Reading Set 2

Dhyey Mavani

Due by 10pm ET on Monday

Reading Set Information

A more thorough reading and light practice of the textbook reading prior to class allows us to jump into things more quickly in class and dive deeper into topics. As you actively read the textbook, you will work through the Reading Sets to help you engage with the new concepts and skills, often by replicating on your own the examples covered in the book.

These should be completed on your own without help from your peers. While most of our work in this class will be collaborative, it is important each individual completes the active readings. The problems should be straightforward based on the textbook readings, but if you have any questions, feel free to ask me!

GitHub Workflow

1. Before editing this file, verify you are working on the copy saved in *your* repo for the course (check the filepath and the project name in the top right corner).
2. Before editing this file, make an initial commit of the file to your repo to add your copy of the problem set.
3. Change your name at the top of the file and get started!
4. You should *save, knit, and commit* the .Rmd file each time you've finished a question, if not more often.
5. You should occasionally *push* the updated version of the .Rmd file back onto GitHub. When you are ready to push, you can click on the Git pane and then click **Push**. You can also do this after each commit in RStudio by clicking **Push** in the top right of the *Commit* pop-up window.
6. When you think you are done with the assignment, save the pdf as "*Name_thisfilename_date*.pdf" (it's okay to leave out the date if you don't need it) before committing and pushing (this is generally good practice but also helps me in those times where I need to download all student homework files).

Gradescope Upload

For each question (e.g., 3.1), allocate all pages associated with the specific question. If your work for a question runs onto a page that you did not select, you may not get credit for the work. If you do not allocate *any* pages when you upload your pdf, you may get a zero for the assignment.

You can resubmit your work as many times as you want before the deadline, so you should not wait until the last minute to submit some version of your work. Unexpected delays/crises that occur on the day the assignment is due do not warrant extensions (please submit whatever you have done to receive partial credit).

Problem 1 **NYC Flights** In Section 5.1, the `flights` and `carrier` tables within the `nycflights13` package are joined together.

1.1 Recreate the `flights_joined` dataset from Section 5.1, being sure to *glimpse* the data in the Console to verify the join worked.

```
library(tidyverse)
library(mdsr)
library(nycflights13)
glimpse(flights)
```

```
Rows: 336,776
Columns: 19
$ year           <int> 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2~
$ month          <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
$ day            <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
$ dep_time       <int> 517, 533, 542, 544, 554, 554, 555, 557, 557, 558, 558, ~
$ sched_dep_time <int> 515, 529, 540, 545, 600, 558, 600, 600, 600, 600, 600, ~
$ dep_delay      <dbl> 2, 4, 2, -1, -6, -4, -5, -3, -3, -2, -2, -2, -2, -2, -1~
$ arr_time       <int> 830, 850, 923, 1004, 812, 740, 913, 709, 838, 753, 849,~
$ sched_arr_time <int> 819, 830, 850, 1022, 837, 728, 854, 723, 846, 745, 851,~
$ arr_delay      <dbl> 11, 20, 33, -18, -25, 12, 19, -14, -8, 8, -2, -3, 7, -1~
$ carrier        <chr> "UA", "UA", "AA", "B6", "DL", "UA", "B6", "EV", "B6", "~
$ flight         <int> 1545, 1714, 1141, 725, 461, 1696, 507, 5708, 79, 301, 4~
$ tailnum        <chr> "N14228", "N24211", "N619AA", "N804JB", "N668DN", "N394~
$ origin         <chr> "EWR", "LGA", "JFK", "JFK", "LGA", "EWR", "EWR", "LGA",~
$ dest           <chr> "IAH", "IAH", "MIA", "BQN", "ATL", "ORD", "FLL", "IAD",~
$ air_time       <dbl> 227, 227, 160, 183, 116, 150, 158, 53, 140, 138, 149, 1~
$ distance       <dbl> 1400, 1416, 1089, 1576, 762, 719, 1065, 229, 944, 733, ~
$ hour           <dbl> 5, 5, 5, 5, 6, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 5, 6, 6, 6~
$ minute         <dbl> 15, 29, 40, 45, 0, 58, 0, 0, 0, 0, 0, 0, 0, 0, 0, 59, 0~
$ time_hour      <dttm> 2013-01-01 05:00:00, 2013-01-01 05:00:00, 2013-01-01 0~
```

```
flights_joined <- flights %>%
  inner_join(airlines, by = c("carrier" = "carrier"))
glimpse(flights_joined)
```

```
Rows: 336,776
Columns: 20
$ year           <int> 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2~
$ month          <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
$ day            <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
$ dep_time       <int> 517, 533, 542, 544, 554, 554, 555, 557, 557, 558, 558, ~
$ sched_dep_time <int> 515, 529, 540, 545, 600, 558, 600, 600, 600, 600, 600, ~
$ dep_delay      <dbl> 2, 4, 2, -1, -6, -4, -5, -3, -3, -2, -2, -2, -2, -2, -1~
$ arr_time       <int> 830, 850, 923, 1004, 812, 740, 913, 709, 838, 753, 849,~
$ sched_arr_time <int> 819, 830, 850, 1022, 837, 728, 854, 723, 846, 745, 851,~
$ arr_delay      <dbl> 11, 20, 33, -18, -25, 12, 19, -14, -8, 8, -2, -3, 7, -1~
$ carrier        <chr> "UA", "UA", "AA", "B6", "DL", "UA", "B6", "EV", "B6", "~
```

```
$ flight         <int> 1545, 1714, 1141, 725, 461, 1696, 507, 5708, 79, 301, 4~
$ tailnum        <chr> "N14228", "N24211", "N619AA", "N804JB", "N668DN", "N394~
$ origin         <chr> "EWR", "LGA", "JFK", "JFK", "LGA", "EWR", "EWR", "LGA",~
$ dest           <chr> "IAH", "IAH", "MIA", "BQN", "ATL", "ORD", "FLL", "IAD",~
$ air_time       <dbl> 227, 227, 160, 183, 116, 150, 158, 53, 140, 138, 149, 1~
$ distance       <dbl> 1400, 1416, 1089, 1576, 762, 719, 1065, 229, 944, 733, ~
$ hour           <dbl> 5, 5, 5, 5, 6, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 5, 6, 6, 6~
$ minute         <dbl> 15, 29, 40, 45, 0, 58, 0, 0, 0, 0, 0, 0, 0, 0, 0, 59, 0~
$ time_hour      <dttm> 2013-01-01 05:00:00, 2013-01-01 05:00:00, 2013-01-01 0~
$ name           <chr> "United Air Lines Inc.", "United Air Lines Inc.", "Amer~
```

1.2 Now, starting from `flights_joined`, create a new dataset `flights_short` that **(1)** creates a new variable, `distance_km`, which is distance in kilometers (note that 1 mile is about 1.6 kilometers); **(2)** keeps only the variables: `name`, `flight`, `arr_delay`, and `distance_km`; and **(3)** keeps only observations where the distance is less than 500 kilometers.

```
flights_joined$distance_km <- (flights_joined$distance/1.6)
flights_short = filter(flights_joined, distance_km < 500)
select(flights_short, name, flight, arr_delay, distance_km)
```

```
# A tibble: 163,892 x 4
   name                     flight arr_delay distance_km
   <chr>                     <int>     <dbl>       <dbl>
 1 Delta Air Lines Inc.        461       -25        476.
 2 United Air Lines Inc.      1696        12        449.
 3 ExpressJet Airlines Inc.   5708       -14        143.
 4 American Airlines Inc.      301         8        458.
 5 JetBlue Airways            1806        -4        117.
 6 Envoy Air                  4650        12        476.
 7 Envoy Air                  4401        16        314.
 8 Delta Air Lines Inc.       1743        -8        475
 9 Envoy Air                  3768        32        449.
10 Delta Air Lines Inc.        575        -9        466.
# ... with 163,882 more rows
```

1.3 Using the functions introduced in Section 4.1.4, compute the number of flights (call this `N`), the average arrival delay (call this `avg_arr_delay`), and the average distance in kilometers (call this `avg_dist_km`) among these flights with distances less than 500 km (i.e. working off of `flights_short`), grouping by the carrier name. Sort the results in descending order based on `avg_arr_delay`. Save the results in a tibble object called `delay_summary`, and display the table.

```
delay_summary <- flights_short %>%
  drop_na() %>%
  group_by(name) %>%
```

```
  summarize(N = n(),
            avr_arr_delay = mean(arr_delay),
            avg_dist_km = mean(distance_km))
delay_summary
```

```
# A tibble: 12 x 4
   name                           N avr_arr_delay avg_dist_km
   <chr>                      <int>         <dbl>       <dbl>
 1 AirTran Airways Corporation  3175        20.1          415.
 2 American Airlines Inc.       7274        -0.205        391.
 3 Delta Air Lines Inc.        15740         5.15         404.
 4 Endeavor Air Inc.           14357         7.47         260.
 5 Envoy Air                   22715        11.0          326.
 6 ExpressJet Airlines Inc.    40554        15.8          281.
 7 JetBlue Airways             16546         9.26         190.
 8 Mesa Airlines Inc.            544        15.6          235.
 9 SkyWest Airlines Inc.          25        14.2          268.
10 Southwest Airlines Co.       6831         9.84         446.
11 United Air Lines Inc.       12029         5.61         332.
12 US Airways Inc.             17591         2.33         225.
```

1.4 Rename the three columns in the `delay_summary` data table to `Airline`, `"Total flights under 500 km"` and `"Average arrival delay (mins)"`, respectively, then use `kable(booktabs = TRUE, digits = 1)` to make the final table output in the pdf close to publication quality.

```
delay_summary <- flights_short %>%
  drop_na() %>%
  group_by(name) %>%
  summarize(N = n(),
            avr_arr_delay = mean(arr_delay),
            avg_dist_km = mean(distance_km)) %>%
  rename('Airline' = name,
         'Total flights under 500 km' = N,
         'Average arrival delay (mins)' =  avr_arr_delay,
         'Average distance in km' = avg_dist_km) %>%
  kable(booktabs = TRUE, digits = 1)
delay_summary
```

Airline	Total flights under 500 km	Average arrival delay (mins)	Average distance in km
AirTran Airways Corporation	3175	20.1	415.5
American Airlines Inc.	7274	-0.2	391.3
Delta Air Lines Inc.	15740	5.1	403.6
Endeavor Air Inc.	14357	7.5	260.3
Envoy Air	22715	11.0	325.9
ExpressJet Airlines Inc.	40554	15.8	280.7
JetBlue Airways	16546	9.3	190.4
Mesa Airlines Inc.	544	15.6	235.3
SkyWest Airlines Inc.	25	14.2	268.4
Southwest Airlines Co.	6831	9.8	445.9
United Air Lines Inc.	12029	5.6	332.1
US Airways Inc.	17591	2.3	224.6

Problem 2 **Baby names**

2.1 Working with the babynames data in the **babynames** package, create a dataset recent_names that only includes years 2000 to 2017.

```
library(tidyverse)
library(mdsr)
library(babynames)
recent_names = filter(babynames, year < 2018 & year > 1999)
recent_names
```

```
# A tibble: 591,925 x 5
    year sex   name          n    prop
   <dbl> <chr> <chr>     <int>   <dbl>
 1  2000 F     Emily     25953 0.0130
 2  2000 F     Hannah    23080 0.0116
 3  2000 F     Madison   19967 0.0100
 4  2000 F     Ashley    17997 0.00902
 5  2000 F     Sarah     17697 0.00887
 6  2000 F     Alexis    17629 0.00884
 7  2000 F     Samantha  17266 0.00866
 8  2000 F     Jessica   15709 0.00787
 9  2000 F     Elizabeth 15094 0.00757
10  2000 F     Taylor    15078 0.00756
# ... with 591,915 more rows
```

2.2 Following the code presented in Section 6.2.5, create a dataset called recentnames_summary that summarizes the total number of people in recent history (years 2000 to 2017) with each name, grouped by sex.

```
recentnames_summary <- recent_names %>%
  group_by(name, sex) %>%
  summarize(total = sum(n))
recentnames_summary
```

```
# A tibble: 73,332 x 3
# Groups:   name [67,063]
   name      sex   total
   <chr>     <chr> <int>
 1 Aaban     M       107
 2 Aabha     F        35
 3 Aabid     M        10
 4 Aabir     M         5
 5 Aabriella F        32
 6 Aada      F         5
 7 Aadam     M       202
```

```
 8 Aadan     M       130
 9 Aadarsh   M       199
10 Aaden     F         5
# ... with 73,322 more rows
```

2.3 Now, following the fourth and fifth code chunks presented in Section 6.2.5, reshape or *pivot* the summary data from *long* format to *wide* format. Only keep observations where more than 10,000 babies have been named in each sex (`M` and `F`), and find the smaller of the two ratios `M / F` and `F / M` to identify the top three sex-balanced names (and only the top three!). Save the wide data as `recentnames_balanced_wide`. Display the table.

```
recentnames_balanced_wide <- recentnames_summary %>%
  pivot_wider(
    names_from = sex,
    values_from = total,
    values_fill = 0
  ) %>%
  filter(`M` > 10000 & `F` > 10000) %>%
  mutate(ratio = pmin(`M`/ `F`, `F` / `M`)) %>%
  arrange(desc(ratio)) %>%
  head(3)
recentnames_balanced_wide
```

```
# A tibble: 3 x 4
# Groups:   name [3]
  name        M     F ratio
  <chr>   <int> <int> <dbl>
1 Justice 11267 10947 0.972
2 Skyler  22154 17120 0.773
3 Quinn   19080 25022 0.763
```

2.4 Finally, use `pivot_longer()` to put the dataset back into *long* form. Call this dataset `recentnames_balanced` and display the table. Why are the number of observations in `recentnames_balanced` different from that in `recentnames_summary` from Problem 2.2?

As we have done other operations of filtering the data set while generating the `recentnames_balanced_wide`, that's why the `recentnames_summary` has more number of observations than `recentnames_balanced_wide`. Now, if we consider the data set `recentnames_balanced`, we have just made the `recentnames_balanced_wide` longer using `pivot_longer()`, but it still contains the filtered data with name coincidences $> 10{,}000$.

```
recentnames_balanced <- recentnames_balanced_wide %>%
  select(name, `M`, `F`) %>%
  pivot_longer(-name,
               names_to = "sex",
               values_to = "total")

recentnames_balanced
```

```
# A tibble: 6 x 3
# Groups:   name [3]
  name    sex   total
  <chr>   <chr> <int>
1 Justice M     11267
2 Justice F     10947
3 Skyler  M     22154
4 Skyler  F     17120
5 Quinn   M     19080
6 Quinn   F     25022
```

Problem 3 **Ethical conundrums** Each subsection of Section 8.4 discusses an ethical scenario and ends with one or more questions. Choose one of the scenarios provided to reflect on, and *in one paragraph or less* respond to the question(s) posed with your initial thoughts. Please identify the scenario for reference (e.g. "8.4.1 The chief executive officer").

8.4.1 The chief executive officer

By "playing God", here consultant means that it would be unsafe as we will be just playing around with luck. Although the privacy and competition of company are there, they shouldn't randomly tweak the data without evaluating and analyzing the ramifications of it just to make the data unique and more personalized to the company to gain the competitive advantage. I think that consultant should refuse to do so because this can be detrimental to the company along with the entire market because it will encourage such practices which will increase the market risk and will engender imbalance in the market by making someone much better off, and someone much worse off.