Applied Data Science Capstone Final Project Description

Jim Cole

The data science problem I've chosen for the capstone project is to build a tool that can be used to identify neighborhoods in the two New York City boroughs of Manhattan and Brooklyn that might be attractive to people interested in some set of venues, and also have relatively affordable apartment rents. For instance, one person might want to find an affordable neighborhood that has a high proportion of jazz clubs and also movie theaters, because they play saxophone and love to go to the movies. Another person might want to find a neighborhood with lots of burrito shops, good bakeries, and gay bars.

The number of potential **stakeholders** for such a tool is very large. According to USA Today, approximately 14% of the U.S. population moves every year. The U.S. Census Bureau estimates that about 4.2 million people live in Manhattan and Brooklyn, which means that it's likely that over 500,000 people move to or within those two boroughs each year. It's often said that New York City is the "greatest city in the world," offering virtually anything a person could want, but the question for many people is, "where in New York City are the things I'm looking for, and what are the most affordable neighborhoods with those amenities?" This tool is designed to help answer those two questions for any of the 500,000 people who might ask them each year. This tool is personally interesting to me because I have two family members who are considering moving to New York City, and they would find this tool useful.

To complete this project, I'll use the following data:

- The latitude and longitude of sections of Manhattan and Brooklyn, used to request data specific to those two boroughs from FourSquare.
- The list of FourSquare categories and sub-categories.
- Lists of venues from various categories from FourSquare.
- Reverse geocoding data from Bing to retrieve missing zip codes.
- The zip code(s) in each neighborhood in the two boroughs.
- The population of each zip code in the two boroughs.
- A geojson file with the boundaries of each zip code.
- The median cost of a 1-bedroom apartment rental in each neighborhood.

Detailed Description of Data Usage

Calculation Overview

To estimate how attractive a neighborhood is for a person interested in particular categories of venues, I'll first retrieve venue data from FourSquare using those categoryies' FourSquare identifier. I'll calculate the number of venues in those categories per 10,000 residents of each neighborhood:

$$venue\ density\ =\ \frac{Number\ of\ venues\ in\ the\ neighborhood}{10{,}000\ (residents)}$$

I'll then factor in the cost of housing, because a neighborhood with the same density of a venue, but lower median rent, is more attractive to most people. The attractiveness value will be normalized between 0.0 and 1.0.

$$attractiveness\ =\ \frac{venue\ density}{median\ monthly\ rent\ for\ 1{-}bedroom\ apartment\ /\ \$1{,}000}$$

Data Retrieval and Manipulation

Categories

I'll first retrieve a **hierarchical JSON list of all the venue categories FourSquare supports**. I'll flatten this list to make it easy to look up FourSquare's category ID for any category. For example, the "Arts & Entertainment" category has a "Movie Theater" sub-category, and that in turn has three sub-categories of its own: "Drive-in Theater," "Indie Movie Theater," and "Multiplex." My tool will recursively traverse the data and produce a simple list of category name/ID dictionary pairs. That is, it will turn this data:

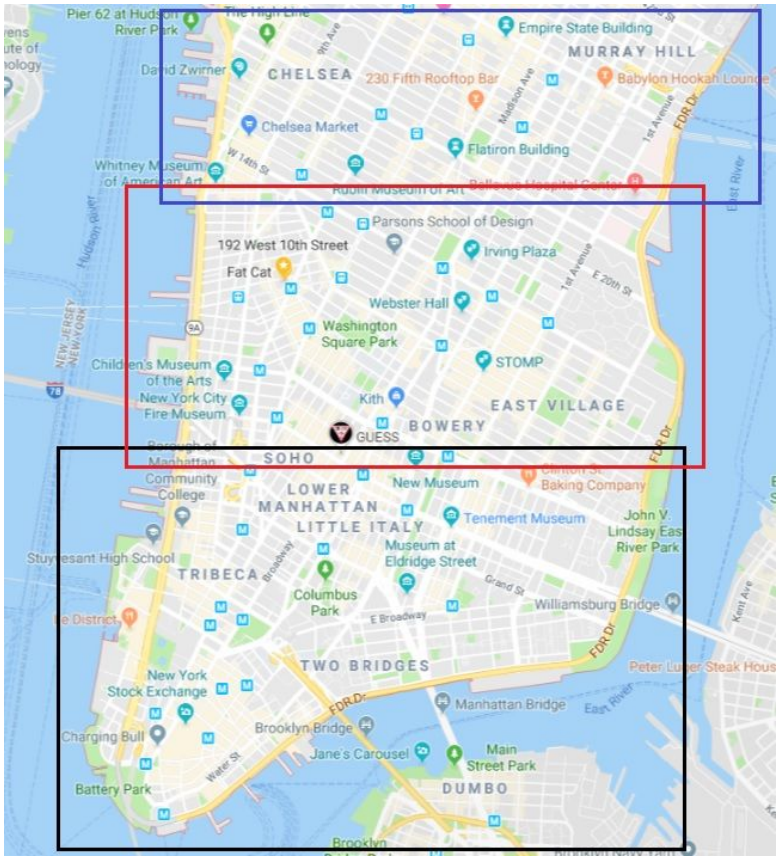
- Arts & Entertainment: 87654
    - Movie Theater: 12345
        - Drive-in Theater: 56781
        - Indie Movie Theater: 90210
        - Multiplex: 44455

Into this data:

- Arts & Entertainment: 87654
- Movie Theater: 12345
- Drive-in Theater: 56781
- Indie Movie Theater: 90210
- Multiplex: 44455

Venues

To retrieve venue data for the two boroughs, I'll use the Southwest (SW) and Northeast (NE) FourSquare search parameters; these two points define a geographic rectangle that FourSquare will return results for. To find these SW/NE points, I'll divide Manhattan and Brooklyn into a series of slightly-overlapping rectangles, as shown in the picture below, and use Google Maps to find the latitude and longitude of the rectangles' lower-left and upper-right corners. I'll create a **CSV file containing these SW/NE pairs**. Because FourSquare will return a maximum of 50 venues per query, the tool will have to slice the rectangles in both directions (latitude and longitude) so that each query represents a very small geographic area. A pandas DataFrame will make it easy to eliminate duplicate entries caused by the overlapping of the rectangles.



One **data cleansing issue** is that FourSquare returns venues that match a specified category, even if that category doesn't represent the main service the venue provides. For instance, FourSquare returns Madison Square Garden for almost any venue category, because that facility is used for so many different things, from basketball games to monster truck rallies to political conventions.

This means that **FourSquare will return an unworkable number of results** for most queries, for two reasons. First, many of the venues in the results won't match what the user is looking for, and second, Folium cannot easily produce cluster maps with thousands of points. If, for instance, you query FourSquare for jazz clubs, it returns literally thousands of results; pretty much every music club of any type, plus restaurants that have music, plus any venue somehow related to music. But there aren't thousands of jazz clubs in Manhattan and Brooklyn, so even if you could show all of those results to the user, that would be misleading, to say the least. To filter out these superfluous results, the tool will examine the first FourSquare category listed for each venue, and will consider that category to be the primary use of, or service provided by, that venue. Thus, Madison Square Garden will show up in the final results if you are looking for venues in the category "Basketball Stadium," but not if you're looking for "Music Venue."

Some of the venues' addresses are returned from FourSquare with a street address, but without a zip code. I'll use **Bing's reverse geocoding capability** to fill in any missing zip codes in the venue data. Manhattan has 12 neighborhoods, and Brooklyn has 18. Using a **CSV file that lists all zip codes** in each neighborhood, I'll use the zip code of each venue returned by FourSquare or Bing to determine which neighborhood a venue is in, and to eliminate venues that are not in any of the two borough's neighborhoods. These irrelevant venues will be returned by FourSquare because both boroughs are irregularly shaped, so some of the rectangles I'll use for lat/long for each borough will include parts of other boroughs. For instance, some of the Brooklyn rectangles in the above picture also include part of another borough, Queens, and Upper Manhattan rectangles include part of the Bronx.

Using **a CSV file that lists the population of, and median rent, for each neighborhood**, I'll calculate the number of venues in each category per 10,000 residents of each neighborhood; i.e., the neighborhood's ***venue density*** for a category, then calculate the neighborhood's ***attractiveness*** value.

I'll use geojson data to draw the boundaries of the neighborhoods on maps. I have not been able to find a geojson file with the boundaries of the neighborhoods, but I have found a **geojson file with the boundaries of zip codes**. I'll **dissolve that geojson file** (combine the polygons of multiple zip codes, eliminate the inner boundaries, leaving just the outer boundary of the combined areas), to create a geopandas DataFrame with the boundaries of the neighborhoods.

I'll also use machine learning, specifically **k-means clustering**, to find clusters of venues, regardless of neighborhood boundaries. I'll print the results of the calculations, and I'll also use the results to create the following maps:

- A **choropleth map of the neighborhoods**, shaded based on their *attractiveness* value, which takes venue density and median rent into account.
- An overlay of neighborhood boundaries on a **heatmap of venues**, making it possible to visually compare the density of the selected venue categories among the neighborhoods.
- A **cluster map of venues,** showing:
    - The location of each venue, color-coded by venue category
    - A circle from the centroid of each cluster, with each circle's radius proportional to the number of venues in that cluster.

  This map enables visual exploration of clusters of venues, regardless of neighborhood boundaries.