**Capstone Project – Battle of Neighborhoods**

**Introduction/Business Problem**

A research team wants to analyze New York city properties sale data and visualize the sale activities for each of its 5 boroughs. As part of this project, they also have a requirement to find out the “Borough” that made the most sales.

New York City has diverse culture and point of interests uniquely spread across 5 boroughs and its distinct neighborhoods make the city so special. Therefore, the city’s data set was chosen for this project.

The team would leverage the Kaggle dataset for properties sold in New York city over a 12-month period from September 2016 to September 2017 and Foursquare location data to explore the most common neighborhood venues and details.

The team would identify the borough with most sales based on sale data, then would utilize the Foursquare APIs/location data to identify the recommendations for common/popular venues in the neighborhoods. Using K means, the common venues will be clustered in groups Finally this data will be projected and visualized on a city neighborhood map.

An analytical approach will be used to solve the problem applying advanced machine learning principles along with data transformation & analysis and data visualization techniques.

**Data**

**1. Kaggle data set will be primarily used for this project.**

* <https://www.kaggle.com/new-york-city/nyc-property-sales/data>

This dataset is a record of every building or building unit (apartment, etc.) sold in the New York City property market over a 12-month period, from September 2016 to September 2017

This dataset contains the location, address, type, sale price, and sale date of building units sold. A reference on the trickier fields:

* + **BOROUGH**: A digit code for the borough the property is located in; in order these are Manhattan (1), Bronx (2), Brooklyn (3), Queens (4), and Staten Island (5).
  + **BLOCK**; **LOT**: The combination of borough, block, and lot forms a unique key for property in New York City. Commonly called a BBL
  + **BUILDING CLASS AT PRESENT** and **BUILDING CLASS AT TIME OF SALE**: The type of building at various points in time

**2. NY City Geo spatial data set.**

* Neighborhood has a total of 5 boroughs and 306 neighborhoods. In order to segment the neighborhoods and explore them, we will essentially need a dataset that contains the 5 boroughs and the neighborhoods that exist in each borough as well as the latitude and longitude coordinates of each neighborhood.
* This dataset exists for free on the web. The link to the dataset: <https://geo.nyu.edu/catalog/nyu_2451_34572>

**3. Top Picks/Common venues in the most transacted borough’s neighborhood of New York city.**

* Foursquare API
  + GET [https://api.foursquare.com/v2/venues/explore/\*](https://api.foursquare.com/v2/venues/explore/*).
  + Response – The following attributes are retrieved from the API response
    - Neighborhood: Name of the neighborhood
    - Neighborhood - latitude & longitude
    - Venue: Venue Name
    - Venue – latitude & longitude
    - Venue Category: category of the venue
* By using the API, we will explore the “top Picks” or recommended venues in the borough

The raw data is first loaded into a **dataframe** using python and transformations applied such as converting the Borough codes to names. The transformed data is then analyzed and grouped to determine the Borough where most sales occurred.

This grouped data is then visualized and plotted in a Bar chart for easy inference. Also, the top borough in terms of sales is identified during the data analysis stage.

NY geospatial data is then loaded into another dataframe in python and used to visualize the neighborhood data in a map visual.

Foursquare API calls are made to get venue details for the neighborhoods and statistical analysis will output the list of venue recommendations for the borough and its neighborhoods.

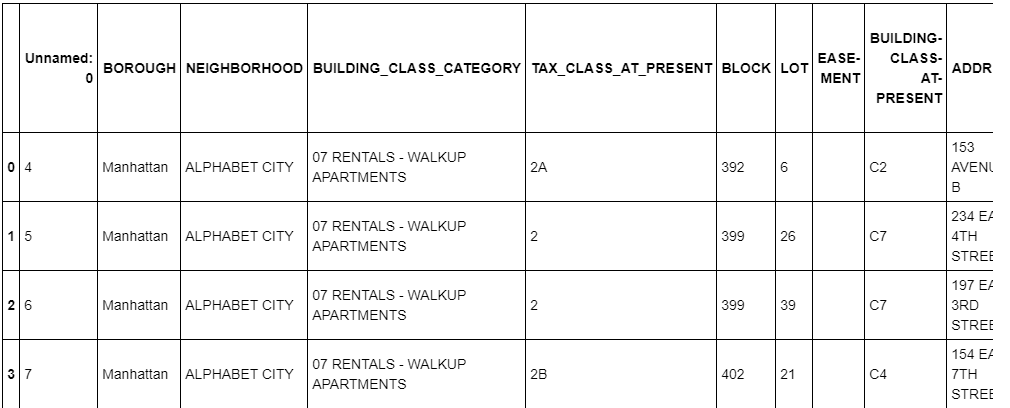
**Methodology**

A statistical exploration methodology was used to analyze the problem and datasets along with visualization techniques. This data analysis and exploration also involved a main machine learning technique using which the data was clustered i.e. most popular neighborhood venues. K-means clustering implemented in Python to produce the results.

The main goal is to solve the problem by running exploratory and analysis steps on the data set. Firstly, we group the NY property sales for the various New York city boroughs and find the borough where most property sales occurred.

**Step 1: Explore Kaggle’s New York city property sale data.**

**Example of the Data:**



The dataset is a record of every property (building, apartment, etc.) sold in the New York City market over a 12-month period from September 2016 to September 2017.

This dataset is a concatenated and slightly cleaned-up version of the New York City Department of Finance's [Rolling Sales dataset](http://www1.nyc.gov/site/finance/taxes/property-rolling-sales-data.page).

**Columns**

* BOROUGH
* NEIGHBORHOOD
* BUILDING CLASS CATEGORY
* TAX CLASS AT PRESENT
* BLOCK
* LOT
* EASE-MENT
* BUILDING CLASS AT PRESENT
* ADDRESS
* APARTMENT NUMBER
* ZIP CODE
* RESIDENTIAL UNITS
* COMMERCIAL UNITS
* TOTAL UNITS
* LAND SQUARE FEET
* GROSS SQUARE FEET
* YEAR BUILT
* TAX CLASS AT TIME OF SALE
* BUILDING CLASS AT TIME OF SALE
* SALE PRICE
* SALE DATE

The main objective of this Data Science project is to identify the top borough in the city which had the most sales based on the raw analysis of NY property sales data. We also wanted to explore the neighborhoods in this borough and identify the most common venues. In addition, we will be using ***matplotlib*** and ***folium*** libraries in Python to plot and visualize the results.

**Data Transformation/Analysis**

As we attempt to explore the raw data, we have a need to transform and prepare this data for subsequent steps and processing.

Raw data is loaded into a ***dataframe*** and the borough code values in the data set is replaced with actual borough names as below.

*# read the csv data file*

df\_ny\_sale\_data = pd.read\_csv(body)

*# replace Borough values as String*

df\_ny\_sale\_data['BOROUGH'].replace(to\_replace=[1,2,3,4,5], value=['Manhattan','Brooklyn', 'Queens','Bronx','Staten Island'],inplace=**True**)

df\_ny\_sale\_data.head()

|  |  |
| --- | --- |
| **Borough Code** | **Borough Name** |
| 1 | Manhattan |
| 2 | Brooklyn |
| 3 | Queens |
| 4 | Bronx |
| 5 | Staten Island |

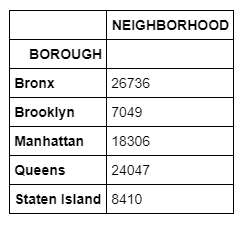
We will find the borough with the most sales

*# find the top/maximum # of sales in NY city Boroughs*

df\_ny\_sale\_data\_grouped=df\_ny\_sale\_data[['BOROUGH','NEIGHBORHOOD']].groupby('BOROUGH').count()

df\_ny\_sale\_data\_grouped=df\_ny\_sale\_data\_grouped.sort\_values(by=['BOROUGH'], ascending=**True**)

df\_ny\_sale\_data\_grouped



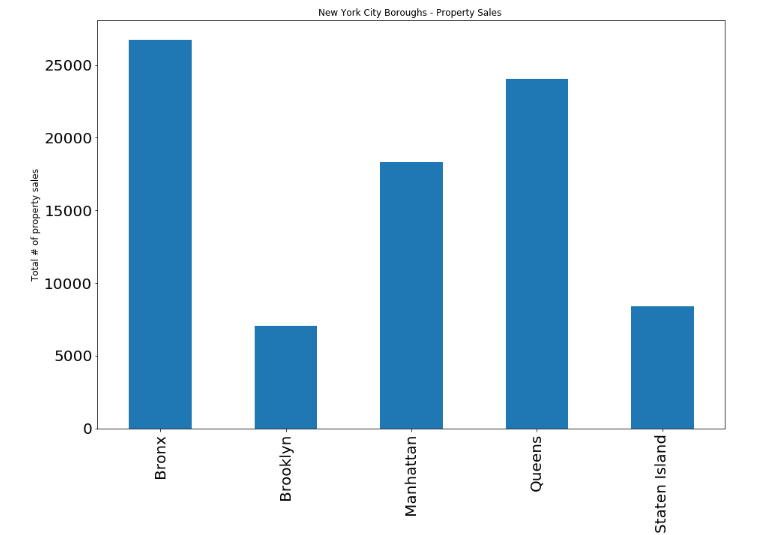
After the raw data is grouped and transformed, it is then plotted in a bar chart using matplotlib libraries to show the property sales for each of the 5 boroughs

ax = df\_ny\_sale\_data\_grouped.plot(kind='bar', title ="New York City Boroughs - Property Sales ", figsize=(15, 10), legend=**False**, fontsize=20)

ax.set\_xlabel("Borough", fontsize=12)

ax.set\_ylabel("Total # of property sales", fontsize=12)

plt.show()



Then we find the borough with most sales from the grouped dataset.

*# Find the top Borough with most sales*

top\_borough = df\_ny\_sale\_data\_grouped.iloc[0].name

print("Top Borough with most property sales : ",top\_borough)

Top Borough with most property sales : Bronx

**Step 2: Explore and merge New York city geo spatial data**

NY neighborhood has a total of 5 boroughs and 306 neighborhoods. In order to segment the neighborhoods and explore them, we will essentially need a dataset that contains the 5 boroughs and the neighborhoods that exist in each borough as well as the latitude and longitude coordinates of each neighborhood.

Luckily, this dataset exists for free on the web, here is the link to the dataset: <https://geo.nyu.edu/catalog/nyu_2451_34572>

The data is in the features key, which is basically a list of the neighborhoods. So, let's define a new variable that includes this data.

**with** open('newyork\_data.json') **as** json\_data:

newyork\_data = json.load(json\_data)

neighborhoods\_data = newyork\_data['features']

neighborhoods\_data[0]

{'type': 'Feature',

'id': 'nyu\_2451\_34572.1',

'geometry': {'type': 'Point',

'coordinates': [-73.84720052054902, 40.89470517661]},

'geometry\_name': 'geom',

'properties': {'name': 'Wakefield',

'stacked': 1,

'annoline1': 'Wakefield',

'annoline2': None,

'annoline3': None,

'annoangle': 0.0,

'borough': 'Bronx',

'bbox': [-73.84720052054902,

40.89470517661,

-73.84720052054902,

40.89470517661]}}

Now we will focus our analysis on the borough with most sales. The data is again transformed to include the neighborhood coordinates (latitude and longitude)

*# define the dataframe columns*

column\_names = ['Borough', 'Neighborhood', 'Latitude', 'Longitude']

*# instantiate the dataframe*

neighborhoods = pd.DataFrame(columns=column\_names)

*# loop through the data and fill the dataframe one row at a time*

**for** data **in** neighborhoods\_data:

borough = neighborhood\_name = data['properties']['borough']

neighborhood\_name = data['properties']['name']

neighborhood\_latlon = data['geometry']['coordinates']

neighborhood\_lat = neighborhood\_latlon[1]

neighborhood\_lon = neighborhood\_latlon[0]

neighborhoods = neighborhoods.append({'Borough': borough,

'Neighborhood': neighborhood\_name,

'Latitude': neighborhood\_lat,

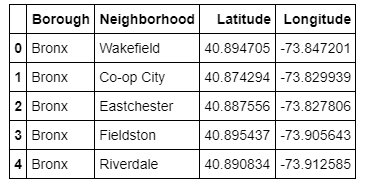
'Longitude': neighborhood\_lon}, ignore\_index=**True**)

neighborhoods.head()

*# only Bronx neighborhoods*

bronx\_data = neighborhoods[neighborhoods['Borough'] == 'Bronx'].reset\_index(drop=**True**)

bronx\_data.head()



All the borough neighborhoods are then loaded onto a NY city map using folium library. This visualization will plot the borough geographically along with its neighborhoods.

*# create map of Bronx using latitude and longitude values*

map\_bronx = folium.Map(location=[latitude, longitude], zoom\_start=11)

*# add markers to map*

**for** lat, lng, label **in** zip(bronx\_data['Latitude'], bronx\_data['Longitude'], bronx\_data['Neighborhood']):

label = folium.Popup(label, parse\_html=**True**)

folium.CircleMarker(

[lat, lng],

radius=5,

popup=label,

color='blue',

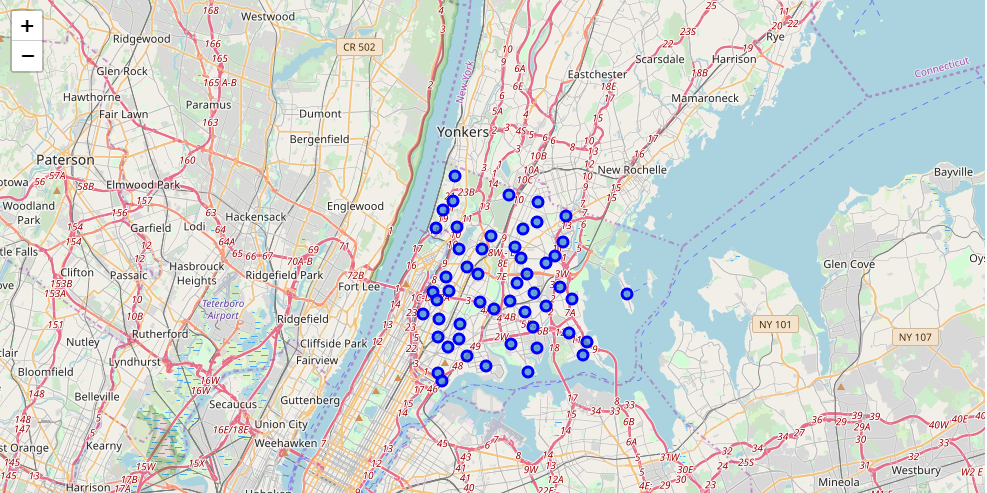
fill=**True**,

fill\_color='#3186cc',

fill\_opacity=0.7,

parse\_html=**False**).add\_to(map\_bronx)

map\_bronx



**Step 3: Foursquare APIs to get venue recommendations.**

The Foursquare APIs is used to explore the Bronx neighborhood data and the top recommended venues are retrieved. This step will fetch the top recommendations with their coordinates. This data is then used to segment and cluster locations

*# create url to explore topPicks in the neighborhood*

LIMIT = 100 *# limit of number of venues returned by Foursquare API*

radius = 1000 *# define radius*

section = 'topPicks' *# search for topPicks(recommendations)*

url = 'https://api.foursquare.com/v2/venues/search?client\_id=**{}**&client\_secret=**{}**&ll=**{}**,**{}**&v=**{}**&radius=**{}**&limit=**{}**&section=**{}**'.format(

CLIENT\_ID,CLIENT\_SECRET,

latitude, longitude,

VERSION,radius, LIMIT,section)

url

The resulting data frame from the above API call with the venue recommendations and details

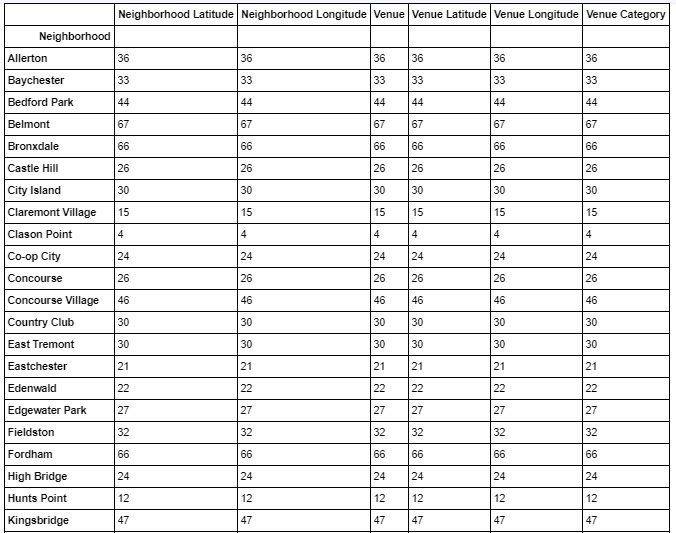
bronx\_top\_venues.head(10)



The API query returns the possible venues, their coordinates and the category of each venue as shown in the sample above.

The results are then grouped together

bronx\_top\_venues.groupby('Neighborhood').count()



This is an important step in the data segmentation process where all the key venue data is derived from here.

Next we will use one hot encoding to analyze the neighborhood venue recommendations.

*# one hot encoding*

bronx\_onehot = pd.get\_dummies(bronx\_top\_venues[['Venue Category']], prefix="", prefix\_sep="")

*# add neighborhood column back to dataframe*

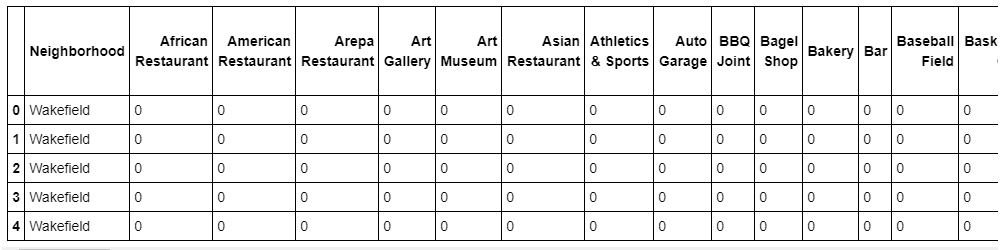
bronx\_onehot['Neighborhood'] = bronx\_top\_venues['Neighborhood']

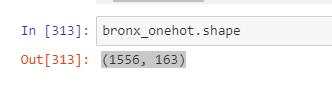
*# move neighborhood column to the first column*

fixed\_columns = [bronx\_onehot.columns[-1]] + list(bronx\_onehot.columns[:-1])

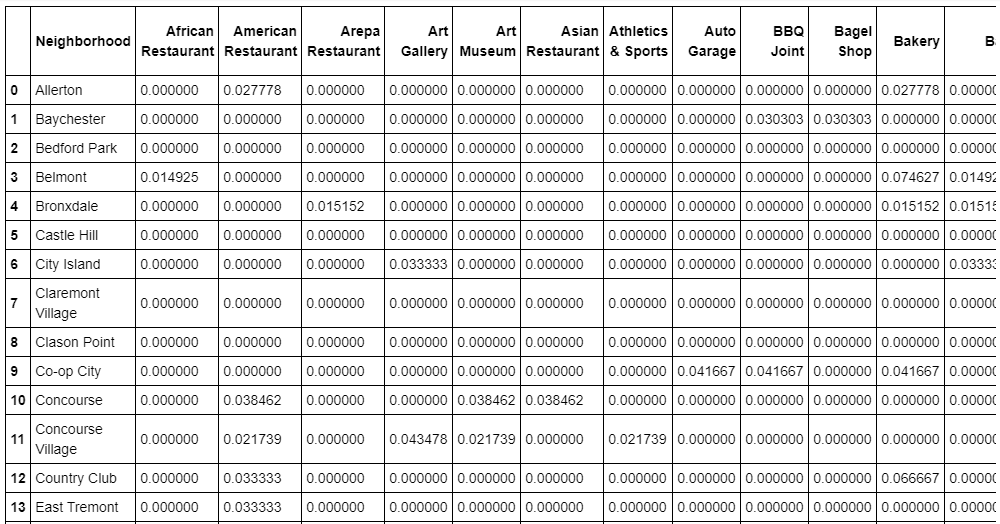
bronx\_onehot = bronx\_onehot[fixed\_columns]

bronx\_onehot.head()





We then calculate the mean occurrence of each venue for the neighborhood:



Let’s print each neighborhood with top 5 venues from the analysis.

num\_top\_venues =5

**for** hood **in** bronx\_grouped['Neighborhood']:

print("----"+hood+"----")

temp = bronx\_grouped[bronx\_grouped['Neighborhood'] == hood].T.reset\_index()

temp.columns = ['venue','freq']

temp = temp.iloc[1:]

temp['freq'] = temp['freq'].astype(float)

temp = temp.round({'freq': 2})

print(temp.sort\_values('freq', ascending=**False**).reset\_index(drop=**True**).head(num\_top\_venues))

print('**\n**')

----Allerton----

venue freq

0 Chinese Restaurant 0.08

1 Pizza Place 0.08

2 Mexican Restaurant 0.08

3 Supermarket 0.08

4 Deli / Bodega 0.08

----Baychester----

venue freq

0 Shopping Mall 0.09

1 Pizza Place 0.09

2 Chinese Restaurant 0.09

3 Supermarket 0.06

4 Fried Chicken Joint 0.06

----Bedford Park----

venue freq

0 Pizza Place 0.11

1 Diner 0.09

2 Baseball Field 0.09

3 Park 0.09

4 Café 0.07

----Belmont----

venue freq

0 Italian Restaurant 0.12

1 Deli / Bodega 0.10

2 Pizza Place 0.09

3 Bakery 0.07

4 Café 0.06

----Bronxdale----

venue freq

0 Pizza Place 0.20

1 Chinese Restaurant 0.09

2 Deli / Bodega 0.06

3 Coffee Shop 0.05

4 Italian Restaurant 0.05

----Castle Hill----

venue freq

0 Deli / Bodega 0.12

1 Diner 0.08

2 Clothing Store 0.08

3 Baseball Field 0.08

4 Latin American Restaurant 0.04

----City Island----

venue freq

0 Harbor / Marina 0.23

1 Seafood Restaurant 0.13

2 Italian Restaurant 0.07

3 Thrift / Vintage Store 0.07

4 Baseball Field 0.03

----Claremont Village----

venue freq

0 Discount Store 0.27

1 Donut Shop 0.13

2 Ice Cream Shop 0.13

3 Pizza Place 0.13

4 Caribbean Restaurant 0.07

----Clason Point----

venue freq

0 Park 0.75

1 Gym / Fitness Center 0.25

2 African Restaurant 0.00

3 Nightclub 0.00

4 Optical Shop 0.00

----Co-op City----

venue freq

0 Pizza Place 0.17

1 Shopping Mall 0.12

2 Liquor Store 0.12

3 Electronics Store 0.08

4 Dumpling Restaurant 0.04

----Concourse----

venue freq

0 Deli / Bodega 0.15

1 Park 0.12

2 Chinese Restaurant 0.08

3 Italian Restaurant 0.04

4 Caribbean Restaurant 0.04

----Pelham Bay----

venue freq

0 Italian Restaurant 0.14

1 Pizza Place 0.14

2 Deli / Bodega 0.10

3 Bakery 0.07

4 Chinese Restaurant 0.03

----Pelham Gardens----

venue freq

0 Caribbean Restaurant 0.1

1 Electronics Store 0.1

2 Supermarket 0.1

3 Golf Course 0.1

4 Deli / Bodega 0.1

----Pelham Parkway----

venue freq

0 Pizza Place 0.21

1 Deli / Bodega 0.10

2 Chinese Restaurant 0.10

3 Italian Restaurant 0.05

4 Coffee Shop 0.05

----Westchester Square----

venue freq

0 Bar 0.08

1 Pizza Place 0.08

2 Food Truck 0.08

3 Italian Restaurant 0.04

4 Spanish Restaurant 0.04

----Williamsbridge----

venue freq

0 Caribbean Restaurant 0.14

1 Pizza Place 0.11

2 Fried Chicken Joint 0.07

3 Supermarket 0.07

4 Pet Store 0.07

----Woodlawn----

venue freq

0 Pizza Place 0.11

1 Deli / Bodega 0.11

2 Pub 0.11

3 American Restaurant 0.07

4 Diner 0.07

The above display indicates the frequency of venue occurrences. Now, it should be possible for us to know the most common venues.

num\_top\_venues = 5

indicators = ['st', 'nd', 'rd']

*# create columns according to number of top venues*

columns = ['Neighborhood']

**for** ind **in** np.arange(num\_top\_venues):

**try**:

columns.append('**{}{}** Most Common Venue'.format(ind+1, indicators[ind]))

**except**:

columns.append('**{}**th Most Common Venue'.format(ind+1))

*# create a new dataframe*

neighborhoods\_venues\_sorted = pd.DataFrame(columns=columns)

neighborhoods\_venues\_sorted['Neighborhood'] = bronx\_grouped['Neighborhood']

**for** ind **in** np.arange(bronx\_grouped.shape[0]):

neighborhoods\_venues\_sorted.iloc[ind, 1:] = return\_most\_common\_venues(bronx\_grouped.iloc[ind, :], num\_top\_venues)



Now we can move on the final stages of this exploration. We have the data that is needed to apply segmentation and create the grouping clusters. Typically, we would need to determine the appropriate number of clusters for segmenting using elbow method k-means.

However, for this project we would assume 5 as the optimal number of clusters. The clusters are built as below.

*# import k-means from clustering stage*

**from** **sklearn.cluster** **import** KMeans

*# set number of clusters*

kclusters = 5

bronx\_grouped\_clustering = bronx\_grouped.drop('Neighborhood', 1)

*# run k-means clustering*

kmeans = KMeans(n\_clusters=kclusters, random\_state=0).fit(bronx\_grouped\_clustering)

*# check cluster labels generated for each row in the dataframe*

kmeans.labels\_[0:10]

array([1, 1, 0, 3, 0, 0, 3, 4, 2, 0], dtype=int32)

*# add clustering labels*

*neighborhoods\_venues\_sorted.insert(0, 'Cluster Labels', kmeans.labels\_)*

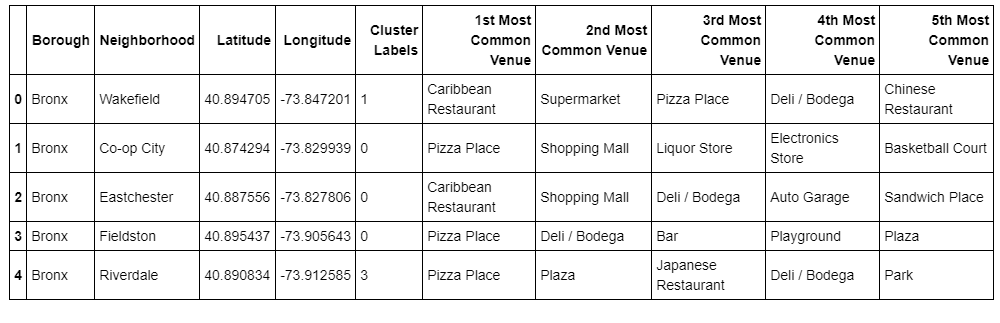
bronx\_merged = bronx\_data

*# merge bronx\_grouped with bronx\_data to add latitude/longitude for each neighborhood*

bronx\_merged = bronx\_merged.join(neighborhoods\_venues\_sorted.set\_index('Neighborhood'), on='Neighborhood')

bronx\_merged.head()

The below table lists the top 5 most common venues amongst the borough neighborbood.



Now let’s visualize the findings in a NY city map and display the venue clusters

*# create map*

map\_clusters = folium.Map(location=[latitude, longitude], zoom\_start=11)

*# set color scheme for the clusters*

x = np.arange(kclusters)

ys = [i + x + (i\*x)\*\*2 **for** i **in** range(kclusters)]

colors\_array = cm.rainbow(np.linspace(0, 1, len(ys)))

rainbow = [colors.rgb2hex(i) **for** i **in** colors\_array]

*# add markers to the map*

markers\_colors = []

**for** lat, lon, poi, cluster **in** zip(bronx\_merged['Latitude'], bronx\_merged['Longitude'], bronx\_merged['Neighborhood'], bronx\_merged['Cluster Labels']):

label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse\_html=**True**)

folium.CircleMarker(

[lat, lon],

radius=5,

popup=label,

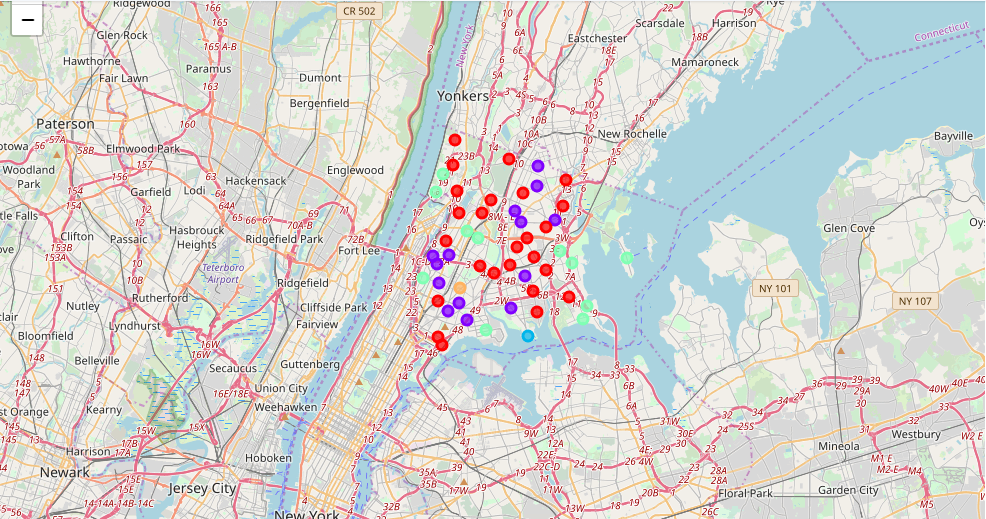
color=rainbow[cluster-1],

fill=**True**,

fill\_color=rainbow[cluster-1],

fill\_opacity=0.7).add\_to(map\_clusters)

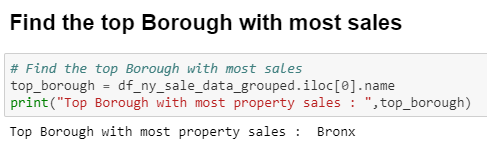
map\_clusters



With this data exploration and methodology, we performed an analysis of the borough by exploring its neighborhoods and venue details. Now we should be able to document the results and write up the conclusions.

**Results**

After our Step 1 analysis using Kaggle data set for NY property sales we were able to determine the total number of sales for each of the boroughs and find out the borough where most sales occurred.



Subsequently, we merged the NY geospatial data set to add the coordinates for each of the neighborhood locations. We were able to then make Foursquare API calls to get a list of recommendations on the most common neighborhood venues in the borough.



Finally, we used the common venue/recommendations data from Foursquare to segment/cluster these venues and visualize it in a geographical location map.

When we review all these findings, we can clearly infer that the borough neighborhood is so diverse and the popular venues are mostly food joints, recreation and outdoors. For anyone who likes to move into the neighborhood towns or interested in buying a property home would benefit from the broader choices available and proximity to Manhattan.

**Discussions**

Note that because this is a financial transaction dataset, there are some points that need to be kept in mind:

* Many sales occur with a nonsensically small dollar amount: $0 most commonly. These sales are transferring of deeds between parties: for example, parents transferring ownership to their home to a child after moving out for retirement.
* This dataset uses the financial definition of a building/building unit, for tax purposes. In case a single entity owns the building in question, a sale covers the value of the entire building. In case a building is owned piecemeal by its residents (a condominium), a sale refers to a single apartment (or group of apartments) owned by some individual.

This project used k-means clustering technique, typically, we would determine the appropriate number of clusters for segmenting using elbow method. This would produce a more accurate segmentation and clustering groups.

**Conclusions**

The purpose of this project was to determine the borough that had most sales in a year based on the NY city Kaggle data set. **Bronx** was identified as the top borough. We also wanted to explore the neighborhood around Bronx and get recommendations/top picks for the most common venues.

We were able to figure out the most common venues and cluster them using k-means. Finally, we visualized these findings in a NY location map and project these venues.