Capstone Project - A Restaurant In Boston

Applied Data Science Capstone, IBM/Coursera Data Science Professional Certification

1 Table of contents

2 Introduction

For my Capstone Project I'll be examining the question of where to put a restaurant in Boston, Massachusetts. The city of Boston has a rich culinary history, covering the gamut from New England seafood ("say chowdah!") to the Italian restaurants and bakeries of the North End to Chinatown to the ubiquitous Dunkin' Donuts, based in nearby Canton MA. With roughly 1700 restaurant venues (including 74 Dunkin' Donuts), there's no shortage of options. So where might one squeeze in another venue?

3 Data/Exploration

I didn't initially have any specific agenda in terms of how I'd solve the problem, so I pulled data from a host of different sources, demographics, venues, etc.; mapped it, sliced and diced it, looked for something to jump out, and eventually I found an angle. Before we get to that in the Methodology section, let's take a look at the data sources used…

Venue info

Foursquare EXPLORE endpoint
https://api.foursquare.com/v2/venues/explore

Census Tract and Population info

US Census Bureau geocoder, using benchmark 4 (Public_AR_Current) and vintage 410 (Census2010_Current)
https://geocoding.geo.census.gov/geocoder/

Household Income info

US Census Bureau American Community Survey 5-Year Data (2009-2019) (aka ACS5), report B19013_001E (Estimate Median household income in the past 12 months in 2019 inflation-adjusted dollars)
https://api.census.gov/data/2019/acs/acs5

Boston Census Tracts

Analyze Boston
https://data.boston.gov/dataset/census-2010-tracts1 via hub.arcgis.com

Boston Census Tract to Neighborhood key

Boston Planning & Development Agency; Neighborhood-Map-by-Census-Tract_Updated-March-2014-(1)-(2).pdf
http://www.bostonplans.org/

MBTA Rapid Transit, Commuter Rail, and Bus info

Massachusetts Bureau of Geographic Information (MassGIS)
https://docs.digital.mass.gov/dataset/massgis-data-mbta-rapid-transit via hub.arcgis.com
https://docs.digital.mass.gov/dataset/massgis-data-trains via hub.arcgis.com
https://docs.digital.mass.gov/dataset/massgis-data-mbta-bus-routes-and-stops via hub.arcgis.com

4 Methodology

I started this project with a clean sheet and no preconceived notions about the direction I'd go. I could focus on a particular restaurant type, perhaps look for an under-represented cuisine (although that could raise questions on whether a given category is under-represented and an opportunity, or under-represented because the market has already spoken regarding that type of restaurant). Going a different direction, I could focus on a particular part of town which seemed ripe for growth.

One decision I made early on was that as part of the project, I'll be breaking up the city into neighborhoods as we did in the NYC and Toronto exercises, but with a twist. In those exercises we used a list of points as centroids of city neighborhoods, and defined venues as being "in" those neighborhoods based on being within a certain radius of the centroid. I understand that this was done for the sake of simplicity and keeping us focused on the task at hand, but in reality a neighborhood is not typically a point or defined simply by proximity to a point. It is a bounded area, usually irregularly shaped. For this project I'll be dividing the map of Boston into real neighborhoods and placing the venues pulled from Foursquare into these irregularly shaped areas. The City of Boston Planning and Development Agency maintains all manner of maps and charts and so on regarding the defined neighborhoods of the city. Most critically for our purposes, they also maintain a "decoder ring" of what U.S. Census tracts correspond to what neighborhoods. Combining the city info with the census info will allow us to create our geographic plots.

4.0.1 File Imports

I took the census tract map of Boston from ArcGis and used GeoPandas "dissolve" to remove some of the internal boundaries of the geo data and aggregate the 180 census tracts into the defined neighborhoods, as well as dissolving all internal features to create a single city-sized object, we'll need that later.

4.0.2 Venue Data Work Begins

Regardless of how large you set the "limit" parameter in your API request, Foursquare's EXPLORE endpoint only returns up to the first 100* recommended venues. Fortunately these results are paginated, so you can use the OFFSET variable to re-run the request and get the next 100, and so on.

**Foursquare's documentation says EXPLORE will return up to 50, but it's returning 100, so…*

The SEARCH endpoint will only retrieve 50 results no matter how high you set the limit, and there is no OFFSET parameter. The makeshift solution suggested by the internet is to repeatedly re-run the search with slightly shifted lat/lon coordinates to try to blanket the area, then remove the duplicates. That sounds extra messy, so we'll be sticking with the EXPLORE endpoint. EXPLORE can use a parameter to limit the type of venues returned, either "categoryId=categorynumber" or "section=sectionname". Category number requests automatically include all subcategories of that category number, so using category number 4d4b7105d754a06374d81259 (Food) will capture all the varieties of restaurant nested under that category. Another alternative is to use one of the section names, which are high level categories like food, drinks, shops, arts, outdoors, sights, and trending. We'll just use "section=food".

So we'll use EXPLORE with "section=food" and loop thru while incrementing the "offset=" parameter to fill out our universe. The next question is how many loops? One option would be to simply keep incrementing the offset until the request gave an error message, but that seems rather crude. When you put in an API call with an offset so high that there are no more results, rather than responding with a code 400 Bad Request, Foursquare returns a code 200 message stating that nothing was found and indicating the max number of venues meeting the query request. By running the coordinates of each of our 180 census tracts thru the EXPLORE with an excessively high offset, we can pull a list of the total number of venues within our specified radius of each point.

Looping thru the census tracts and pulling the max venue count tells us that the maximum number of venues in the radius of any of our 180 input locations is 250, so it looks like three loops should cover it. We'll loop thru our data requests to the Foursquare API three times, incrementing the offset at 0, 100, and 200, then combine the results.

…FIRST LOOP RETRIEVAL COMPLETE, 14631 ROWS
…SECOND LOOP RETRIEVAL COMPLETE, 6785 ROWS
…THIRD LOOP RETRIEVAL COMPLETE, 1937 ROWS

Now time to combined the loop results. It's likely that many venues will be within the specified radius of more than one of our 180 census tract coordinates, so after combining need to sort the results and drop duplicates;

before 23353 / after 2355

Twenty-three thousand before, twenty-three hundred after. Yup, there were a few duplicates.

The US Census Bureau's geocoder API will accept a set of coordinates and send a response indicating what census tract those coordinates are in. We'll loop the coordinates of our venues thru the geocoder and assign each to a specific census tract. This particular chunk of code is a bit slow, as it's making twenty-three hundred requests one at a time. The Census Bureau geocoder page will accept bulk upload of CSV files of up to 10000 locations per file, but I decided it was more in the spirit of this assignment to run as much of it within Python as possible, so we'll go for the API. All things being equal, just running the cell and getting a beverage is still probably quicker than downloading the list of coordinates, uploading it to the geocoder webpage, downloading those results, pulling them back into the notebook, and fixing any formatting issues.

…CENSUS TRACT RETRIEVAL COMPLETE, 2348 ROWS

Now that we have our venues and we have census tract data to go with each, we can eliminate entries that were within our specified radius but are not actually in the City of Boston. Every venue location will now be tagged with a GEOID10 number from the Census Bureau. The format of the number is:

SSCCCTTTTT with **S**tate: 2 digits, **C**ounty: 3 digits, **T**ract: 6 digits

So a GEOID10 of 25025020101 = State 25, County 025, Tract 020101 (201.01). Massachusetts is state 25, and Suffolk County is county 025, so if it doesn't start "25025" then it's not in Suffolk County MA.

Next is filtering out the three communities in Suffolk County that aren't Boston; Chelsea (tracts 1601.01, 1602, 1603, 1604, 1605.01, 1605.02, 1606.01, 1606.02), Revere (tracts 1701, 1702, 1703, 1704, 1705.01, 1705.02, 1706.01, 1707.01, 1707.02, 1708), and Winthrop (tracts 1801.01, 1802, 1803.01, 1804, 1805). We'll pass our geocoded venue data thru a few filters, limiting to just those which start with '25025' to dump the entries from the counties of Middlesex, Essex, and Norfolk, then remove entries starting with '25025160', '25025170', and '25025180' to pull out Chelsea, Revere, and Winthrop.

before 2355 / after 1678

This is our final venue count. Next we'll take our cleaned up list of venues and use the associated census tracts to assign the venues to neighborhoods as defined by the Boston Planning & Development Agency.

4.0.3 Population Data Work Begins

Using a couple of different parts of the Census Bureau data resources, we'll pull in population, housing, and income info for each of the census tracts in the city. For various purposes, we'll arrange this into DataFrames on a tract-by-tract basis, a neighborhood-by-neighborhood basis, and a whole-city basis.

In so doing, I found there were a handful of tracts which had significant populations but no venues at all. As we'll see in a subsequent chart, letting these tracts fall to "NaN" (not a number) on the "population per venue" count would be a missed opportunity, so instead for any tract with population greater than 1000 and 0 venues, I assigned a "population per venue" figure equal to the population of the tract.

4.0.4 Looking At The Data

Now we start looking at these piles of data we've accumulated and arranged, getting a sense of the shape of the city and it's neighborhoods. We'll start with a list of the neighborhoods with population, housing units, and total number of venues in each.

Neighborhood	Population	Housing Units	Venue Count
Allston	22,312	7,898	71
Back Bay	16,634	11,339	72
Beacon Hill	9,023	6,013	30
Brighton	52,685	24,014	121
Charlestown	16,439	8,648	38
Dorchester	114,249	45,140	169
Downtown/Chinatown/Leather	15,992	8,033	167
East Boston	40,517	15,857	170
Fenway	33,883	13,503	105
Harbor Islands	535	0	3
Hyde Park	32,317	12,481	45
Jamaica Plain	35,541	15,846	84
Longwood Medical Area	4,861	416	11
Mattapan	22,500	9,112	28
Mission Hill	16,874	6,790	47
North End	8,608	5,795	53
Roslindale	26,368	11,138	51
Roxbury	49,111	19,373	70
South Boston	31,110	16,086	61
South Boston Waterfront	2,564	1,530	90
South End	29,612	16,665	121
West End	5,423	3,261	20
West Roxbury	30,445	13,546	51

No, I don't know where those 535 people on the Harbor Islands are living, given the absence of housing units. The Census Bureau defines a housing unit as "*A house, an apartment, a mobile home or trailer, a group of rooms, or a single room occupied as separate living quarters, or if vacant, intended for occupancy as separate living quarters. Separate living quarters are those in which the occupants live separately from any other individuals in the building and which have direct access from outside the building or through a common hall. For vacant units, the criteria of separateness and direct access are applied to the intended occupants whenever possible.*" So perhaps they sleep on boats… Next we'll crunch some numbers on the categories of venues we pulled from Foursquare.

Venue Categories sorted by frequency:
Pizza Place 191
American Restaurant 107
Café 97
Italian Restaurant 94
Donut Shop 81
…
Eastern European Restaurant 1
Empanada Restaurant 1
Israeli Restaurant 1
German Restaurant 1
Afghan Restaurant 1

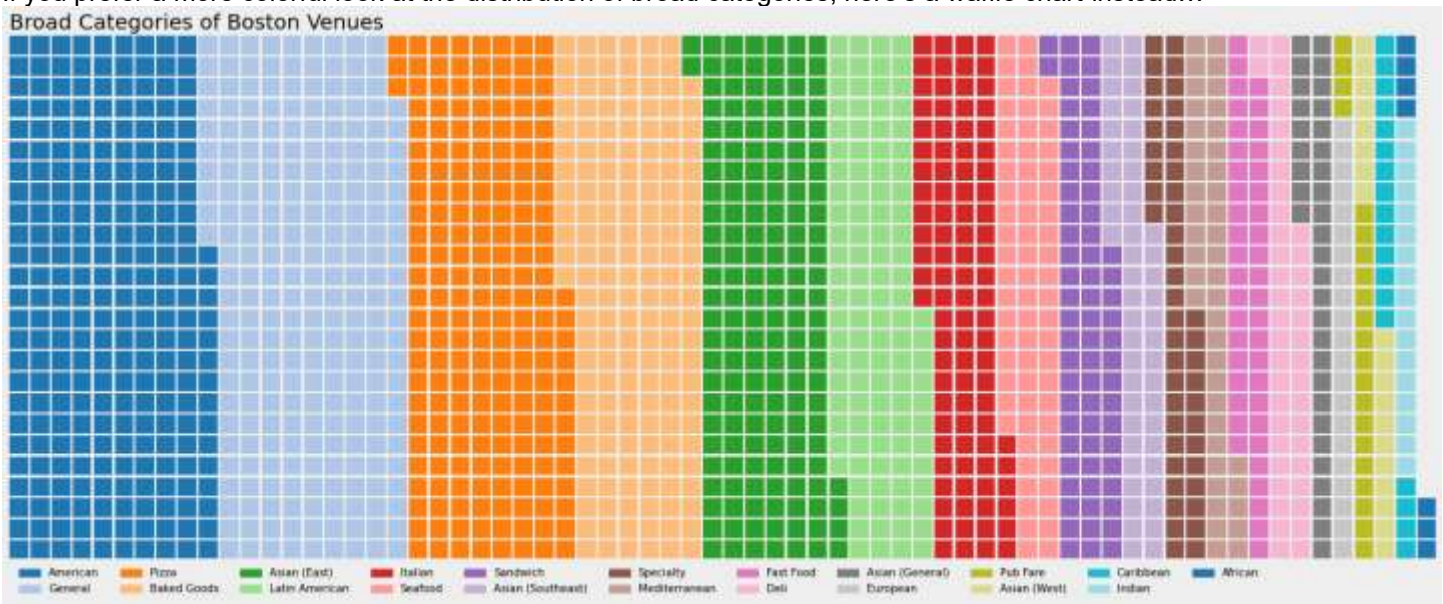
Total Length: 90

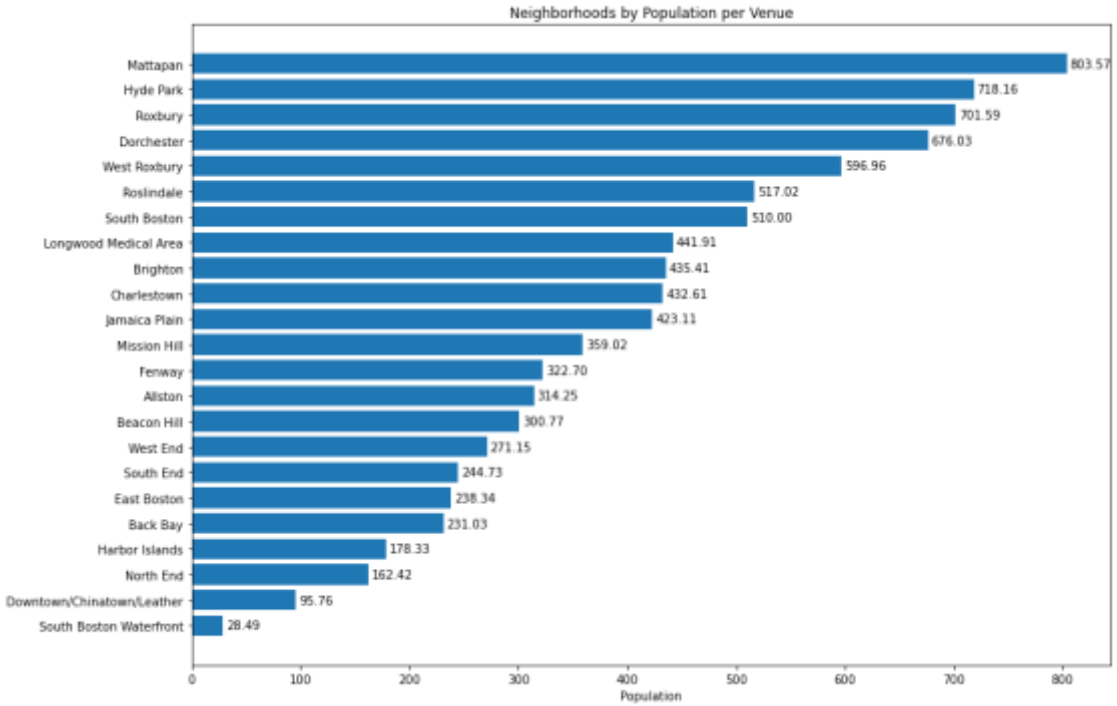
As we see above, Foursquare lists 90 categories for food venues in Boston. To try to flatten that down to something more manageable, I've introduced a higher level categorization called "Broad Category". The broad category groupings I've made are listed in the table below. One could quibble about what should go into what category, but this is my working set. Part of my motivation is that Foursquare allows varying degrees of specificity in categorization, resulting in 34 venues just being categorized as "Asian", 80 more specifically labeled as "Chinese", and then some labeled even more specifically as "Szechuan". Likewise there are "Japanese", then also "Sushi", "Udon", etc. The uneven degrees of specificity / granularity prove bothersome when trying to get accurate aggregate views; broadly speaking, East Asian restaurants are one the most common types in Boston, but split out into a dozen subgroups buries them in any ranking list.

Another issue is that a rather unhelpful number of entries in Foursquare's data have generic categorizations like "Food" or "Restaurant". There are also a decent number of "Café" entries, which traditionally might mean "Coffeehouse", but upon closer inspection the label has been applied to everything from coffee shops to greasy spoon diners. I've lumped all these together in as "General".

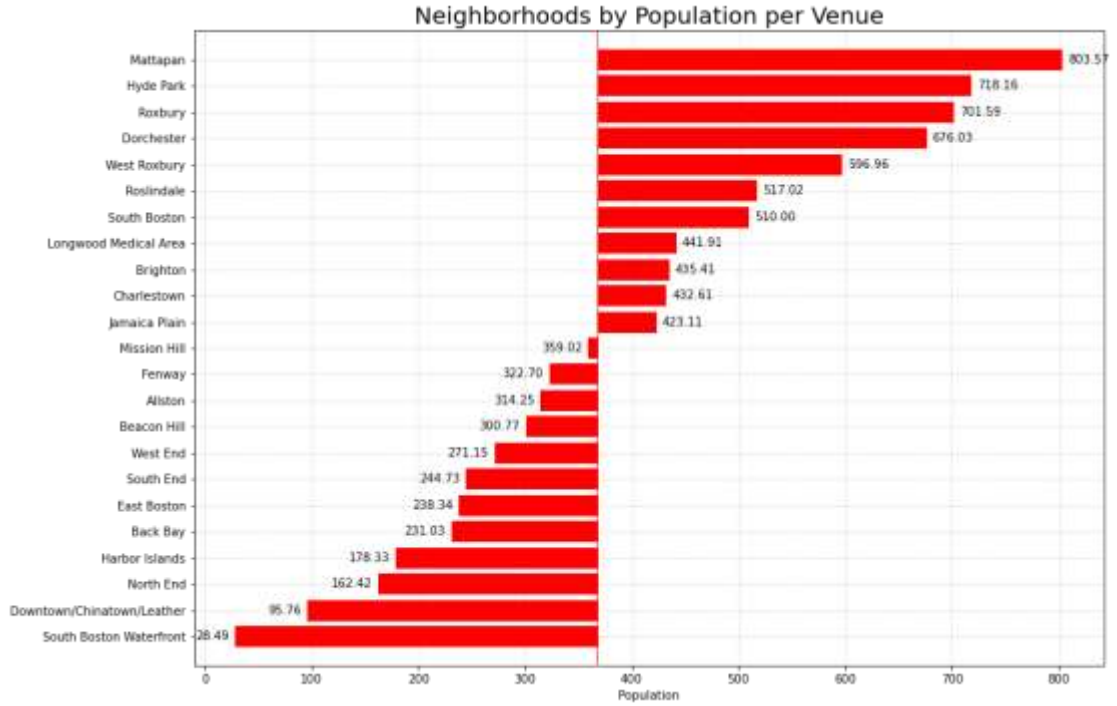
Broad Category	Venues	Venue Category
African	7	African Restaurant, Ethiopian Restaurant
American	240	American Restaurant, Steakhouse, New American Restaurant, Burger Joint, Southern / Soul Food Restaurant, Diner, BBQ Joint, Fried Chicken Joint, Comfort Food Restaurant, Hot Dog Joint, Wings Joint, Cajun / Creole Restaurant
Asian (East)	156	Ramen Restaurant, Sushi Restaurant, Japanese Restaurant, Hotpot Restaurant, Chinese Restaurant, Dim Sum Restaurant, Korean Restaurant, Noodle House, Szechuan Restaurant, Udon Restaurant, Dumpling Restaurant
Asian (General)	34	Asian Restaurant
Asian (Southeast)	51	Vietnamese Restaurant, Thai Restaurant, Malay Restaurant, Burmese Restaurant, Cambodian Restaurant
Asian (West)	19	Falafel Restaurant, Middle Eastern Restaurant, Afghan Restaurant, Israeli Restaurant, Turkish Restaurant, Halal Restaurant
Baked Goods	160	Bakery, Bagel Shop, Donut Shop, Creperie
Caribbean	18	Caribbean Restaurant, Cuban Restaurant
Deli	43	Deli / Bodega
European	21	French Restaurant, Eastern European Restaurant, Belgian Restaurant, Russian Restaurant, Polish Restaurant
Fast Food	43	Fast Food Restaurant
General	232	Food Truck, Café, Restaurant, Bistro, Food, Snack Place, Food Court, Cafeteria, Buffet
Indian	17	Indian Restaurant, North Indian Restaurant
Italian	94	Italian Restaurant
Latin American	108	Brazilian Restaurant, Mexican Restaurant, South American Restaurant, Burrito Place, Taco Place, Latin American Restaurant, Arepa Restaurant, Peruvian Restaurant, Empanada Restaurant, Colombian Restaurant
Mediterranean	43	Mediterranean Restaurant, Greek Restaurant, Tapas Restaurant, Moroccan Restaurant, Spanish Restaurant, Paella Restaurant, Portuguese Restaurant
Pizza	191	Pizza Place
Pub Fare	21	Gastropub, Irish Pub, Fish & Chips Shop
Sandwich	67	Sandwich Place
Seafood	67	Seafood Restaurant
Specialty	46	Salad Place, Breakfast Spot, Vegetarian / Vegan Restaurant, Mac & Cheese Joint, Australian Restaurant

If you prefer a more colorful look at the distribution of broad categories, here's a waffle chart instead…





This is the where the metaphorical lightbulb went on for me on how to approach this “where to put a restaurant” problem. As you’ll see in subsequent maps, the Census Bureau data supplied land area and population for each tract, from which we can calculate population density. Add in the Foursquare data and we can also calculate venue density. Looking at areas of the city with higher population density and lower venue density got me thinking of how to tie those two stats together, and what I came up with was Population per Venue, the population of a tract or neighborhood divided by the number venues in a tract or neighborhood. Where in the city are there way more people than venues? I took the chart above and re-worked it to show the bars as being above and below the city-wide mean value:



The red dividing line in the middle represents the city-wide average for people per venue, 370.04. Those in the bottom half have less people per venue, those in the top half have more people per venue. One thing which stands out immediately is that the neighborhoods in the southwest of the city (Mattapan, Hyde Park, Roxbury, Dorchester, West Roxbury) have by far the highest Population per Venue numbers, or to put it another way, far fewer venues per capita, meaning this is the area of the city is under-served in terms of restaurants and should be where we focus for placing a new venue. But first, some maps…

4.0.5 Maps

It might be useful to select a location that isn't already surrounded by existing venues, so we'll use GeoPandas to add a buffer around every venue in Boston. Adding circles to a Folium map is easy, however there are a couple of drawbacks to going down that path. First, it just looks messy; the cumulative opacity of overlapping circles in areas with high venue density creates areas that are completely dark and blotted out. Second, it results only in viewable decoration on a map, the buffer zone that results isn't a separate geo-object that you can perform operations with (such as programatically checking if coordinates fall within the shadow of the buffer zone, etc). Thus rather than just using Folium to project this buffer zone round venues, we'll create an object representing the buffer zone. Before that, an aside about Coordinate Reference Systems (CRS)…

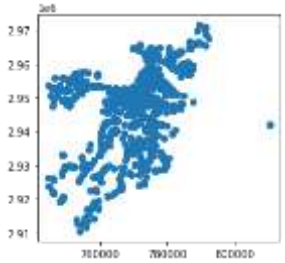
Although Python and Folium do a decent job of not burying you in detail when it's not needed, some geodata transformations do require knowing the EPSG number of the CRS your geodata is projected in. EPSG stands for European Petroleum Survey Group, which no longer exists, but their survey data has evolved into a public repository of spatial reference systems, geodetic datums, etc. EPSG numbers have become the de facto industry standard for labeling coordinate reference systems, used by most GIS systems and libraries, including Folium and GeoPandas.

A common standard CRS is EPSG:4326, which uses WGS1984 for both datum and ellipsoid. WGS means it's the World Geodetic System, a coordinate standard used in cartography and navigation which is maintained by the US National Geospatial-Intelligence Agency (NGA). The latest rev is WGS84, first published in 1984 and last revised in 2014. WGS84 is the reference coordinate system used by the Global Positioning System (GPS) and is also the base used by Folium as a default. Unfortunately WGS84 has one drawback when it comes to our buffer zone plans, which I'll get to in a moment.

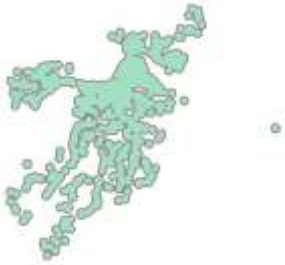
The GeoPandas geometry.buffer method to create buffers around locations uses a single float64 as input, with no means to specify what unit of measure that number refers to. Instead it uses whatever unit of measure is specified by the CRS which the geodata is projected in. Where this becomes problematic is that EPSG:4326 uses degrees as a unit of measure. Degrees of latitude are fairly consistent, 1° equals approximately 69.1 miles. Longitude is another story, 1° of longitude ranges from 69.1 miles at the equator to 0 miles at the poles. At Boston's coordinates, 1° of latitude and longitude is approximately 69 miles wide by 51 miles tall. Given that geometry.buffer only takes one number as the distance parameter, any buffer made in Boston in the WGS84 CRS will be oval shaped. And given the scale of 1°, getting a reasonable buffer of 1000 feet would also be… some really tiny number I'm not going to bother figuring out.

GeoPandas supports re-projecting from one CRS to another, so for the purposes of adding the buffer to each point we'll switch to a different CRS. EPSG:2249 uses North American Datum 1983 (NAD83) and ellipsoid GRS 1980. NAD83 is from the US Defense Mapping Agency and uses one survey foot as the standard of measure, which makes getting a 1000 foot buffer from geometry.buffer simply a matter of putting in '1000' as the distance. It isn't necessary to re-project it back from EPSG:2249 to EPSG:4326, as Folium will do that for us when reading the data.

The first step is taking a copy of the 'venue_master' dataframe and stripping it to just a few fields, creating a geodataframe from it in CRS EPSG:4326, and converting it to EPSG:2249. This gives us a plot of all the venue locations in a CRS that uses feet as a unit of measure:

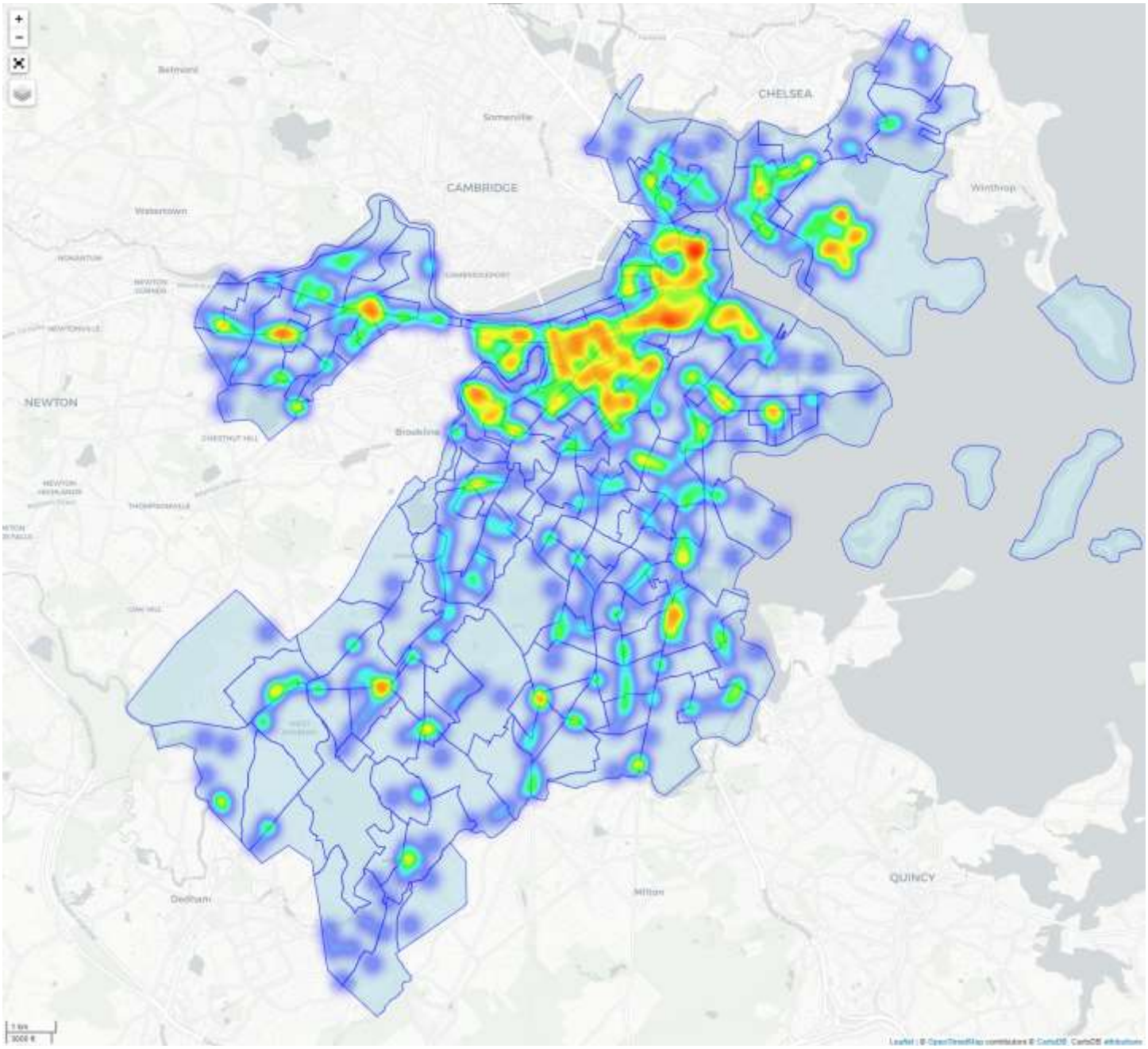


The next step is creating the buffer zone around existing venues using geometry.buffer, then using geometry.unary_union to merge these ~1700 circles into a single multipolygon object. We'll make it projected into a few different CRS in case we need them down the line.



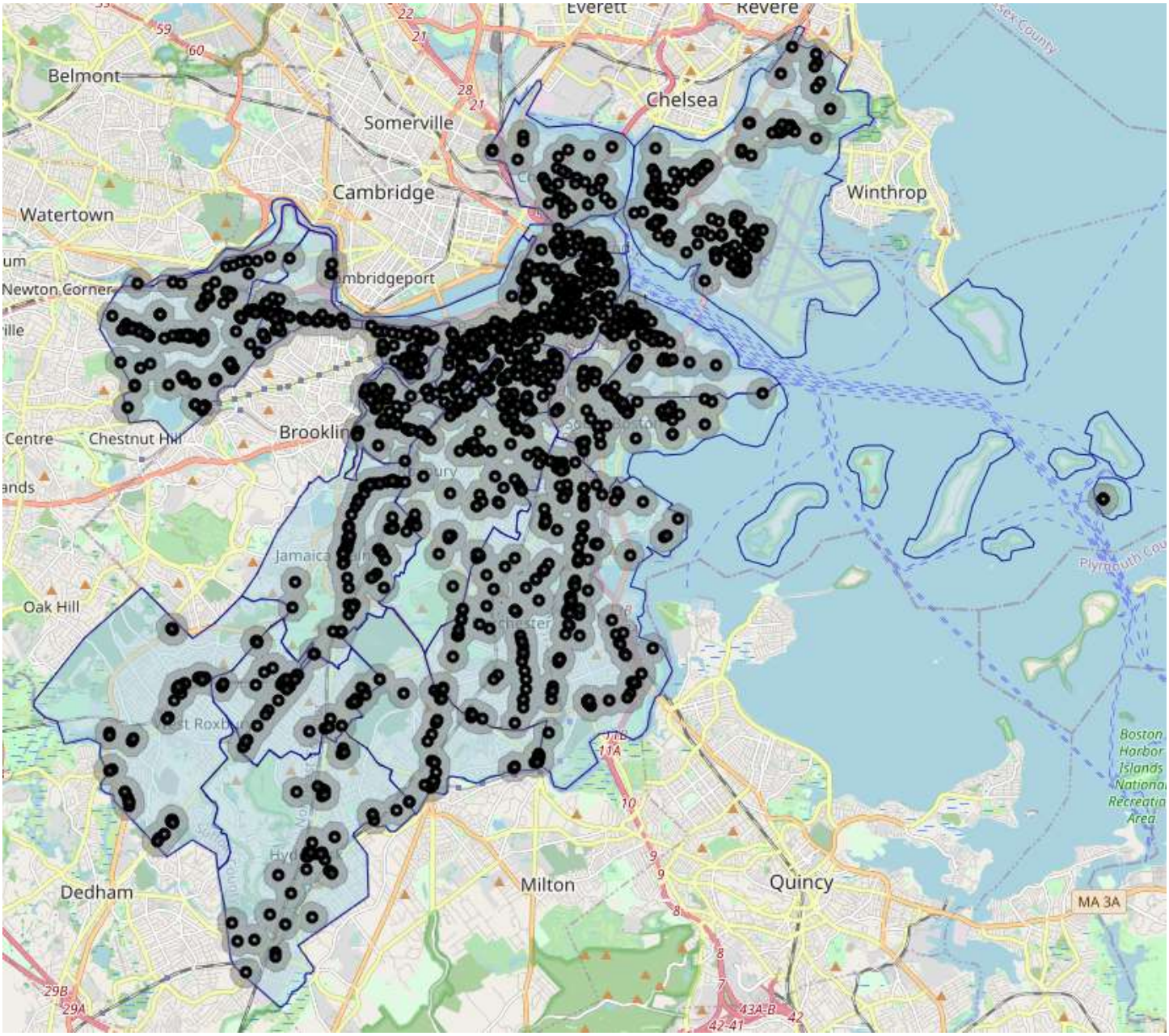
Then we take the new multipolygon we've created, and convert it back to DataFrame then GeoDataFrame, and we're ready to use it in a map layer. Next we'll take these polygons and all the various bits of venue data, census data, buffer data, etc and create the various map layers and feature groups that will be applied to subsequent maps.

First up is the popular "heat map", which represents density of venues via hotter colors:

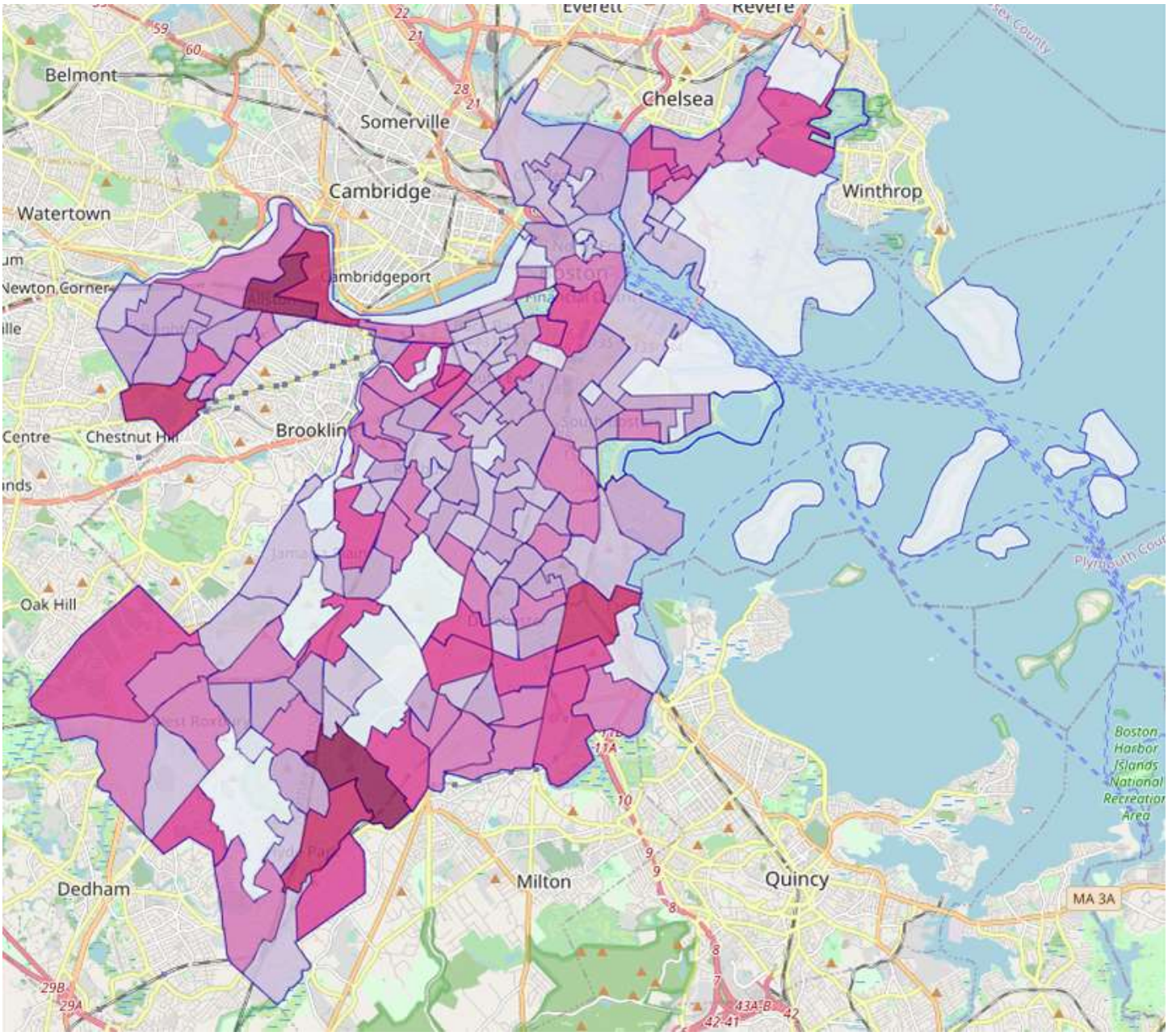


Below is a series of maps which cover all the various bits of data we've collected thus far; The location of each dining venue in Boston and 1000 foot buffers around each venue, neighborhood and census tract boundaries, T lines and stops, Commuter Rail lines and stops, bus lines, a variety of demographic stats; population, population density, venue count, venue density, population per venue, median household income, and population per housing unit.

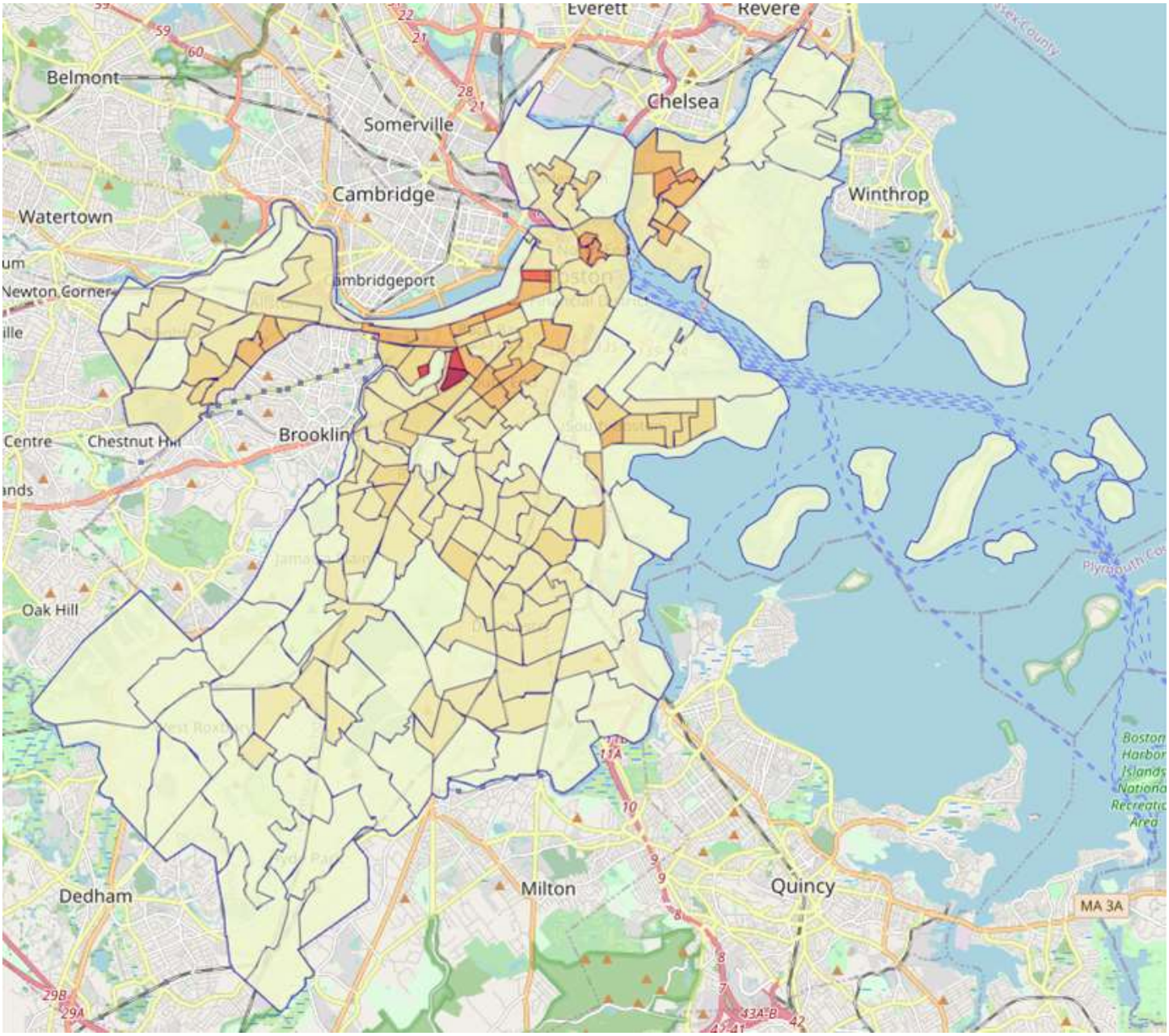
Venues with buffer zones



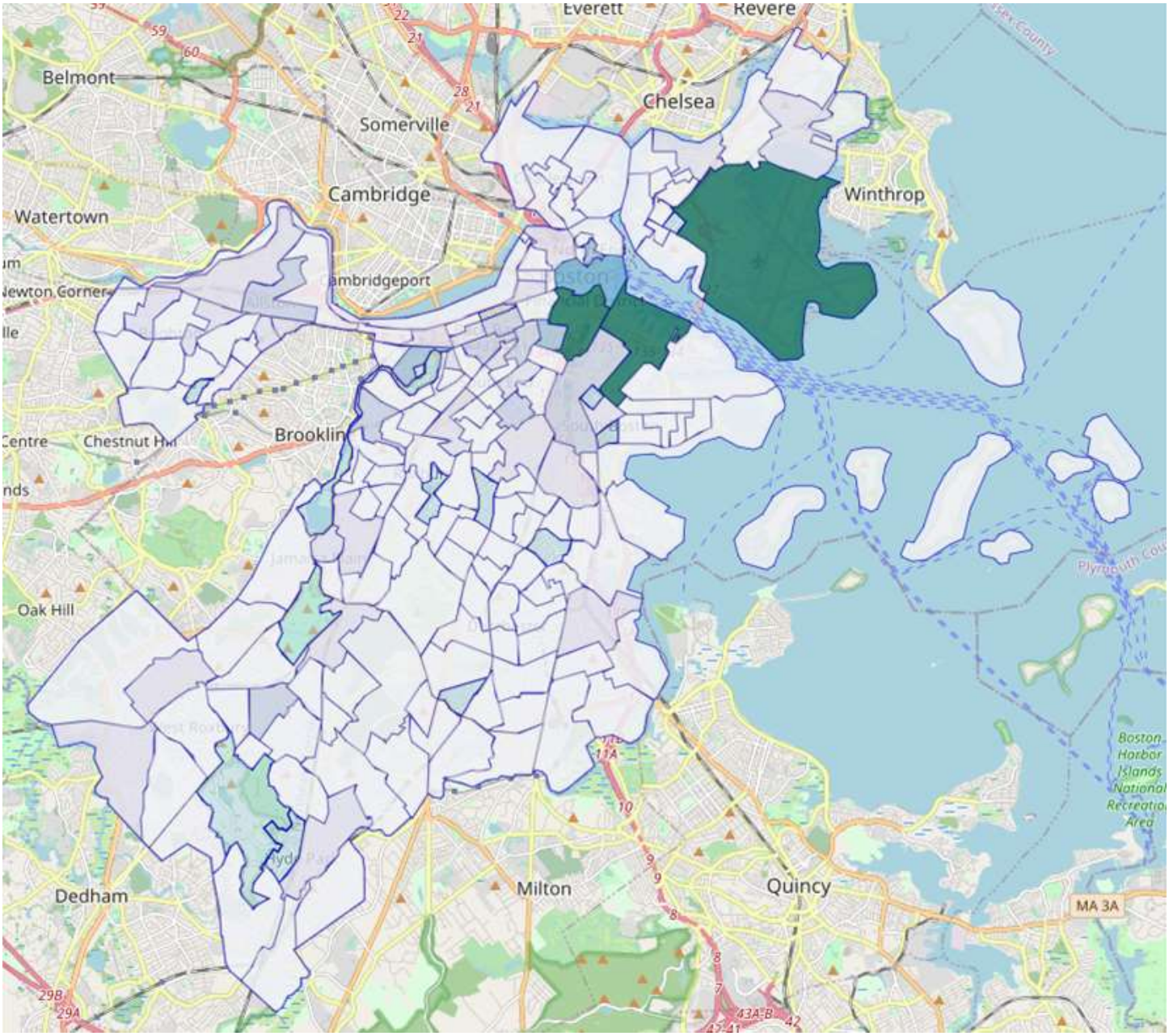
Population (darker is higher)



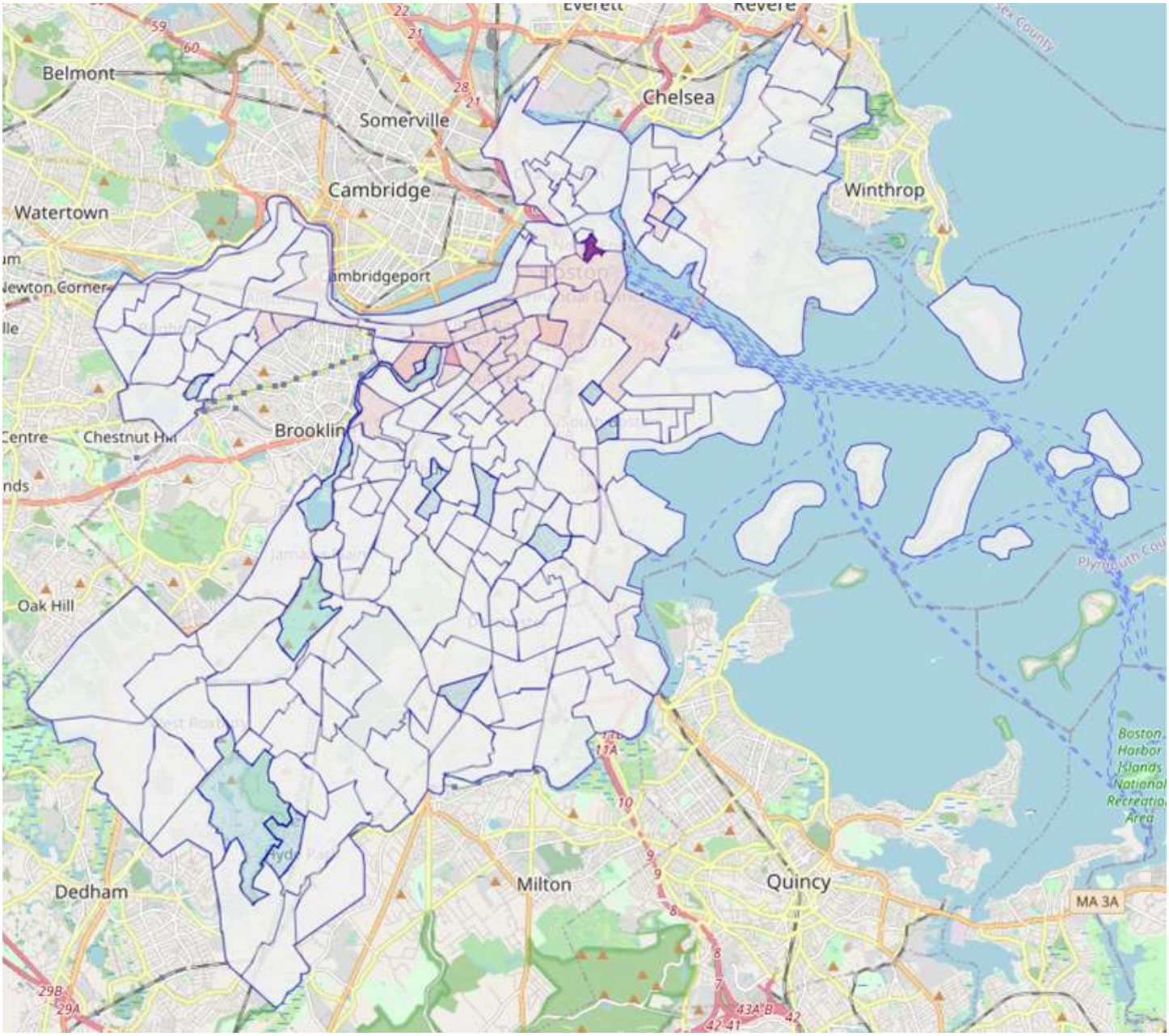
Population density (darker is higher)



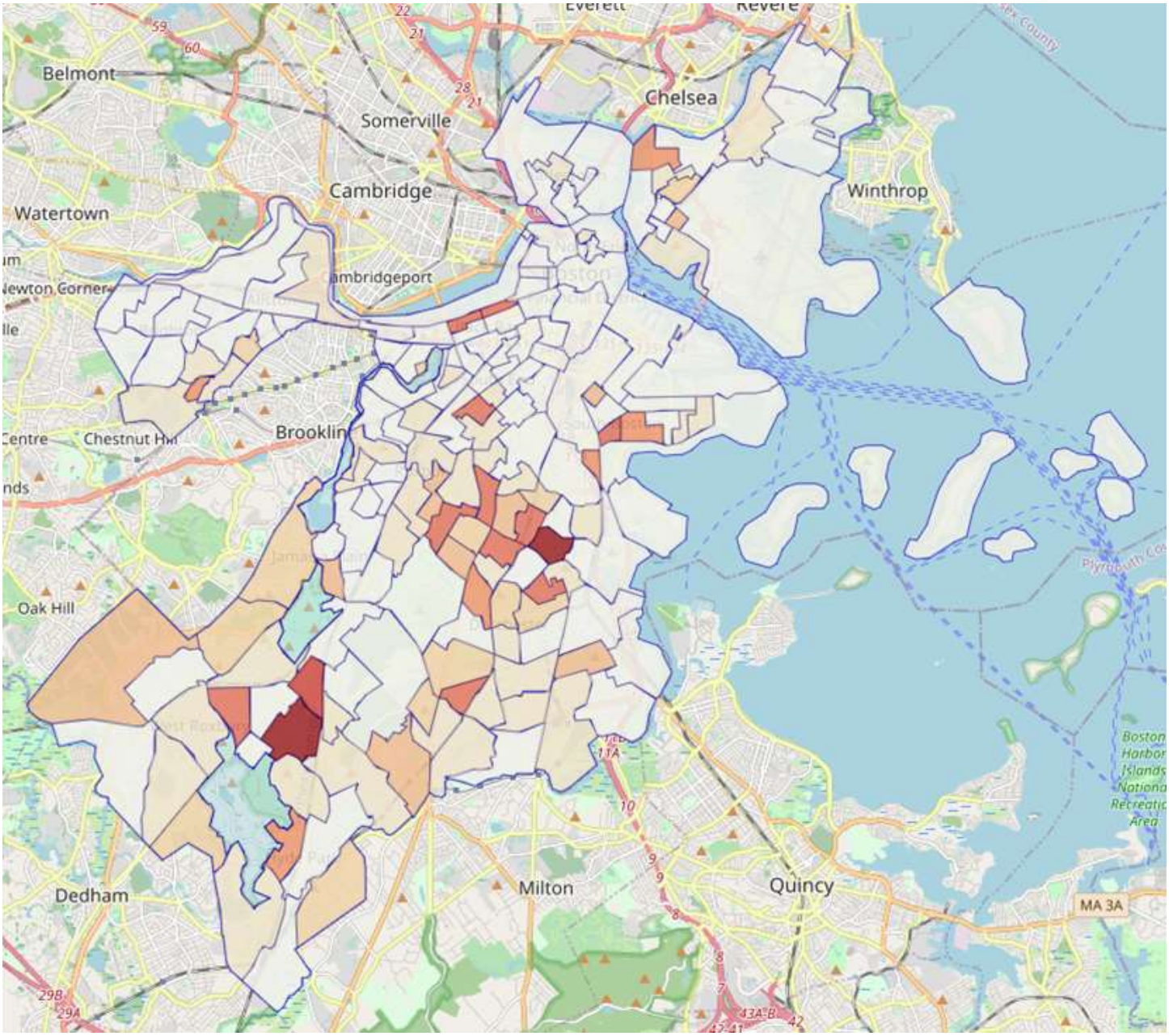
Venue count (darker is higher)



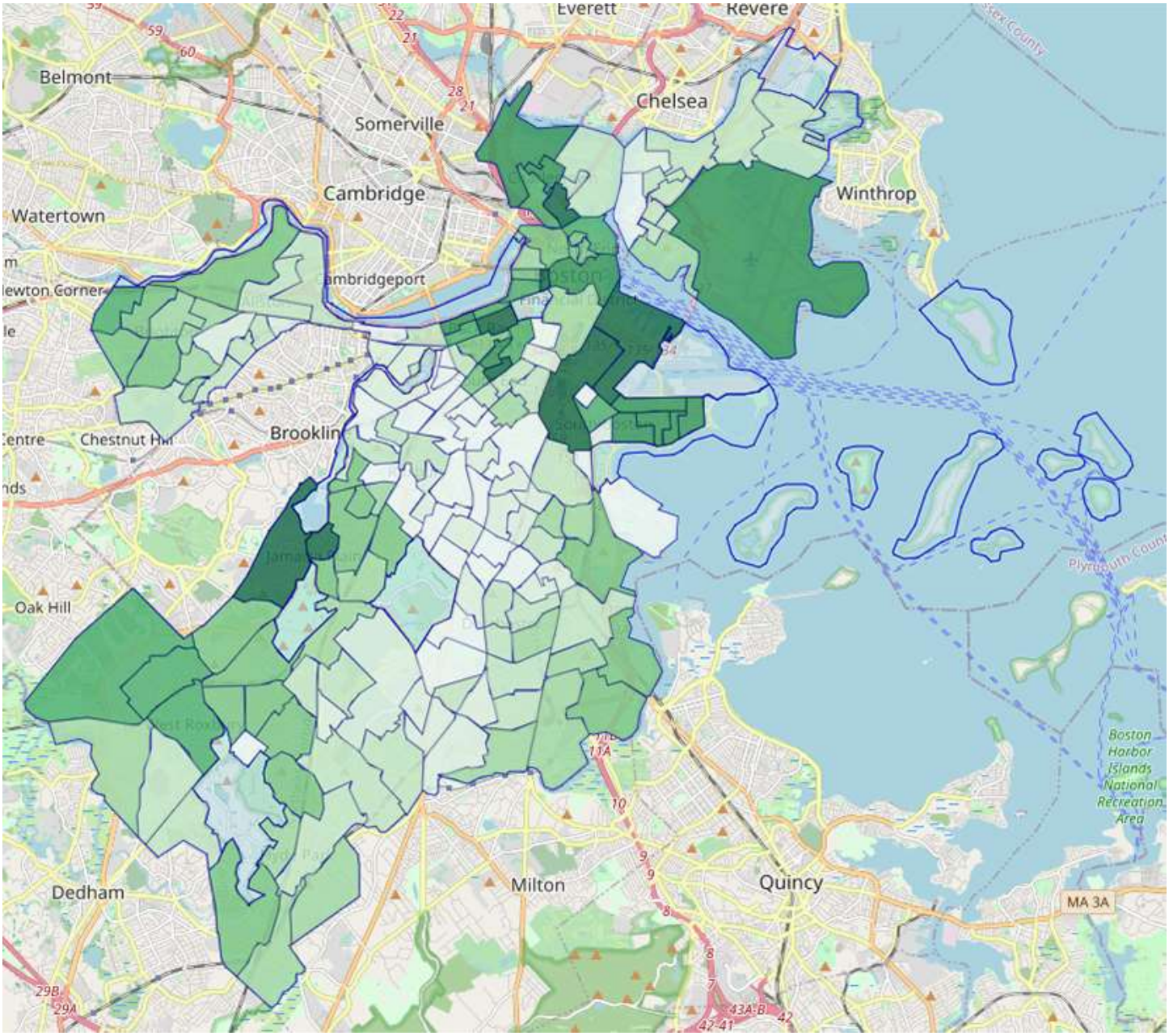
Venue density (darker is higher)



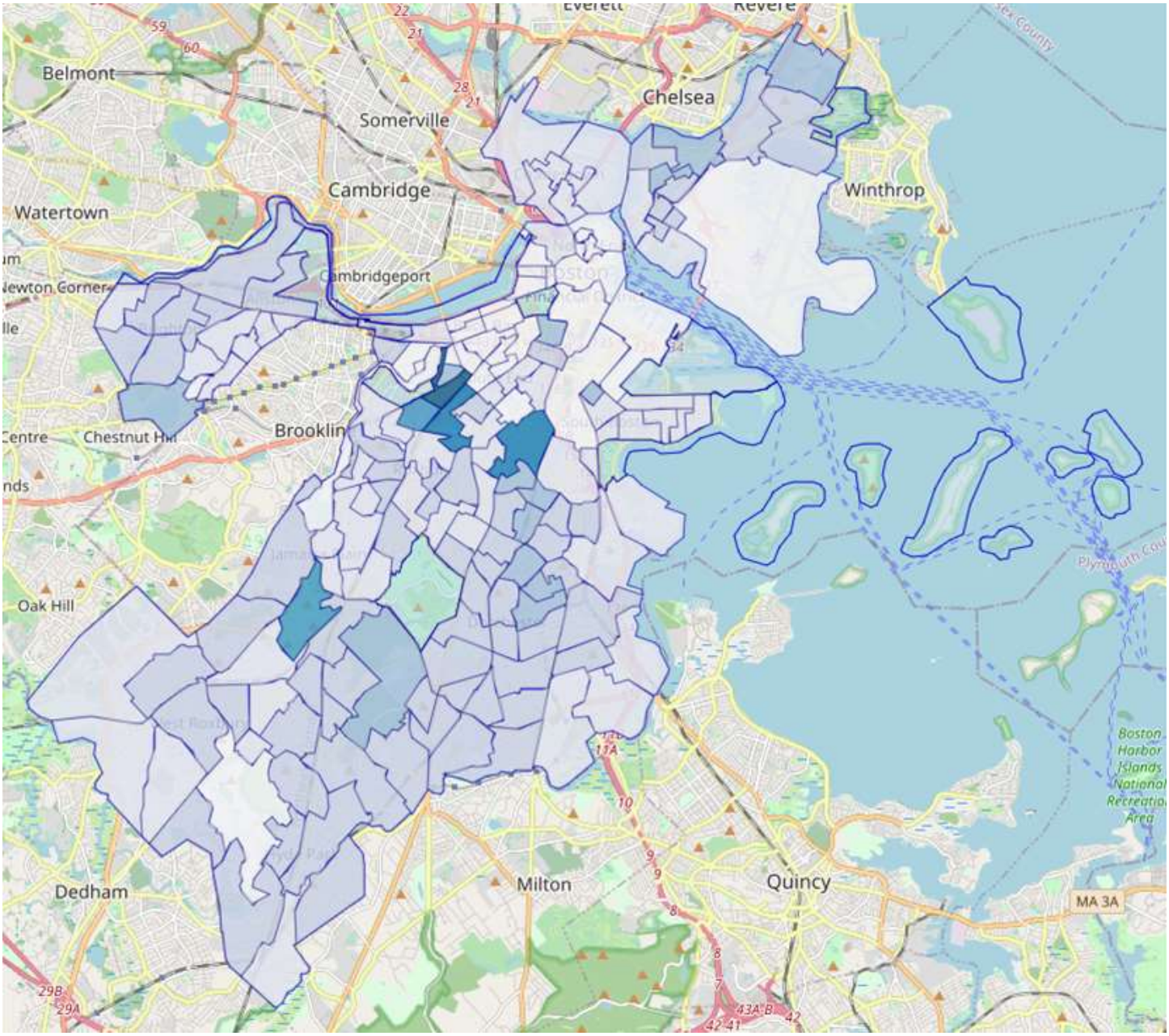
Population per venue (darker is higher)



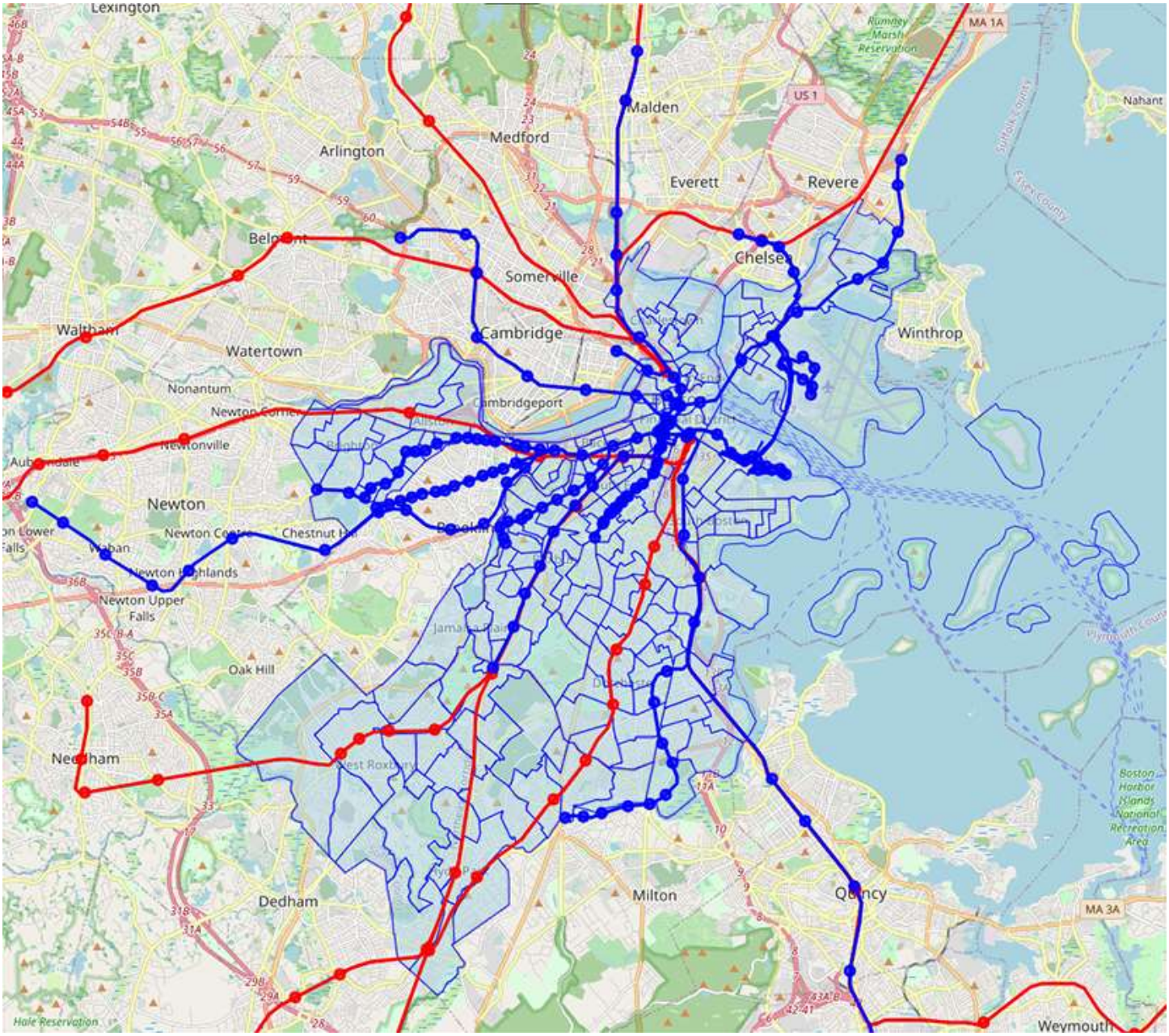
Median household income (darker is higher)



Population per housing unit (darker is higher)



Transit map (blue is the T, red is Commuter Rail)



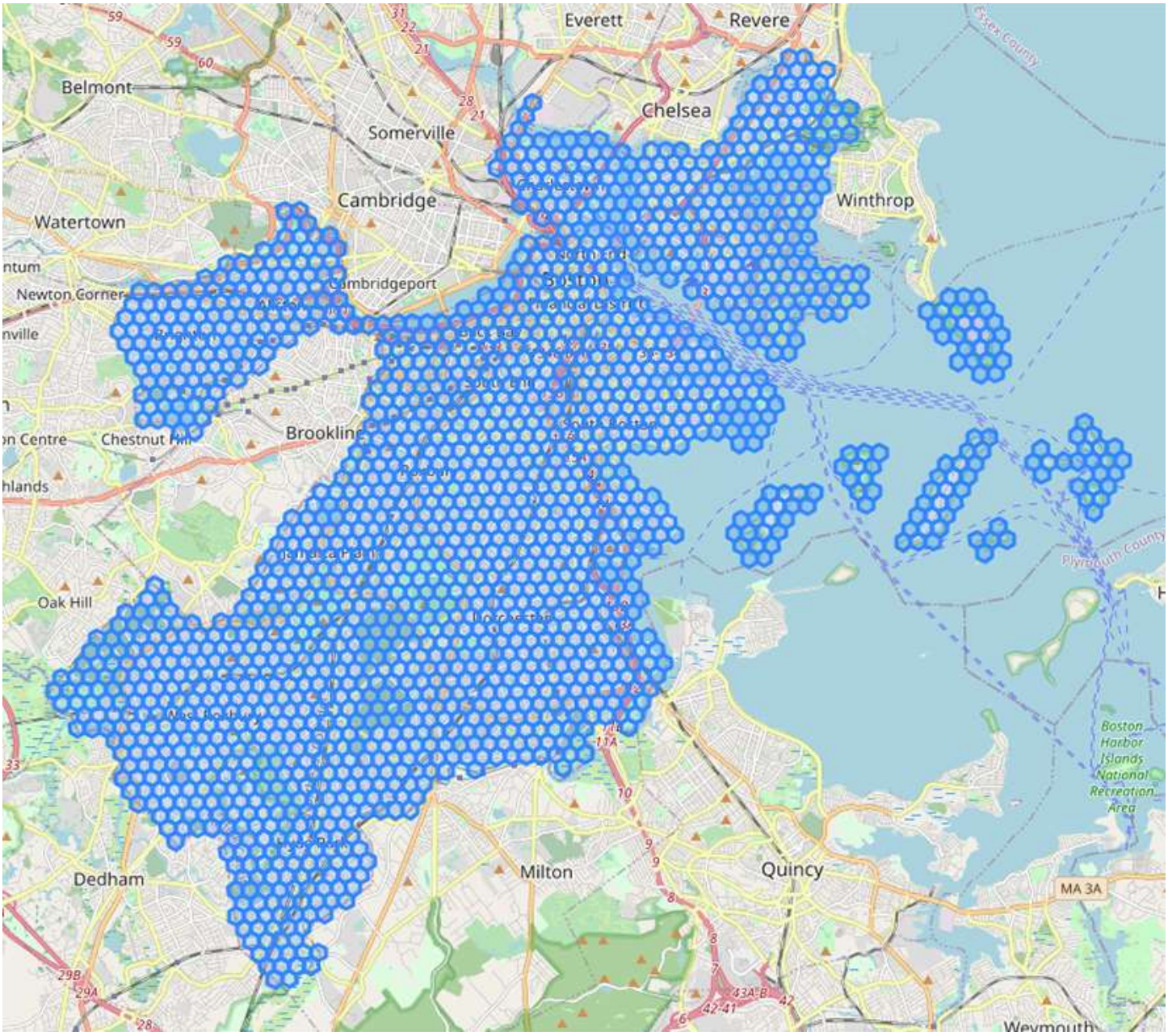
5 Analysis

Okay, so we've mapped where everything in Boston is, the people, the venues, the transit lines. We've found our starting point, areas which are under-represented in terms of venue per capita. Now it's time to focus in on identifying areas meeting a defined list of criteria. We'll blanket the city with a grid of hexagons surrounding centroids 1000 feet apart. We'll eliminate any centroids that landed within the gray buffer zones we mapped. We'll eliminate centroids that landed in the water (a possibility given that census tracts do contain water areas). We'll eliminate centroids that landed in unlikely areas for placing a restaurant (school grounds, cemeteries, parks, wetlands). Then we'll eliminate any centroids in tracts with population per venue of less than 700. This will give us locations which are:

1) not near existing venues
2) not in obviously implausible locations
3) in parts of the city under-populated with food venues

5.0.1 Time For A Hexy Party

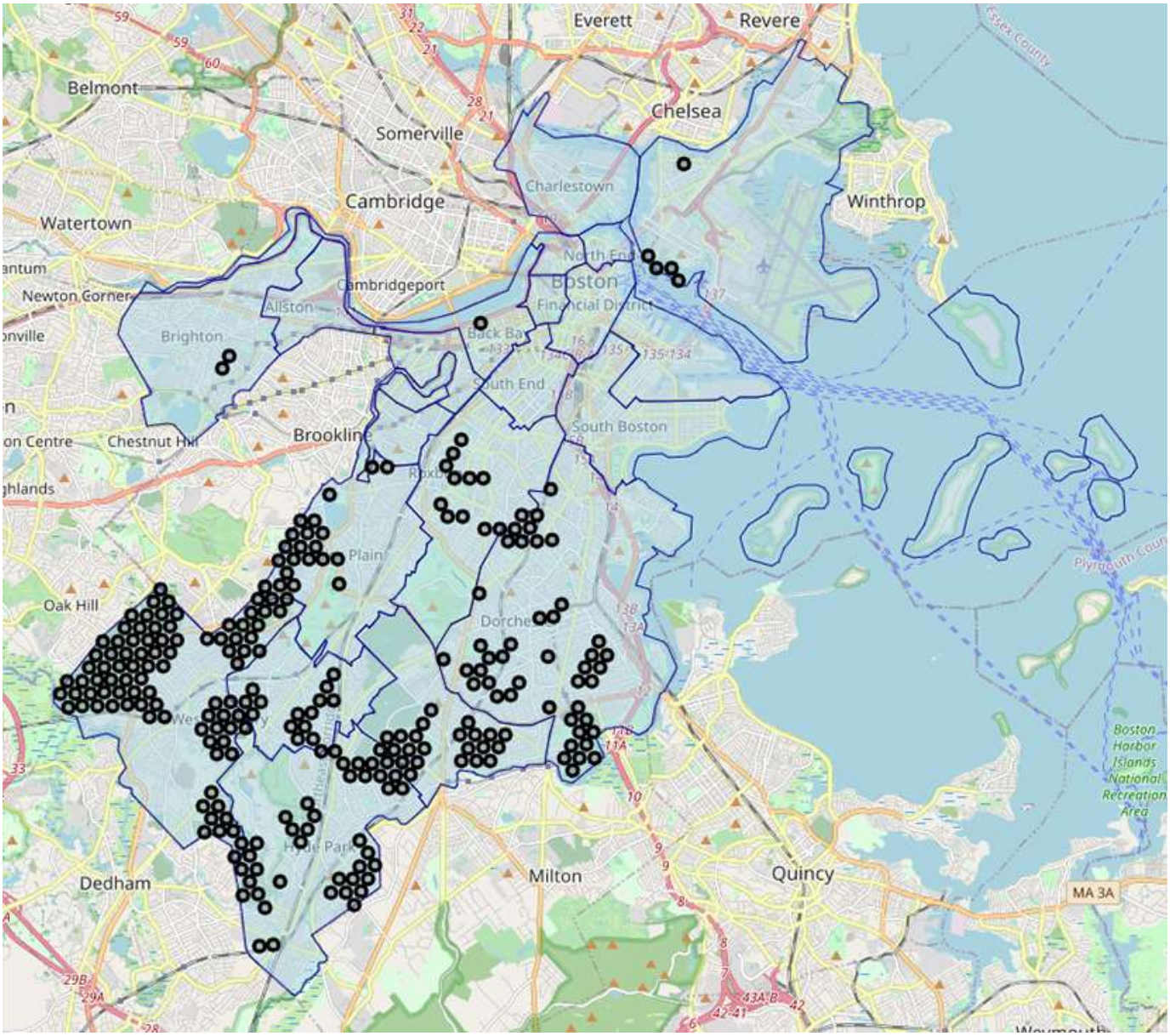
We'll use the "whole-city" polygon of Boston we created earlier and loop a function to create our evenly spaced centroirds within the boundaries of said polygon. Then we'll write a function to draw hexagonal polygons around each centroid, and make our baseline hexmap of Boston, without any of the aforementioned exclusions applied.



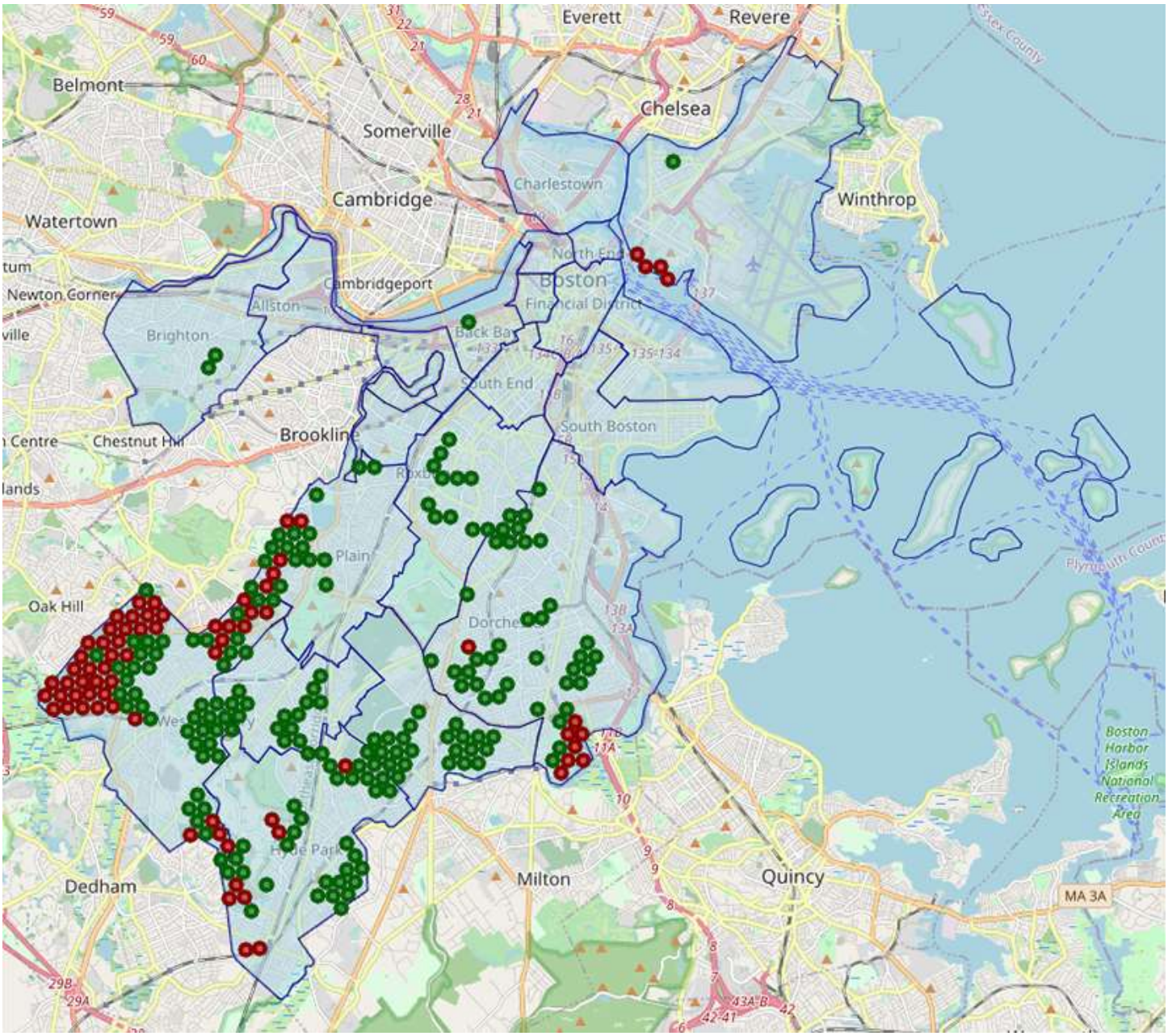
5.0.2 Removing Points in Buffer from Hex List
Now it's time to start carving chunks out of the master hexmap we just made, based on the rules we've laid out. The first step is to take each centroid in the grid and check if it is within the buffer zone multipolygon we made earlier. A new column will be added to the point list to tell us if a given row is in the buffer or not, then we'll remove the offending rows.

5.0.3 Checking Census Tract of Hex List
The Census Bureau uses tract codes in the 9800 range to identify special land-use areas with minimal population, such as parks, wetlands, nature preserves, cemeteries, airports, industrial land, etc. The Boston data bears this out, with 98XX tracts representing Harbor Islands, Franklin Park, Stony Brook Reservation, Arnold Arboretum, Forest Hills Cemetery, Marine Park and Castle Island, Massport Terminal, Logan Airport, Charles River Reservation, Irving Oil Terminal, Belle Isle Marsh Reservation, Boston Commons, and Muddy River / Jamaica Pond. Of these only the Logan Airport tract has a significant restaurant density, however the entire tract is controlled by Massport, and the 80+ venues (including nine Dunkin' shops) are all in the various terminals of the airport. Definitely doesn't meet our "not near existing venues" rule. So we'll remove the hexagon centroids that are in the 98XX tracts.

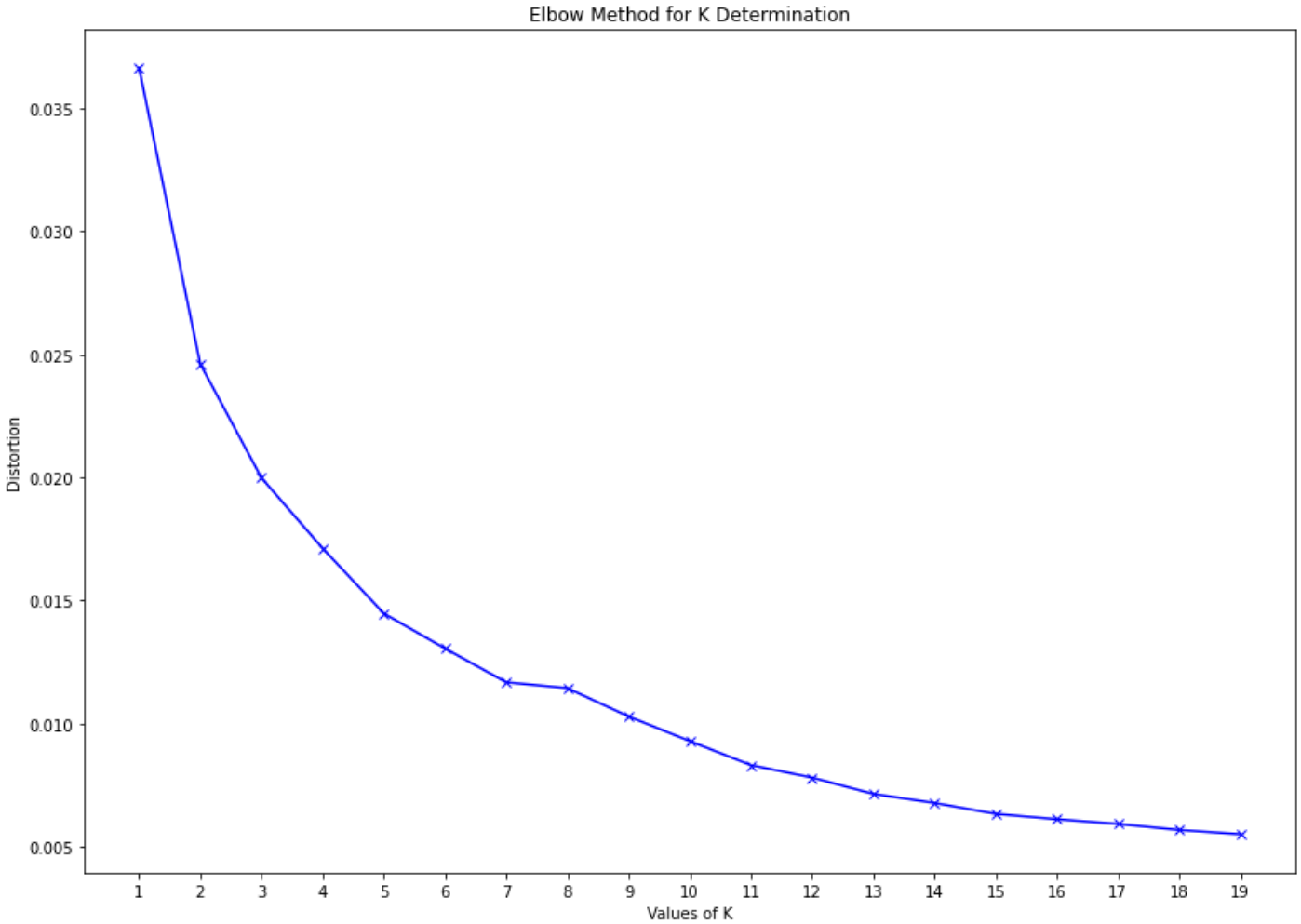
Using the census tract ID we've pulled, we can tie back to the various bits of demographic etc data we previously assembled, and based on that we'll remove any centroids that are in census tracts below 700 people per venue. We started with 1816 centroids, removing those in the buffer zone dropped it to 819, removing those in non-viable tracts dropped it to 460, and limiting to those with Pop per Ven of 700+ dropped it to 253. Let's take a quick look to check if any of the remaining centroids are sketchy, eg. just off shore, in a park or schoolyard, etc. We'll map our remaining centroids and take a quick look at where the centroids landed. For example, that zig-zag of four points in the river just south of the airport… those should probably go.



As one would expect, the matrix of centroids we generated has some “problem children”. The coordinates for those locs have been passed to a separate list and next we’ll split the centroids into “ok” and “problem” for one final inspection. Of our 253 remaining centroids, 72 are in places like schoolyards, marshland, etc. Dropping those leaves 181 viable centroids remaining. In the map below, the bad ones are in red, the survivors are in green.



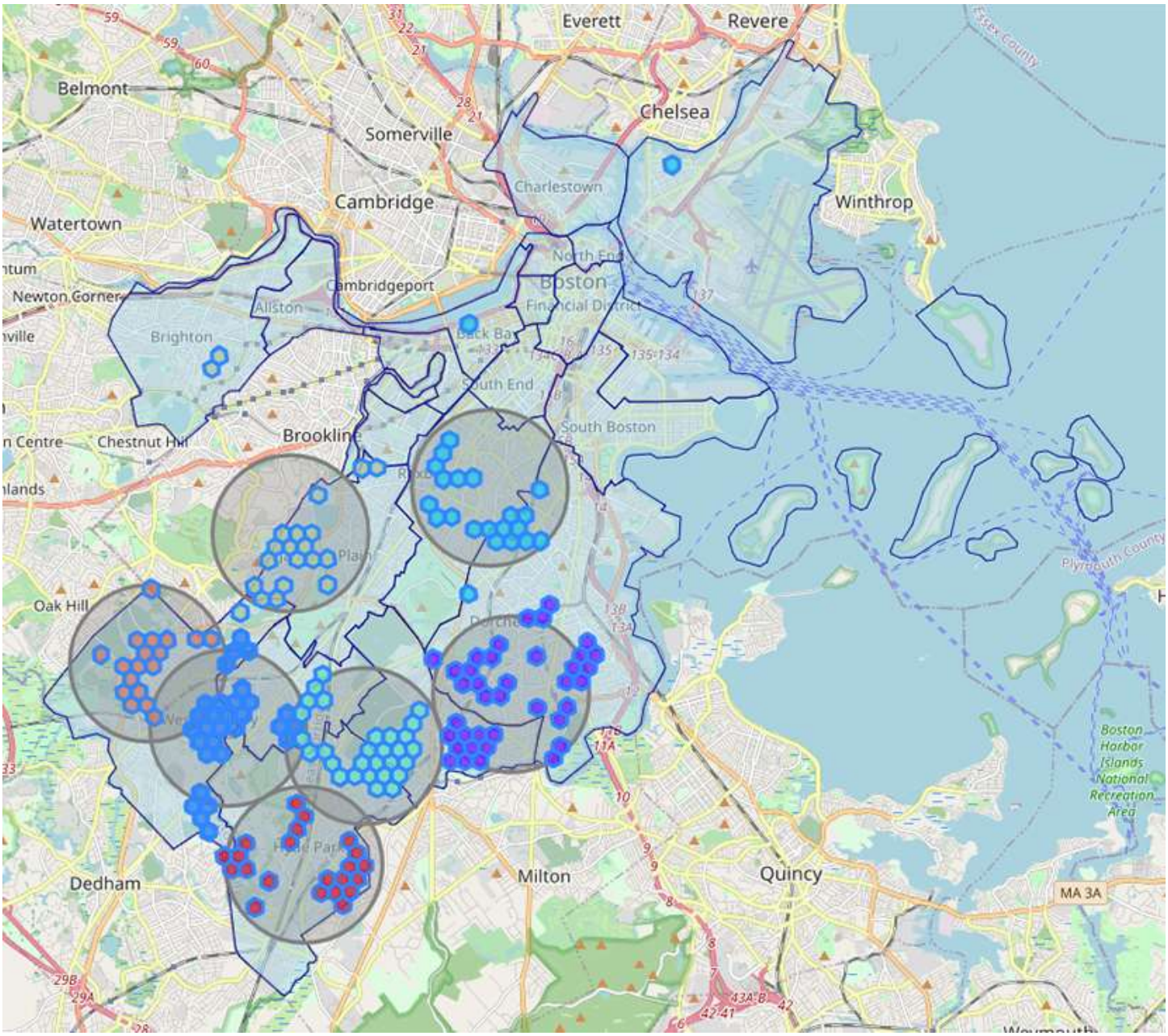
Next we'll take our 181 remaining centroids and use K-Means clustering to group them. First we'll chart a series of runs using K=1 thru K=20 to determine how many clusters to make.



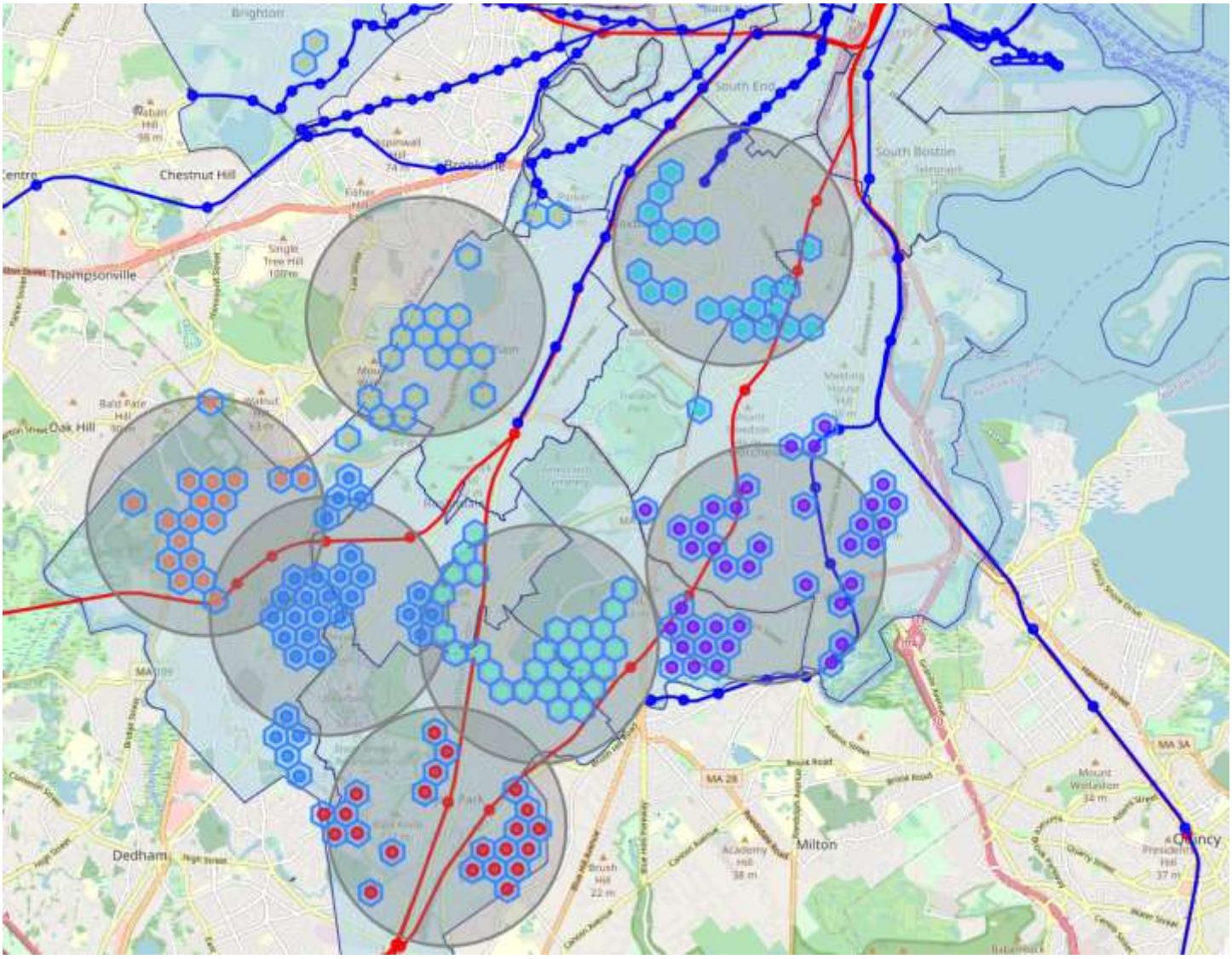
It's not the most clear-cut elbow in the world, but at 7 there's a decent flattening, so we'll set our K at 7 and make our clusters.

5.0.4 Making The Final Hexagons

Okay, we've got our list of centroids filtered down to 181 and assigned to one of seven clusters, now it's time to revist our hexagon function and make our final hexagons. Below are our final clusters and hexes.



Close-up of main southwest clusters



As expected when we first made the “Population per Venue” bar chart, the neighborhoods in the southwest of the city dominate. There are several areas in the these clusters which are adjacent to T/Commuter Rail stations, which is another plus. Before moving on to results and conclusion, we’ll take a quick look at what sort of venues are currently present in our target areas.

5.0.5 One-Hot Encoding on Categories by Census Tract
For purposes of ranking category types within tracts, I decided to use top three as the number of venue categories to use. In previous one-hot encoding exercises we’d gone as far as using the top 10, however looking at the number of different categories in each tract, I found that going that deep introduced more bad data than it was worth.

The way the standard “most common category” function works, once it runs out of venues and/or distinct categories, it starts filling in empty categories alphabetically. This means that if a tract only has three different categories and you ask for the five most common, number 4 and number 5 aren’t real. This became apparent when running the function with “num_top_venues” at 10 and finding that “African” made the top ten for quite a few census tracts even though there are only seven African restaurants in the entire city. Looking at the tracts, 87% of tracts have less than 10 distinct categories, 44% of tracts have less than 5 distinct categories, and 23% of tracts have less than 3 distinct categories. So “top ten” is definitely out.

The rows below list the three most common venue types, along with some demographic basics, for any census tract which hosts one of our 181 final points. You'll notice a handful of tracts that have no venues at all, which seems like an opportunity.

Cluster	Neighborhood	Census Tract	1st Most Common Cat	2nd Most Common Cat	3rd Most Common Cat	Pop	Venue Count	Pop per Venue	Pop Density	Venue Density	Housing Units	Median Household Income
1	Dorchester	920	Pizza	Sandwich	Asian (Southea	4,945	4	1,236.25	26,151.77	21.15	1,693	$57,214
1	Dorchester	923	Fast Food	General	Sandwich	2,893	4	723.25	21,846.66	30.21	1,077	$55,469
1	Dorchester	1001	Pizza	Mediterranean	Caribbean	5,510	4	1,377.50	13,761.51	9.99	2,047	$28,659
1	Dorchester	1002	(none)	(none)	(none)	2,787	(none)	2,787.00	20,588.88	(none)	1,013	$53,269
1	Dorchester	1003	American	General	Indian	3,303	4	825.75	17,391.94	21.06	1,262	$51,645
1	Dorchester	1004	General	Pizza	Baked Goods	4,865	6	810.83	20,122.70	24.82	1,830	$68,194
1	Dorchester	1005	General	Specialty	Pizza	5,989	7	855.57	18,058.98	21.11	2,383	$59,913
1	Dorchester	1006	Asian (East)	Pizza	Deli	5,154	3	1,718.00	24,157.00	14.06	2,192	$76,593
1	Dorchester	1008	Pizza	Baked Goods	Seafood	5,546	7	792.29	8,134.68	10.27	2,702	$89,500
1	Mattapan	1010	Baked Goods	Fast Food	Pizza	4,979	7	711.29	11,067.13	15.56	2,149	$49,947
2	Roslindale	1104	General	(none)	(none)	4,309	1	4,309.00	11,836.49	2.75	1,767	$100,833
2	Roslindale	1105	Pizza	(none)	(none)	3,253	1	3,253.00	13,004.95	4.00	1,371	$94,615
2	West Roxbury	1106	Pizza	Latin American	(none)	2,969	2	1,484.50	5,466.92	3.68	1,295	$107,012
2	West Roxbury	1303	Indian	Asian (East)	Baked Goods	4,419	5	883.80	7,936.01	8.98	1,811	$123,241
2	West Roxbury	1304.1	Latin American	Pizza	American	5,107	4	1,276.75	12,611.86	9.88	2,192	$53,265
3	Back Bay	108.02	American	(none)	(none)	3,059	1	3,059.00	50,961.13	16.66	2,106	$131,518
3	Dorchester	901	Pub Fare	Pizza	(none)	4,571	2	2,285.50	26,485.56	11.59	1,863	$34,195
3	Dorchester	903	Latin American	(none)	(none)	3,179	1	3,179.00	21,644.85	6.81	1,268	$30,975
3	Dorchester	913	Mediterranean	African	(none)	2,499	2	1,249.50	24,651.05	19.73	868	$41,020
3	Dorchester	914	Pizza	(none)	(none)	2,741	1	2,741.00	21,842.22	7.97	939	$45,388
3	Dorchester	915	(none)	(none)	(none)	4,370	(none)	4,370.00	28,003.96	(none)	1,653	$53,651
3	East Boston	501.01	Baked Goods	Latin American	(none)	5,115	2	2,557.50	44,192.88	17.28	1,850	$51,230
3	Roxbury	814	African	General	Pizza	3,003	4	750.75	12,535.01	16.70	1,462	$55,909
3	Roxbury	815	(none)	(none)	(none)	2,134	(none)	2,134.00	16,435.18	(none)	905	$36,464
3	Roxbury	817	General	American	Fast Food	3,820	5	764.00	15,616.03	20.44	1,599	$32,183
3	Roxbury	819	Pizza	(none)	(none)	3,115	1	3,115.00	18,615.81	5.98	1,397	$27,745
3	Roxbury	820	Fast Food	Deli	(none)	2,815	2	1,407.50	18,591.53	13.21	1,278	$43,077
3	Roxbury	904	Mediterranean	Caribbean	(none)	3,659	2	1,829.50	22,291.19	12.18	1,294	$38,750
4	Hyde Park	1404	General	Pizza	Pub Fare	7,650	9	850.00	10,549.49	12.41	2,906	$62,743
4	Mattapan	1010	Fast Food	Pizza	American	5,480	3	1,826.67	10,949.10	5.99	2,266	$64,185
4	Mattapan	1011	Baked Goods	Caribbean	(none)	3,155	2	1,577.50	21,516.85	13.64	1,160	$50,625
4	Roslindale	1104	Asian (East)	(none)	(none)	3,566	1	3,566.00	17,678.12	4.96	1,362	$64,494
4	Roslindale	1104	General	(none)	(none)	4,309	1	4,309.00	11,836.49	2.75	1,767	$100,833
5	Brighton	4.01	Asian (East)	Asian (General)	Asian (Southea	5,672	7	810.29	34,081.08	42.06	2,965	$72,739
5	Jamaica Plain	1201	Baked Goods	(none)	(none)	2,095	1	2,095.00	18,691.35	8.92	1,060	$100,977
5	Jamaica Plain	1201.1	General	African	(none)	2,444	2	1,222.00	3,443.94	2.82	953	$161,473
5	Jamaica Plain	1207	African	Sandwich	(none)	2,014	2	1,007.00	17,534.45	17.41	1,044	$96,711
5	West Roxbury	1106	Pizza	Latin American	(none)	2,969	2	1,484.50	5,466.92	3.68	1,295	$107,012
6	West Roxbury	1106	Pizza	Latin American	(none)	2,969	2	1,484.50	5,466.92	3.68	1,295	$107,012
6	West Roxbury	1301	Asian (East)	Baked Goods	Italian	5,906	3	1,968.67	3,520.97	1.79	2,816	$112,468
7	Hyde Park	1401	Baked Goods	Italian	Deli	4,401	6	733.50	5,479.90	7.47	1,654	$98,750
7	Hyde Park	1401.1	(none)	(none)	(none)	2,531	(none)	2,531.00	10,448.96	(none)	1,051	$82,750
7	Hyde Park	1402	General	Pizza	African	5,600	4	1,400.00	9,426.10	6.73	2,081	$77,250
7	West Roxbury	1304.1	Latin American	Pizza	American	5,107	4	1,276.75	12,611.86	9.88	2,192	$53,265

6 Results and Discussion

The analysis shows that the tourist-heavy northeast of the city around the North End and Downtown is already thick with venues. Where there are tourists, there are always possibilities, but I decided to go a different direction. There are a couple different types of restaurant, and I don't mean cuisine categories; First, there are restaurants you go to as an "occasion", for a birthday or special night out or that sort of thing. For those times when you're willing to spend half an hour or more getting across town just for dinner. Second, there are restaurants you go to because they're convenient or nearby. Doesn't mean they're bad restaurants, it just means sometimes you want to get something to eat without the hassle of travel, tourists, reservations, etc. This is the sort of place I'd suggest for the areas we've identified. A decent local place for the people who live in the area. Maybe just a sandwich shop or bodega, maybe a general "American" restaurant. Pizza places and donut shops already seem well-represented in the target areas, perhaps give the locals a convenient nearby dining option that isn't another pizza shop or Dunkin'.

7 Conclusion

The intent of this datawork has been to identify candidate areas for a new restaurant in Boston. Using the data available, we settled on using a metric of population per venue as a guide to locating these candidate areas. We've identified several areas with high population, low venue options, and rapid transit access. A stakeholder would need to then take these candidate areas and do additional investigation (eg. zoning requirements, real estate cost and availability, etc.), however I believe we've created a good jumping-off point for further investigation.