dplyr workshops

Natalia Potocka

March 16, 2017

```
library(dplyr)
library(nycflights13)

data(flights)

glimpse(flights)
```

`dplyr::%>%` passes object on left hand side as first argument (or `.` argument) of function on righthand side. "Piping" with `%>%` makes code more readable,

```
x %>% f(y)
```

is the same as

```
f(x, y)
y %>% f(x, ., z)
```

is the same as

```
f(x, y, z)
```

e.g.

```
sum(c(1,2,3))
c(1,2,3) %>% sum()
```

A shortcut in RStudio for `%>%` is **CTRL + SHIFT + M** or **CMD + SHIFT + M**.

dplyr's main functions:

- `select` - select columns/variables
- `arrange` - sort the observations
- `mutate` - create a new column/variable
- `filter` - choose rows that satisfy certain condition
- `summarise` - summarise
- `group_by` - count in goups
- `join` - join two data sets

Cheatsheet or Help -> Cheatsheets -> Data manipulation with dplyr, tidyr

select

Selecting one column:

```
flights %>%
  select(dep_delay)
```

Selecting more than one column:

```
flights %>%
  select(dep_delay, dep_time)
```

Selecting columns that contains `dep` in the name

```
flights %>%
  select(contains("dep"))
```

Exercise 1. Choose columns that have `arr` int the name.

arrange

Sorting columns by departure delay

```
flights %>%
  arrange(dep_delay)
```

The same but choosing only two columns

```
flights %>%
  select(dep_delay, arr_delay) %>%
  arrange(dep_delay)
```

The same but sorting desceding

```
flights %>%
  select(dep_delay, arr_delay) %>%
  arrange(desc(dep_delay))
```

Exercise 2. Choose two columns: `distance` and `arr_delay` and sort data by distance.

mutate

Create new variable indicating whether there was a delay or not

```
flights %>%
  mutate(if_arr_delay = (arr_delay > 0)) %>%
  select(if_arr_delay, arr_delay)
```

Create more variables

```
flights %>%
  mutate(if_arr_delay = (arr_delay > 0), air_time_hour = air_time/60) %>%
  select(if_arr_delay, arr_delay, air_time_hour, air_time)
```

Exercise 3. Create new variable which tells us what is the sum of departure and arrival delay.

filter

Choose rows that had `dep_delay` greater than 20 mins.

```
flights %>%
  filter(dep_delay > 20) %>%
  arrange(dep_delay)
```

Choose rows that had `dep_` **AND** `arr_delay` greater than 20 mins.

```
flights %>%
  filter(dep_delay > 20, arr_delay > 20) %>%
  select(dep_delay, arr_delay) %>%
  arrange(dep_delay, desc(arr_delay))
```

Choose rows that had `dep_` or `arr_delay` greater than 20 mins.

```
flights %>%
  filter(dep_delay > 20 | arr_delay > 20) %>%
  select(dep_delay, arr_delay) %>%
  arrange(dep_delay, desc(arr_delay))
```

Choose rows that had `dep_delay` greater than 20 mins and the plane started from JFK.

```
flights %>%
  filter(dep_delay > 20, origin == "JFK") %>%
  select(dep_delay, origin)
```

Choose rows that had `dep_delay` greater than 20 mins and the plane didn't start from JFK.

```
flights %>%
  filter(dep_delay > 20, origin != "JFK") %>%
  select(dep_delay, origin)
```

Exercise 4. Choose rows that had distance more than 800 miles and arrived in ORD. Then check out the same arrival port but flights that had distance **less** than 800 miles.

summarise

Count mean departure delay

```
flights %>%
  summarise(dep_delay = mean(dep_delay, na.rm = TRUE))
```

Count mean and median departure delay

```
flights %>%
  summarise(dep_delay_mean = mean(dep_delay, na.rm = TRUE),
            dep_delay_median = median(dep_delay, na.rm = TRUE))
```

The same but with the number of rows in the dataset

```
flights %>%
  filter(!is.na(dep_delay)) %>%
  summarise(dep_delay_mean = mean(dep_delay, na.rm = TRUE),
            dep_delay_median = median(dep_delay, na.rm = TRUE),
            n = n())
flights <- flights %>% filter(!is.na(dep_delay))
```

Exercise 5. Count mean, median and max arrival delay.

group_by

Count mean departure delay when in groups by origin port

```
flights %>%
  group_by(origin) %>%
  summarise(dep_delay_mean = mean(dep_delay))
```

Count how many flights there where arriving in a certain city and departing from a certain city

```
flights %>%
  group_by(origin, dest) %>%
  summarise(n = n())
```

Choose which destination had the greatest delay depending on the origin

```
flights %>%
  group_by(origin) %>%
  filter(dep_delay == max(dep_delay)) %>%
  select(origin, dep_delay, dest)
```

Exercise 6. Count mean arrival delay by month.

Exercise 7. Count max `air_time` by origin.

join

Join information about planes with the flights to know what was the delay by the type of engine

```
data(planes)
flights %>%
  left_join(planes, by = c("tailnum" = "tailnum")) %>%
  group_by(year.y) %>%
  summarise(dep_delay = mean(dep_delay))
```

Join information about planes with the flights to know what was the delay by the number of seats

```
data(planes)
flights %>%
  left_join(planes, by = c("tailnum" = "tailnum")) %>%
  group_by(seats) %>%
  summarise(dep_delay = mean(dep_delay))
```

Exercise 8. Join information about planes with the flights to know what was the delay by the manufacturer