Ibrahim A. Rasheed
April 16, 2017

SI 330 Final Project: Relationship Between Michigan County Per Capita Income and Number of Hospitals

Introduction

My final project proposal is to see whether there is a relationship between a Michigan county's per capita income or population and the number of hospitals in that county. My goal of this project is to begin gaining exposure to analyzing public healthcare data. I want to go into healthcare when I graduate so this project will give me an introduction to considering the context of these variables and how they change the meaning.

For example, Wayne County, which is the home of Detroit, has a lower per capita income ($22,125) than Oakland county which has the largest ($36,138), however, it has over 600,000 more people (1,820,584 vs. 1,202,362 respectively). Lower income is generally associated with poorer health, and do those people have enough facilities to address that? Or are there more hospitals in more wealthy areas for those who can afford them?

Data Sources

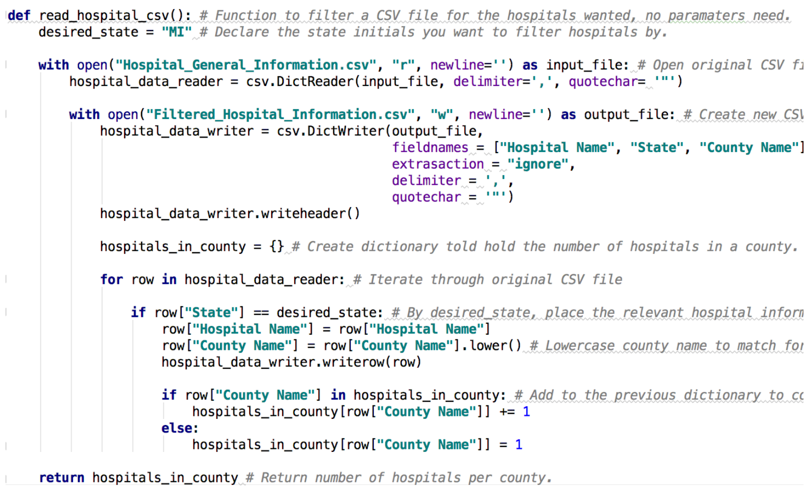
I use per capita income compared to the other representations of income because it is already normalized for population.

Data Set 1: List of Hospitals with Medicare	Data Set 2: Michigan Counties with Income Data
Size: 2.4 MB	Size: 83 county records
URL: https://data.medicare.gov/Hospital-Compare/Hospital-General-Information/xubh-q36u	URL: https://en.wikipedia.org/wiki/List_of_Michigan_locations_by_per_capita_income - cite_note-2
CSV file downloaded from Medicare website	Webpage scraping using BeautifulSoup

Data Processing Steps

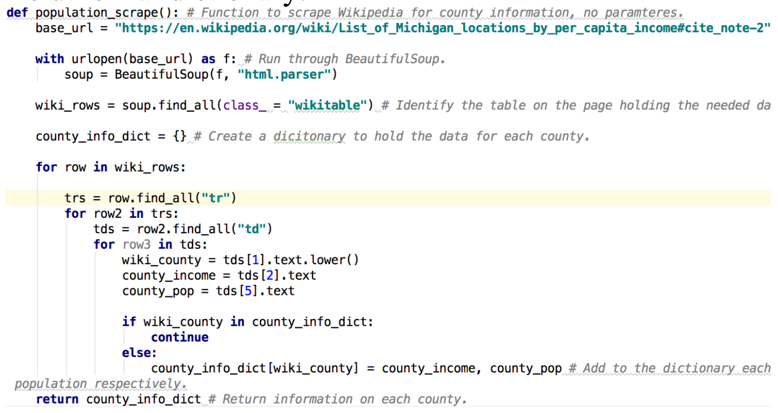
- Step 1:
  - Download the "Hospital General Information" file as a CSV from the Medicare Data website.
- Step 2: Function read_hospital_csv
  - Purpose: Reads the Hospital CSV file to find the number of hospitals per county.
  - Input/Output:
    - Takes no input.
    - Reads the Hospital csv file.
    - Outputs a dictionary hospitals_in_county which has the county name as a key and the number of hospitals in that county as a value.
  - Details:
    - Uses csv.DictReader and csv.DictWriter.
    - Hospitals are filtered by the desired_state, Michigan.
    - Michigan hospitals' names, state, and county are written into a new csv file, "Filtered Hospital Information".

        - The county names are lowercased to match with the web scraping data later.
        - Add the county names as keys to a dictionary which count the number of hospitals in that county as a value.
        - Returns that dictionary, hospitals_in_county.



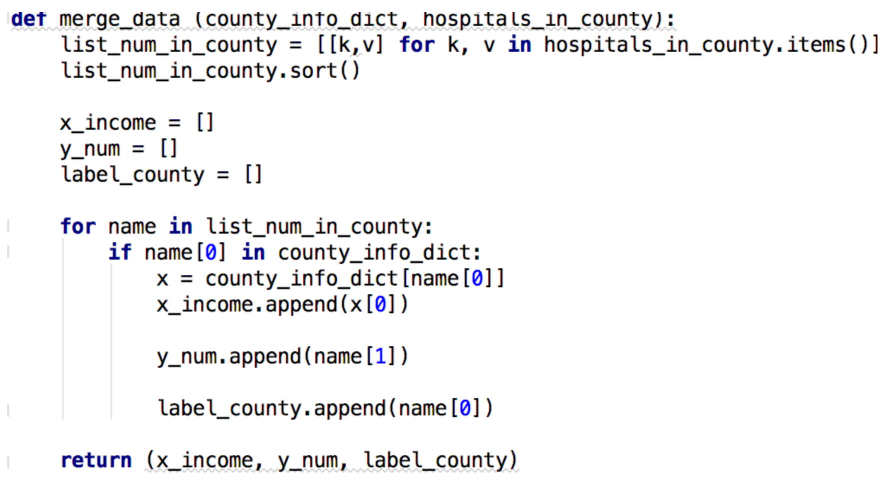
```python
def read_hospital_csv(): # Function to filter a CSV file for the hospitals wanted, no paramaters need.
    desired_state = "MI" # Declare the state initials you want to filter hospitals by.

    with open("Hospital_General_Information.csv", "r", newline='') as input_file: # Open original CSV fi
        hospital_data_reader = csv.DictReader(input_file, delimiter=',', quotechar= '"')

        with open("Filtered_Hospital_Information.csv", "w", newline='') as output_file: # Create new CSV
            hospital_data_writer = csv.DictWriter(output_file,
                                                  fieldnames = ["Hospital Name", "State", "County Name"]
                                                  extrasaction = "ignore",
                                                  delimiter = ',',
                                                  quotechar = '"')
            hospital_data_writer.writeheader()

            hospitals_in_county = {} # Create dictionary told hold the number of hospitals in a county.

            for row in hospital_data_reader: # Iterate through original CSV file

                if row["State"] == desired_state: # By desired_state, place the relevant hospital inform
                    row["Hospital Name"] = row["Hospital Name"]
                    row["County Name"] = row["County Name"].lower() # Lowercase county name to match for
                    hospital_data_writer.writerow(row)

                    if row["County Name"] in hospitals_in_county: # Add to the previous dictionary to co
                        hospitals_in_county[row["County Name"]] += 1
                    else:
                        hospitals_in_county[row["County Name"]] = 1

    return hospitals_in_county # Return number of hospitals per county.
```

- Step 3: Function population_scrape
    - Purpose: Get a Michigan county's per capita income and population.
    - Input/Output:
        - Takes no input.
        - Uses the link to the webpage as the base_url.
        - Outputs a dictionary county_info_dict which has the county name as a key and the county's per capita income and county population as values.
    - Details:
        - Uses BeautifulSoup and urllib.requets.
        - Finds the class "wikitable" to identify the tables on the page.
        - Iterates through the table, finding a row, "tr".
        - Iterates through the row, finding a cell, "td".
        - Collects the county name (lowercase), the county's per capita income, and the county's population.
        - Adds the values to the dictionary county_info_dict.
        - Returns that dictionary.



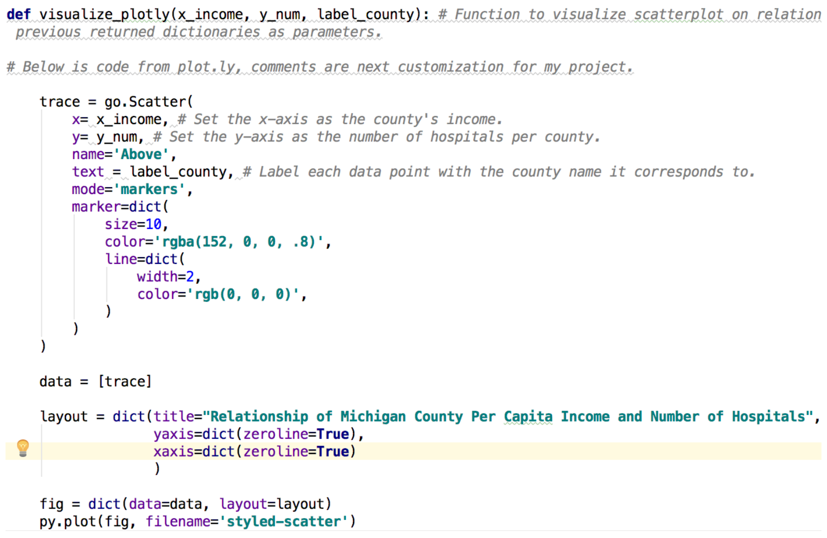
```python
def population_scrape(): # Function to scrape Wikipedia for county information, no paramteres.
    base_url = "https://en.wikipedia.org/wiki/List_of_Michigan_locations_by_per_capita_income#cite_note-2"

    with urlopen(base_url) as f: # Run through BeautifulSoup.
        soup = BeautifulSoup(f, "html.parser")

    wiki_rows = soup.find_all(class_ = "wikitable") # Identify the table on the page holding the needed da

    county_info_dict = {} # Create a dicitonary to hold the data for each county.

    for row in wiki_rows:

        trs = row.find_all("tr")
        for row2 in trs:
            tds = row2.find_all("td")
            for row3 in tds:
                wiki_county = tds[1].text.lower()
                county_income = tds[2].text
                county_pop = tds[5].text

                if wiki_county in county_info_dict:
                    continue
                else:
                    county_info_dict[wiki_county] = county_income, county_pop # Add to the dictionary each
 population respectively.
    return county_info_dict # Return information on each county.
```

- Step 4: Function merge_data
  - Purpose: Arrange the data to be read by plot.ly visualization.
  - Input/Output:
    - Takes county_info_dict and hospitals_in_county dictionaries as inputs respectively from population_scrape and read_hospital_csv functions.
    - Outputs three lists – x_income, y_num, and label_county – as three items in a tuple. These will be the data for the x axis, y axis, and label respectively. The value in each index corresponds to the values in the other two lists.
  - Details:
    - Convert the dictionary with the number of hospital per county to a list using .items() for a dictionary.
    - Sort the list alphabetically.
    - Create the three lists.
    - Access the county name in the new list (name[0]).
    - If that county name appears in the dictionary with the county information, append the values to their respective lists in that iteration loop.
    - To test for per capita income, change line 76 to x[0]; for population, change to x[1].
    - Return the three lists as values in a tuple.



```python
def merge_data (county_info_dict, hospitals_in_county):
    list_num_in_county = [[k,v] for k, v in hospitals_in_county.items()]
    list_num_in_county.sort()

    x_income = []
    y_num = []
    label_county = []

    for name in list_num_in_county:
        if name[0] in county_info_dict:
            x = county_info_dict[name[0]]
            x_income.append(x[0])

            y_num.append(name[1])

            label_county.append(name[0])

    return (x_income, y_num, label_county)
```

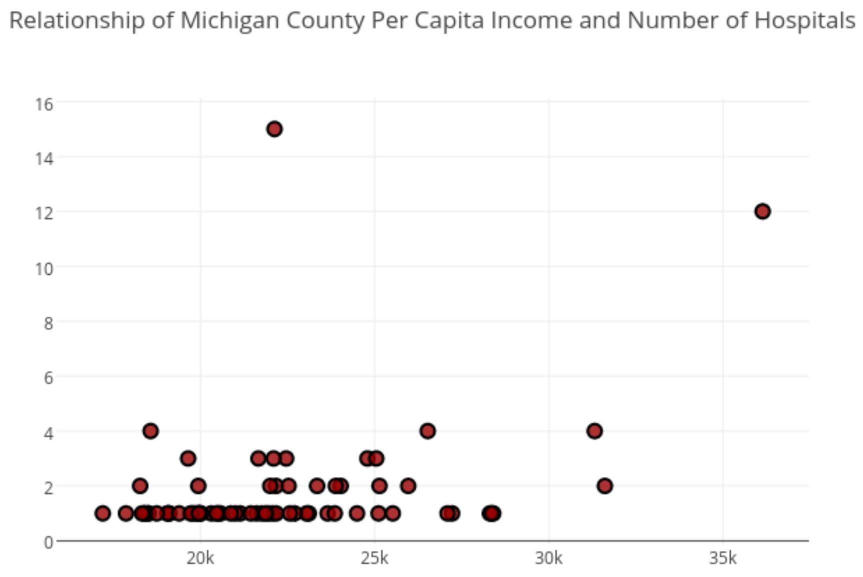
- Step 5: Function visualize_plotly
  - Purpose: Use plot.ly code to visualize the data as a scatterplot.
  - Input/Output:
    - Takes the three previous lists (as values in a tuple) as inputs from the merge_data function.
    - Output directs to the plot.ly website which shows a visualization of the graphs.
  - Details:
    - X (x_income) = either the county's per capita income or population, depending on whether one chooses to append the income or population in Step 4 to the list.
    - Y = the number of hospitals in that county.
    - "text" was added so I could apply the county names as labels to each point.
    - An option for a second style points was removed.
    - The title was added as the "Relationship of Michigan County's Income and Number of Hospitals". It was changed to "Population" when population was used as the x-axis.



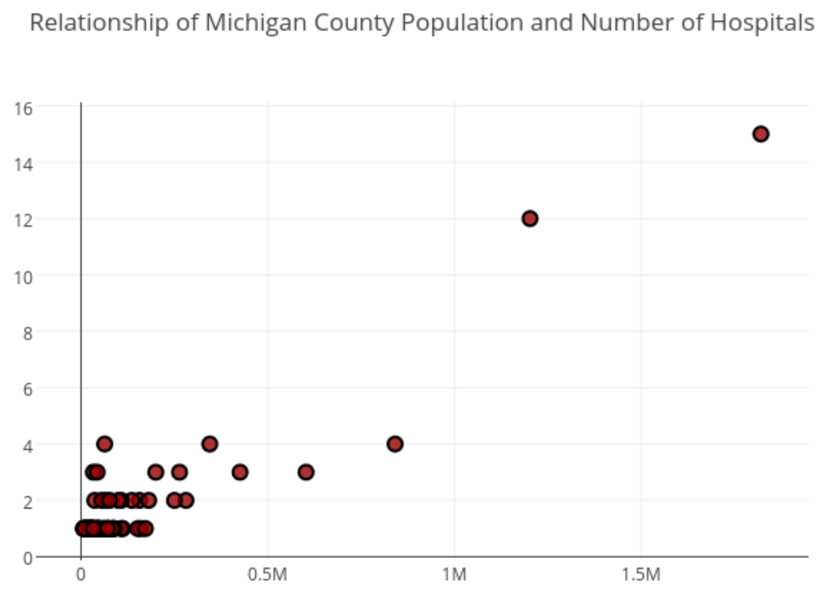
```
def visualize_plotly(x_income, y_num, label_county): # Function to visualize scatterplot on relation
 previous returned dictionaries as parameters.

# Below is code from plot.ly, comments are next customization for my project.

    trace = go.Scatter(
        x= x_income, # Set the x-axis as the county's income.
        y= y_num, # Set the y-axis as the number of hospitals per county.
        name='Above',
        text = label_county, # Label each data point with the county name it corresponds to.
        mode='markers',
        marker=dict(
            size=10,
            color='rgba(152, 0, 0, .8)',
            line=dict(
                width=2,
                color='rgb(0, 0, 0)',
            )
        )
    )

    data = [trace]

    layout = dict(title="Relationship of Michigan County Per Capita Income and Number of Hospitals",
                  yaxis=dict(zeroline=True),
                  xaxis=dict(zeroline=True)
                  )

    fig = dict(data=data, layout=layout)
    py.plot(fig, filename='styled-scatter')
```

Results

I began assuming a county's income would be a greater indication to the number of hospitals. For the results, the county with the greatest number of hospitals was Wayne County (where Detroit is) with 15 hospitals and a per capita income of $22,125. The county with the highest per capita income was Oakland County with 12 hospitals. This is not reinventing the wheel – clearly, the county with more money can afford more medical care. I assumed Wayne County and Detroit to have more hospitals because it is a lower SES area that receives a lot of attention because of its significance. Furthermore, it is a historical area so more sites of medical care would have built up over the year. Otherwise, the data is consisted but one could see more hospitals as the per capita income of the counties get higher. I'm sure I would see a stronger trend if I ran the data nationally.



When I tested for population instead of per capita income, I received results which were more reasonable. Wayne County now had the most hospitals and the most people, with Oakland coming in second with three less hospitals and 618,222 less people. This trend is stronger and more consistent. It's assuring that higher populated areas at least have access to more hospital options.



Discussion and Conclusion
I chose this project because I wanted to introduce myself to healthcare data. This project introduced me to data sites such as data.medicare.gov and to be critical of the statistics and values I was using; for example, per capita income vs. median household income (per capita income is already normalized for population).

If there's one thing about healthcare data I learned, is that it's private. I could not do many of the projects I wanted to because that data was not available online publicly. If there was any, then it came from journals or in a format that was already worked upon, never raw. However, the data.medicare website has many other specific datasets which would be great to explore, such as malpractice in hospitals.

Within the project, there are many other things I would like to take to the next level. For example, finding a way to account for county name formats. St. Joseph, for example, has two hospitals, but this was not shown because the Wikipedia page spelled the county "St. Joseph" while the CSV file had it as "Saint Joseph". This was one reason why I did not extend my search into the nation, because I was not confident yet on how to account for the many county name overlaps and different spellings. Otherwise, I would love to learn more about how this trend extends nationally. Next, I would wish to record a hospital's address and map it along with the population. A geospatial representation would place the results into a greater context among Michigan and I'd be able to better evaluate whether there is greater advanced healthcare options, especially in rural areas. Lastly, I would have liked to make my project more efficient by identifying slower function and improving them. For example, arranging the data into dictionaries and lists beforehand so I won’t have to later and using the data.medicare.gov API so one doesn’t need the csv file to run the program.

In conclusion, I’m glad I chose this project because I was able to gain an introduction to hospital and public healthcare data because I wish to go into healthcare as a career. Furthermore, I learned to be mindful about understanding the significance of these statistics and how they should be formatted on a graph and understood in context. Lastly, I was able to display my technical skills such as iterating through and writing csv files, scraping a webpage using BeautifulSoup, organizing data into different structures, and creating visuals right from my code using plot.ly.