TA Session 9: Grouped Analysis

Harris Coding Camp

Summer 2023

Background and data

> One thing the email said was that the University is taking steps toward transparency.
>
> What caught my eye: hyperlinks to publicly available data on all traffic stops and "field interviews" (e.g. questioning or searching people) done by University of Chicago Police (UCPD)
>
> — Damon Jones (@nomadj1s) August 14, 2020

First, follow the tweet thread and you'll see that Prof. Damon Jones, of Harris, gets that data and does some analysis. In this exercise, you're going to follow his lead and dig into traffic stop data from the University of Chicago Police Department, one of the largest private police forces in the world.

Download the data here. You can save the file directly from your browser using `ctrl + s` or `cmd + s`. Alternatively, you can read the csv directly from the internet using the link https://github.com/harris-coding-lab/harris-coding-lab.github.io/raw/master/data/data_traffic.csv

Warm-up

1. Open a new `Rmd` and save it in your coding lab folder; if you downloaded the data, move your data file to your preferred data location.
2. In your `Rmd`, write code to load your packages. If you load packages in the console, you will get an error when you knit because knitting starts a fresh R session.
3. Load `data_traffic.csv` and assign it to the name `traffic_data`. This data was scrapped from the UCPD website and partially cleaned by Prof. Jones.
4. Recall that `group_by()` operates silently. Below I create a new data frame called grouped_data.

```
grouped_data <-
  traffic_data %>%
    group_by(Race, Gender)
```

   a. How can you tell `grouped_data` is different from `traffic_data`?
   b. How many groups (Race-Gender pairs) are in the data? (This information should be available without writing additional code!)
   c. Without running the code, predict the dimensions (number of rows by number of columns) of the tibbles created by `traffic_data %>% summarize(n = n())` and `grouped_data %>% summarize(n = n())`.
   d. Now check you intuition by running the code.

5. Use `group_by()` and `summarize()` to recreate the following table.

   ```
   #> # A tibble: 6 x 2
   #>   Race                                         n
   #>   <chr>                                    <int>
   #> 1 African American                          3278
   #> 2 American Indian/Alaskan Native              12
   #> 3 Asian                                      226
   #> 4 Caucasian                                  741
   #> 5 Hispanic                                   217
   #> 6 Native Hawaiian/Other Pacific Islander       4
   ```

6. Use `count()` to produce the same table.

Moving beyond counts

Raw counts are okay, but frequencies (or proportions) are easier to compare across data sets. We want to add a column with frequencies.

1. On your first attempt to write the code, you try this. Explain why the frequencies are all 1.[1]

   ```
   traffic_stop_freq_bad <-
   traffic_data |>
     group_by(Race) |>
     summarize(n = n(),
               freq = n / sum(n))

   traffic_stop_freq_bad
   ```

2. Adjust the code above to get the result and assign the new tibble to the name `traffic_stop_freq`.

```
#> # A tibble: 6 x 3
#>   Race                                         n     freq
#>   <chr>                                    <int>    <dbl>
#> 1 African American                          3278 0.732
#> 2 American Indian/Alaskan Native              12 0.00268
#> 3 Asian                                      226 0.0505
#> 4 Caucasian                                  741 0.165
#> 5 Hispanic                                   217 0.0485
#> 6 Native Hawaiian/Other Pacific Islander       4 0.000893
```

1. The frequencies out of context are not super insightful. What additional information do we need to argue the police are disproportionately stopping members of a certain group? (Hint: Prof. Jones shares the information in his tweets.)[2]

2. Now we want to go a step further.[3] Do outcomes differ by race? In the first code block below, I provide code so you can visualize disposition by race. "Disposition" is police jargon that means the current status or final outcome of a police interaction.

[1] Hint: This is a lesson about `group_by()`!

[2] To be fair, even with this information, this is crude evidence that can be explained away in any number of ways. One job of a policy analyst is to bring together evidence from a variety of sources to better understand the issue.

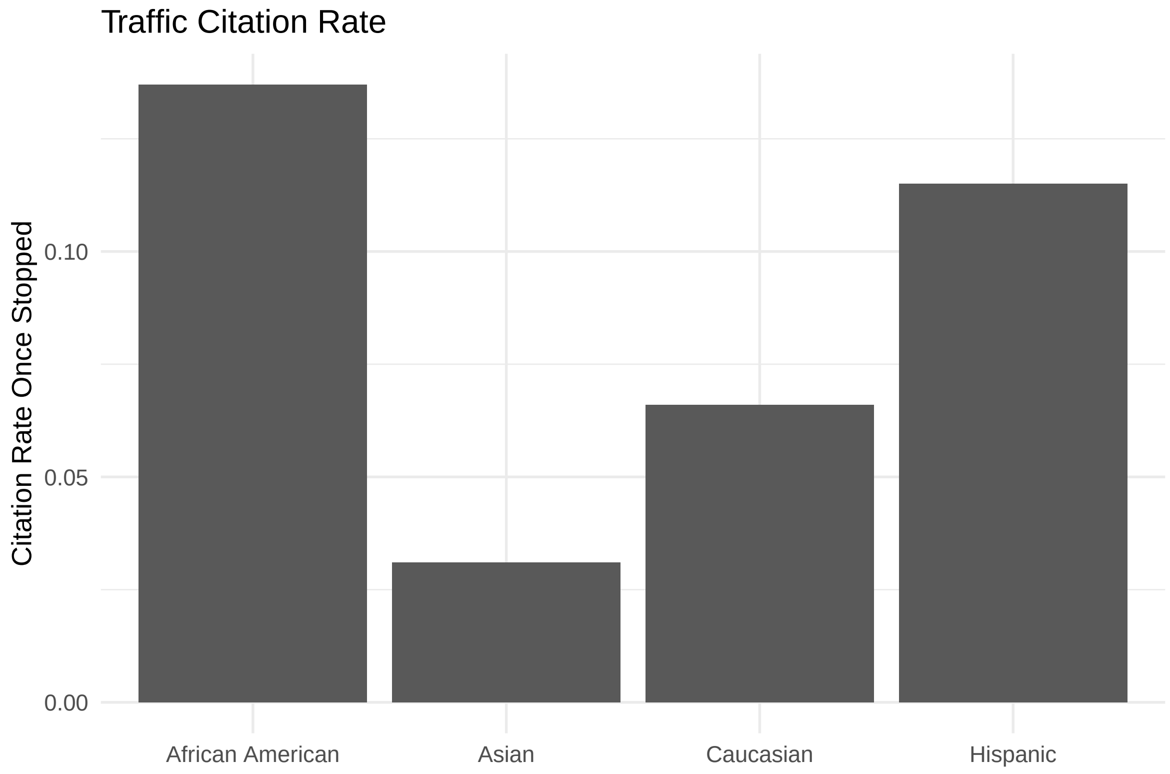
[3] The analysis that follows is partially inspired by Eric Langowski, a Harris alum, who was also inspired to investigate by the existence of this data (You may have seen Prof. Jones retweet him at the end of the thread.)

```
citation_strings <- c("citation issued", "citations issued", "citation  issued" )

arrest_strings <- c("citation issued, arrested on active warrant",
                "citation issued; arrested on warrant",
                "arrested by cpd",
                "arrested on warrant",
                "arrested",
                "arrest")

disposition_by_race <-
    traffic_data %>%
      mutate(Disposition = str_to_lower(Disposition),
             Disposition = case_when(Disposition %in% citation_strings ~ "citation",
                                     Disposition %in% arrest_strings ~ "arrest",
                                     TRUE ~ Disposition)) %>%
      count(Race, Disposition) %>%
      group_by(Race) %>%
      mutate(freq = round(n / sum(n), 3))


disposition_by_race %>%
  filter(n > 5, Disposition == "citation") %>%
  ggplot(aes(y = freq, x = Race)) +
  geom_col() +
  labs(y = "Citation Rate Once Stopped", x = "", title = "Traffic Citation Rate") +
  theme_minimal()
```



Let's break down how we got to this code. First, I ran `traffic_data %>% count(Race, Disposition)` and noticed that we have a lot of variety in how officers enter information into the system.[4] I knew I could deal with some of the issue by standardizing capitalization.

a. In the console, try out `str_to_lower(...)` by replacing the `...` with different strings. The name may be clear enough, but what does `str_to_lower()` do?[5]

After using `mutate` with `str_to_lower()`, I piped into `count()` again and looked for strings that represent the same `Disposition`. I stored terms in character vectors (e.g. `citation_strings`). The purpose is to make the `case_when()` easier to code and read. Once I got that right, I added frequencies to finalize `disposition_by_race`.

3. To make the graph, I first tried to get all the disposition data on the same plot.

```
disposition_by_race %>%
  ggplot(aes(y = freq, x = Race, fill = Disposition)) +
  geom_col()
```

   By default, the bar graph is stacked. Look at the resulting graph and discuss the pros and cons of this plot with your group.

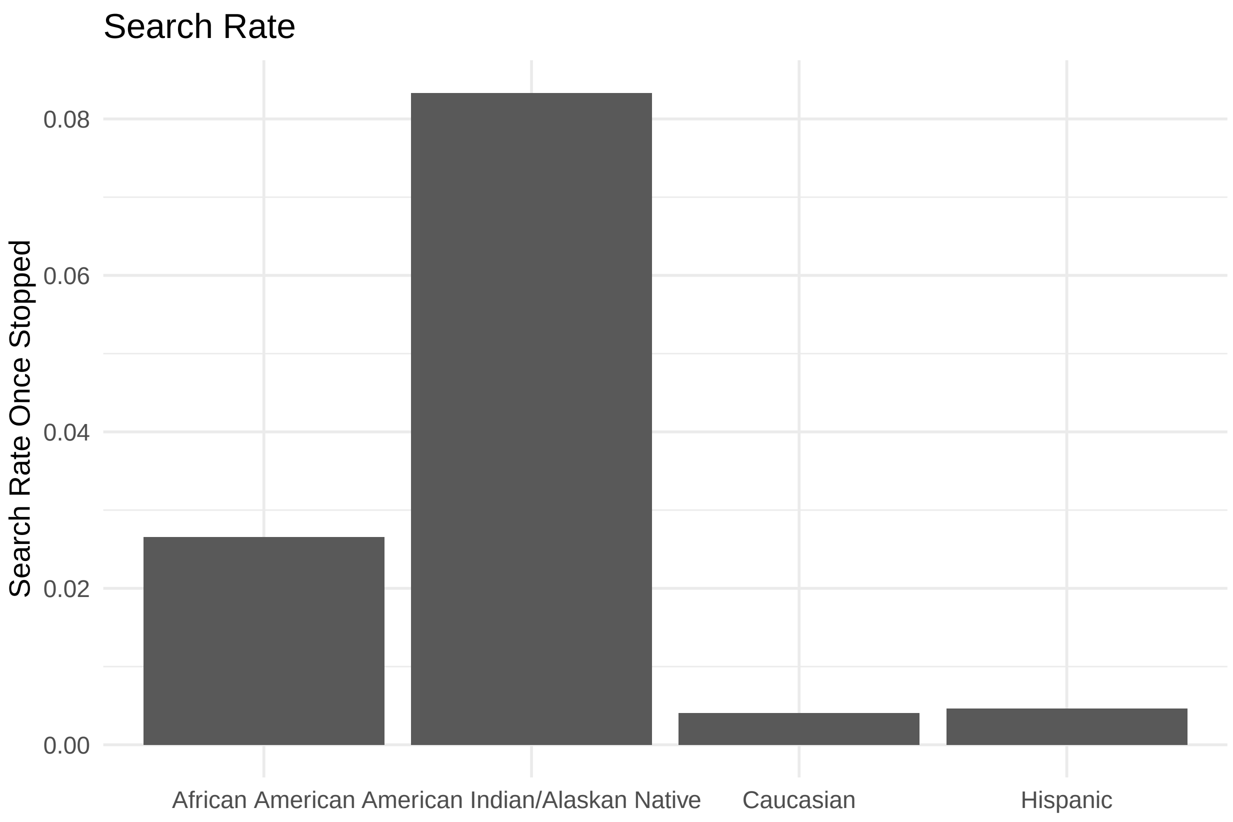
4. I decided I would focus on citations only and added the `filter(n > 5, Disposition == "citation")` to the code.[6] What is the impact of filtering based on `n > 5`? Would you make the same choice? This question doesn't have a "right" answer. You should try different options and reflect.

5. Now, you can create a similar plot based called "Search Rate" using the `Search` variable. Write code to reproduce this plot.

[4] Try it yourself!

[5] This code comes from the `stringr` package. Checkout `?str_to_lower` to learn about some related functions.

[6] Notice that I get the data exactly how I want it using `dplyr` verbs and then try to make the graph.


Search Rate
Search Rate Once Stopped
0.08
0.06
0.04
0.02
0.00
African American
American Indian/Alaskan Native
Caucasian
Hispanic