dplyr tutorial

What is dplyr?

dplyr is a powerful R-package to transform and summarize tabular data with rows and columns. For another explanation of dplyr see the dplyr package vignette: Introduction to dplyr

Why Is It Useful?

The package contains a set of functions (or "verbs") that perform common data manipulation operations such as filtering for rows, selecting specific columns, re-ordering rows, adding new columns and summarizing data.

In addition, dplyr contains a useful function to perform another common task which is the "split-apply-combine" concept. We will discuss that in a little bit.

How Does It Compare To Using Base Functions R?

If you are familiar with R, you are probably familiar with base R functions such as split(), subset(), apply(), sapply(), lapply(), tapply() and aggregate(). Compared to base functions in R, the functions in dplyr are easier to work with, are more consistent in the syntax and are targeted for data analysis around data frames, instead of just vectors.

How Do I Get dplyr?

To install dplyr:

```r
install.packages("dplyr")
```

To load dplyr:

```r
library(dplyr)
```

Data: Mammals Sleep

The msleep (mammals sleep) data set contains the sleep times 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.

Download the msleep data set in CSV format from here, and then load into R:

```r
dir <- ("/Users/guillermotorres/Documents/Proyectos/2017/12PastoWorkshop/CourseLab/ws/extdata/") # defi
file <- paste0(dir,"msleep_ggplot2.csv")
msleep <- read.csv(file)
head(msleep)
```

```
##                         name      genus  vore        order conservation
## 1                    Cheetah   Acinonyx carni    Carnivora           lc
## 2                 Owl monkey      Aotus  omni     Primates         <NA>
## 3            Mountain beaver Aplodontia herbi     Rodentia           nt
## 4 Greater short-tailed shrew    Blarina  omni Soricomorpha           lc
## 5                        Cow        Bos herbi Artiodactyla domesticated
## 6           Three-toed sloth   Bradypus herbi       Pilosa         <NA>
##   sleep_total sleep_rem sleep_cycle awake brainwt  bodywt
## 1        12.1        NA          NA  11.9      NA  50.000
## 2        17.0       1.8          NA   7.0 0.01550   0.480
## 3        14.4       2.4          NA   9.6      NA   1.350
## 4        14.9       2.3   0.1333333   9.1 0.00029   0.019
## 5         4.0       0.7   0.6666667  20.0 0.42300 600.000
## 6        14.4       2.2   0.7666667   9.6      NA   3.850
```

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
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)
```

```
##                         name sleep_total
## 1                    Cheetah        12.1
## 2                 Owl monkey        17.0
## 3            Mountain beaver        14.4
## 4 Greater short-tailed shrew        14.9
## 5                        Cow          4.0
## 6             Three-toed sloth        14.4
```

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

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

```
##        genus  vore        order conservation sleep_total sleep_rem
## 1   Acinonyx carni    Carnivora           lc        12.1        NA
## 2      Aotus  omni     Primates         <NA>        17.0       1.8
## 3 Aplodontia herbi     Rodentia           nt        14.4       2.4
## 4    Blarina  omni Soricomorpha           lc        14.9       2.3
## 5        Bos herbi Artiodactyla domesticated         4.0       0.7
## 6   Bradypus herbi       Pilosa         <NA>        14.4       2.2
##   sleep_cycle awake brainwt  bodywt
## 1          NA  11.9      NA  50.000
## 2          NA   7.0 0.01550   0.480
## 3          NA   9.6      NA   1.350
## 4   0.1333333   9.1 0.00029   0.019
## 5   0.6666667  20.0 0.42300 600.000
## 6   0.7666667   9.6      NA   3.850
```

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

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

```
##                         name      genus  vore        order
## 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()`:

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

```
##   sleep_total sleep_rem sleep_cycle
## 1        12.1        NA          NA
## 2        17.0       1.8          NA
## 3        14.4       2.4          NA
## 4        14.9       2.3   0.1333333
## 5         4.0       0.7   0.6666667
## 6        14.4       2.2   0.7666667
```

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

1. `ends_with()` = Select columns that end with a character string
2. `contains()` = Select columns that contain a character string
3. `matches()` = Select columns that match a regular expression
4. `one_of()` = Select column 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.

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

```
##                        name        genus    vore           order conservation
## 1                Owl monkey        Aotus    omni        Primates         <NA>
## 2      Long-nosed armadillo      Dasypus   carni       Cingulata           lc
## 3   North American Opossum    Didelphis    omni Didelphimorphia           lc
## 4             Big brown bat    Eptesicus insecti      Chiroptera           lc
## 5      Thick-tailed opposum   Lutreolina   carni Didelphimorphia           lc
## 6           Little brown bat       Myotis insecti      Chiroptera         <NA>
## 7           Giant armadillo   Priodontes insecti       Cingulata           en
## 8  Arctic ground squirrel Spermophilus   herbi        Rodentia           lc
##   sleep_total sleep_rem sleep_cycle awake brainwt bodywt
## 1        17.0       1.8          NA   7.0 0.01550  0.480
## 2        17.4       3.1   0.3833333   6.6 0.01080  3.500
## 3        18.0       4.9   0.3333333   6.0 0.00630  1.700
## 4        19.7       3.9   0.1166667   4.3 0.00030  0.023
## 5        19.4       6.6          NA   4.6      NA  0.370
## 6        19.9       2.0   0.2000000   4.1 0.00025  0.010
## 7        18.1       6.1          NA   5.9 0.08100 60.000
## 8        16.6        NA          NA   7.4 0.00570  0.920
```

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

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

```
##                        name      genus    vore           order conservation
## 1      Long-nosed armadillo    Dasypus   carni       Cingulata           lc
## 2   North American Opossum  Didelphis    omni Didelphimorphia           lc
## 3           Giant armadillo Priodontes insecti       Cingulata           en
##   sleep_total sleep_rem sleep_cycle awake brainwt bodywt
## 1        17.4       3.1   0.3833333   6.6  0.0108    3.5
## 2        18.0       4.9   0.3333333   6.0  0.0063    1.7
## 3        18.1       6.1          NA   5.9  0.0810   60.0
```

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

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

```
##                 name         genus  vore          order conservation
## 1       Owl monkey          Aotus  omni       Primates         <NA>
## 2           Grivet Cercopithecus  omni       Primates           lc
## 3            Horse         Equus herbi Perissodactyla domesticated
## 4           Donkey         Equus herbi Perissodactyla domesticated
## 5     Patas monkey  Erythrocebus  omni       Primates           lc
## 6           Galago        Galago  omni       Primates         <NA>
## 7            Human          Homo  omni       Primates         <NA>
## 8   Mongoose lemur         Lemur herbi       Primates           vu
## 9          Macaque        Macaca  omni       Primates         <NA>
## 10       Slow loris    Nyctibeus carni       Primates         <NA>
## 11      Chimpanzee           Pan  omni       Primates         <NA>
## 12           Baboon         Papio  omni       Primates         <NA>
## 13            Potto  Perodicticus  omni       Primates           lc
## 14 Squirrel monkey       Saimiri  omni       Primates         <NA>
## 15 Brazilian tapir       Tapirus herbi Perissodactyla           vu
##    sleep_total sleep_rem sleep_cycle awake brainwt  bodywt
## 1         17.0       1.8          NA   7.0  0.0155   0.480
## 2         10.0       0.7          NA  14.0      NA   4.750
## 3          2.9       0.6   1.0000000  21.1  0.6550 521.000
## 4          3.1       0.4          NA  20.9  0.4190 187.000
## 5         10.9       1.1          NA  13.1  0.1150  10.000
## 6          9.8       1.1   0.5500000  14.2  0.0050   0.200
## 7          8.0       1.9   1.5000000  16.0  1.3200  62.000
## 8          9.5       0.9          NA  14.5      NA   1.670
## 9         10.1       1.2   0.7500000  13.9  0.1790   6.800
## 10        11.0        NA          NA  13.0  0.0125   1.400
## 11         9.7       1.4   1.4166667  14.3  0.4400  52.200
## 12         9.4       1.0   0.6666667  14.6  0.1800  25.235
## 13        11.0        NA          NA  13.0      NA   1.100
## 14         9.6       1.4          NA  14.4  0.0200   0.743
## 15         4.4       1.0   0.9000000  19.6  0.1690 207.501
```

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

Pipe Operator: %>%

Before we go any further, let's introduce the pipe operator: %>%. dplyr imports this operator from another package (magrittr).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 piping is to read the functions from left to right.

Here's an example you have seen:

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

```
##                          name sleep_total
## 1                     Cheetah        12.1
## 2                  Owl monkey        17.0
## 3             Mountain beaver        14.4
## 4  Greater short-tailed shrew        14.9
## 5                         Cow         4.0
## 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.

```r
msleep %>%
    select(name, sleep_total) %>%
    head
```

```
##                          name sleep_total
## 1                     Cheetah        12.1
## 2                  Owl monkey        17.0
## 3             Mountain beaver        14.4
## 4  Greater short-tailed shrew        14.9
## 5                         Cow         4.0
## 6            Three-toed sloth        14.4
```

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
```

```
##       name     genus  vore        order conservation sleep_total sleep_rem
## 1   Tenrec    Tenrec  omni Afrosoricida         <NA>        15.6       2.3
## 2      Cow       Bos herbi Artiodactyla domesticated         4.0       0.7
## 3 Roe deer Capreolus herbi Artiodactyla           lc         3.0        NA
## 4     Goat     Capri herbi Artiodactyla           lc         5.3       0.6
## 5  Giraffe   Giraffa herbi Artiodactyla           cd         1.9       0.4
## 6    Sheep      Ovis herbi Artiodactyla domesticated         3.8       0.6
##   sleep_cycle awake brainwt  bodywt
## 1          NA   8.4  0.0026   0.900
## 2   0.6666667  20.0  0.4230 600.000
## 3          NA  21.0  0.0982  14.800
## 4          NA  18.7  0.1150  33.500
## 5          NA  22.1      NA 899.995
## 6          NA  20.2  0.1750  55.500
```

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
```

```
##       name        order sleep_total
## 1   Tenrec Afrosoricida        15.6
## 2  Giraffe Artiodactyla         1.9
## 3 Roe deer Artiodactyla         3.0
## 4    Sheep Artiodactyla         3.8
## 5      Cow Artiodactyla         4.0
## 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)
```

```
##                      name           order sleep_total
## 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        18.0
## 6   Thick-tailed opposum Didelphimorphia        19.4
## 7              Owl monkey        Primates        17.0
## 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)
```

```
##                      name           order sleep_total
## 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        18.0
## 7              Owl monkey        Primates        17.0
## 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) %>%
    head
```

```
##                          name      genus  vore        order conservation
## 1                     Cheetah   Acinonyx carni    Carnivora           lc
## 2                  Owl monkey      Aotus  omni     Primates         <NA>
## 3             Mountain beaver Aplodontia herbi     Rodentia           nt
## 4 Greater short-tailed shrew    Blarina  omni Soricomorpha           lc
## 5                         Cow        Bos herbi Artiodactyla domesticated
## 6            Three-toed sloth   Bradypus herbi       Pilosa         <NA>
##   sleep_total sleep_rem sleep_cycle awake brainwt  bodywt rem_proportion
## 1        12.1        NA          NA  11.9      NA  50.000             NA
## 2        17.0       1.8          NA   7.0 0.01550   0.480      0.1058824
## 3        14.4       2.4          NA   9.6      NA   1.350      0.1666667
## 4        14.9       2.3   0.1333333   9.1 0.00029   0.019      0.1543624
## 5         4.0       0.7   0.6666667  20.0 0.42300 600.000      0.1750000
## 6        14.4       2.2   0.7666667   9.6      NA   3.850      0.1527778
```

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) %>%
    head
```

```
##                          name      genus  vore        order conservation
## 1                     Cheetah   Acinonyx carni    Carnivora           lc
## 2                  Owl monkey      Aotus  omni     Primates         <NA>
## 3            Mountain beaver Aplodontia herbi     Rodentia           nt
## 4 Greater short-tailed shrew    Blarina  omni Soricomorpha           lc
## 5                        Cow        Bos herbi Artiodactyla domesticated
## 6           Three-toed sloth   Bradypus herbi       Pilosa         <NA>
##   sleep_total sleep_rem sleep_cycle awake brainwt  bodywt rem_proportion
## 1        12.1        NA          NA  11.9      NA  50.000             NA
## 2        17.0       1.8          NA   7.0 0.01550   0.480      0.1058824
## 3        14.4       2.4          NA   9.6      NA   1.350      0.1666667
## 4        14.9       2.3   0.1333333   9.1 0.00029   0.019      0.1543624
## 5         4.0       0.7   0.6666667  20.0 0.42300 600.000      0.1750000
## 6        14.4       2.2   0.7666667   9.6      NA   3.850      0.1527778
##   bodywt_grams
## 1        50000
## 2          480
## 3         1350
## 4           19
## 5       600000
## 6         3850
```

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))
```

```
##   avg_sleep
## 1  10.43373
```

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())
```

```
##   avg_sleep min_sleep max_sleep total
## 1  10.43373       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 x 5
##              order avg_sleep min_sleep max_sleep total
##             <fctr>     <dbl>     <dbl>     <dbl> <int>
##  1    Afrosoricida 15.600000      15.6      15.6     1
##  2    Artiodactyla  4.516667       1.9       9.1     6
##  3       Carnivora 10.116667       3.5      15.8    12
##  4         Cetacea  4.500000       2.7       5.6     3
##  5      Chiroptera 19.800000      19.7      19.9     2
##  6       Cingulata 17.750000      17.4      18.1     2
##  7 Didelphimorphia 18.700000      18.0      19.4     2
##  8   Diprotodontia 12.400000      11.1      13.7     2
##  9  Erinaceomorpha 10.200000      10.1      10.3     2
## 10      Hyracoidea  5.666667       5.3       6.3     3
## 11      Lagomorpha  8.400000       8.4       8.4     1
## 12     Monotremata  8.600000       8.6       8.6     1
## 13  Perissodactyla  3.466667       2.9       4.4     3
## 14          Pilosa 14.400000      14.4      14.4     1
## 15        Primates 10.500000       8.0      17.0    12
## 16     Proboscidea  3.600000       3.3       3.9     2
## 17        Rodentia 12.468182       7.0      16.6    22
## 18      Scandentia  8.900000       8.9       8.9     1
## 19    Soricomorpha 11.100000       8.4      14.9     5
```