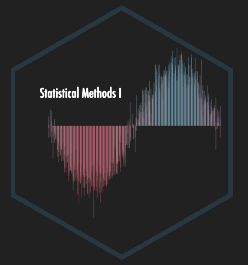
Distributions and Central Tendency

EDP 613

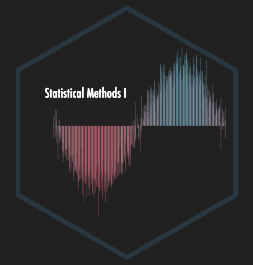
Week 4

Prepping a New R Script

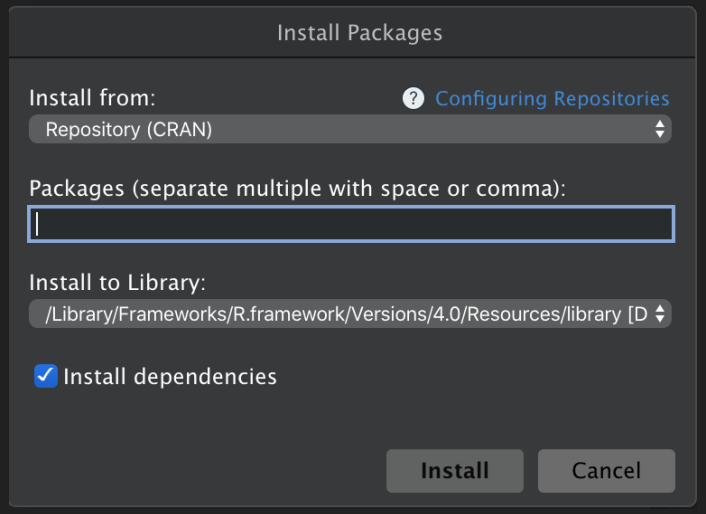


1. Open up a blank R script using the menu path **File > New File > R Script**.
2. Save this script as `whatever.R` (replacing the term `whatever`) in your R folder. Remember to note where the file is!
3. After you have saved this file as `whatever.R`, go to the menu and select **Session > Set Working Directory > To Source File Location**.

Getting ready for this session



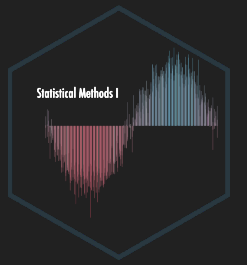
- Get the file `teampolview.csv` and save it in the same location as this script.
- Install the package `pacman`. Remember you can download it using **Tools > Install Packages** and typing in the name. Please make sure the **Install Dependencies** option has a checkmark beside of it. The install may take a minute.



- `pacman` will automatically install a package if you don't have it and load it up for you.


```
pacman::p_load(tidyverse)
```

Use the Pipe



- Here's what it looks like: `%>%`.
- In RStudio, you can take a shortcut:
  - For Windows: `Ctrl+Shift+M` (Windows)
  - For Macs: `Cmd+Shift+M` (Mac)

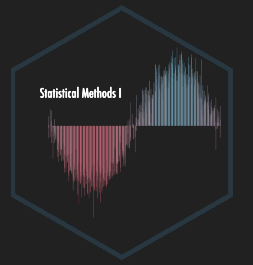
Basic Logic


```
"get up in the morning" %>%
  "drink a lot of coffee" %>%
  "come to work" %>%
  "do stuff" %>%
  "go home "%>%
  "eat" %>%
  "sleep (maybe)"
```

- works like layers
- you can highlight parts of it to run

```r
starwars %>%
  select(name, species, homeworld) %>%
  head()
```

```
# A tibble: 6 × 3
  name           species homeworld
  <chr>          <chr>   <chr>
1 Luke Skywalker Human   Tatooine
2 C-3PO          Droid   Tatooine
3 R2-D2          Droid   Naboo
4 Darth Vader    Human   Tatooine
5 Leia Organa    Human   Alderaan
6 Owen Lars      Human   Tatooine
```

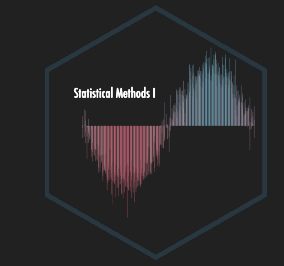


Run a smaller chunk

Highlight the first two lines and run it

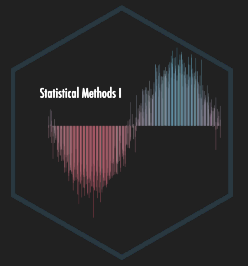
```
starwars %>%
  select(name, species, homeworld) %>%
  head()
```

Output



Now on to Descriptives

Frequency distributions



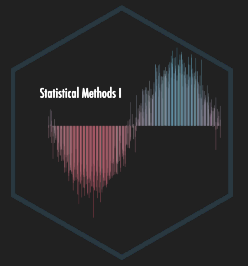
- *Frequency distribution* tells us how many observations there are at different values of a variable.
- You could count manually...but why?
- We can have R do the work for us using a *frequency table*

```r
starwars %>%
  select(name, species, homeworld) %>%
  count(species)
```

```
# A tibble: 38 × 2
   species       n
   <chr>     <int>
 1 Aleena        1
 2 Besalisk      1
 3 Cerean        1
 4 Chagrian      1
 5 Clawdite      1
 6 Droid         6
 7 Dug           1
 8 Ewok          1
 9 Geonosian     1
10 Gungan        3
# … with 28 more rows
```

```
starwars %>%
  select(name, species, homeworld) %>%
  count(species, homeworld)
```

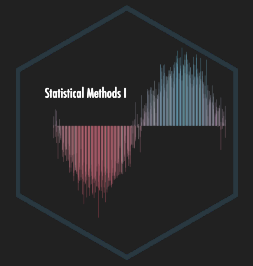
```
# A tibble: 58 × 3
   species  homeworld       n
   <chr>    <chr>       <int>
 1 Aleena   Aleen Minor     1
 2 Besalisk Ojom            1
 3 Cerean   Cerea           1
 4 Chagrian Champala        1
 5 Clawdite Zolan           1
 6 Droid    Naboo           1
 7 Droid    Tatooine        2
 8 Droid    <NA>            3
 9 Dug      Malastare       1
10 Ewok     Endor           1
# … with 48 more rows
```



Better but a large table is difficult to picture...

```
starwars %>%
  select(name, species, homeworld) %>%
  count(species, homeworld) %>%
  arrange(-n)
```

```
# A tibble: 58 × 3
   species  homeworld     n
   <chr>    <chr>     <int>
 1 Human    Tatooine      8
 2 Human    Naboo         5
 3 Human    <NA>          5
 4 Droid    <NA>          3
 5 Gungan   Naboo         3
 6 Human    Alderaan      3
 7 Droid    Tatooine      2
 8 Human    Corellia      2
 9 Human    Coruscant     2
10 Kaminoan Kamino        2
# … with 48 more rows
```



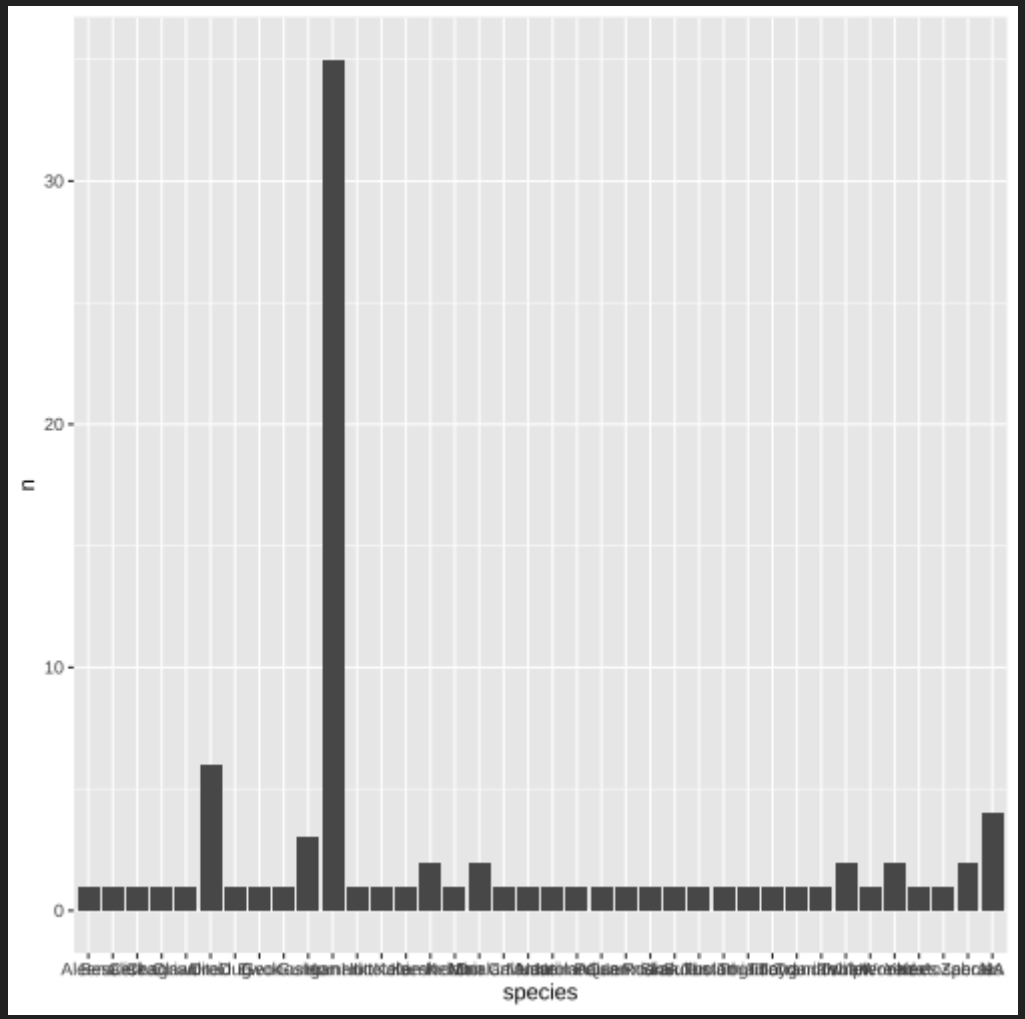
Well that's better but nothing really beats a picture so...

Let's Make a Bar Plot

1. Assign the data to a variable

```
sw_counts <-
  starwars %>%
  select(name, species, homeworld) %>%
  count(species)
```

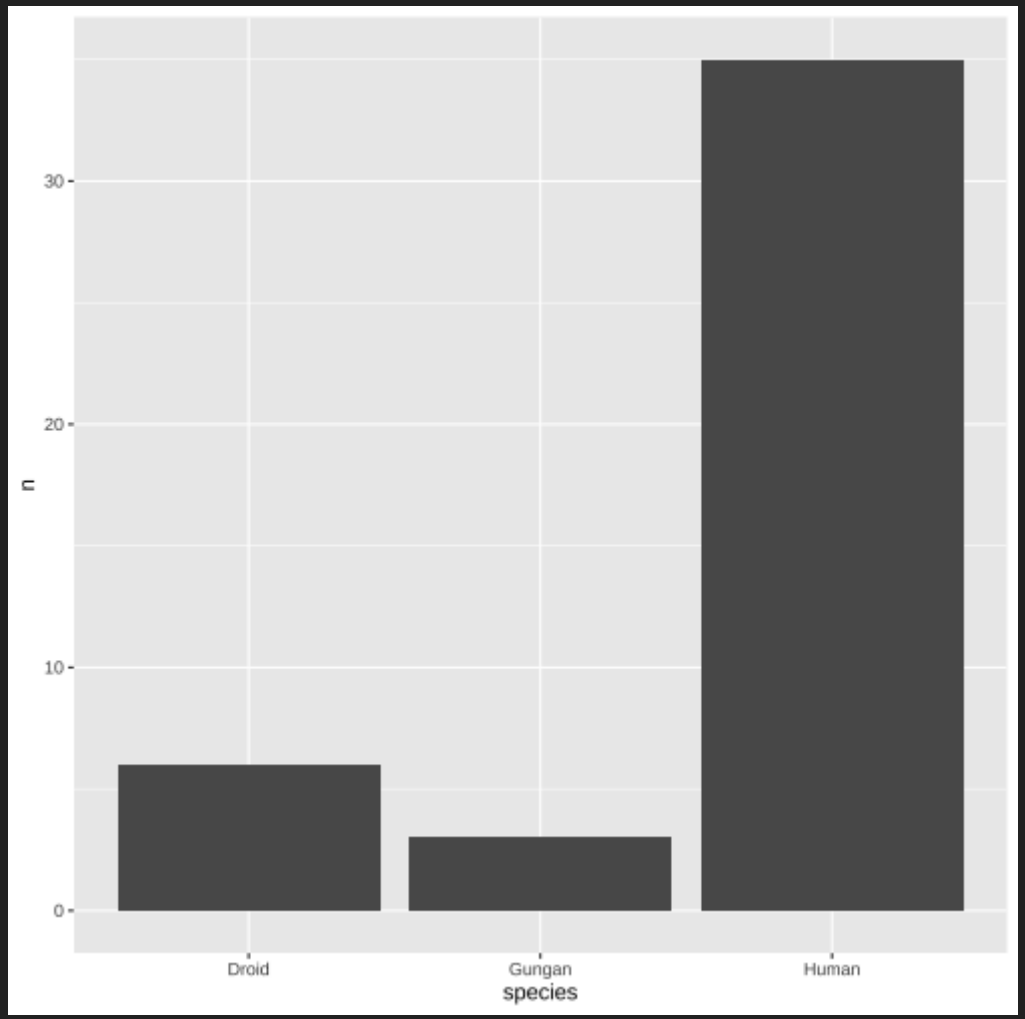
```r
ggplot(data = sw_counts,
       aes(x = species, y = n)) +
  geom_bar(stat = "identity")
```



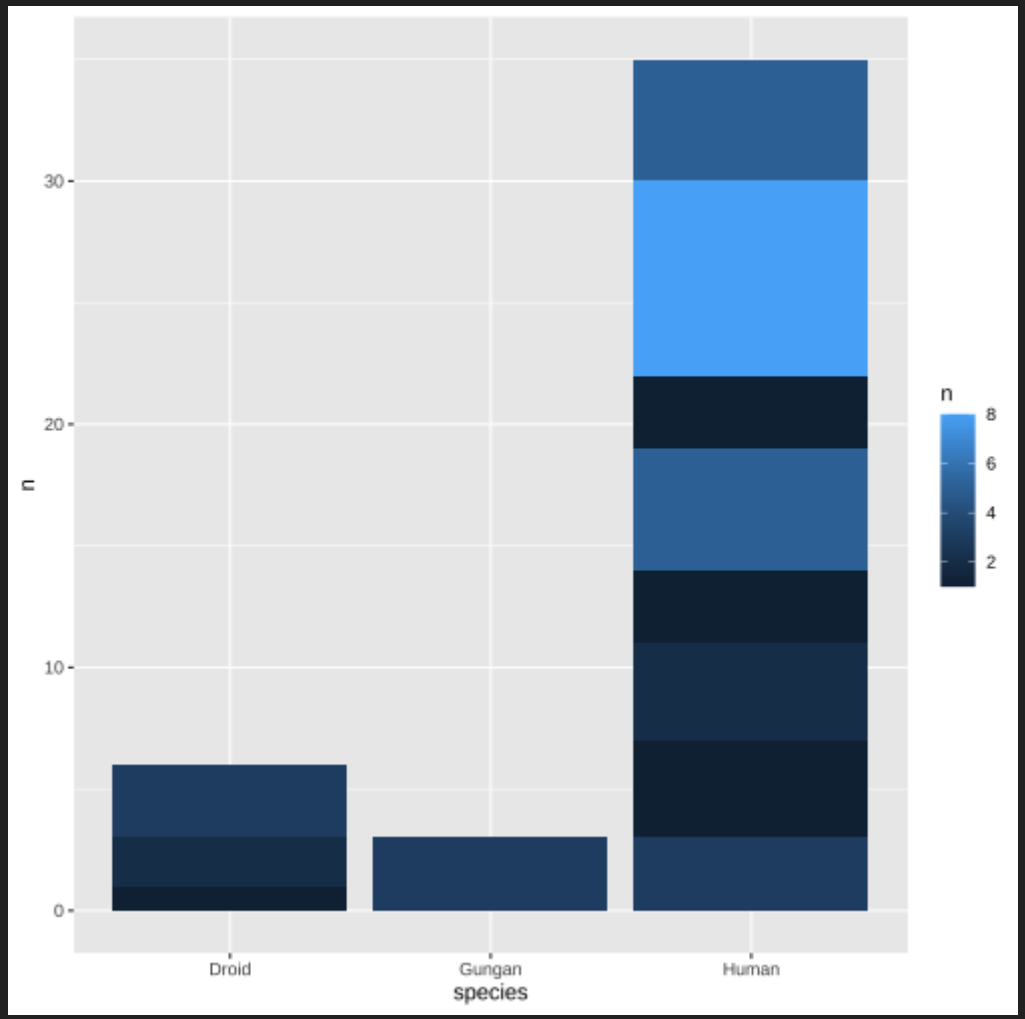
Well that looks terrible

- Maybe we can just look at a few of them

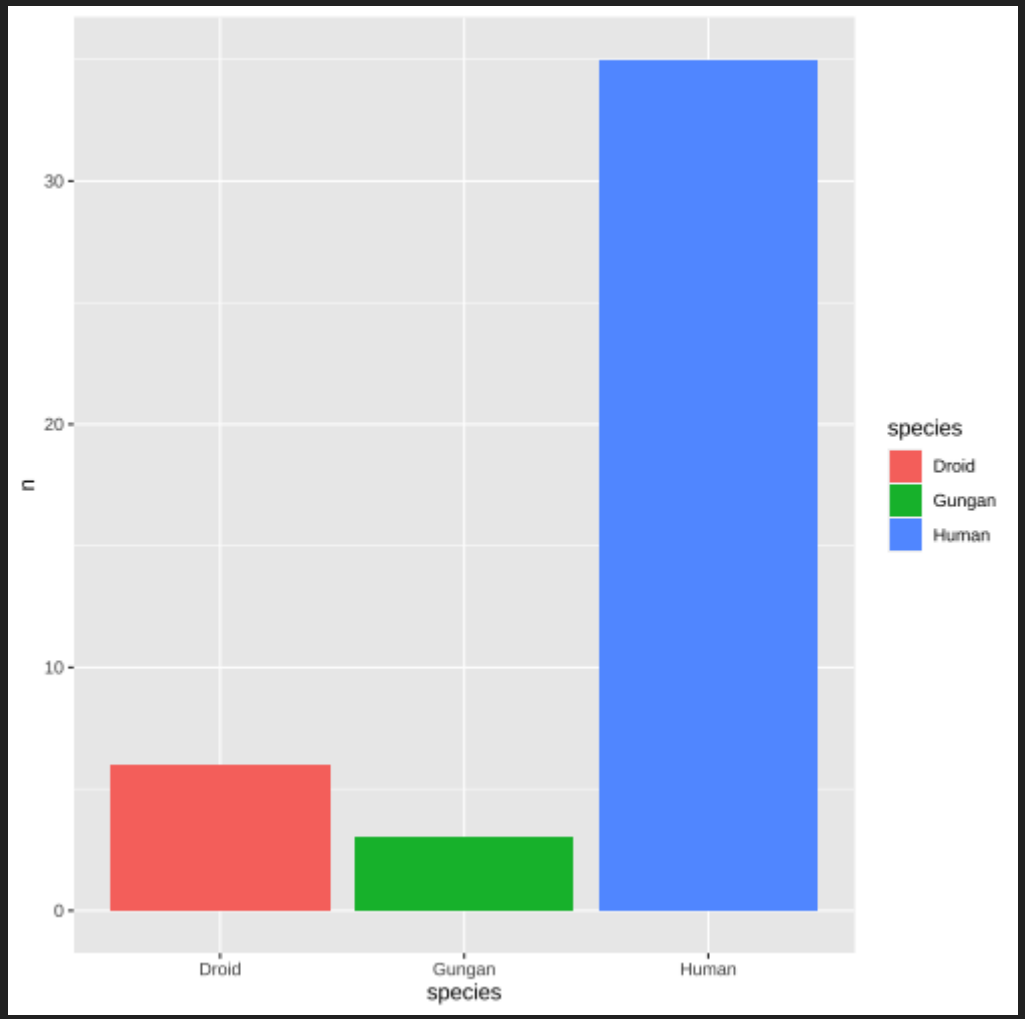
```r
ggplot(data = sw_filtered,
       aes(
         x = species,
         y = n
         )
       ) +
  geom_bar(stat = "identity")
```



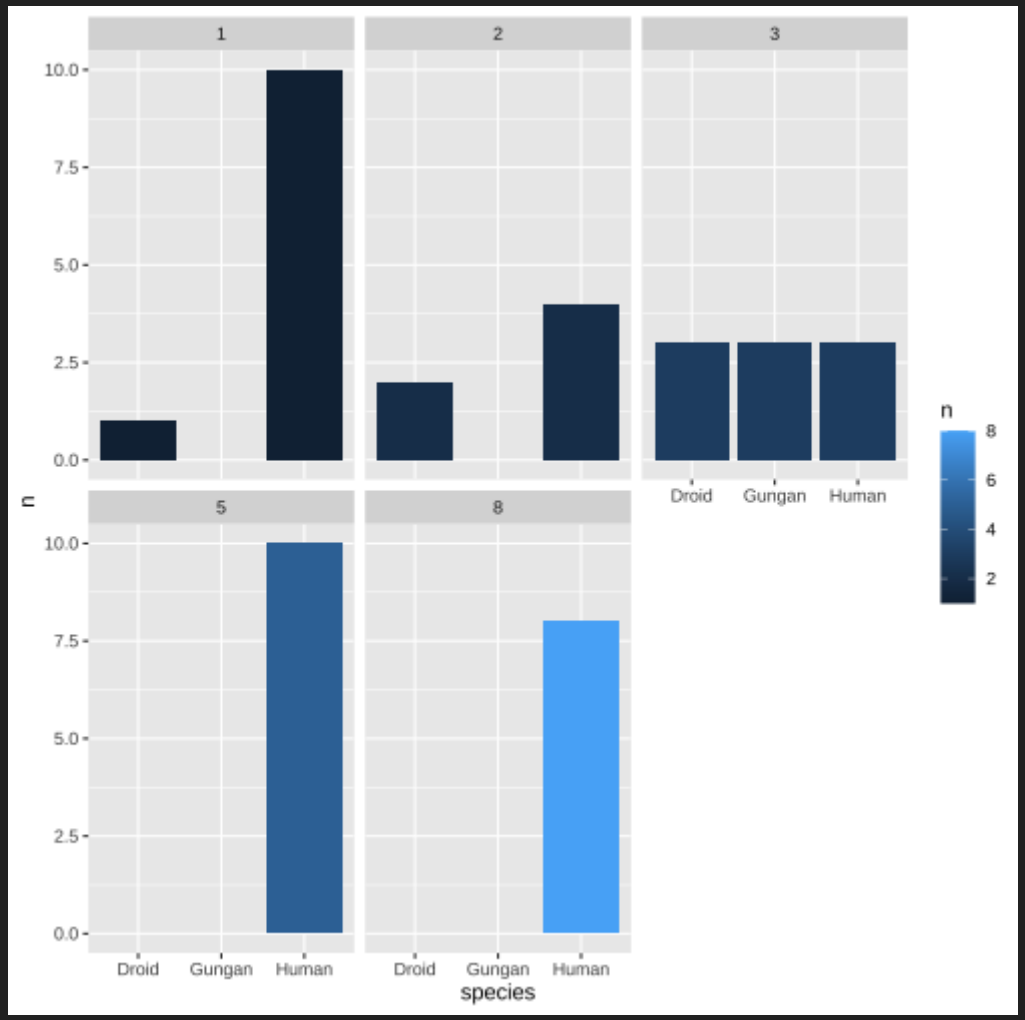
```
ggplot(data = sw_filtered,
       aes(
         x = species,
         y = n,
         fill = n
         )
       ) +
  geom_bar(stat = "identity")
```



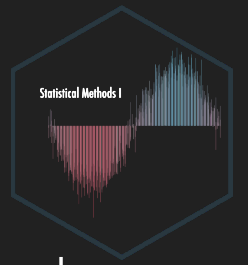
```r
ggplot(data = sw_filtered,
       aes(
         x = species,
         y = n,
         fill = species
         )
       ) +
  geom_bar(stat = "identity")
```



```
ggplot(data = sw_filtered,
       aes(
         x = species,
         y = n,
         fill = n
         )
       ) +
  geom_bar(stat = "identity") +
  facet_wrap(n ~ .)
```



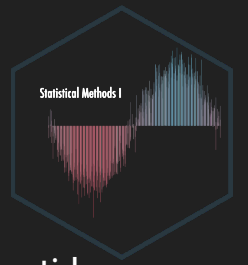
Loading up local data



To explore this, let's load the 2012 voter fraud file first and assign it to a variable. We can do this using the `read_csv()` command from the `readr` package within `tidyverse`.

```
voter_fraud <- read_csv("2012_Voter_Fraud.csv")
```

Side note

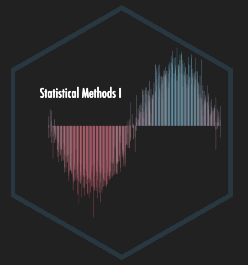


R itself uses `read.csv` which can be a royal pain if you don't know what you're doing. Its strongly advised that you stick with the tidy way of loading data.

Remember:

> - `read_csv` with a `_` is tidy
> - `read.csv` with a `.` is messy

Measures of Central Tendency

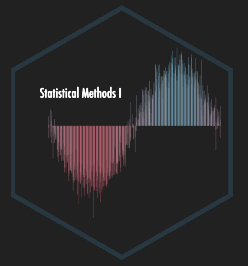


To take a look at how we assess the mean, median, and mode, let's use our original data set and first look at the `total` column which has the raw data counts.

```
 voter_fraud %>%
   select(total)
```

```
# A tibble: 50 × 1
   total
   <dbl>
 1     6
 2    11
 3     1
 4    24
 5    20
 6    21
 7     3
 8     9
 9    48
10    20
# … with 40 more rows
```

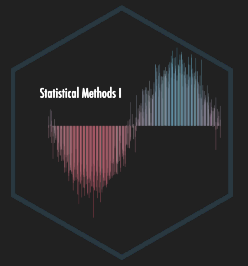
For the mean, we use




```
 voter_fraud %>%
   summarize(Average = mean(total))
```

```
# A tibble: 1 × 1
  Average
    <dbl>
1    13.3
```

For the median, we use




```
 voter_fraud %>%
   summarize(Average = median(total))
```

```
# A tibble: 1 × 1
  Average
    <dbl>
1      11
```

For the mode, we use

```
 voter_fraud %>%
   summarize(Average = mode(total))
```

```
# A tibble: 1 × 1
  Average
  <chr>
1 numeric
```

`mode` still doesn't work!

A Mode You Can Use

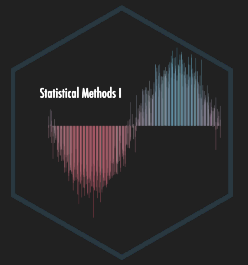

```r
Mode <- function(x) {
  ux <- unique(x)
  ux[which.max(tabulate(match(x, ux)))]
}

# Notice that 'Mode' is capitalized so that R won't confuse it
# with its internal command 'mode'.
```

```r
 voter_fraud %>%
   summarize(Average = Mode(total))
```

```
# A tibble: 1 × 1
  Average
    <dbl>
1       4
```

On Your Own



This is your chance to get some practice in and to ask questions. You won't get the opportunity to get help during quizzes and exams so take advantage now!

Open up a new script and load up the `Box Office.csv` data set in R. This set was scraped from Rotten Tomatoes prior to Avengers: Endgame becoming the highest grossing movie of all time.

Now try answering the following questions using R:

1. What is the average number of positive reviews for the top five movies?
2. What are the average number of negative reviews for the bottom five movies?
3. How were movies released over the years? Provide counts and a visualization.
4. Which measure of central tendency is the best to describe the average number of movies over the years?
5. Which year has the most number of ranked movies?

I'll post the solutions next week!

That's it for today!