × 11
##   name    genus vore  order conservation sleep_total sleep_rem sleep_cycle awake
##   <chr>   <chr> <chr> <chr> <chr>              <dbl>     <dbl>       <dbl> <dbl>
## 1 Tenrec  Tenr… omni  Afro… <NA>                15.6       2.3      NA       8.4
## 2 Cow     Bos   herbi Arti… domesticated         4         0.7       0.667  20
## 3 Roe de… Capr… herbi Arti… lc                   3        NA        NA      21
## 4 Goat    Capri herbi Arti… lc                   5.3       0.6      NA      18.7
## 5 Giraffe Gira… herbi Arti… cd                   1.9       0.4      NA      22.1
## 6 Sheep   Ovis  herbi Arti… domesticated         3.8       0.6      NA      20.2
## # ℹ 2 more variables: brainwt <dbl>, bodywt <dbl>
```

Now, we will select three columns from msleep, arrange the rows by the taxonomic order and then arrange the rows by sleep_total. Finally show the head of the final data frame

```
msleep |>
    select(name, order, sleep_total) |>
    arrange(order, sleep_total) |>
    head()
```

```
## # A tibble: 6 × 3
##   name     order        sleep_total
##   <chr>    <chr>              <dbl>
## 1 Tenrec   Afrosoricida        15.6
## 2 Giraffe  Artiodactyla         1.9
## 3 Roe deer Artiodactyla         3
## 4 Sheep    Artiodactyla         3.8
## 5 Cow      Artiodactyla         4
## 6 Goat     Artiodactyla         5.3
```

Same as above, except here we filter the rows for mammals that sleep for 16 or more hours instead of showing the head of the final data frame

```
msleep |>
    select(name, order, sleep_total) |>
    arrange(order, sleep_total) |>
    filter(sleep_total >= 16)
```

```
## # A tibble: 8 × 3
##   name                   order           sleep_total
##   <chr>                  <chr>                 <dbl>
## 1 Big brown bat          Chiroptera             19.7
## 2 Little brown bat       Chiroptera             19.9
## 3 Long-nosed armadillo   Cingulata              17.4
## 4 Giant armadillo        Cingulata              18.1
## 5 North American Opossum Didelphimorphia        18
## 6 Thick-tailed opposum   Didelphimorphia        19.4
## 7 Owl monkey             Primates               17
## 8 Arctic ground squirrel Rodentia               16.6
```

Something slightly more complicated: same as above, except arrange the rows in the sleep_total column in a descending order. For this, use the function desc()

```
msleep |>
    select(name, order, sleep_total) |>
    arrange(order, desc(sleep_total)) |>
    filter(sleep_total >= 16)
```

```
## # A tibble: 8 × 3
##   name                   order           sleep_total
##   <chr>                  <chr>                 <dbl>
## 1 Little brown bat       Chiroptera             19.9
## 2 Big brown bat          Chiroptera             19.7
## 3 Giant armadillo        Cingulata              18.1
## 4 Long-nosed armadillo   Cingulata              17.4
## 5 Thick-tailed opposum   Didelphimorphia        19.4
## 6 North American Opossum Didelphimorphia        18
## 7 Owl monkey             Primates               17
## 8 Arctic ground squirrel Rodentia               16.6
```

Create new columns using `mutate()`

The `mutate()` function will add new columns to the data frame. Create a new column called rem_proportion which is the ratio of rem sleep to total amount of sleep.

```
msleep |>
    mutate(rem_proportion = sleep_rem / sleep_total) |>
    select(name, sleep_rem, sleep_total, rem_proportion) |>
    head()
```

```
## # A tibble: 6 × 4
##   name                        sleep_rem sleep_total rem_proportion
##   <chr>                           <dbl>       <dbl>          <dbl>
## 1 Cheetah                          NA          12.1         NA
## 2 Owl monkey                        1.8        17            0.106
## 3 Mountain beaver                   2.4        14.4          0.167
## 4 Greater short-tailed shrew        2.3        14.9          0.154
## 5 Cow                               0.7         4            0.175
## 6 Three-toed sloth                  2.2        14.4          0.153
```

You can many new columns using mutate (separated by commas). Here we add a second column called bodywt_grams which is the bodywt column in grams.

```
msleep |>
    mutate(rem_proportion = sleep_rem / sleep_total,
           bodywt_grams = bodywt * 1000, true_column = TRUE) |>
    select(name, sleep_rem, sleep_total,
           rem_proportion, bodywt, bodywt_grams, true_column) |>
    head()
```

```
## # A tibble: 6 × 7
##   name     sleep_rem sleep_total rem_proportion  bodywt bodywt_grams true_column
##   <chr>        <dbl>       <dbl>          <dbl>   <dbl>        <dbl> <lgl>
## 1 Cheetah       NA          12.1         NA      50            50000 TRUE
## 2 Owl mon…       1.8        17            0.106   0.48            480 TRUE
## 3 Mountai…       2.4        14.4          0.167   1.35           1350 TRUE
## 4 Greater…       2.3        14.9          0.154   0.019            19 TRUE
## 5 Cow            0.7         4            0.175 600            600000 TRUE
## 6 Three-t…       2.2        14.4          0.153   3.85           3850 TRUE
```

Create summaries of the data frame using `summarise()`

The `summarise()` function will create summary statistics for a given column in the data frame such as finding the mean. For example, to compute the average number of hours of sleep, apply the `mean()` function to the column sleep_total and call the summary value avg_sleep.

```
msleep |>
    summarise(avg_sleep = mean(sleep_total))
```

```
## # A tibble: 1 × 1
##   avg_sleep
##       <dbl>
## 1      10.4
```

There are many other summary statistics you could consider such `sd()`, `min()`, `max()`, `median()`, `sum()`, `n()` (returns the length of vector), `first()` (returns first value in vector), `last()` (returns last value in vector) and `n_distinct()` (number of distinct values in vector).

```
msleep |>
    summarise(avg_sleep = mean(sleep_total),
              min_sleep = min(sleep_total),
              max_sleep = max(sleep_total),
              total = n())
```

```
## # A tibble: 1 × 4
##   avg_sleep min_sleep max_sleep total
##       <dbl>     <dbl>     <dbl> <int>
## 1      10.4       1.9      19.9    83
```

Group operations using `group_by()`

The `group_by()` verb is an important function in dplyr. As we mentioned before it's related to concept of "split-apply-combine". We literally want to split the data frame by some variable (e.g. taxonomic order), apply a function to the individual data frames and then combine the output.

Let's do that: split the msleep data frame by the taxonomic order, then ask for the same summary statistics as above. We expect a set of summary statistics for each taxonomic order.

```
msleep |>
    group_by(order) |>
    summarise(avg_sleep = mean(sleep_total),
              min_sleep = min(sleep_total),
              max_sleep = max(sleep_total),
              total = n())
```

```
## # A tibble: 19 × 5
##    order           avg_sleep min_sleep max_sleep total
##    <chr>               <dbl>     <dbl>     <dbl> <int>
##  1 Afrosoricida        15.6       15.6      15.6     1
##  2 Artiodactyla         4.52       1.9       9.1     6
##  3 Carnivora           10.1        3.5      15.8    12
##  4 Cetacea              4.5        2.7       5.6     3
##  5 Chiroptera          19.8       19.7      19.9     2
##  6 Cingulata           17.8       17.4      18.1     2
##  7 Didelphimorphia     18.7       18        19.4     2
##  8 Diprotodontia       12.4       11.1      13.7     2
##  9 Erinaceomorpha      10.2       10.1      10.3     2
## 10 Hyracoidea           5.67       5.3       6.3     3
## 11 Lagomorpha           8.4        8.4       8.4     1
## 12 Monotremata          8.6        8.6       8.6     1
## 13 Perissodactyla       3.47       2.9       4.4     3
## 14 Pilosa              14.4       14.4      14.4     1
## 15 Primates            10.5        8         17      12
## 16 Proboscidea          3.6        3.3       3.9     2
## 17 Rodentia            12.5        7         16.6    22
## 18 Scandentia           8.9        8.9       8.9     1
## 19 Soricomorpha        11.1        8.4      14.9     5
```