Tyler Cartwright

Methods

I am working in Linux which means my resources are a bit different than the Windows resources I have used in the past. Some resources I had used in the past and chose to use here were: Python, Python libaries including pandas, numpy, matplotlib, and csv. Some new resources I used were LibreOffice Calc, PyPlot, and Plotly Express.

The two types of visualizations I ultimately chose to create were a choropleth map and some pie charts.

Choropleth Map

Because of the sheer size of the data, spreadsheet work with the initial data was out of the question. I used sqlite3 in Python to convert the CSV file into a DB file that could be queried more reliably. I spent a few hours running queries to get a feel for the data. This included identifying all the distinct provider types, sorting to see where payments were highest, getting only doctors named David, and evaluating the different HCPCS codes and noting the provider types. Eventually the most compelling piece of data was the provider type and decided to focus on this particular aspect of the data.

My first interpretation of the provider type was latching on to a specific one and using the thorough location-based data to visualize where that particular type of service was most common; of all the specialties here, Dermatology stood out as one with obvious implications and preconceived notions. I assumed that warm, coastal regions would demand the most dermatologists.

I chose to group by ZIP code and get a sum of both the distinct services column or beneficiaries column. These were unsurprisingly similar, with many of the same cities at the top of the list: Little Rock, Delray Beach, Houston, The Villages, Norfolk, Naples, Tyler, Newport Beach, and New York were all top 10 cities using either methods. So while this initial query did certainly support the theory that states like California and Florida were particularly prone to skin disease, seeing Little Rock and New York rank so highly reminded me that without accounting for population, this data would emphasize larger cities everywhere over small towns where dermatological concerns were just as prevalent.

Failed Detour

I wanted to implement population into this data before I made any conclusions about it.

My 2020 population data was supplied by census.gov. The xlsx file provided already broke down annual population estimates by city/town, but it was formatted as Townname classification (town/city), State. I used find/replace to remove the town/city suffix, split the columns on the comma, and used VLOOKUP and a table with state names and abbreviations to fill a column with state abbreviations. Finally, I concatenated the city and state with a comma separator to unify into one column which I would also do for the results of the query, so I could do another VLOOKUP to combine the data. After this process was complete and I compared results, the population data unsurprisingly paired with only half the data. I determined it would be best to discard the population numbers at this point, but return to it later along with other data points to draw conclusions about the findings without.

Back to the Map

I settled on the following query:

```
SELECT city, state, zip, SUM(beneficiaries) FROM cms_data WHERE
providertype='Dermatology' GROUP BY zip
```

This data was mostly acceptable as it generated, but I did have to re-introduce the leading zeros to the ZIP codes here.

I decided that the visualization I would create for this data would be a choropleth map, which could be done with FIPS codes. While the CMS data did contain FIPS codes, these were two digit state prefixes; I needed five digit FIPS codes. The best way to generate these seemed to be a ZIP to FIPS crosswalk from Kaggle. I also had to re-introduce leading zeros to this data, which came in a CSV. This was much more manageable than the population file; only 15 records were missing a matching FIPS code. I was able to manually input these for 13 records, however I was left with Apo AE and Fpo ZZ. Because these correspond with overseas military addresses, they would not be visualized with a US map anyway, so I chose to discard them. Once I felt comfortable with this data, I made a new csv with just the fips and beneficiaries columns and ran one last query to group by FIPS code. The map would use these two data points.

I created another Python script for generating the python using a URL and JSON library to access a json document hosted by Plotly that pairs FIPS codes with a map of the US and plugged in my csv file. The visualization generated without issue, but the main concern was determining what range to use; because of the nature of this data, some FIPS codes accounted for less than 100 beneficiaries, while the highest was just over 500,000; because of this discrepancy, using the minimum and maximum number to define the range for color scaling meant that a lot of insight was lost when the majority of the counties visualized were an almost identical shade of purple. I tweaked this number until it was extreme outliers that looked more alike; the number I settled on as my max was 75,000.

Pie Charts

I went back to the start to work with data that more directly targets the area we're interested in. Since the data I was given references locations by ZIP code, I determined the best way to get data on Volusia, Flagler, Seminole, Brevard, St. Johns, Marion, and Putnam County would be to pull a list of ZIP codes for these counties. Zillow proved to be a great resource here. I collected all the listed ZIP codes, used find and replace in gedit to get a comma separated list I could implement into a query. It should be noted that some ZIP codes, such as 32792 for Winter Park, can cross into multiple counties; I am choosing to include this data as well.

The data I intended to work with would be the provider specialties once again, but this time I would focus strictly on counting the number of providers in each specialty and representing this with a display of some sort (most likely a pie chart). I also thought it would be interesting to do a separate breakdown for male and female doctors to see trends in what sorts of specialization a woman would prefer over a man and vice versa.

I did two queries here, one to get the actual data I'd be using and one to check my work:

To check my work:

```
SELECT * FROM cms_data WHERE zip IN ([LONG LIST OF ZIP CODES]) GROUP BY npi
ORDER BY providertype
```

To get a count of provider types:

```
SELECT providertype, COUNT(DISTINCT npi) FROM cms_data WHERE zip IN ([LONG LIST OF ZIP CODES]) GROUP BY providertype ORDER BY COUNT(DISTINCT npi) DESC
```

By comparing the two result sets I was able to see that the counts generated were accurate. I then decided to do two additional variants of the count query for pulling specialties by gender.

Once I had the data in csv files, it was relatively easy to generate a Matplotlib pie chart, however the formatting did prove to be challenging. Because of the sheer volume of specializations, particularly ones with low counts, the labels output overlapped so dramatically that it was impossible to tell what they said at first. For the sake of practicality, I chose to manually manipulate the data and combine all small values (sub-20 in the main data, sub-10 in the gendered data) into an ‘Other’ category directly in the CSV.