Exercise 6: Pivoting to the Presidential Election

ECON 256 Data Analysis and Visualization

Objective

Practice Pivots and Joins.

Make an election map for the 2020 Presidential Election.

1 Set up Your R Workspace

Set up a R script with the normal preface. eg:

```
setwd ("/Users/justintyndall/Desktop/myproject")
library(tidyverse)
```

2 Load Data Sets

The MIT Election Lab publishes detailed data on US elections. All the data is freely available here:

MIT Election Lab Data.

I've provided a .csv version of the county level 2020 Presidential election results data on Laulima (election2020.csv). Download the file to your working directory.

I have also posted a data set containing the latitude and longitude coordinates of every county in the US (latlon.csv). Download this data set to your working directory as well.

Write two `read_csv()` commands to load the data sets into R as objects.

3 Clean the Election Data

Take a look at the election data. The Trump column shows the number of votes Trump received in that county. The Biden column shows the number of votes Biden received.

Think about whether this is "Tidy Data." One of the conditions was: Each variable must have its own column. In this case, we have one variable (number of votes) spread across two columns, we would prefer the data to be "longer" so that number of votes is contained by one column.

Make the data "long" by using the `pivot_longer()` function.

Your command should have this form, with the blanks filled in:

```
_____<-pivot_longer(_____,cols=c("_____","Biden"),names_to="_____",values_to="_____")
```

Make use of the help file (`?pivot_longer`) to recall the arguments. You should experiment with the `names_to=` and `values_to=` arguments to generate variable names that make sense.

4 Keep the Winner from Each County

Lets make a plot that shows the winner of each county. Let's filter the data to only include one observation per county, where that observation indicates the county's winning candidate.

You can accomplish this in three lines of code. (Optional: Try to use piping (%>%)):

1. use the `group_by()` function to group the observations by `county_fips`.
2. use the `mutate()` function to create a variable that is equal to the maximum number of votes received in each county. For example: `newobject<-mutate(oldobject,maxvotes = max(votes))`. Because we have grouped the data, the `max()` function will only take the maximum from within each group.
3. use the `filter()` function to keep only those observations where the number of votes won equals (==) the maximum for that county.

You should end up with a data set of 3,153 observations, one observation for each county.

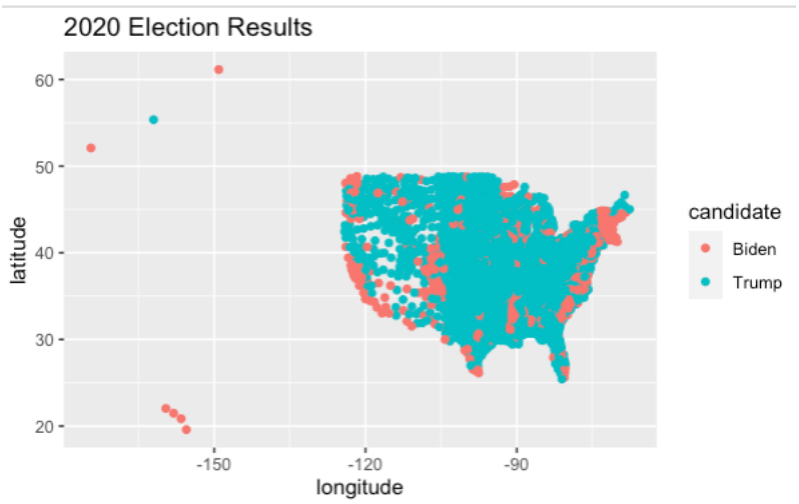
5 Join on Latitude/Longitude Data

Now use an `inner_join()` function to join the object you created in step 4 to the latitude and longitude object. You can join the two objects by the "county_fips" variable that they have in common. We use an `inner_join()` (rather than a `left_join()` or `right_join()`) to only keep observations for which we have both location and voting data.

6 Plot the Data Spatially

Finally, use `ggplot()` with `geom_point()` to plot the data. For aesthetics: make longitude your x axis variable and latitude your y axis variable. Additionally, set the "color" aesthetic to correspond to the variable containing the candidate name.

You should end up with something that looks similar to this:



(Note that plotting points by latitude and longitude essentially gives you a map. Next week we will start discussing how to make maps that look a bit nicer.)

7 Send me Your Code

Save your R script. Name it with your last name, followed by the exercise number and submit it on Laulima.