Lab-02

Write your name here

February 02, 2023

Preface

The goal of this assignment is to help you gain familiarity with data frames – think "spreadsheets" – and how to use **dplyr** functions to transform data. In this lab we are providing some code snippets to serve as "scaffolding" to help guide you through each step. As always, please come to office hours and reach out to your teaching staff if you have any questions.

We will work with the data table `flights` provided in the package **nycflights13**. The data table includes all domestic flights that departed NYC (i.e. JFK, LGA or EWR) in 2013. It has 19 variables. Details of the package **nycflights13** are available here.

```
flights
```

```
# A tibble: 336,776 x 19
    year month   day dep_time sched_de~1 dep_d~2 arr_t~3 sched~4 arr_d~5 carrier
   <int> <int> <int>    <int>      <int>   <dbl>   <int>   <int>   <dbl> <chr>
 1  2013     1     1      517        515       2     830     819      11 UA
 2  2013     1     1      533        529       4     850     830      20 UA
 3  2013     1     1      542        540       2     923     850      33 AA
 4  2013     1     1      544        545      -1    1004    1022     -18 B6
 5  2013     1     1      554        600      -6     812     837     -25 DL
 6  2013     1     1      554        558      -4     740     728      12 UA
 7  2013     1     1      555        600      -5     913     854      19 B6
 8  2013     1     1      557        600      -3     709     723     -14 EV
 9  2013     1     1      557        600      -3     838     846      -8 B6
10  2013     1     1      558        600      -2     753     745       8 AA
# ... with 336,766 more rows, 9 more variables: flight <int>, tailnum <chr>,
#   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>,
#   minute <dbl>, time_hour <dttm>, and abbreviated variable names
#   1: sched_dep_time, 2: dep_delay, 3: arr_time, 4: sched_arr_time,
#   5: arr_delay
```

1. In this data set, `arr_delay` is a variable that records the arrival delays in minutes. Negative times represent early arrivals. Use dplyr::filter to find: (1) the flights that arrived more than two hours late, and (2) the flights that arrived earlier than scheduled. What is the proportion of flights that arrived more than two hours late? What is the proportion of flights that arrived earlier than scheduled time?

```
# Use dplyr::filter to find and count the flights that arrived more than two hours late
two_hour_late <- flights |>
  filter(FALSE) |>  # Replace FALSE with your code
  count()

# Use dplyr::filter to find and count the flights that arrived earlier than scheduled
early_arr <- flights |>
  filter(FALSE) |>   # Replace FALSE with your code
  count()

# Count the total number of flights
total <- count(flights)
```

A proportion of 0 of the flights arrived more than two hours late. A proportion of 0 of the flights arrived earlier than scheduled time.

2. How many flights have a missing `dep_time`? What other variables are missing? What might these rows represent?

```
flights |>
  filter(FALSE) |> # Replace FALSE with your code
  count()
```

```
# A tibble: 1 x 1
      n
  <int>
1     0
```

These rows probably represent...

3. Use at least two ways to select variables of `dep_time`, `sched_dep_time`, `dep_delay`, `arr_time`, `sched_arr_time`, `arr_delay`. Put `arr_delay` in the first column.

```
# Method 1
# Your code goes here

# Method 2
# Your code goes here
```

4. Use dplyr::arrange to sort flights by arrival delays in descending order. How long was the worst arrival delay?

```
worst_delay <- flights |>
  arrange(desc(FALSE)) |> # sort flights by arrival delays in descending order
  filter(FALSE) |> # choose the first row
  pull(arr_delay)
```

The worst arrival delay was `inline code` minutes.

5. Select `air_time` and `distance`. Generate a new varible `speed` that is calculated as `distance` divided by `air_time` (in miles/min). Then create a variable `mph` that contains speed in miles/hour.

```
flights |>
  select() |>  # select `air_time` and `distance` here
  mutate( # create a new variable `speed`
    )  # create a new variable `mph`
```

```
# A tibble: 336,776 x 0
```

6. Select `dep_time`. Currently `dep_time` is convenient to look at, but hard to compute with because it is not really a continuous number. Convert it to a more convenient representation of number of minutes since midnight. Add a new column `dep_time_min` to store the converted values.

```
# Your code goes here
flights |>
  select() |>
  mutate(dep_hour = FALSE) |> # split out the hour-digits
  mutate(dep_min = FALSE) |> # split out the minute-digits
  mutate(dep_time_min = dep_hour * 60 + dep_min) # generate dep_time in the number of minu
```

```
# A tibble: 336,776 x 3
   dep_hour dep_min dep_time_min
   <lgl>    <lgl>          <dbl>
 1 FALSE    FALSE              0
 2 FALSE    FALSE              0
 3 FALSE    FALSE              0
 4 FALSE    FALSE              0
 5 FALSE    FALSE              0
 6 FALSE    FALSE              0
 7 FALSE    FALSE              0
 8 FALSE    FALSE              0
 9 FALSE    FALSE              0
10 FALSE    FALSE              0
# ... with 336,766 more rows
```

7. Calculate the average arrival delay by carrier. Which carrier has the worst delays?

```
worst_delay_carrier <- flights |>
  group_by() |>
  summarise() |>
  arrange() |>
  filter()
```

Carrier `inline code` has the worst delays.

8. What time of day should you fly if you want to avoid delays as much as possible?

```
# Your code goes here
```

The best time to fly to avoid delays is...