Tutorial and Laboratories

Week 6

The purpose of this week's tutorial and lab exercises is to practise manipulating data with the grammar verbs in the `dplyr` package:

The objectives for this tutorial and lab are to:

1- Get yourself familiar with filtering, sub-setting and summarising data stored in data frames

2- Get yourself familiar with using the pipes in performing multiple functions on the same data frame.

Data Manipulation in R

The `*dplyr*` package in `*tidyverse*` library provides five *verbs*, which represent the grammar of the data manipulation in R. These verbs are as follows:

1. filter() returns a subset of rows based on some conditions
2. select() returns a subset of columns (i.e., features, variables) based on some conditions.
3. mutate() add or modify existing columns
4. arrange() sort the rows based on some variables
5. summarise() aggregate the data across rows (e.g., group it according to some criteria)

Each of these functions takes a data frame as its first argument, and returns a data frame as the result of the operation.

Exercise 1 (filter verb)

Use the package called `nycflights13` and the flights table to find the flights that:

1. Had an arrival delay of two or more hours
2. Flew to Houston (IAH or HOU)
3. Were operated by United, American, or Delta
4. Departed in summer (July, August, and September)
5. Arrived more than two hours late, but didn't leave late
6. Were delayed by at least an hour, but made up over 30 minutes in flight
7. Departed between midnight and 6am (inclusive)
8. Their departure time is not recorded (i.e. NA).

Exercise 2

Use the `nycflights13` package and the flights data frame to answer the following questions:

- What month had the highest proportion of cancelled flights?
- What month had the lowest?

Pipes (%>%)

What if you wanted to apply multiple of these verbs on the same data? To do so, you may use one of the following three ways: intermediate steps, nested functions, or pipes.

A- Intermediate steps:

You will need to create temporary data frames and use them as input to the next operation. This results in creating many non-needed objects in the workspace.

```
sorted_flights <- arrange(flights, month)
delayed_flights <- filter(sorted_flights, arr_delay >= 120)
Select(delayed_flights, day, month, year)
```

B- Nested functions

You may nest functions or call them inside each other. This would solve the problem of creating extra objects. However, this is hard to read and to understand.

```
Select(filter(arrange(flights, month), arr_delay >= 120), day,
month, year)
```

C- Pipes

Your last option is using the pipes, which provide a solution to the problem of creating extra data frames as well as are quite easy to read to understand. You just need to list the functions that you want to apply on a data frame on the right side of the pipe operator (%>%) after each other and the data frame in itself in the left-side of the pipe.

Pipes are made available via the `magrittr` package, installed automatically with `dplyr` package. In RStudio, you can type the pipe with (Ctrl + Shift + M) if you have a PC or (Cmd + Shift + M) if you have a Mac.

```
flights %>%
    arrange(flights, month) %>%
    filter(sorted_flights, arr_delay >= 120) %>%
    Select(delayed_flights, day, month, year)
```

Exercise 3 (Unsupervised activity)

Use the nycflights13 package and the flights data frame to answer the following question:

- What plane (specified by the `tailnum` variable) traveled the most times from New York City airports in 2013?
- (Optional) Visualise the number of trips per week over the year.

Exercise 4 (Unsupervised activity)

Use the `nycflights13` package and the flights to answer the following questions:

- What is the average departure delay for the flights per their carriers?
- What is the minimum, maximum and the average arrival delay of the flights per their arrival town (i.e. destination)?