An Exploration of Parks and Gyms Across the Houston and Dallas Metropolitan Areas

Coleman Cunningham

February 28, 2021

Prepared as the report component of the Applied Data Science Capstone course

1. Introduction

1.1 Background

With the World in turmoil as COVID-19 turned into a pandemic, the physical activity routines of many were put in question. The shutdowns of gatherings extended to gyms, shared worked spaces, and other enclosed areas, cutting off the facilities that many relied on to maintain their physical wellbeing. As the pandemic took hold, gym attendance dropped due to concerns for amplified risk of transmission, as well as gathering restrictions put in place. As an article in the *Washington Post* noted, there has been "widespread evidence of a surge in outdoor activity since the pandemic hit."

1.2 Problem

Depending on the type of area one lives in, the options for exercise friendly outdoor spaces can be limited. As the impacts of COVID-19 pushed people to explore options other than gyms, outdoor venues became the logical destination for people seeking space to exercise. Within Metropolitan areas, parks are not present on every block, just the same as gyms. With the varied locations of parks and gyms as a limiting factor, some groups of people may experience more obstacles than others when attempting to find alternative places to carry out their physical activity. This paper aims to take into account the distribution of locations of parks and gyms across two similar cities and explore the possibility of correlation with variables regarding the surrounding area such as per capita income, population, race, and age distributions.

1.3 Interest

With the recent impact of COVID-19 on everyday life, the issue of physical wellbeing has been propelled to the forefront of many peoples' minds. Taking a look at the ease of access to outdoor venues is something that many would find important. The locations of gyms as they compare to parks may also provide some insight for investors that are considering opening a gym facility in one of these two cities.

2. Data

2.1 Data Sources

Majority of the data used in this analysis was obtained through the National Historical Geographic Information System (NHGIS). The NHGIS website describes their service as providing "free online access to summary statistics and GIS files for U.S. censuses and other nationwide surveys from 1790 through the present." For this analysis, the datafiles selected from NHGIS were limited to the year of 2019, and held data on the Place, County and Tract levels. The tables used were sourced by NHGIS through the American Community Survey, and the matching GIS files downloaded from NHGIS were generated based primarily on U.S. Census Bureau TIGER/Line files. The names of the tables used, as well as their parent datasets, can be found in the Appendix, under Appendix A (NHGIS Data – Table Names and Source Datasets). The other data source that was heavily relied upon was Foursquare. Using the Search API, I was able to gather information on venues that held the Foursquare category titles of "Park," or "Gym," across the location areas of interest.

2.2 Import of Census Data and GIS Files

2.2.1 Initial Import of Datafiles

Due to the size and saving format of the datafiles downloaded from NHGIS, they had to be imported using SQL. After setting the column types for each file in the DB2 interface and completing the import of the data, SQL was then used through a Jupyter Notebook to transfer that data into the Python workspace. The full list of column names can be found in **Appendix: Breakdown of Raw Data**.

2.2.2 Initial Import of GIS Files/Coordinate Reference System Note

The GIS files, as downloaded from NHGIS, were classified as .shp, which required them to be read into the Jupyter notebook as GeoDataFrames using GeoPandas. Each GIS file had an initial Coordinate Reference System (CRS) that they were formatted with. Following a later merge of data, all datasets were converted to the same CRS, so as to make sure that the geospatial representations were consistent.

2.3 Cleaning of Census Data and GIS Files

2.3.1 Cleaning of Datafiles

Once the Census files were imported into the Jupyter Notebook, many columns were dropped. The files each had over 70 columns initially, which was trimmed down to the variables of interest. On the place level, following renaming, those columns were:

'GISJOIN', 'State FIPS Code', 'Place Area Name',
'Total Place Pop', 'Total Place Median Age', 'Place Median Age M', 'Place Median Age F',
'Place Pop White', 'Place Pop Black or African American',
'Place Pop American Indian and Alaska Native', 'Place Pop Asian',
'Place Per Capita Income - 12 Months', 'Total Place Pop ME',
'Total Place Median Age ME', 'Place Median Age ME M', 'Place Median Age ME F',
'Place Pop ME White', 'Place Pop ME Black or African American',
'Place Pop ME American Indian and Alaska Native', 'Place Pop ME Asian',
'Place Per Capita Income ME - 12 Months'

On the county level, after renaming, those columns were:

'GISJOIN', 'State FIPS Code', 'County Name', 'County FIPS Code', 'County Area Name',
'Total County Pop', 'Total County Median Age', 'County Median Age M',
'County Median Age F', 'County Pop White', 'County Pop Black or African American',
'County Pop American Indian and Alaska Native', 'County Pop Asian',
'County Per Capita Income - 12 Months', 'Total County Pop ME',
'Total County Median Age ME', 'County Median Age ME M', 'County Median Age ME F',
'County Pop ME White', 'County Pop ME Black or African American',
'County Pop ME American Indian and Alaska Native', 'County Pop ME Asian',
'County Per Capita Income ME - 12 Months'

On the tract level, after renaming, those columns were:

'GISJOIN', 'State FIPS Code', 'County Name', 'County FIPS Code', 'Tract FIPS Code',
'Tract Area Name', 'Total Tract Pop', 'Total Tract Median Age', 'Tract Median Age M',
'Tract Median Age F', 'Tract Pop White', 'Tract Pop Black or African American',
'Tract Pop American Indian and Alaska Native', 'Tract Pop Asian',
'Tract Per Capita Income - 12 Months', 'Total Tract Pop ME',
'Total Tract Median Age ME', 'Tract Median Age ME M', 'Tract Median Age ME F',

'Tract Pop ME White', 'Tract Pop ME Black or African American',

'Tract Pop ME American Indian and Alaska Native', 'Tract Pop ME Asian',

'Tract Per Capita Income ME - 12 Months'

The details of these columns can be found in **Appendix: Breakdown of Cleaned Data**.

2.3.2 Cleaning of GIS Files

The GIS files held a lot of valuable information for each geographic sect, part of which was in the "geometry," column. This column, for each geographic piece (Place, County, Tract) held the coordinate systems that mapped out the complex shape bounding that area. However, there was also plenty of information that was able to be dropped due to its irrelevance to the rest of the process. On the place level, the following 10 columns were dropped:

'PLACEFP', 'PLACENS', 'PLACEFP', 'PLACENS', 'GEOID', 'NAMELSAD', 'LSAD', 'CLASSFP', 'PCICBSA', 'PCINECTA', 'MTFCC', 'FUNCSTAT'

Similarly, on the County Level, the following 12 columns were dropped:

'STATEFP', 'COUNTYNS', 'GEOID', 'NAME', 'NAMELSAD', 'LSAD', 'CLASSFP', 'MTFCC', 'CSAFP', 'CBSAFP', 'METDIVFP', 'FUNCSTAT'

Finally, on the Tract Level, the following 8 columns were dropped:

'STATEFP', 'COUNTYFP', 'TRACTCE', 'GEOID', 'NAME', 'NAMELSAD', 'MTFCC', 'FUNCSTAT'

The details of these columns can be found in **Appendix Breakdown of Raw Data**. After dropping those columns, many of the remaining columns were renamed, resulting in the following columns on the Place Level:

'GISJOIN', 'State FIPS Code', 'Place Name', 'Land Area', 'Water Area', 'Place Center Lat', 'Place Center Long', 'Place Perimeter', 'Place Area', 'geometry'

Similarly, the following columns, some renamed, remained at the county level:

'GISJOIN', 'County FIPS Code', 'Land Area', 'Water Area', 'County Center Lat', 'County Center Long', 'County Perimeter', 'County Area', 'geometry'

Finally, on the Tract Level, the following columns remained:

'GISJOIN', 'Land Area', 'Water Area', 'Tract Center Lat', 'Tract Center Long', 'Tract Perimeter', 'Tract Area', 'geometry'

The details of these columns can be found in **Appendix: Breakdown of Cleaned Data**.

2.4 Merging GIS and Datafiles

After filtering the data to the columns of interest, the GIS and shapefiles were merged as GeoPandas DataFrames. The column they were merged on was GISJOIN, as it served as a unique value in each dataset but matched regarding the areas they represented. With the columns of interest together with their corresponding geospatial data, the next step could begin

2.5 Sourcing Foursquare Data

The Foursquare data, a required part of this project, was acquired using the Foursquare Search API. Due to rate limits and retrieval limits, the data had to be gathered in parts. The method of this is explored in the Methodology section, but the data gathered was in two parts for each city. Each Foursquare usage resulted in sets of longitude and latitude points representing search results for specific venue categories: "Park" and "Gym/Fitness Center." These points were gathered through many searches throughout the cities, laid out further in the Methodology section.

3. Methodology

3.1 Creating More Variables of Interest

In order to look more in depth at the data, more columns were created that held variables calculated from existing variables. The following were created:

3.1.1 Pop Density per sq km

Population Density per sq km, calculated by dividing 'Land Area' by 1000000, then dividing 'Total Tract Pop' by that. The reason for the first operation is that 'Land Area' was in square meters, so this was required to convert it to square kilometers.

3.1.2 Population Race Percentages

For each race, the population of each race was simply divided by the total population of the area and multiplied by 100.

3.2 Filtering to Cities

To get down to the cities of interest (Dallas & Houston), the DataFrame containing the place level GIS data was trimmed twice into separate instances, each of which contained only place data on the city specified. Following the trimming of the place level data, this file was then merged with the county and tract levels, individually. This was done with a GeoPandas spatial join, with the "how" set to an inner join, and the "op" set to intersects. The options for inner join and intersects meant that the geometry and data of the tract level or county level file would be preserved but would be limited to those areas that are either within or intersecting with the geometry (outline) of the city. If the join had been done instead using the within operation, many tracts/counties that were majority inside of the city would have been discounted. It was decided that keeping in those was more important than severely limiting the data to the arbitrary lines of the city, especially since the surrounding tracts were still in the general metropolitan area and had at least parts of them inside the city limits.

3.3 NaN Values

3.3.1 Discovering NaN Values

Once the data was merged for each city (one large DataFrame to use for each city), the data had to be explored for NaN values that could impact it. In the exploration of the NaN values that were revealed, I found two categories of them. During the initial search for NaN values, I found the following counts for Houston and Dallas, respectively:

Houston

```
Total Tract Median Age                          1
Tract Median Age M                              1
Tract Median Age F                              2
Tract Per Capita Income - 12 Months             1
Total Tract Median Age ME                       1
Tract Median Age ME M                           1
Tract Median Age ME F                           2
Tract Per Capita Income ME - 12 Months          1
Tract Income ME Upper                           1
Tract Income ME Lower                           1
dtype: int64
```

Dallas

```
Tract Median Age M                              1
Tract Median Age F                              1
Tract Per Capita Income - 12 Months             1
Tract Median Age ME M                           1
Tract Median Age ME F                           1
Tract Per Capita Income ME - 12 Months          1
Tract Income ME Upper                           1
Tract Income ME Lower                           1
dtype: int64
```

The counts are extremely similar, the main difference being that the Houston dataset had two NaN values for both 'Tract Median Age F' and 'Tract Median Age ME F' as opposed to the one NaN value shown in the Dallas dataset. Upon further investigation, the Houston NaN values were contained to two observations, while the Dallas NaN values were contained to one observation. One of the Houston NaN value observations only had the two age variables related to Females, while the other had the remaining NaN values.

3.3.2 Changing NaN Values of First Type

For the Houston observation that only had the Female age values as NaN, it was determined that there were no Females in that tract according to the Census Data. Because the variable regarded an age, substituting a value of zero would skew the data. This combined with the distribution of the median age, shown in the image in **Appendix: Distribution Plots**, made it clear that for Houston, the curve was decently symmetric. With those factors in mind, the decision was made to replace the NaN values in 'Tract Median Age F' and 'Tract Median Age ME F' respectively, with the average across Houston for each of their columns.

3.3.3 Changing NaN Values of Second Type

For the remaining NaN values, the tracts were analyzed in many ways, but an amazing discovery was made when looking at it them on a geospatial level. Upon viewing the two tracts on maps, in both the Houston and the Dallas cases, the remaining observations with the NaN values were actually airports in each of those cities. As each of these tracts were limited to solely the area of the airports, the decision was made to drop these observations from the datasets. Following the actions on both types, there were no longer any NaN values in the Datasets.

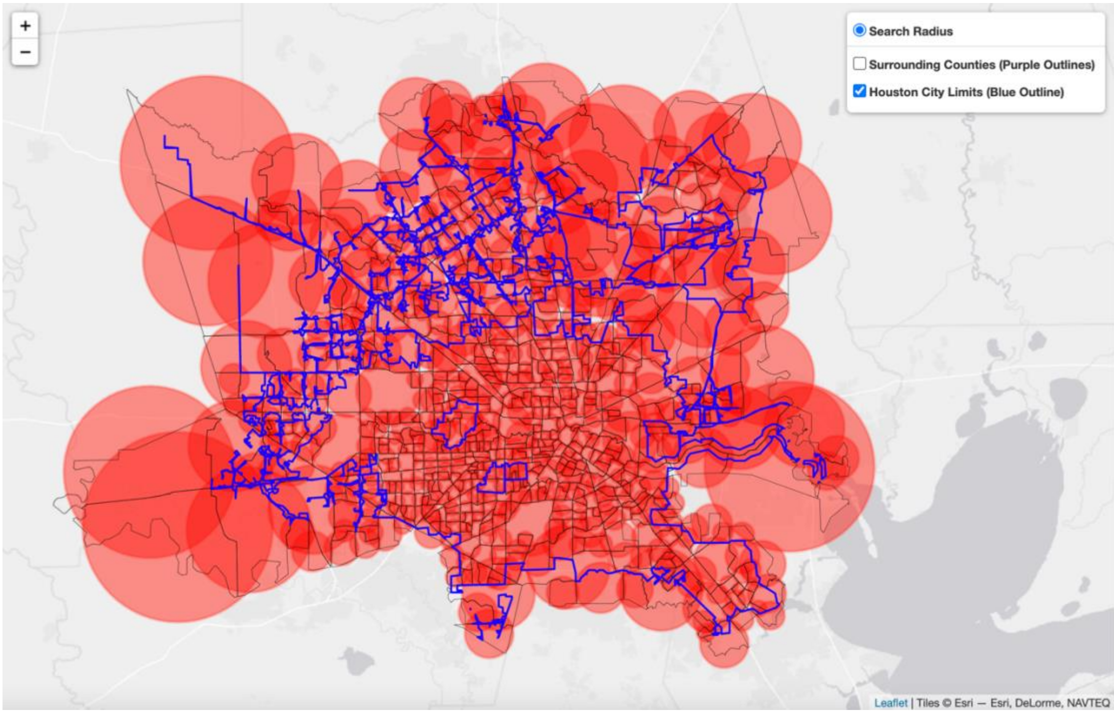
3.4 Collecting Foursquare Data

3.4.1 Limitations of Search API

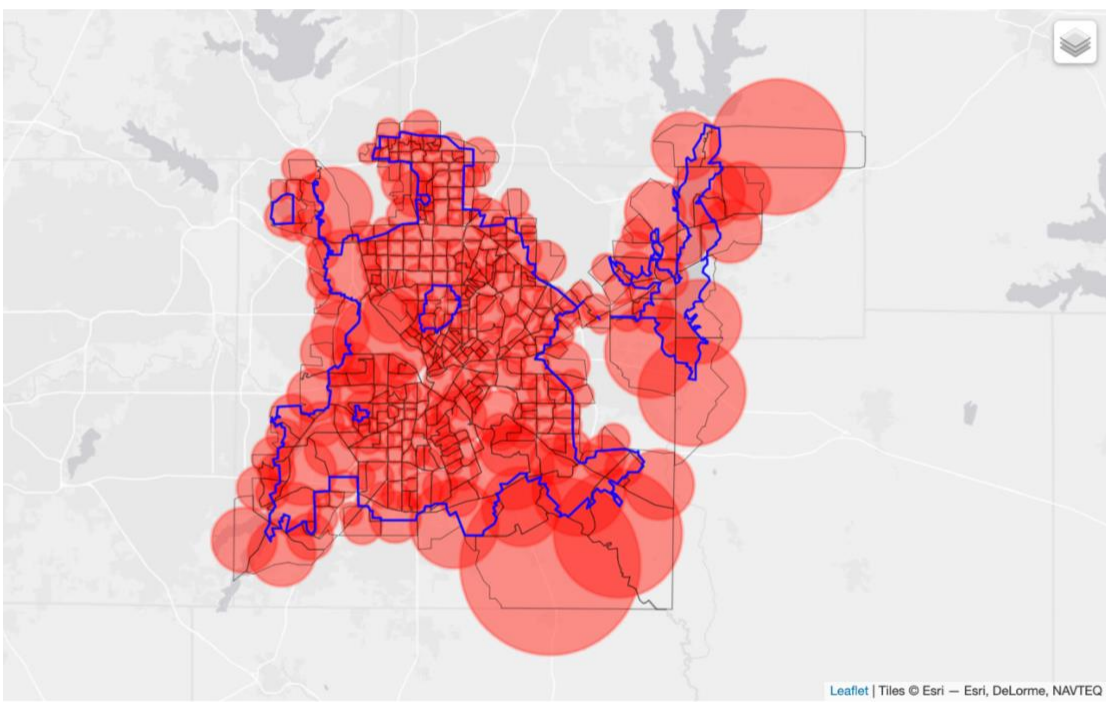
The Foursquare Search API limits the results of any search to 50, no matter how large the area, and also limits the number of searches per hour. In order to collect the data necessary to include the locations of each venue categorized as "Park" or "Gym/Fitness Center" across each city, I had to search in a slightly different manner. I was able to use the following variables to search smaller areas that would end up covering a significant amount of each city:

'Tract Perimeter', 'Tract Center Lat', 'Tract Center Long'

The perimeter of each tract was used to create a functional circle to search for each tract. The perimeter (meters) was divided by 6.283 (a simplification of 2pi), in order to get a radius to use for each search. That radius was then used, along with the latitude and longitude pairs given by the Census Data that represented the center of mass of each tract, to search small areas that spanned the cities. See images on the next page displaying visual representations of the areas covered by the searches for each city.



Houston Foursquare Searches – Visual Representation
Red circles represent search areas; blue outline denotes city limits; grey outlines denote tract shapes



Houston Foursquare Searches – Visual Representation
Red circles represent search areas; blue outline denotes city limits; grey outlines denote tract shapes

3.5 Utilizing Foursquare Results

3.5.1 Appending Existing DataFrames

After getting the results of the searches for each city (and each of the two categories), the data was cleaned to be added into the other DataFrames. The number of parks for each tract, and similarly that of gyms in each tract, was counted utilizing another spatial join and some manipulations to count the occurrences and creating the columns ‘Number of Parks’ and ‘Number of Gyms’ respectively. In anticipation of future exploratory analysis, other columns were also created that converted each of those counts into per capita and per km values.

3.5.2 Mapping points

Using the Foursquare search results, the points for each venue, returned as belonging to the “Park” or “Gym/Fitness Center” categories, were overlayed onto the maps created with the geospatial data gathered. The next section explores this further.

3.6 Visualizing Data – Preparation for Maps

Successful visualization of all of the data was a significant focus throughout this process, and with that came the need to find appropriate strategies to effectively display each dataset/data type. One piece of that was color schemes, but another was how to scale those color schemes. The main variables that would be displayed were Population Density, Per Capita Income, Number of Parks and Number of Gyms. Other variables that could be useful to show visually were the Percentage Breakdown of Race and the Median Age.

3.6.1 Population Density

Finding an appropriate scale to use for both cities (as they were comparable), came down to looking at the quantiles of Population Density for each city. After looking at the minimum and maximum values for both, and subsequently looking at where in the data the .05, .95 as well as the .25, .5, .75 quantiles were located, it was decided to make the scales start at 0 and cap between the .90 and the .95 quantiles at 5,000 (people/sq km) so as not to overshadow the distribution with the few areas that were extremely dense. The color used was Red.

3.6.2 Per Capita Income

Similar to that of the Population Density, the Per Capita Income scales were created after looking at various quantiles of each city. The scale decided upon was from 0 to 100000, which similarly fell around the .95 quartile range. The color used was Green.

3.6.3 Combining Population Density and Per Capita Income

In addition to the option for a side-by-side comparison with multiple maps, the ability to overlay both the Population Density and the Per Capita Income color scales simultaneously was created. Setting the opacity lower for the colors, the overlay creates an interesting, blended view of the colors, giving an interesting view into how both extremes cluster in different areas.

3.6.4 Foursquare – Clusters and Points

Two separate views were created for each of the Foursquare Search results. Using an automatic clustering algorithm that solely clusters based on proximity depending on Zoom levels on the map, one view was created. As the viewer zooms in, the clusters separate, and do so every time the user increases the Zoom level, until the concentration of points is small enough that they separate into individual icons. This method was used separately on maps for Gyms (white icon with a red heart) and maps for Parks (green icon with a white tree) as a way to show the concentration of venues without cluttering the maps too much. With that in mind, however, showing the clutter of individual points was important as well. One set of maps gives the option to overlay all points that Foursquare returned as Gyms (yellow dots) and those returned as Parks (blue dots). This set of maps used a DataFrame that was scrubbed of duplicates by dropping duplicate values that had the same "id" value, which Foursquare uses as a unique identifier for venues. See **Appendix: Breakdown of Raw Data** for more on those DataFrames.

3.6.5 Other Data to Include

In order to keep the other data included in the maps, the decision was made to include, in one map for each city, a pop-up that listed the following whenever someone hovers over a tract:

> *'Tract Area Name', 'Tract Per Capita Income - 12 Months', 'Pop Density per sq km', 'Number of Parks', 'Number of Gyms', 'Tract Percentage White',*
> *'Tract Percentage Black or African American', 'Tract Percentage Asian',*
> *'Tract Percentage American Indian and Alaskan Native', 'Tract Percentage Other Race'*

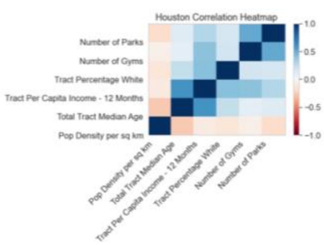
3.7 A Look into Correlation

As part of the exploratory process, looking at Correlation was a natural step. The Pearson method was used. There were no unrelated features that had a Pearson Coefficient close to 1. However, there were a couple variables that had Pearson Coefficients greater than 0.5, and their corresponding p-values were very low (well below .01), making this statistically significant. The correlation tables, along with correlation heatmaps generated from them, can be found in **Appendix: Correlation**. Although there was no significant correlation found between Parks and some of the other variables of interest, and same for Gyms in relation to other variables of interest such as Per Capita Income, there were some interesting correlations that came to light. This will be discussed in the results section.

4. Results

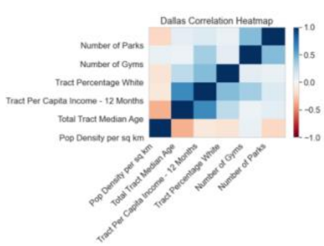
4.1 Correlation

The below heatmaps, along with their corresponding tables, were generated from the datasets.



	Pop Density per sq km	Total Tract Median Age	Tract Per Capita Income - 12 Months	Tract Percentage White	Number of Gyms	Number of Parks
Pop Density per sq km	1.000000	-0.257947	-0.093624	-0.120279	-0.057542	-0.182958
Total Tract Median Age	-0.257947	1.000000	0.600379	0.236048	0.094414	0.124181
Tract Per Capita Income - 12 Months	-0.093624	0.600379	1.000000	0.418977	0.413948	0.295231
Tract Percentage White	-0.120279	0.236048	0.418977	1.000000	0.189665	0.164675
Number of Gyms	-0.057542	0.094414	0.413948	0.189665	1.000000	0.514283
Number of Parks	-0.182958	0.124181	0.295231	0.164675	0.514283	1.000000

<u>Houston Correlation</u>



	Pop Density per sq km	Total Tract Median Age	Tract Per Capita Income - 12 Months	Tract Percentage White	Number of Gyms	Number of Parks
Pop Density per sq km	1.000000	-0.355919	-0.110906	-0.136441	0.068977	-0.209612
Total Tract Median Age	-0.355919	1.000000	0.634072	0.261805	0.059398	0.097585
Tract Per Capita Income - 12 Months	-0.110906	0.634072	1.000000	0.430823	0.340224	0.147975
Tract Percentage White	-0.136441	0.261805	0.430823	1.000000	0.109296	0.063511
Number of Gyms	0.068977	0.059398	0.340224	0.109296	1.000000	0.429802
Number of Parks	-0.209612	0.097585	0.147975	0.063511	0.429802	1.000000

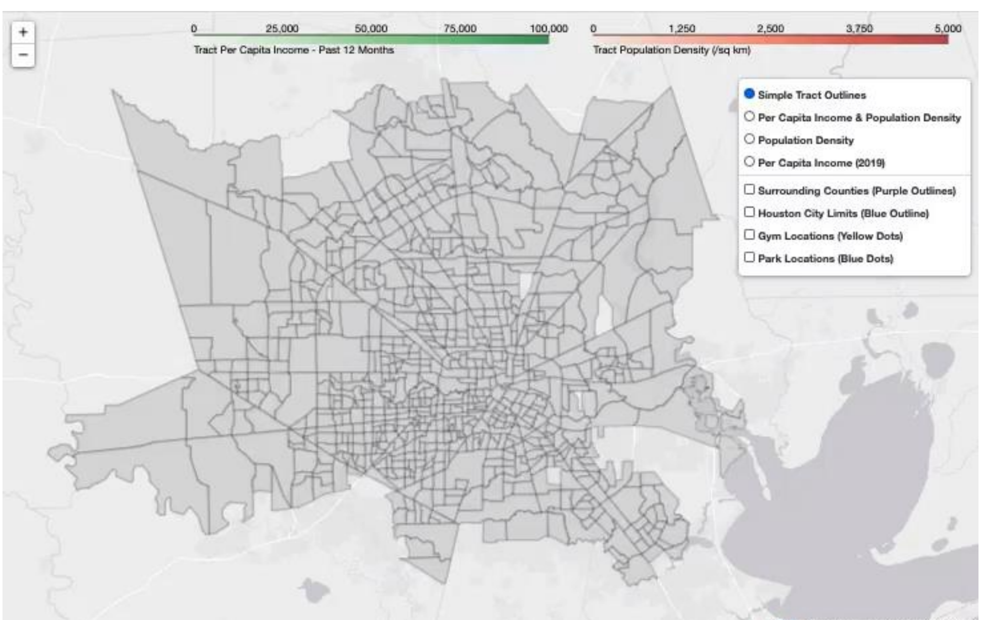
<u>Dallas Correlation</u>

The notable correlations that were shown from these tables were twofold – firstly, the correlation between parks and gyms themselves. For Houston, the correlation coefficient between Number of Parks and Number of Gyms showed a positive correlation of 0.5142383, statistically significant when viewed along with a p_value well under .001 (see p_value table in **Appendix: Correlation**). Although this number shows only minor correlation, it is an interesting look. The Dallas dataset showed correlation slightly below 0.5, so it was not mentioned. Another correlation that was revealed was that of Median Age and Per Capita Income. For Houston, the Pearson Coefficient was 0.600379, with a p_value well under .001. Similarly, for Dallas, the Pearson Coefficient was 0.634072, with a p_value well under .001. As shown by the Heatmaps, correlation between many of the variables was extremely similar across the two cities, which was hypothesized due to the cities being so similar.

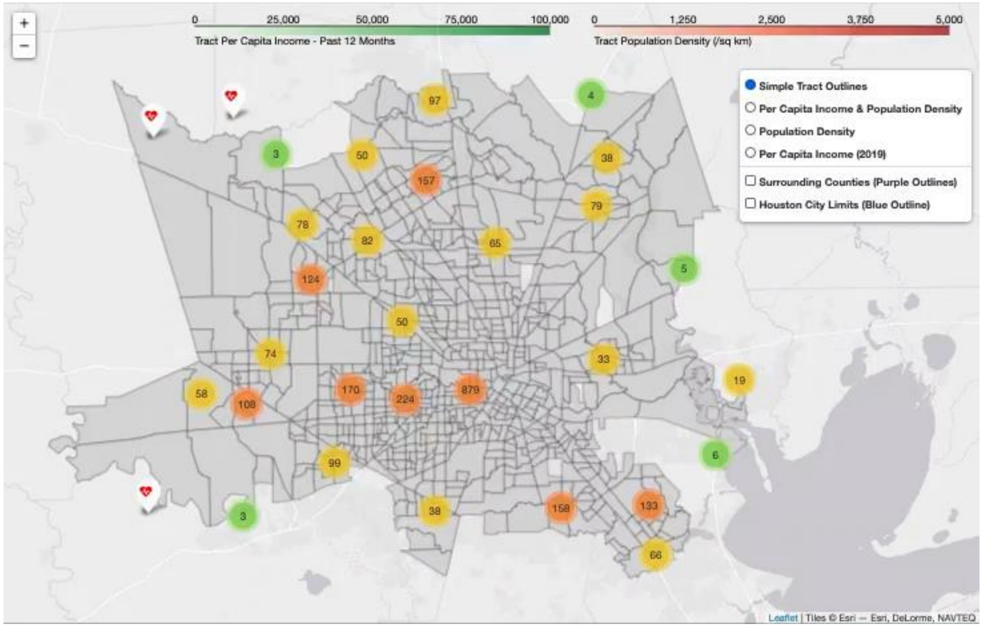
4.2 Maps

The maps generated from this project were interactive in nature and can be viewed on github. The videos below, and the stills in the appendix, are simply screen recordings and screenshots of the exported html maps from the notebook used to generate them, taken to give the reader an idea of what the maps have to offer.

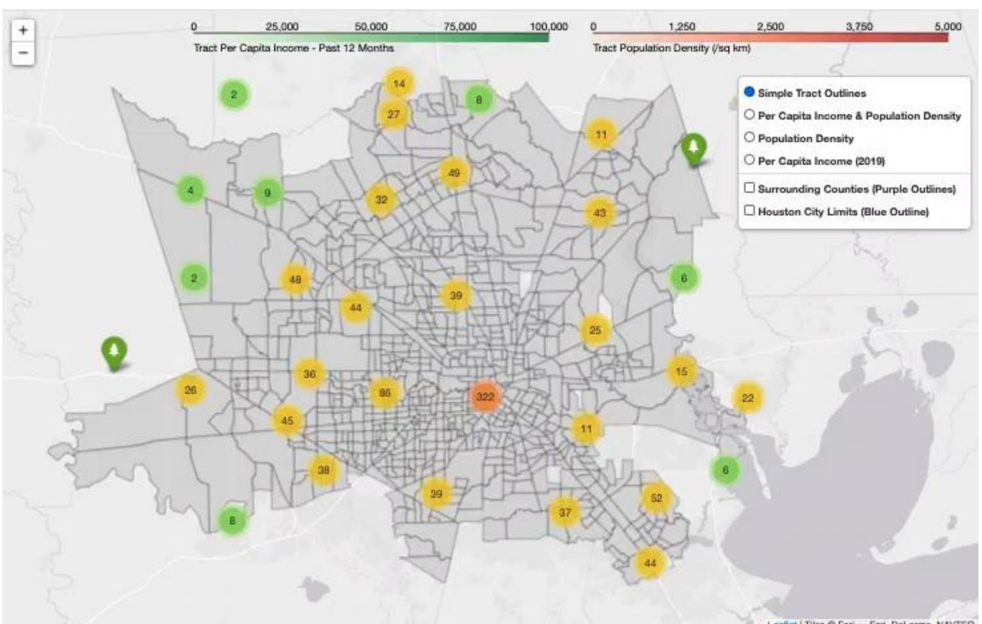
4.2.1 Houston Points



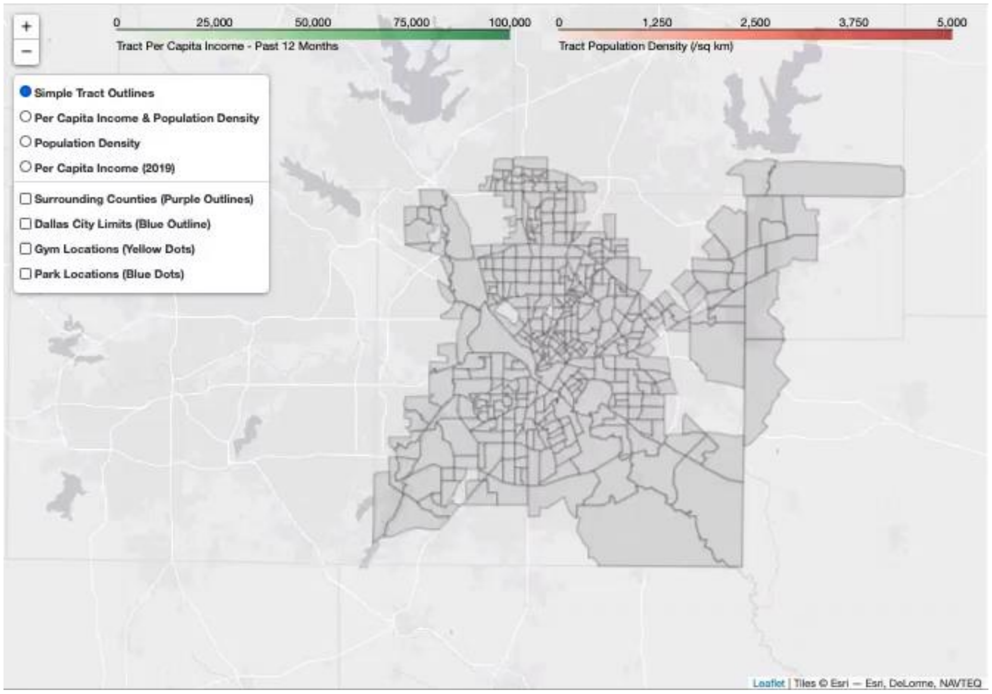
4.2.2 Houston Gym Clusters



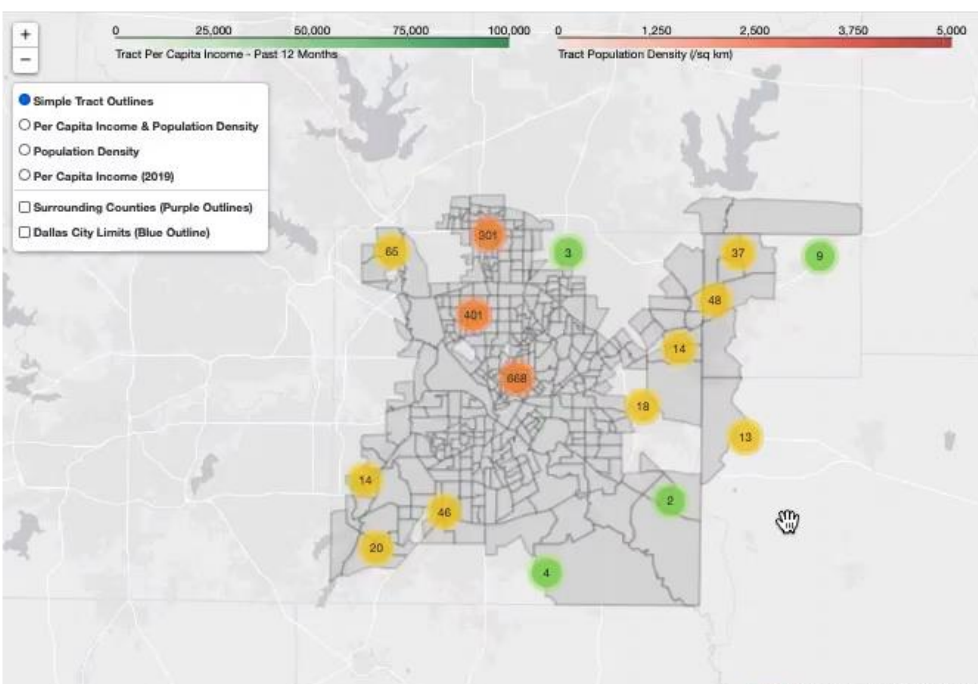
4.2.3 Houston Park Clusters



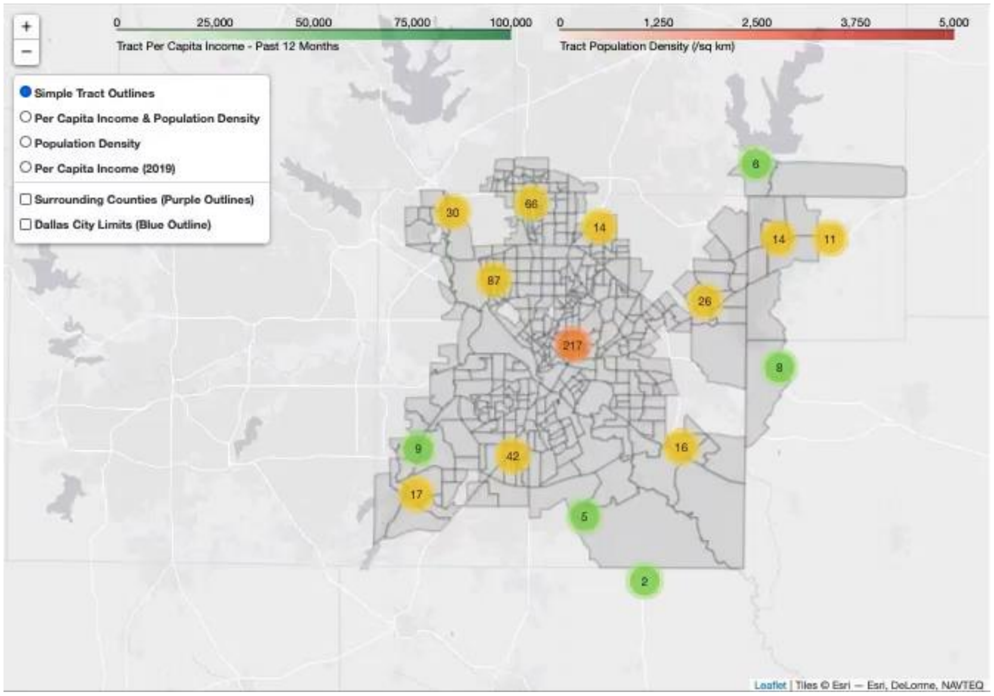
4.2.4 Dallas Points



4.2.5 Dallas Gym Clusters



4.2.6 Dallas Park Clusters



5. Discussion

5.1 Maps

Although the correlation tests did not provide many interesting potential relationships to explore, the visualizations of the data gave fascinating pictures to consider. The maps, visually, show high concentrations of Parks and Gyms in specific parts of the cities. In the future, perhaps the clustering function could be specified to not go solely off of distance from other venues/numbers of surrounding venues but could take a machine learning approach and cluster based on the combination of other categorical values.

5.2 Foursquare Data Problems

The Foursquare data component of this project was a required one. That being said, there are many possible issues to consider with the Foursquare data. First of all, not all venues that should actually be categorized as parks or gyms might be on the database. Similarly, there may be multiple entries for the same venues under different id's. Although duplicate values for id were dropped, that places faith in the platform for actually maintaining the uniqueness of that category. Finally, there could be false venues that were created by Foursquare users. With the data on hand, the analysis was performed. In the future, other APIs could be used to crosscheck or append the Foursquare data.

6. Conclusion

This analysis could provide insight on many levels to many different groups. The types of data shown together on the maps could be utilized by potential investors in gyms, city planners, even health workers. The multifaceted visualizations offer the ability for creativity in terms of the possibilities for usage, but also provide a unique look at the cities themselves.

6.1 Future Ideas for Consideration

6.1.1 Time Series

The most obvious addition to this project/analysis would be a time series component. At the moment, while there is a lot of data, it is hard to gather many meaningful conclusions given that it is all across one time period. Adding a time series aspect to this data could potentially explore the movement/development of clusters of venues matching up with the clustering of other variables.

6.1.2 Unemployment Numbers

Unemployment rates, while potentially related heavily to income numbers, could add another level to the data looked at in this project. These numbers would provide more meaning across different expanses of time, so this may work in tandem with time series.

6.1.3 Trending Locations

Sourcing trending location data over long periods of time could be another way to gather data pertaining to park/gym visitation. Although this would require a long-term dedication to the exploratory process, the data cached (if the API allows it), could potentially hold tremendous investigative value. This could feasibly be done using a number of APIs, including Foursquare, Google, and others.

7. Appendix

7.1 Appendix: NHGIS Data – Table Names and Source Datasets

7.1.1 Median Age by Sex

- 2015_2019_ACS5a
- 2019 American Community Survey: 5-Year Data [2015-2019, Block Groups & Larger Areas]

7.1.2 Total Population

- 2015_2019_ACS5a
- 2019 American Community Survey: 5-Year Data [2015-2019, Block Groups & Larger Areas]

7.1.3 Race

- 2015_2019_ACS5a
- 2019 American Community Survey: 5-Year Data [2015-2019, Block Groups & Larger Areas]

7.1.4 Per Capita Income in the Past 12 Months (in 2019 Inflation-Adjusted Dollars)

- 2015_2019_ACS5a
- 2019 American Community Survey: 5-Year Data [2015-2019, Block Groups & Larger Areas]

7.2 Appendix: Breakdown of Raw Data

7.2.1 GIS Data

```
<class 'geopandas.geodataframe.GeoDataFrame'>
RangeIndex: 29574 entries, 0 to 29573
Data columns (total 20 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   GISJOIN     29574 non-null  object
 1   STATEFP     29574 non-null  object
 2   PLACEFP     29574 non-null  object
 3   PLACENS     29574 non-null  object
 4   GEOID       29574 non-null  object
 5   NAME        29574 non-null  object
 6   NAMELSAD    29574 non-null  object
 7   LSAD        29574 non-null  object
 8   CLASSFP     29574 non-null  object
 9   PCICBSA     29574 non-null  object
 10  PCINECTA    29574 non-null  object
 11  MTFCC       29574 non-null  object
 12  FUNCSTAT    29574 non-null  object
 13  ALAND       29574 non-null  float64
 14  AWATER      29574 non-null  float64
 15  INTPTLAT    29574 non-null  object
 16  INTPTLON    29574 non-null  object
 17  Shape_Leng  29574 non-null  float64
 18  Shape_Area  29574 non-null  float64
 19  geometry    29574 non-null  geometry
dtypes: float64(4), geometry(1), object(15)
memory usage: 4.5+ MB
```

Place Level

```
<class 'geopandas.geodataframe.GeoDataFrame'>
RangeIndex: 3220 entries, 0 to 3219
Data columns (total 21 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   GISJOIN     3220 non-null   object
 1   STATEFP     3220 non-null   object
 2   COUNTYFP    3220 non-null   object
 3   COUNTYNS    3220 non-null   object
 4   GEOID       3220 non-null   object
 5   NAME        3220 non-null   object
 6   NAMELSAD    3220 non-null   object
 7   LSAD        3220 non-null   object
 8   CLASSFP     3220 non-null   object
 9   MTFCC       3220 non-null   object
 10  CSAFP       1255 non-null   object
 11  CBSAFP      1915 non-null   object
 12  METDIVFP    110 non-null    object
 13  FUNCSTAT    3220 non-null   object
 14  ALAND       3220 non-null   float64
 15  AWATER      3220 non-null   float64
 16  INTPTLAT    3220 non-null   object
 17  INTPTLON    3220 non-null   object
 18  Shape_Leng  3220 non-null   float64
 19  Shape_Area  3220 non-null   float64
 20  geometry    3220 non-null   geometry
dtypes: float64(4), geometry(1), object(16)
memory usage: 528.4+ KB
```

County Level

```
<class 'geopandas.geodataframe.GeoDataFrame'>
RangeIndex: 73666 entries, 0 to 73665
Data columns (total 16 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   GISJOIN     73666 non-null  object
 1   STATEFP     73666 non-null  object
 2   COUNTYFP    73666 non-null  object
 3   TRACTCE     73666 non-null  object
 4   GEOID       73666 non-null  object
 5   NAME        73666 non-null  object
 6   NAMELSAD    73666 non-null  object
 7   MTFCC       73666 non-null  object
 8   FUNCSTAT    73666 non-null  object
 9   ALAND       73666 non-null  float64
 10  AWATER      73666 non-null  float64
 11  INTPTLAT    73666 non-null  object
 12  INTPTLON    73666 non-null  object
 13  Shape_Leng  73666 non-null  float64
 14  Shape_Area  73666 non-null  float64
 15  geometry    73666 non-null  geometry
dtypes: float64(4), geometry(1), object(11)
memory usage: 9.0+ MB
```

Tract Level

7.2.2 Census Data

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1746 entries, 0 to 1745
Data columns (total 74 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   gisjoin    1746 non-null   object
 1   YEAR       1746 non-null   object
 2   stusab     1746 non-null   object
 3   regiona    0 non-null      object
 4   divisiona  0 non-null      object
 5   state      1746 non-null   object
 6   statea     1746 non-null   int64
 7   countya    0 non-null      object
 8   cousuba    0 non-null      object
 9   place      1746 non-null   object
 10  placea     1746 non-null   int64
 11  tracta     0 non-null      object
 12  blkgrpa    0 non-null      object
 13  concita    0 non-null      object
 14  aianhha    0 non-null      object
 15  res_onlya  0 non-null      object
 16  trusta     0 non-null      object
 17  aihhtli    0 non-null      object
 18  aits       0 non-null      object
 19  anrca      0 non-null      object
 20  cbsaa      0 non-null      object
 21  csaa       0 non-null      object
 22  metdiva    0 non-null      object
 23  memi       0 non-null      object
 24  nectaa     0 non-null      object
 25  cnectaa    0 non-null      object
 26  nectadiva  0 non-null      object
 27  uaa        0 non-null      object
 28  cdcurra    0 non-null      object
 29  sldua      0 non-null      object
 30  sldla      0 non-null      object
 31  zcta5a     0 non-null      object
 32  submcda    0 non-null      object
 33  sdelma     0 non-null      object
 34  sdseca     0 non-null      object
 35  sdunia     0 non-null      object
 36  ur         0 non-null      object
 37  pci        0 non-null      object
 38  puma5a     0 non-null      object
 39  geoid      1746 non-null   object
 40  bttra      0 non-null      object
 41  btbga      0 non-null      object
 42  name_e     1746 non-null   object
 43  alt1e001   1678 non-null   object
 44  alt1e002   1638 non-null   object
 45  alt1e003   1634 non-null   object
 46  alube001   1746 non-null   int64
 47  aluce001   1746 non-null   int64
 48  aluce002   1746 non-null   int64
 49  aluce003   1746 non-null   int64
 50  aluce004   1746 non-null   int64
 51  aluce005   1746 non-null   int64
 52  aluce006   1746 non-null   int64
 53  aluce007   1746 non-null   int64
 54  aluce008   1746 non-null   int64
 55  aluce009   1746 non-null   int64
 56  aluce010   1746 non-null   int64
 57  alx5e001   1655 non-null   float64
 58  name_m     1746 non-null   object
 59  alt1m001   1677 non-null   object
 60  alt1m002   1638 non-null   object
 61  alt1m003   1634 non-null   object
 62  alubm001   1746 non-null   int64
 63  alucm001   1746 non-null   int64
 64  alucm002   1746 non-null   int64
 65  alucm003   1746 non-null   int64
 66  alucm004   1746 non-null   int64
 67  alucm005   1746 non-null   int64
 68  alucm006   1746 non-null   int64
 69  alucm007   1746 non-null   int64
 70  alucm008   1746 non-null   int64
 71  alucm009   1746 non-null   int64
 72  alucm010   1746 non-null   int64
 73  alx5m001   1655 non-null   float64
dtypes: float64(2), int64(24), object(48)
memory usage: 1009.5+ KB
```

Place Level

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 254 entries, 0 to 253
Data columns (total 74 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   gisjoin    254 non-null    object
 1   YEAR       254 non-null    object
 2   stusab     254 non-null    object
 3   regiona    0 non-null      object
 4   divisiona  0 non-null      object
 5   state      254 non-null    object
 6   statea     254 non-null    int64
 7   county     254 non-null    object
 8   countya    254 non-null    int64
 9   cousuba    0 non-null      object
 10  placea     0 non-null      object
 11  tracta     0 non-null      object
 12  blkgrpa    0 non-null      object
 13  concita    0 non-null      object
 14  aianhha    0 non-null      object
 15  res_onlya  0 non-null      object
 16  trusta     0 non-null      object
 17  aihhtli    0 non-null      object
 18  aits       0 non-null      object
 19  anrca      0 non-null      object
 20  cbsaa      0 non-null      object
 21  csaa       0 non-null      object
 22  metdiva    0 non-null      object
 23  memi       0 non-null      object
 24  nectaa     0 non-null      object
 25  cnectaa    0 non-null      object
 26  nectadiva  0 non-null      object
 27  uaa        0 non-null      object
 28  cdcurra    0 non-null      object
 29  sldua      0 non-null      object
 30  sldla      0 non-null      object
 31  zcta5a     0 non-null      object
 32  submcda    0 non-null      object
 33  sdelma     0 non-null      object
 34  sdseca     0 non-null      object
 35  sdunia     0 non-null      object
 36  ur         0 non-null      object
 37  pci        0 non-null      object
 38  puma5a     0 non-null      object
 39  geoid      254 non-null    object
 40  bttra      0 non-null      object
 41  btbga      0 non-null      object
 42  name_e     254 non-null    object
 43  alt1e001   254 non-null    object
 44  alt1e002   254 non-null    object
 45  alt1e003   254 non-null    object
 46  alube001   254 non-null    int64
 47  aluce001   254 non-null    int64
 48  aluce002   254 non-null    int64
 49  aluce003   254 non-null    int64
 50  aluce004   254 non-null    int64
 51  aluce005   254 non-null    int64
 52  aluce006   254 non-null    int64
 53  aluce007   254 non-null    int64
 54  aluce008   254 non-null    int64
 55  aluce009   254 non-null    int64
 56  aluce010   254 non-null    int64
 57  alx5e001   254 non-null    int64
 58  name_m     254 non-null    object
 59  alt1m001   254 non-null    object
 60  alt1m002   254 non-null    object
 61  alt1m003   254 non-null    object
 62  alubm001   254 non-null    int64
 63  alucm001   254 non-null    int64
 64  alucm002   254 non-null    int64
 65  alucm003   254 non-null    int64
 66  alucm004   254 non-null    int64
 67  alucm005   254 non-null    int64
 68  alucm006   254 non-null    int64
 69  alucm007   254 non-null    int64
 70  alucm008   254 non-null    int64
 71  alucm009   254 non-null    int64
 72  alucm010   254 non-null    int64
 73  alx5m001   254 non-null    int64
dtypes: int64(26), object(48)
memory usage: 147.0+ KB
```

County Level

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5265 entries, 0 to 5264
Data columns (total 74 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   gisjoin    5265 non-null   object
 1   YEAR       5265 non-null   object
 2   stusab     5265 non-null   object
 3   regiona    0 non-null      object
 4   divisiona  0 non-null      object
 5   state      5265 non-null   object
 6   statea     5265 non-null   int64
 7   county     5265 non-null   object
 8   countya    5265 non-null   int64
 9   cousuba    0 non-null      object
 10  placea     0 non-null      object
 11  tracta     5265 non-null   int64
 12  blkgrpa    0 non-null      object
 13  concita    0 non-null      object
 14  aianhha    0 non-null      object
 15  res_onlya  0 non-null      object
 16  trusta     0 non-null      object
 17  aihhtli    0 non-null      object
 18  aits       0 non-null      object
 19  anrca      0 non-null      object
 20  cbsaa      0 non-null      object
 21  csaa       0 non-null      object
 22  metdiva    0 non-null      object
 23  memi       0 non-null      object
 24  nectaa     0 non-null      object
 25  cnectaa    0 non-null      object
 26  nectadiva  0 non-null      object
 27  uaa        0 non-null      object
 28  cdcurra    0 non-null      object
 29  sldua      0 non-null      object
 30  sldla      0 non-null      object
 31  zcta5a     0 non-null      object
 32  submcda    0 non-null      object
 33  sdelma     0 non-null      object
 34  sdseca     0 non-null      object
 35  sdunia     0 non-null      object
 36  ur         0 non-null      object
 37  pci        0 non-null      object
 38  puma5a     0 non-null      object
 39  geoid      5265 non-null   object
 40  bttra      0 non-null      object
 41  btbga      0 non-null      object
 42  name_e     5265 non-null   object
 43  alt1e001   5227 non-null   object
 44  alt1e002   5224 non-null   object
 45  alt1e003   5216 non-null   object
 46  alube001   5265 non-null   int64
 47  aluce001   5265 non-null   int64
 48  aluce002   5265 non-null   int64
 49  aluce003   5265 non-null   int64
 50  aluce004   5265 non-null   int64
 51  aluce005   5265 non-null   int64
 52  aluce006   5265 non-null   int64
 53  aluce007   5265 non-null   int64
 54  aluce008   5265 non-null   int64
 55  aluce009   5265 non-null   int64
 56  aluce010   5265 non-null   int64
 57  alx5e001   5224 non-null   float64
 58  name_m     5265 non-null   object
 59  alt1m001   5227 non-null   object
 60  alt1m002   5224 non-null   object
 61  alt1m003   5216 non-null   object
 62  alubm001   5265 non-null   int64
 63  alucm001   5265 non-null   int64
 64  alucm002   5265 non-null   int64
 65  alucm003   5265 non-null   int64
 66  alucm004   5265 non-null   int64
 67  alucm005   5265 non-null   int64
 68  alucm006   5265 non-null   int64
 69  alucm007   5265 non-null   int64
 70  alucm008   5265 non-null   int64
 71  alucm009   5265 non-null   int64
 72  alucm010   5265 non-null   int64
 73  alx5m001   5224 non-null   float64
dtypes: float64(2), int64(25), object(47)
memory usage: 3.0+ MB
```

Tract Level

7.2.3 Foursquare Data

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1152 entries, 0 to 2836
Data columns (total 16 columns):
 #   Column                     Non-Null Count  Dtype  
---  ------                     --------------  -----  
 0   name                       1152 non-null   object 
 1   categories                 1152 non-null   object 
 2   location.lat               1152 non-null   float64
 3   location.lng               1152 non-null   float64
 4   location.labeledLatLngs    1152 non-null   object 
 5   location.distance          1152 non-null   int64  
 6   location.cc                1152 non-null   object 
 7   location.state             1152 non-null   object 
 8   location.country           1152 non-null   object 
 9   location.formattedAddress  1152 non-null   object 
 10  location.city              1043 non-null   object 
 11  location.address           438 non-null    object 
 12  location.postalCode        759 non-null    object 
 13  location.crossStreet       194 non-null    object 
 14  location.neighborhood      5 non-null      object 
 15  id                         1152 non-null   object 
dtypes: float64(2), int64(1), object(13)
memory usage: 153.0+ KB
```

Houston Parks

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2903 entries, 0 to 6897
Data columns (total 16 columns):
 #   Column                     Non-Null Count  Dtype  
---  ------                     --------------  -----  
 0   name                       2903 non-null   object 
 1   categories                 2903 non-null   object 
 2   location.address           1717 non-null   object 
 3   location.lat               2903 non-null   float64
 4   location.lng               2903 non-null   float64
 5   location.labeledLatLngs    2902 non-null   object 
 6   location.distance          2903 non-null   int64  
 7   location.postalCode        2287 non-null   object 
 8   location.cc                2902 non-null   object 
 9   location.city              2722 non-null   object 
 10  location.state             2903 non-null   object 
 11  location.country           2902 non-null   object 
 12  location.formattedAddress  2903 non-null   object 
 13  location.crossStreet       508 non-null    object 
 14  location.neighborhood      17 non-null     object 
 15  id                         2903 non-null   object 
dtypes: float64(2), int64(1), object(13)
memory usage: 385.6+ KB
```

Houston Gyms

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 570 entries, 0 to 1305
Data columns (total 16 columns):
 #   Column                     Non-Null Count  Dtype  
---  ------                     --------------  -----  
 0   name                       570 non-null    object 
 1   categories                 570 non-null    object 
 2   location.address           254 non-null    object 
 3   location.lat               570 non-null    float64
 4   location.lng               570 non-null    float64
 5   location.labeledLatLngs    570 non-null    object 
 6   location.distance          570 non-null    int64  
 7   location.postalCode        379 non-null    object 
 8   location.cc                570 non-null    object 
 9   location.city              523 non-null    object 
 10  location.state             570 non-null    object 
 11  location.country           570 non-null    object 
 12  location.formattedAddress  570 non-null    object 
 13  location.crossStreet       147 non-null    object 
 14  location.neighborhood      5 non-null      object 
 15  id                         570 non-null    object 
dtypes: float64(2), int64(1), object(13)
memory usage: 75.7+ KB
```

Dallas Parks

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1663 entries, 0 to 3494
Data columns (total 16 columns):
 #   Column                     Non-Null Count  Dtype  
---  ------                     --------------  -----  
 0   name                       1663 non-null   object 
 1   categories                 1663 non-null   object 
 2   location.lat               1663 non-null   float64
 3   location.lng               1663 non-null   float64
 4   location.labeledLatLngs    1663 non-null   object 
 5   location.distance          1663 non-null   int64  
 6   location.cc                1663 non-null   object 
 7   location.state             1663 non-null   object 
 8   location.country           1663 non-null   object 
 9   location.formattedAddress  1663 non-null   object 
 10  location.postalCode        1338 non-null   object 
 11  location.city              1603 non-null   object 
 12  location.address           1032 non-null   object 
 13  location.crossStreet       338 non-null    object 
 14  location.neighborhood      7 non-null      object 
 15  id                         1663 non-null   object 
dtypes: float64(2), int64(1), object(13)
memory usage: 220.9+ KB
```

Dallas Gyms

7.3 Appendix: Breakdown of Cleaned Data

7.3.1 GIS Data

```
<class 'geopandas.geodataframe.GeoDataFrame'>
RangeIndex: 29574 entries, 0 to 29573
Data columns (total 10 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   GISJOIN            29574 non-null  object
 1   State FIPS Code    29574 non-null  object
 2   Place Name         29574 non-null  object
 3   Land Area          29574 non-null  float64
 4   Water Area         29574 non-null  float64
 5   Place Center Lat   29574 non-null  object
 6   Place Center Long  29574 non-null  object
 7   Place Perimeter    29574 non-null  float64
 8   Place Area         29574 non-null  float64
 9   geometry           29574 non-null  geometry
dtypes: float64(4), geometry(1), object(5)
memory usage: 2.3+ MB
```

Place Level

```
<class 'geopandas.geodataframe.GeoDataFrame'>
RangeIndex: 3220 entries, 0 to 3219
Data columns (total 9 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   GISJOIN             3220 non-null   object
 1   County FIPS Code    3220 non-null   object
 2   Land Area           3220 non-null   float64
 3   Water Area          3220 non-null   float64
 4   County Center Lat   3220 non-null   object
 5   County Center Long  3220 non-null   object
 6   County Perimeter    3220 non-null   float64
 7   County Area         3220 non-null   float64
 8   geometry            3220 non-null   geometry
dtypes: float64(4), geometry(1), object(4)
memory usage: 226.5+ KB
```

County Level

```
<class 'geopandas.geodataframe.GeoDataFrame'>
RangeIndex: 73666 entries, 0 to 73665
Data columns (total 8 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   GISJOIN            73666 non-null  object
 1   Land Area          73666 non-null  float64
 2   Water Area         73666 non-null  float64
 3   Tract Center Lat   73666 non-null  object
 4   Tract Center Long  73666 non-null  object
 5   Tract Perimeter    73666 non-null  float64
 6   Tract Area         73666 non-null  float64
 7   geometry           73666 non-null  geometry
dtypes: float64(4), geometry(1), object(3)
memory usage: 4.5+ MB
```

Tract Level

7.3.2 Census Data

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1746 entries, 0 to 1745
Data columns (total 21 columns):
 #   Column                                       Non-Null Count  Dtype
---  ------                                       --------------  -----
 0   GISJOIN                                      1746 non-null   object
 1   State FIPS Code                              1746 non-null   int64
 2   Place Area Name                              1746 non-null   object
 3   Total Place Pop                              1746 non-null   int64
 4   Total Place Median Age                       1678 non-null   float32
 5   Place Median Age M                           1638 non-null   float32
 6   Place Median Age F                           1634 non-null   float32
 7   Place Pop White                              1746 non-null   int64
 8   Place Pop Black or African American          1746 non-null   int64
 9   Place Pop American Indian and Alaska Native  1746 non-null   int64
 10  Place Pop Asian                              1746 non-null   int64
 11  Place Per Capita Income - 12 Months          1655 non-null   float64
 12  Total Place Pop ME                           1746 non-null   int64
 13  Total Place Median Age ME                    1677 non-null   float32
 14  Place Median Age ME M                        1638 non-null   float32
 15  Place Median Age ME F                        1634 non-null   float32
 16  Place Pop ME White                           1746 non-null   int64
 17  Place Pop ME Black or African American       1746 non-null   int64
 18  Place Pop ME American Indian and Alaska Native 1746 non-null   int64
 19  Place Pop ME Asian                           1746 non-null   int64
 20  Place Per Capita Income ME - 12 Months       1655 non-null   float64
dtypes: float32(6), float64(2), int64(11), object(2)
memory usage: 245.7+ KB
```

Place Level

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 254 entries, 0 to 253
Data columns (total 23 columns):
 #   Column                                        Non-Null Count  Dtype
---  ------                                        --------------  -----
 0   GISJOIN                                       254 non-null    object
 1   State FIPS Code                               254 non-null    int64
 2   County Name                                   254 non-null    object
 3   County FIPS Code                              254 non-null    int64
 4   County Area Name                              254 non-null    object
 5   Total County Pop                              254 non-null    int64
 6   Total County Median Age                       254 non-null    float32
 7   County Median Age M                           254 non-null    float32
 8   County Median Age F                           254 non-null    float32
 9   County Pop White                              254 non-null    int64
 10  County Pop Black or African American          254 non-null    int64
 11  County Pop American Indian and Alaska Native  254 non-null    int64
 12  County Pop Asian                              254 non-null    int64
 13  County Per Capita Income - 12 Months          254 non-null    int64
 14  Total County Pop ME                           254 non-null    int64
 15  Total County Median Age ME                    254 non-null    float32
 16  County Median Age ME M                        254 non-null    float32
 17  County Median Age ME F                        254 non-null    float32
 18  County Pop ME White                           254 non-null    int64
 19  County Pop ME Black or African American       254 non-null    int64
 20  County Pop ME American Indian and Alaska Native 254 non-null    int64
 21  County Pop ME Asian                           254 non-null    int64
 22  County Per Capita Income ME - 12 Months       254 non-null    int64
dtypes: float32(6), int64(14), object(3)
memory usage: 39.8+ KB
```

County Level

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5265 entries, 0 to 5264
Data columns (total 24 columns):
 #   Column                                       Non-Null Count  Dtype
---  ------                                       --------------  -----
 0   GISJOIN                                      5265 non-null   object
 1   State FIPS Code                              5265 non-null   int64
 2   County Name                                  5265 non-null   object
 3   County FIPS Code                             5265 non-null   int64
 4   Tract FIPS Code                              5265 non-null   int64
 5   Tract Area Name                              5265 non-null   object
 6   Total Tract Pop                              5265 non-null   int64
 7   Total Tract Median Age                       5227 non-null   float32
 8   Tract Median Age M                           5224 non-null   float32
 9   Tract Median Age F                           5216 non-null   float32
 10  Tract Pop White                              5265 non-null   int64
 11  Tract Pop Black or African American          5265 non-null   int64
 12  Tract Pop American Indian and Alaska Native  5265 non-null   int64
 13  Tract Pop Asian                              5265 non-null   int64
 14  Tract Per Capita Income - 12 Months          5224 non-null   float64
 15  Total Tract Pop ME                           5265 non-null   int64
 16  Total Tract Median Age ME                    5227 non-null   float32
 17  Tract Median Age ME M                        5224 non-null   float32
 18  Tract Median Age ME F                        5216 non-null   float32
 19  Tract Pop ME White                           5265 non-null   int64
 20  Tract Pop ME Black or African American       5265 non-null   int64
 21  Tract Pop ME American Indian and Alaska Native 5265 non-null   int64
 22  Tract Pop ME Asian                           5265 non-null   int64
 23  Tract Per Capita Income ME - 12 Months       5224 non-null   float64
dtypes: float32(6), float64(2), int64(13), object(3)
memory usage: 863.9+ KB
```

Tract Level

7.3.3 Foursquare Data

```
<class 'geopandas.geodataframe.GeoDataFrame'>
Int64Index: 1152 entries, 0 to 1150
Data columns (total 4 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   id        1152 non-null   object
 1   name      1152 non-null   object
 2   geometry  1152 non-null   geometry
 3   GISJOIN   1152 non-null   object
dtypes: geometry(1), object(3)
memory usage: 45.0+ KB
```

Houston Parks

```
<class 'geopandas.geodataframe.GeoDataFrame'>
Int64Index: 2903 entries, 0 to 2902
Data columns (total 4 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   id        2903 non-null   object
 1   name      2903 non-null   object
 2   geometry  2903 non-null   geometry
 3   GISJOIN   2903 non-null   object
dtypes: geometry(1), object(3)
memory usage: 113.4+ KB
```

Houston Gyms

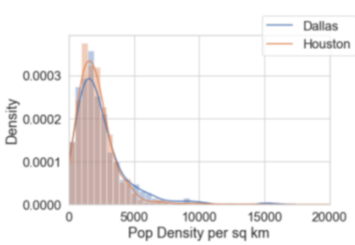
```
<class 'geopandas.geodataframe.GeoDataFrame'>
Int64Index: 570 entries, 0 to 569
Data columns (total 4 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   id        570 non-null    object
 1   name      570 non-null    object
 2   geometry  570 non-null    geometry
 3   GISJOIN   570 non-null    object
dtypes: geometry(1), object(3)
memory usage: 22.3+ KB
```

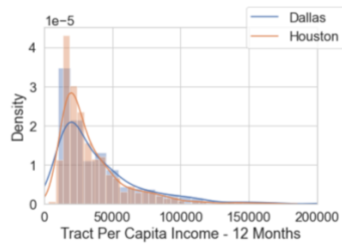
Dallas Parks

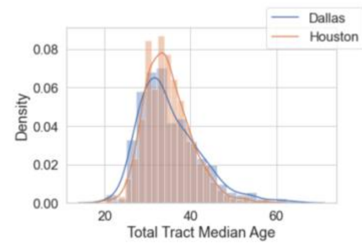
```
<class 'geopandas.geodataframe.GeoDataFrame'>
Int64Index: 1663 entries, 0 to 1662
Data columns (total 4 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   id        1663 non-null   object
 1   name      1663 non-null   object
 2   geometry  1663 non-null   geometry
 3   GISJOIN   1663 non-null   object
dtypes: geometry(1), object(3)
memory usage: 65.0+ KB
```

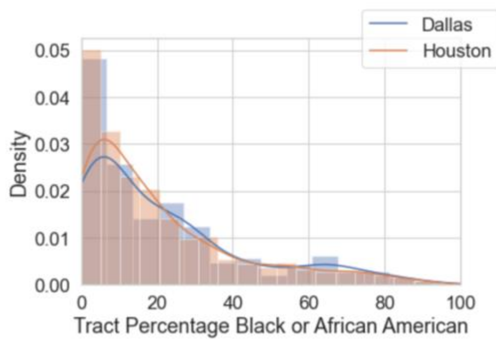
Dallas Gyms

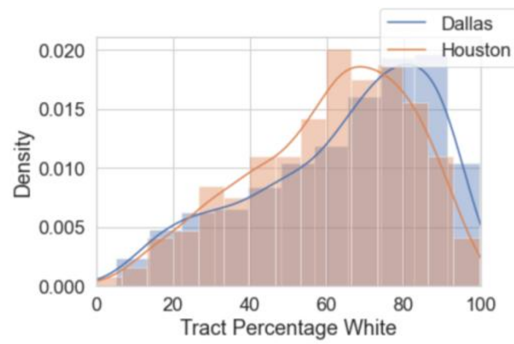
7.4 Appendix: Distribution Plots

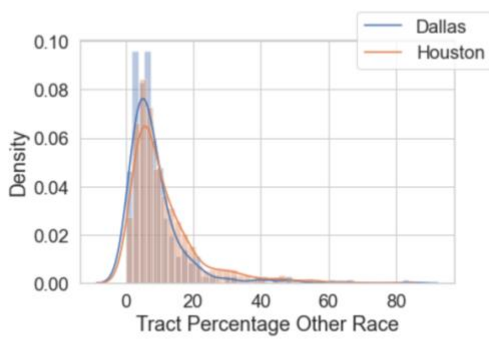


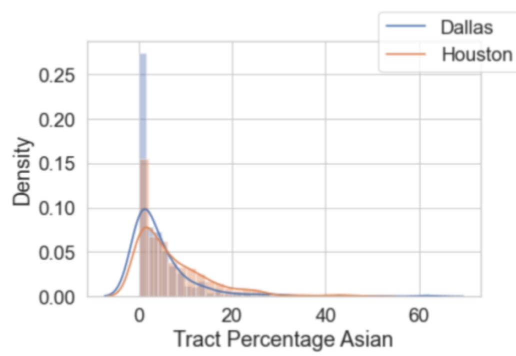






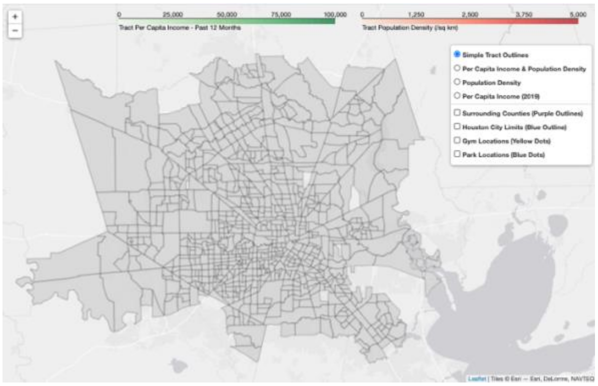


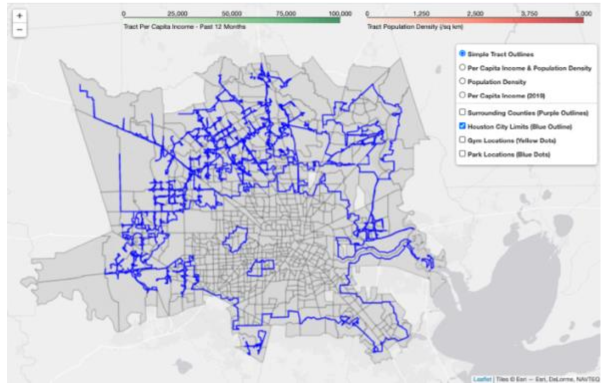


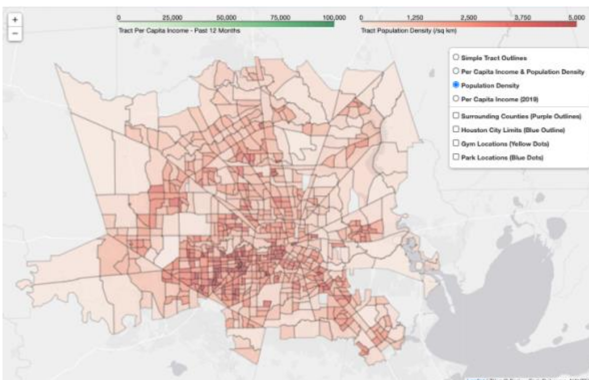


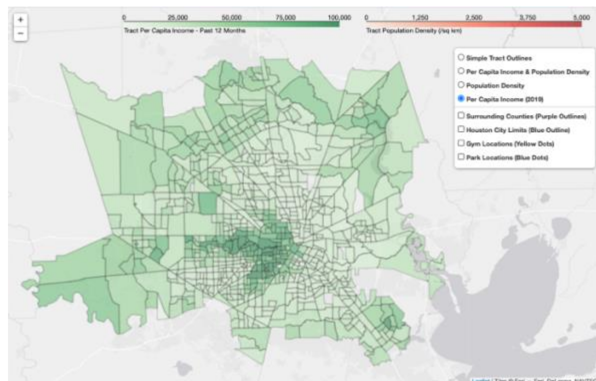
7.5 Appendix: Stills of Maps

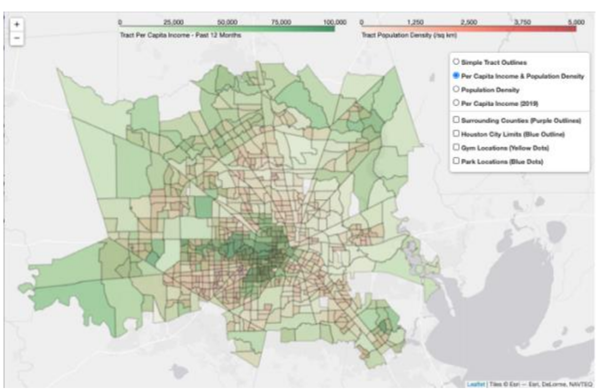
7.5.1 Houston

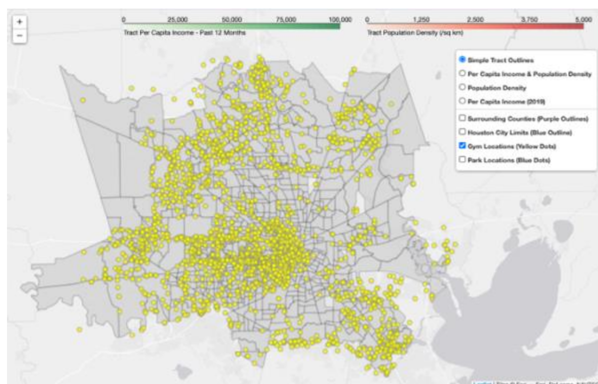


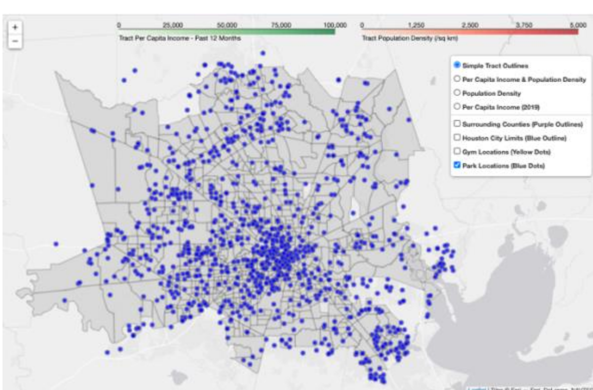


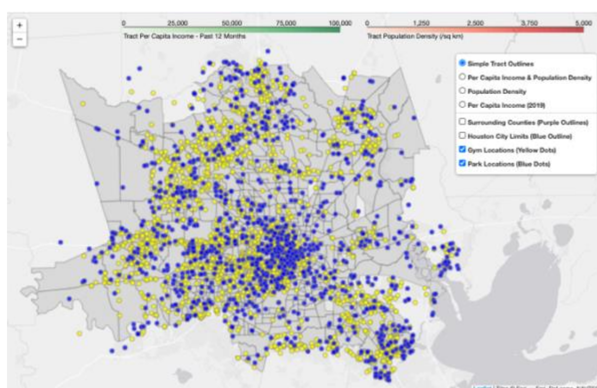




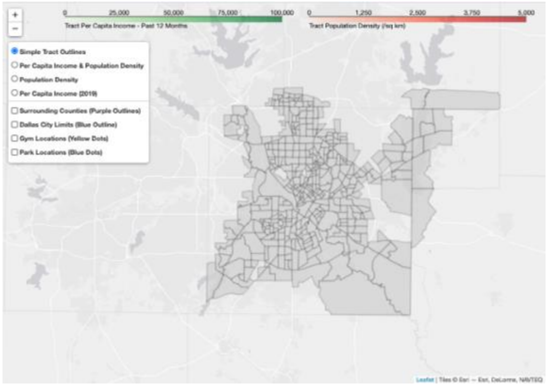


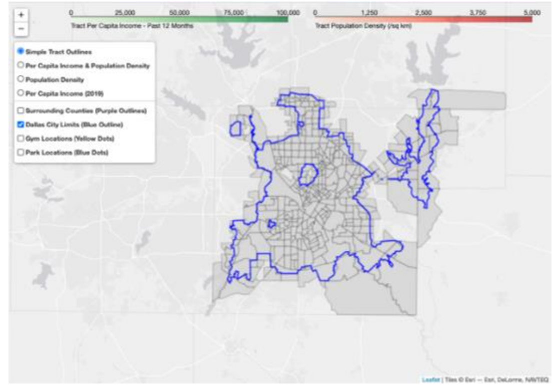


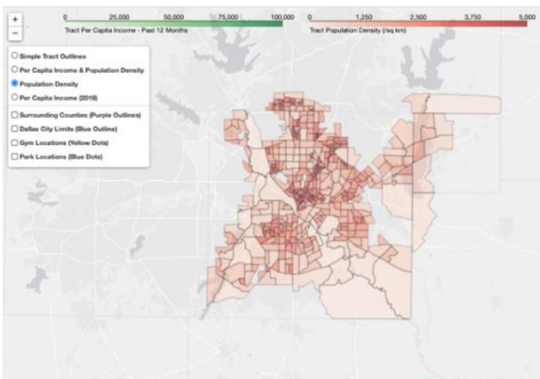


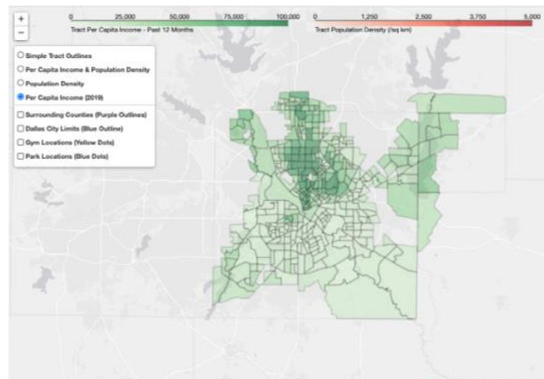


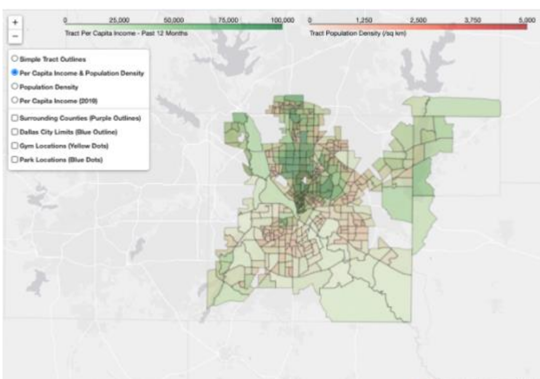
7.5.2 Dallas

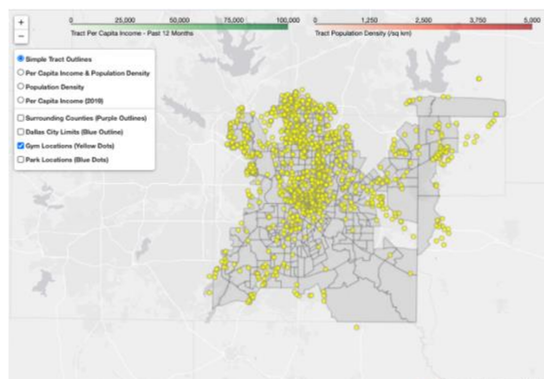


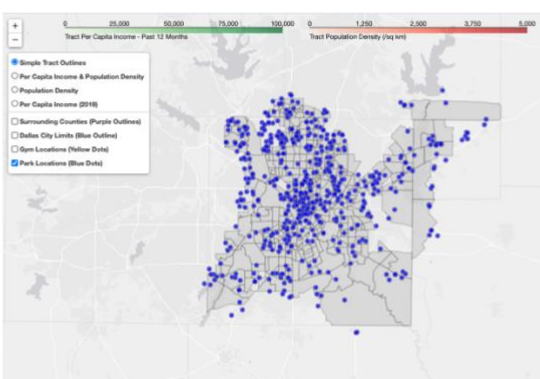


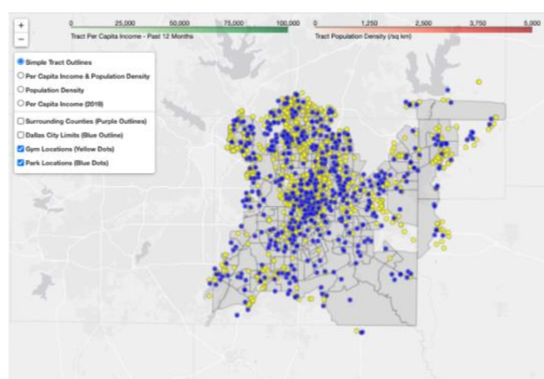






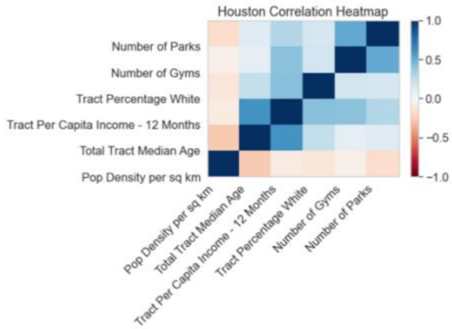


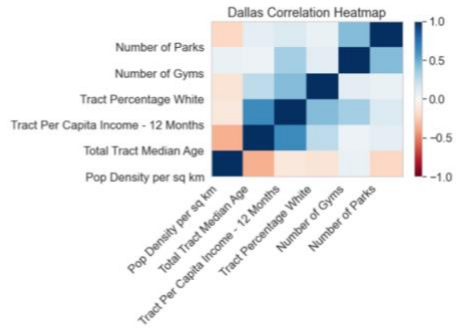




7.6 Appendix: Correlation

7.6.1 Correlation Heatmaps





7.6.2 Correlation Tables

<u>Houston (1. Pearson Correlation Coefficient, 2. P-Value)</u>

	Pop Density per sq km	Total Tract Median Age	Tract Per Capita Income - 12 Months	Tract Percentage White	Number of Gyms	Number of Parks
Pop Density per sq km	1.000000	-0.257947	-0.093624	-0.120279	-0.057542	-0.182958
Total Tract Median Age	-0.257947	1.000000	0.600379	0.236048	0.094414	0.124181
Tract Per Capita Income - 12 Months	-0.093624	0.600379	1.000000	0.418977	0.413948	0.295231
Tract Percentage White	-0.120279	0.236048	0.418977	1.000000	0.189665	0.164675
Number of Gyms	-0.057542	0.094414	0.413948	0.189665	1.000000	0.514283
Number of Parks	-0.182958	0.124181	0.295231	0.164675	0.514283	1.000000

	Pop Density per sq km	Total Tract Median Age	Tract Per Capita Income - 12 Months	Tract Percentage White	Number of Gyms	Number of Parks
Pop Density per sq km	1.000000e+00	4.453533e-13	9.618152e-03	8.646308e-04	1.120134e-01	3.546552e-07
Total Tract Median Age	4.453533e-13	1.000000e+00	5.227433e-76	3.910988e-11	9.022157e-03	5.814512e-04
Tract Per Capita Income - 12 Months	9.618152e-03	5.227433e-76	1.000000e+00	7.875489e-34	5.494561e-33	7.853204e-17
Tract Percentage White	8.646308e-04	3.910988e-11	7.875489e-34	1.000000e+00	1.278601e-07	4.750150e-06
Number of Gyms	1.120134e-01	9.022157e-03	5.494561e-33	1.278601e-07	1.000000e+00	8.296446e-53
Number of Parks	3.546552e-07	5.814512e-04	7.853204e-17	4.750150e-06	8.296446e-53	1.000000e+00

<u>Dallas (1. Pearson Correlation Coefficient, 2. P-Value)</u>

	Pop Density per sq km	Total Tract Median Age	Tract Per Capita Income - 12 Months	Tract Percentage White	Number of Gyms	Number of Parks
Pop Density per sq km	1.000000	-0.355919	-0.110906	-0.136441	0.068977	-0.209612
Total Tract Median Age	-0.355919	1.000000	0.634072	0.261805	0.059398	0.097585
Tract Per Capita Income - 12 Months	-0.110906	0.634072	1.000000	0.430823	0.340224	0.147975
Tract Percentage White	-0.136441	0.261805	0.430823	1.000000	0.109296	0.063511
Number of Gyms	0.068977	0.059398	0.340224	0.109296	1.000000	0.429802
Number of Parks	-0.209612	0.097585	0.147975	0.063511	0.429802	1.000000

	Pop Density per sq km	Total Tract Median Age	Tract Per Capita Income - 12 Months	Tract Percentage White	Number of Gyms	Number of Parks
Pop Density per sq km	1.000000e+00	4.328240e-13	2.852801e-02	6.965659e-03	1.740065e-01	3.010547e-05
Total Tract Median Age	4.328240e-13	1.000000e+00	2.998268e-45	1.559360e-07	2.418853e-01	5.415849e-02
Tract Per Capita Income - 12 Months	2.852801e-02	2.998268e-45	1.000000e+00	4.675941e-19	5.043335e-12	3.400248e-03
Tract Percentage White	6.965659e-03	1.559360e-07	4.675941e-19	1.000000e+00	3.093067e-02	2.107632e-01
Number of Gyms	1.740065e-01	2.418853e-01	5.043335e-12	3.093067e-02	1.000000e+00	5.776413e-19
Number of Parks	3.010547e-05	5.415849e-02	3.400248e-03	2.107632e-01	5.776413e-19	1.000000e+00

Works Cited

Steven Manson, Jonathan Schroeder, David Van Riper, Tracy Kugler, and Steven Ruggles. IPUMS National Historical Geographic Information System: Version 15.0 [dataset]. Minneapolis, MN: IPUMS. 2020. **http://doi.org/10.18128/D050.V15.0**

Foursquare. *Foursquare Developers Search API*. n.d.