Reading Set 2

Ziji Zhou

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.

```
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_shorts <- flights_joined %>%
  filter(distance <= 500) %>%
  select(name, flight, arr_delay, distance) %>%
  rename(distance_km = distance)
glimpse(flights_shorts)
```

```
Rows: 80,327
Columns: 4
```

```
$ name        <chr> "ExpressJet Airlines Inc.", "JetBlue Airways", "Southwest ~
$ flight      <int> 5708, 1806, 4646, 4144, 1002, 102, 20, 44, 4406, 1172, 183~
$ arr_delay   <dbl> -14, -4, -19, 12, -10, 5, -1, 4, -6, -19, -22, -4, 17, -14~
$ distance_km <dbl> 229, 187, 185, 212, 187, 301, 264, 209, 427, 200, 187, 479~
```

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_joined %>%
  drop_na() %>%
  group_by(name) %>%
  filter(distance <= 500) %>%
  summarise(N = n(), avg_arr_delay = mean(arr_delay), avg_dist_km = mean(distance))
delay_summary
```

```
# A tibble: 12 x 4
   name                            N avg_arr_delay avg_dist_km
   <chr>                       <int>         <dbl>       <dbl>
 1 AirTran Airways Corporation   842         19.7         397
 2 American Airlines Inc.       1428          1.88        187
 3 Delta Air Lines Inc.         1861          1.11        284.
 4 Endeavor Air Inc.            8382          7.45        262.
 5 Envoy Air                   11942          9.80        388.
 6 ExpressJet Airlines Inc.    23222         14.2         306.
 7 JetBlue Airways             14321          8.97        251.
 8 Mesa Airlines Inc.            286         18.0         225.
 9 SkyWest Airlines Inc.          24         10.3         417.
10 Southwest Airlines Co.        200          4.92        170.
11 United Air Lines Inc.        5174          4.94        274.
12 US Airways Inc.              9093          2.22        193.
```

1.4 Rename the four columns in the `delay_summary` data table to `Airline`, `"Total flights under 500 km"`, `"Average arrival delay (mins)"` and `"Average distance (km)"`, respectively, then use `kable(booktabs = TRUE, digits = 0)` to make the final table output in the pdf close to publication quality.

```
delay_summary %>%
  rename("Airline" = name, "Total flights under 500 km" = N, "Average arrival delay (mins)" = avg_arr_d
  kable(booktabs = TRUE, digits = c(0,0,0,1))
```

Airline	Total flights under 500 km	Average arrival delay (mins)	Average distance (km)
AirTran Airways Corporation	842	20	397.0
American Airlines Inc.	1428	2	187.0
Delta Air Lines Inc.	1861	1	283.9
Endeavor Air Inc.	8382	7	262.3
Envoy Air	11942	10	388.3
ExpressJet Airlines Inc.	23222	14	306.2
JetBlue Airways	14321	9	251.4
Mesa Airlines Inc.	286	18	225.3
SkyWest Airlines Inc.	24	10	416.8
Southwest Airlines Co.	200	5	170.2
United Air Lines Inc.	5174	5	274.4
US Airways Inc.	9093	2	192.7

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.

```
recent_names <- babynames %>%
  filter(year >= 2000 & year <= 2017)
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.

```
recent_names_summary <- recent_names %>%
  group_by(sex,name) %>%
  summarise(total_people = sum(n))
recent_names_summary
```

```
# A tibble: 73,332 x 3
# Groups:   sex [2]
   sex   name      total_people
   <chr> <chr>            <int>
 1 F     Aabha               35
 2 F     Aabriella           32
 3 F     Aada                 5
 4 F     Aaden                5
 5 F     Aadhira             77
 6 F     Aadhvika             9
 7 F     Aadhya            1478
 8 F     Aadi                16
 9 F     Aadilynn             5
10 F     Aadison             11
# ... 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 <- recent_names_summary %>%
  filter(total_people >= 10000) %>%
  pivot_wider(names_from = sex, values_from = total_people, values_fill = 0)  %>%
  mutate(ratio = pmin(F/M , M/F)) %>%
  arrange(desc(ratio))
recentnames_balanced_wide
```

```
## # A tibble: 1,010 x 4
##    name         F     M ratio
##    <chr>    <int> <int> <dbl>
##  1 Justice 10947 11267 0.972
##  2 Skyler  17120 22154 0.773
##  3 Quinn   25022 19080 0.763
##  4 Amari   11778 15676 0.751
##  5 Casey   12109 16809 0.720
##  6 Riley   89827 59823 0.666
##  7 Peyton  61217 39261 0.641
##  8 Emerson 18592 11742 0.632
##  9 Charlie 13255 21243 0.624
## 10 Dakota  21950 35840 0.612
## # ... with 1,000 more rows
```

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_wide` different from that in `recentnames_summary` from Problem 2.2?

```
recentnames_summary <- recentnames_balanced_wide %>%
  pivot_longer(-name & -ratio, names_to = "sex", values_to = "total") %>%
  arrange(desc(total))
recentnames_summary
```

```
## # A tibble: 2,020 x 4
##    name    ratio sex    total
##    <chr>   <dbl> <chr>  <int>
##  1 Jacob       0 M     413884
##  2 Michael     0 M     372782
##  3 Emma        0 F     339802
##  4 Ethan       0 M     327315
##  5 Matthew     0 M     326922
##  6 William     0 M     325521
```

```
 7 Emily       0 F     324072
 8 Joshua      0 M     323648
 9 Daniel      0 M     315342
10 Olivia      0 F     303477
# ... with 2,010 more rows
```

The number of observations are doubled because each name has a row for both the count of female and male.

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").

8.4.4 Race Prediction

I found this dilemma, along with the "Gaydar" one, to be the most compelling. In a scientific and research mindset this model seems very straightforward, finding a correlation between last names and race through public census. There are plenty of other similar models to this one available open source with less controversy over the subject. The question of morality in this model stems from the sensitive nature of race itself. I personally believe that because of its open source nature, allowing people to analyze the methods used and possible bias in the model, and the source for the data (the US bureau) the model is ethical. The concern for this model is the misuse of it in racist intentions, however it is hard to pin the blame on the model created from public bureau records.