Zach Quinn

DSC 540

Milestone 5: Summary

Initially, cleaning, parsing and combining data files into a database were daunting tasks. I was glad that we had two weeks for each milestone because sometimes I felt like I didn’t necessarily figure out what I needed to know until the last minute. However, I did learn an important lesson about coding, workflow and logic this semester: Find the shortest way to conceive, implement and execute a programmatic task. It’s incredibly simple but I catch my peers (in peer reviews for other courses) and myself overcomplicating processes that can sometimes be solved with a few lines of code or a concise function rather than complex nested loops or recursive programming. This isn’t to say that writing concise code is necessarily a short cut. In fact, I think I’ve spent more time determining how to complete a project in a concise manner than I would if I had just approached it with brute force. However, coding concisely is important because it is much easier to spot errors, communicate intent (with inline comments) and specify objects.

For this project, I wanted to explore the implications of COVID-19 on minority ethnic populations. While I initially intended to conduct a comprehensive analysis other course work and obligations prevented me from doing so. Instead, I completed the requisite tasks. The CSV data was easiest to import and I cleaned it by rearranging and renaming column names to be explicit about what I was intending to analyze. The web data proved to be one of the most challenging portions of the project because it combined both HTML and Python concepts. After failing to write a loop comprehensive enough to capture the data from the embedded table, I discovered Pandas read\_html function and was able to indicate the precise index value of the table, which Pandas scraped seamlessly.

Locating and formatting API data was easier than I anticipated, especially after completing the Twitter API activity. Milestone five was challenging in that I was indecisive regarding my approach. Initially, I thought I would create a table from scratch in SQL Lite. However, then I determined that it would be far more efficient to import each data frame as a CSV file and add them to the existing data base. I was also a bit confused about how exactly I was going to synthesize three separate data sources, especially because my first attempts yielded thousands of NA values. However, I did some post-import cleaning and imputed the NA values with 0 to maintain the position of the columns and rows. From there, I created two queries. The first was selecting values from the API table to determine values with a 14 day cumulative infection rate of greater than 10.00. I plotted the resulting values using Seaborn because I prefer the package to Matplotlib. After making the initial query, I used pd.concat to combine all three data frames and then entered this table into my created database. Finally, I created a query to determine the number of individuals under 50 who were infected with COVID-19.