TA Session 11: Grouped Analysis (solutions)

Harris Coding Camp

Summer 2022

We expect you to **watch the class 4 material**, here prior to lab. In addition, **read the background and data section before lab**.

Background and data

Follow the tweet thread and you'll see that Prof. Damon Jones, of Harris, gets that data and does some analysis. In this lab, 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.

Warm-up

1. Open a new `Rmd` and save it in your coding lab folder. If you have not yet, 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.

```
library("tidyverse")
```

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.

Note: This solution may vary depending on where your csv file is, compared to the Rmd file location. Please refer to Lab 3's Problem Set for more information

```
traffic_data <- read_csv("https://github.com/harris-coding-lab/harris-coding-lab.github.io/raw/master/da
```

4. Recall that `group_by()` operates silently.

a. How can you tell `grouped_data` different from `traffic_data`?

You can use summarise() to check the grouped data:

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

summarise(grouped_data)
```

```
## # A tibble: 14 x 2
## # Groups:   Race [6]
##    Race                                   Gender
##    <chr>                                  <chr>
##  1 African American                       female
##  2 African American                       Female
##  3 African American                       Male
##  4 African American                       <NA>
##  5 American Indian/Alaskan Native         Female
##  6 American Indian/Alaskan Native         Male
##  7 Asian                                  Female
##  8 Asian                                  Male
##  9 Caucasian                              Female
## 10 Caucasian                              male
## 11 Caucasian                              Male
## 12 Hispanic                               Female
## 13 Hispanic                               Male
## 14 Native Hawaiian/Other Pacific Islander Male
```

b. How many groups (Race-Gender pairs) are in the data? (This information should be available without writing additional code!)

SOLUTION:

Fourteen (14) - the number of rows in the tibble.

c. Before 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())`.

SOLUTION:

The traffic_data summary will be a 1x1 tibble and the grouped_data summary will be a 14x3 tibble

d. Now check you intuition by running the code.

SOLUTION:

```
traffic_data %>% summarize(n = n())
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1  4478
```

```
grouped_data %>% summarize(n = n())
```

```
## # A tibble: 14 x 3
## # Groups:   Race [6]
##    Race                                   Gender     n
##    <chr>                                  <chr>  <int>
##  1 African American                       female     4
##  2 African American                       Female  1217
##  3 African American                       Male    2056
##  4 African American                       <NA>       1
##  5 American Indian/Alaskan Native         Female     2
```

```
##   6 American Indian/Alaskan Native         Male      10
##   7 Asian                                  Female    62
##   8 Asian                                  Male     164
##   9 Caucasian                              Female   263
## 10 Caucasian                              male       1
## 11 Caucasian                              Male     477
## 12 Hispanic                               Female    68
## 13 Hispanic                               Male     149
## 14 Native Hawaiian/Other Pacific Islander Male       4
```

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

SOLUTION:

```
traffic_data %>%
  group_by(Race) %>%
  summarize(n = n())
```

```
## # 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.

SOLUTION:

```
traffic_data %>%
  count(Race)
```

```
## # 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
```

Moving beyond counts

1. Raw counts are okay, but frequencies (or proportions) are easier to compare across data sets. Add a column with frequencies and assign the new tibble to the name `traffic_stop_freq`. The result should be identical to Prof. Jones's analysis on twitter.

```
Try on your own first. If you're not sure how to add a frequency though, you could google "add a propor
```

SOLUTION:

```
traffic_stop_freq <- traffic_data %>%
  group_by(Race) %>%
  summarise(n = n()) %>%
  mutate(freq = n / sum(n))

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
```

2. 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.)[1]

SOLUTION:

Prof Jones compares these frequencies with two other frequencies: the demographic breakdown of Hyde Park and the breakdown of UChicago Students races.

3. For the problem above, your groupmate tried the following code. Explain why the frequencies are all 1.[2]

SOLUTION:

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

traffic_stop_freq_bad
```

As explained in the linked stackoverflow post, the last grouping variable is peeled off *after* the summarise function, by default. So, if you calculate frequencies within the summarize function, the data will still be

[1] 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.

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

grouped by race and therefore each frequency must be 1. However, if you calculate frequencies after the summarise function, the whole data will be ungrouped and frequencies can be properly calculated.

4. Now we want to go a step further than Prof. Jones.[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.

````
```r
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()
```

![](TA-session11-solutions_files/figure-latex/unnamed-chunk-11-1.pdf)<!-- -->
````

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]

```
traffic_data %>%
  count(Race, Disposition)
```

```
## # A tibble: 31 x 3
##    Race             Disposition                                      n
```

[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.)

[4] Try it yourself!

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

```
##    <chr>            <chr>                                       <int>
##  1 African American Arrest                                          1
##  2 African American Arrested                                        1
##  3 African American Arrested by CPD                                 1
##  4 African American Arrested on warrant                             1
##  5 African American Citation  Issued                                2
##  6 African American Citation issued                               127
##  7 African American Citation Issued                               297
##  8 African American Citation Issued, Arrested on Active Warrant     1
##  9 African American Citation issued; arrested on warrant            1
## 10 African American Citations issued                                5
## # ... with 21 more rows
```

```
str_to_lower("Citation Issued")
```

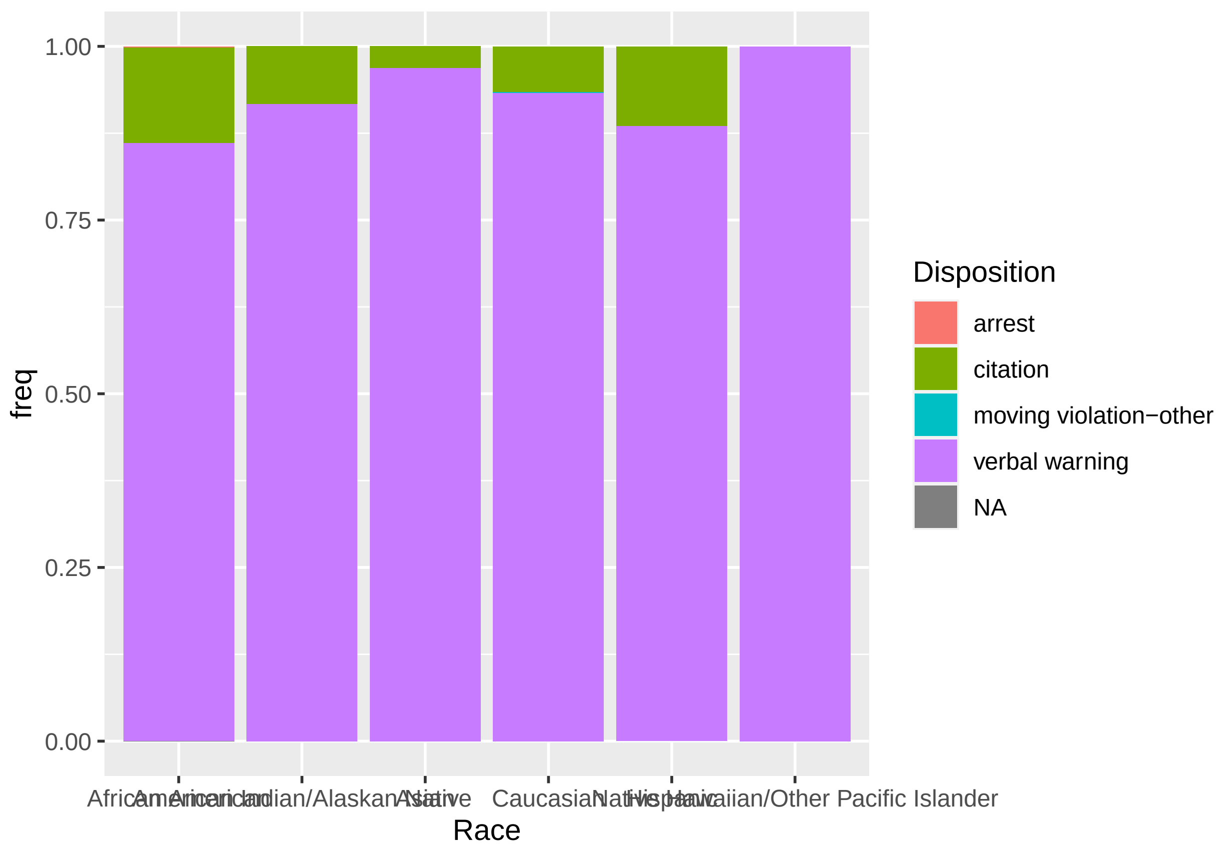
```
## [1] "citation issued"
```

```
After using `mutate` with  `str_to_lower()`, I piped into `count()` again and looked for strings that re
See code above.
```

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

SOLUTION:

```
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.

6. 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.

SOLUTION:

Here are some arguments (not a comprehensive list):

Against:

- We throw away information.
- `n` here is already subdivided based on "Disposition", but it would make more sense to filter based on number of observations for a given race rather than a race-disposition count.

For:

- small `n` groups can be misleading since one interaction can sway the result significantly.
- An alternative is to create an "other" category, though that might bury heterogeneity across the smallest groups.

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

7

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

SOLUTION:

```
search <- traffic_data %>%
  mutate(Search = str_to_lower(Search),
         Search = ifelse(is.na(Search) | Search == "N/A","No" , Search)) %>%
  count(Race, Search) %>%
  group_by(Race) %>%
  mutate(freq = n / sum(n))


search %>%
  filter(Search == "yes", n > 0) %>%
ggplot(aes(y = freq, x = Race)) +
      geom_col() +
      labs(y = "Search Rate Once Stopped", x = "", title = "Search Rate") +
      theme_minimal()
```