Generating Maps with Location Hotspots and Leveraging Foursquare for Tourism

Justin Zheng 21.8.2019
Applied Data Science Capstone

Table of Contents

Introduction and Business Problem

Background

The purpose of this data project is to leverage the Foursquare location and venue data in order to generate maps that would be able to show hotspots within a specific neighborhood, which in this project will be Rittenhouse Square in Philadelphia, PA. Maps as generated is intended to be printed and handed to tourists for guidance to aid their exploration of the city and in particular Rittenhouse Square. The inspiration behind this project is from personal experience whereby through my work as a tutor at my university I was responsible for helping a number of visiting scholars thrive and enjoy Philadelphia. Due to many of them lacking a functioning data-plan for them to access the internet which prevents them from using any online map service such as Google Maps, my office recommended printing regular maps to help them navigate Philadelphia.

Business Problem

Foursquare data will need to be leveraged to find different important venues around the area of Rittenhouse Square. This project aims to generate relevant maps through data analysis. To generate relevant maps, I will need to filter out important categories of venues, for example a shopping mall or a convenience store. Furthermore, distance and overall radius of location search for venues will need to be individually determined through trial and error in order to generate an appropriately extensive list of venues in each category.

Interest and Relevance

The target audience for this particular data science project will be travel agencies, the hospitality industry such as hotels, motels or other forms of accommodation as well as any tourist-centric venues. Such venues often display and allow visitors to take maps with them to help them navigate in the neighborhood Although physical maps are becoming less relevant due to well-run and up to date online navigational systems like Google Maps, they still play a role especially as tourists may not always have quick access to the internet. Furthermore, the kind of cluster-based maps that I will generate in this project will allow extremely quick and simple navigation to different kinds of venues as the maps are separated by venue category. This will help alleviate confusion and the problem who many run into using apps like Google Maps, which is too much information which makes the map unreadable.

Data

Acquisition and Purpose of Data

A majority of location data was scraped directly from Foursquare. This location data included postal codes, names, longitude and latitude values and exact addresses of different venues within the Philadelphia area. For example, one of the examples I scraped of a shopping mall near Rittenhouse Square was the Liberty Place on 1625 Chestnut Street and is within the 19103 postal code area in which Rittenhouse Square is located. Example of a data-frame:

1	The Shops at Liberty Place	Shopping Mall	1625 Chestnut St	39.951919	-75.167833	19103	PA
2	The Gallery at Market East	Shopping Mall	901 Market St	39.952689	-75.158149	19107	PA

Much of the data scraped from Foursquare was omitted, however I will discuss this more under the Data Cleaning section. The purpose of the data I kept and acquired is to generate a map with clusters, hence columns such as “Neighborhood” was negligible since I had the postcode. Furthermore, the map code I ran only required longitude and latitude values hence they were the priority of my data acquisition.

Longitude and latitude values that were acquired through Foursquare were particularly used to form maps out of clustering. Postal codes acquired through Foursquare were further used to identify specific neighborhoods and to selectively clean data to only show different venues of the same post code as Rittenhouse Square.

Other data was also acquired and added to data-frames manually. The Foursquare location data was not fully accurate, hence I also utilized Google Map data and Apple Maps data to manually fill missing addresses and postal codes. This was mainly done for consistency and accuracy in order to generate the utmost updated maps.

Cleaning

The Foursquare data scraped required significant cleaning. I ran a function that extracts venue categories from the original data scraped from Foursquare then filtered each row. The end result was as follows:

	name	categories	address	cc	city	country	crossStreet	distance	formattedAddress	labeledLatLngs	lat	lng	postalCode	state
0	On-Line Shopping Store	Shoe Store	1509 Walnut St	US	Philadelphia	United States	NaN	80	[1509 Walnut St, Philadelphia, PA 19102, Unite...	[{'label': 'display', 'lat': 39.95205126106208...	39.952051	-75.163867	19102	PA
1	The Shops at Liberty Place	Shopping Mall	1625 Chestnut St	US	Philadelphia	United States	at 16th St	378	[1625 Chestnut St (at 16th St), Philadelphia, ...	[{'label': 'display', 'lat': 39.95191851893288...	39.951919	-75.167833	19103	PA
2	The Gallery at Market East	Shopping Mall	901 Market St	US	Philadelphia	United States	at 9th St	458	[901 Market St (at 9th St), Philadelphia, PA 1...	[{'label': 'display', 'lat': 39.95268865552114...	39.952689	-75.158149	19107	PA
3	Chinatown	Neighborhood	NaN	US	Philadelphia	United States	NaN	659	[Philadelphia, PA 19107, United States]	[{'label': 'display', 'lat': 39.95548773443018...	39.955488	-75.156693	19107	PA
4	Walnut Street	Shopping Mall	NaN	US	Philadelphia	United States	NaN	447	[Philadelphia, PA 19103, United	[{'label': 'display', 'lat': 39.949739550253...	39.949740	-75.167047	19103	PA

Furthermore, many columns in the generated pandas data-frame were removed such as distance from town hall or the "cross-street". Both were negligible as they only served to help me locate the venue, which the postal code was already sufficient in providing. Furthermore, the decision to leave out "cross-street" was for clarification purposes as streets in Philadelphia run through the city hence the identification of a perpendicular street was not relevant in locating venues near Rittenhouse Square.

Also, the column names within the data also needed to be transformed for simplicity purposes. Each column name began with *"location."* hence I ran a simple for loop and split at the dot.

Trial and Error for Venue Category

It is also important to note that after data-cleaning, often the data would only contain one or two venues which is unrealistic for most venues I used in my analysis. For example, when I ran a search for "Cafe" through Foursquare and finished all cleaning, only two venues remained. This is unrealistic, hence I needed to run a search again, however this time under the category of "Coffee Shop". Running this search however required additional data-cleaning as the Foursquare data set which is scraped often contain outright errors such as:

13	Groom Barber Shop	Salon / Barbershop	1324 Locust St	US	Philadelphia	United States	btwn Juniper And Broad	510	[1324 Locust St (btwn Juniper And Broad), Phil...	[{'label': 'display', 'lat': 39.9481439748482,...	39.948144	-75.163728	NaN	19107	PA	4b632f

The data also contained many slight inaccuracies, such as listing a shopping mall which contains coffee shops as a coffee shop. This needed to be cleaned as well.

Therefore, for each different category I was required to run and scrape different category data-sets from Foursquare and through trial and error decide upon which to use for further clustering and cleaning. This was done to maintain realism and accuracy.

Methodology

First, to scrape information out of Foursquare, I established my credentials. Then I was required to import the required libraries. I imported requests for requests, pandas and numpy for data analysis. I also imported Nominatim for latitude and longitude transformation, Image and HTML from IPython for images, json_normalize for changing a json file into a pandas data frame library and folium for plotting.

Secondly, I established coordinates for Philadelphia via geolocator. Then moving on to creating clusters, I had to enter the search query and radius. These parameters were then added to a url with my credential to retrieve data from Foursquare. I would then send the GET request and assign the resulting JSON file to venues then transforming it into a pandas data-frame.

I would then clean the data with a function that extracts the category of the venue and filters each row. I would drop unnecessary columns and fill NaN values with the Google Maps data base.

```
CS_clean_columns = ['name', 'categories'] + [col for col in df_CS.columns if col.startswith('location.')]+ ['id']
df_clean_CS= df_CS.loc[:,CS_clean_columns]

# function that extracts the category of the venue
def get_category_type(row):
    try:
        categories_list5 = row['categories']
    except:
        categories_list5 = row['venue.categories']

    if len(categories_list5) == 0:
        return None
    else:
        return categories_list5[0]['name']

# filter the category for each row
df_clean_CS['categories'] = df_clean_CS.apply(get_category_type, axis=1)
```

This would result in a final data frame, fully cleaned like this:

	name	categories	address	lat	lng	postalCode	state
2	Kimpton Hotel Palomar Philadelphia	Hotel	117 S 17th St	39.950809	-75.168642	19103	PA
3	Club Quarters Hotel in Philadelphia	Hotel	1628 Chestnut St	39.951466	-75.168632	19103	PA
21	The Westin Philadelphia	Hotel	99 S 17th St	39.951937	-75.168411	19103	PA
25	Sonesta Philadelphia Rittenhouse Square	Hotel	1800 Market St	39.952834	-75.170345	19103	PA
29	Embassy Suites by Hilton Philadelphia Center City	Hotel	1776 Benjamin Franklin Pkwy	39.956422	-75.168899	19103	PA

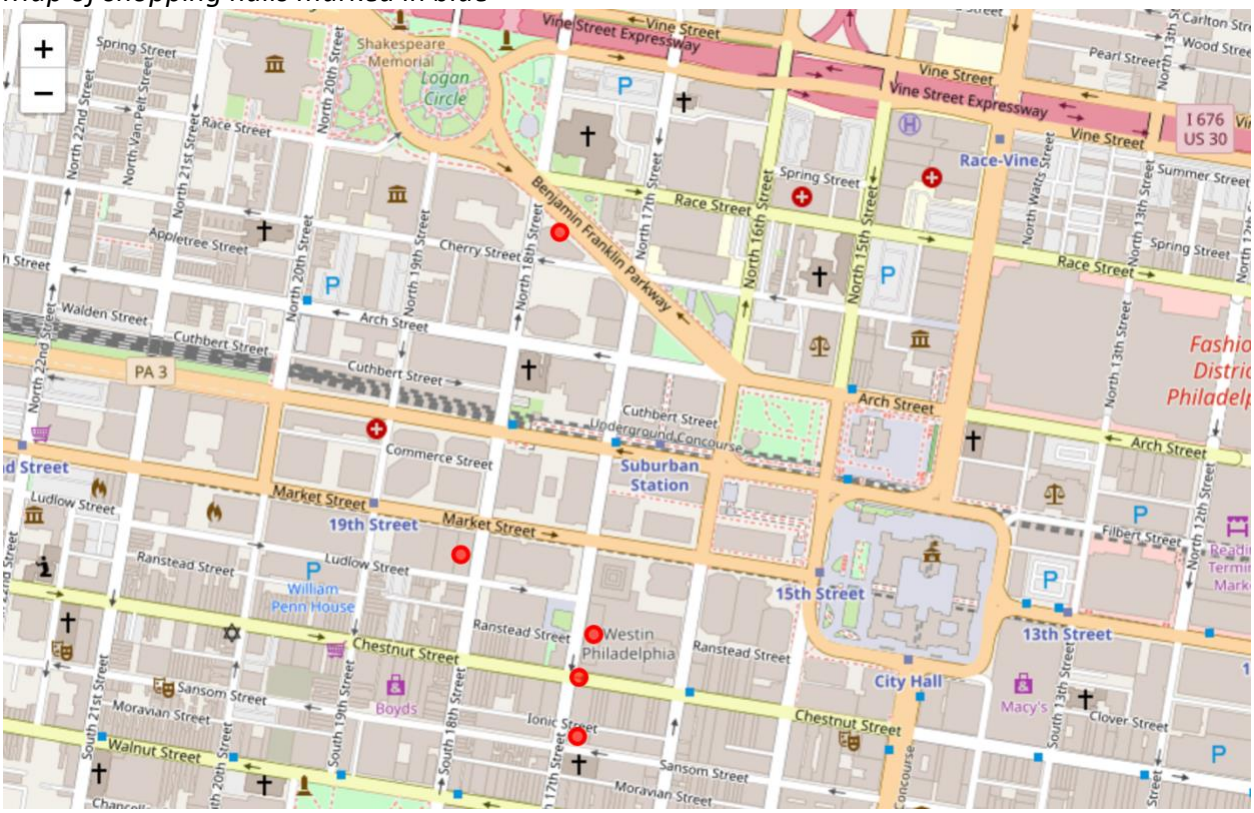
For more exploratory data analysis, I ran searches with different radius amounts ranging from 500, 1000, 1500 and 2000. There were slight inconsistencies with the differences in searches when altering radius, for example, the Shops at Liberty Place was part of the data returned for both the search of “Shopping Malls” and “Coffee Shops”. However, for “Shopping Malls”, it would only be returned if the radius was at 1000, while for “Coffee Shops” it was returned with a radius of 500. Therefore, to avoid such inconsistencies, I ran different searches adjusting radius on every category and the ones shown in the notebook are the ones that returned the most relevant results. 2000 was set at the limit, as beyond that, the searches would return venues with postal codes that were way beyond Rittenhouse Square.

Finally, I combined all remaining and fully cleaned data frames into one larger one that contained all venues that would appear on the maps. I used a simple concat function for that. After that, all I needed to do was use folium and its map function to form a clustered map. The pop-up was set as label and color individually changed depending on the map. Zoom was altered depending on the full radius used in the search. For example for shopping malls, the zoom was set farther due to the larger radius.

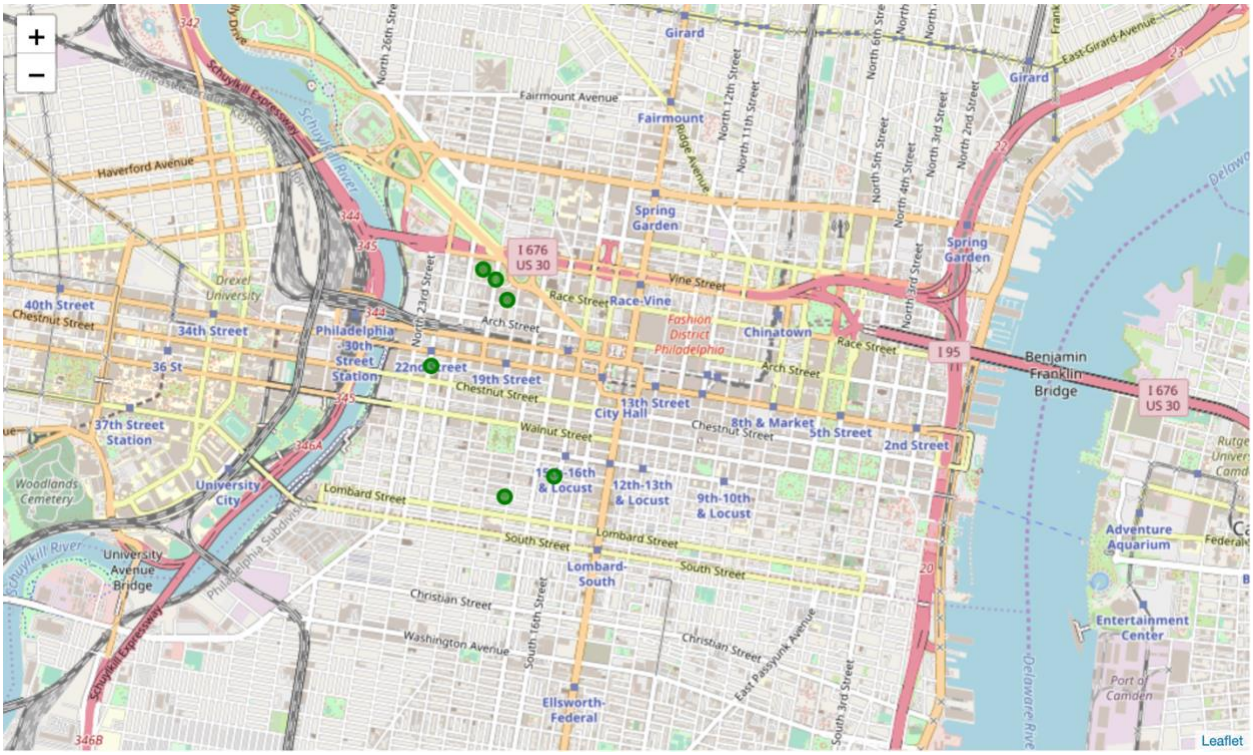
Results

The result was five clustered maps with each containing a different venue category and marked with a different color. I was successful in creating five separate maps with each of their unique venue category marked within Philadelphia and near Rittenhouse Square. The venues were marked by a circle. The following maps are what were generated:

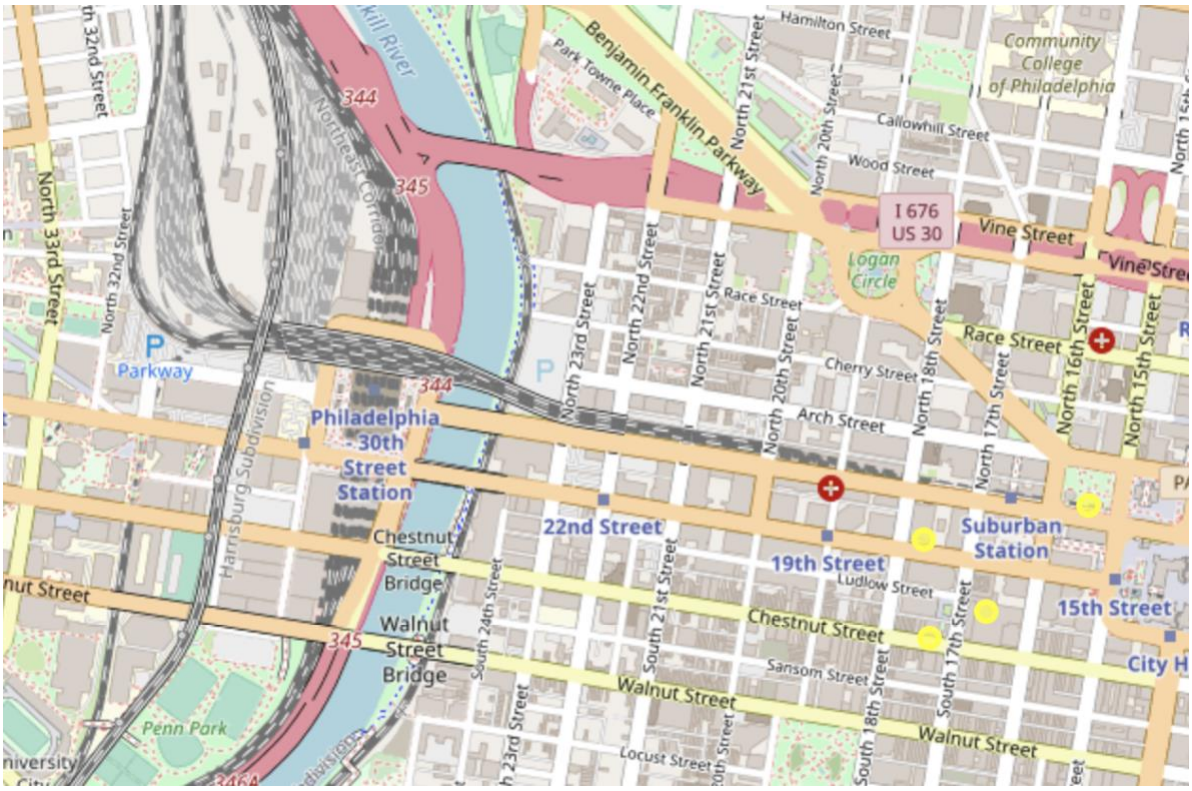
Map of shopping halls marked in blue



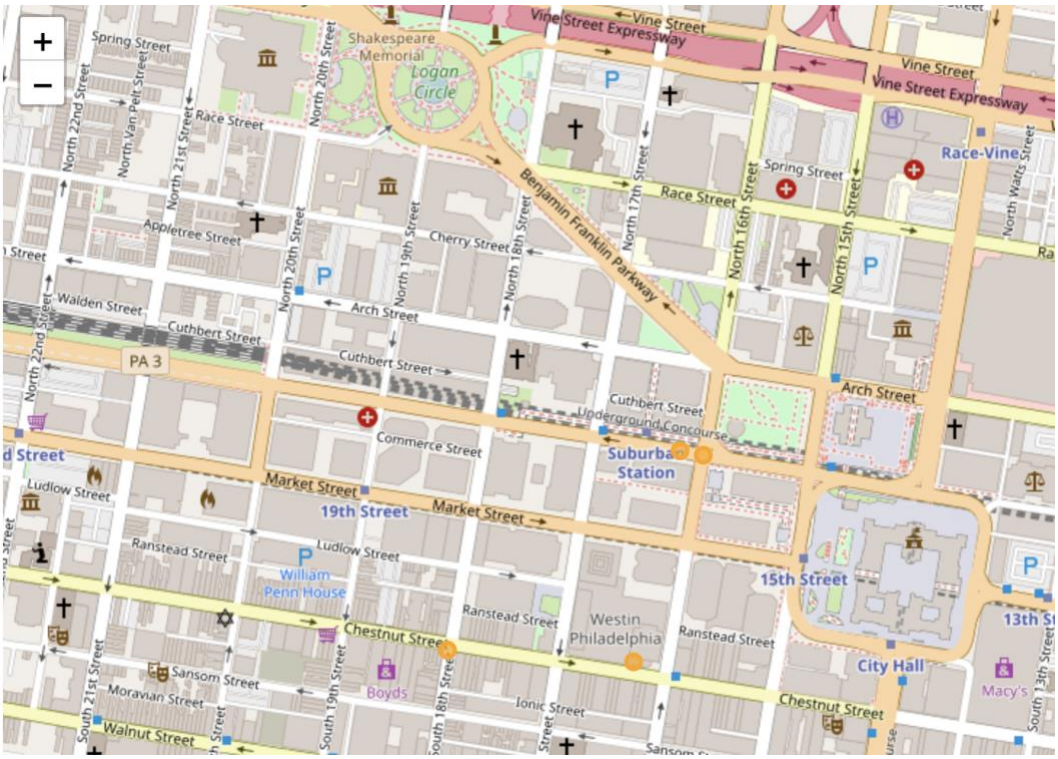
Map of hotels marked in red



Map of museums marked in green



Map of restaurants marked in yellow



Map of coffee shops marked in orange

Discussion

One issue that came up during this data science project was the unpredictability of data bases. Although trial and error is not the most efficient way always to scrape through data, in this case with the Foursquare data base, it was necessary. When it came to listing venues, the Foursquare data base contained many errors hence I needed to send requests for different categories of venues, then after filtering and cleaning, observe whether the data frame made sense or not.

Another observation was how such clusters would look on a map if they were collectively placed. I considered clustering everything into one map, however I quickly realized this was directly contradictory to my objective. My purpose from the beginning was to create simple and readable maps. However, if all the venues were to be clustered together onto one map, it would quickly become unreadable because of the number of marks and color differentiation. Hence, separating maps was still the method to generate the most clear and accurate maps.

For further analysis, one could leverage more databases. I realized in my project with the use of Foursquare and occasionally Google, the venues were still not the utmost accurate and up to date. For future projects, perhaps utilizing more databases and manually checking other databases besides location based such as Yelp or Facebook might help confirm venues.

Furthermore, for future reference, proximity to an area could be measured more accurately. I utilized postal code in this data science project, however there are definitely more accurate means of establishing a perimeter of interest. This could be done by leveraging streets, buildings and perhaps official government distinctions for neighborhoods.

Conclusion

In this project, I leveraged Foursquare location data in order to create five separate and individual cluster maps of different venues near Rittenhouse Square in Philadelphia. The venues were “shopping mall”, “hotel”, “museum”, “food” and “coffee shops”. The purpose of this project was to generate maps that tourists could use to help them navigate in Rittenhouse square. Venue categories were chosen based on what would be significant for travelers. These were based on anecdotal experience.

This data science project involved manipulating parameters for Foursquare location data searches, clustering and mapping using folium, data scraping and cleaning using Python functions as well as my own.