NYC DATA PROPERTY ANALYSIS - CAPSTONE FINAL PROJECT REPORT

Sucheta Gangullapattu

Introduction / Business Problem

New York City (NYC) is one of the most populated cities in USA and second most populous city in North America. Its is comprised of 5 boroughs namely Brooklyn, Manhattan, Staten Island, Bronx and Queens. The 5 Boroughs cover overall land area of about 784 Km square with a population of about roughly 8,398,748 in year 2018. As a Neighbor of New York City, I have chosen this location for my capstone project.

New York City is culturally very diverse with population density of 159 people per square Km and described as financial, and media capital of the world and it's a center for commerce entertainment, research, technology, education, politics, tourism, art, fashion, and sports.

NYC is one of the largest metropolitan cities with over 20 million people and home to headquarters of United Nations. There are more than 100 neighborhoods divided among 5 boroughs with Manhattan titled the most expensive real estate Markets. New York city is most powerful city economically and financially, it is also home to largest stock exchanges the NASDAQ and New York Stock Exchange.

As we can see from the statistics NYC is a very diverse and financial capital, we can derive many ideas and problems like: if I am looking to open a restaurant or business, I would like to explore neighborhoods /areas with low real estate property values? If someone is looking for office / house to rent which area should they prefer and why?

With help of foursquare location data and raw data (NYC property data) and other tools I explore further to cluster based on borough information and venue data obtained using foursquare and come up with a solution to some of the problems mentioned above.

Reference:

https://en.wikipedia.org/wiki/New_York_City

Note: some of the statistical information (population info) has been taken from the link above.

Data Section:

1. https://www1.nyc.gov/site/finance/taxes/property-rolling-sales-data.page
2. Foursquare location data will also be used.

Data consists of rolling property sales data for all 5 boroughs and information about taxes, type of property, neighborhood, date of sale, square footage info etc. The data corresponds to 12-month period (year 2018). The source had sales data recorded per each Borough. I have consolidated the data in one data source.

Description:

The data contain property sales data across all 5 boroughs

Manhattan (1), Brooklyn (3), Staten Island (5), Bronx (2), Queens (4)

Neighborhood info: Name of the Neighborhood where the property dwells

Building Class Category: 01 ONE FAMILY DWELLINGS, 21 OFFICE BUILDINGs, COMMERICAL CONDOS etc. (There are about 44 Categories)

Tax class at Present: There 3 to 4 diff tax classes applied based on building class category

Property Details: 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, T AX TIME OF SALE, SALE PRICE, SALE DATE

Date Understanding and Preparation:

The data is obtained from the following source:

https://www1.nyc.gov/site/finance/taxes/property-rolling-sales-data.page

Sales data of NYC for 12-months sorted by Borough (Manhattan, Bronx, Brooklyn, Queens, Staten Island)

The source page contains a downloadable Excel file for each borough.

New York City Sales Data from May 2018 to April 2019
Manhattan
Bronx
Brooklyn
Queens
Staten Island

I combined the data from 5 boroughs into one Master Excel File.

I used the Master file saved on the IBM Cloud and retrieved using Credential's.

Credentials for accessing the file on IBM Cloud Object Storage

```
# @hidden_cell
# The following code contains the credentials for a file in your IBM Cloud Object Storage.
# You might want to remove those credentials before you share your notebook.
credentials_1 = {
    'IAM_SERVICE_ID': 'iam-ServiceId-aa17df86-647f-4b81-9916-2147be9abcb2',
    'IBM_API_KEY_ID': 'zR3mO0lbGu-Lvo7Z-DGB72MqvT_sMFLQ806fAfa2ZjHk',
    'ENDPOINT': 'https://s3-api.us-geo.objectstorage.service.networklayer.com',
    'IBM_AUTH_ENDPOINT': 'https://iam.bluemix.net/oidc/token',
    'BUCKET': 'courseracapstoneproject-donotdelete-pr-df1ua5hmjvjuua',
    'FILE': 'rollingsales_boroughs_nyc_jan-dec_2018.xlsx'
}
```

Retrieving the file from cloud storage:

```
streaming_body_5 = client_ca4e73bbce0f43b1a9b491eb56886b3c.get_object(Bucket='courseracapstoneproject-donotdelete-pr-df1ua5hm
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(streaming_body_5, "__iter__"): streaming_body_5.__iter__ = types.MethodType( __iter__, streaming_body_5 )

df_data = pd.read_excel(streaming_body_5)
df_data.head()
```

As we can see the dataset contains the following info:

The following snapshot shows the consolidated data from all five boroughs and their corresponding neighborhood's.

	BOROUGH	NEIGHBORHOOD	BUILDING CLASS CATEGORY	TAX CLASS AT PRESENT	BLOCK	LOT	EASE-MENT	BUILDING CLASS AT PRESENT	ADDRESS	APARTMENT NUMBER	...	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
0	1	ALPHABET CITY	01 ONE FAMILY DWELLINGS	1	390	61		A4	189 EAST 7TH STREET		...	1	0	1	987	2183	1860	1	A4
1	1	ALPHABET CITY	01 ONE FAMILY DWELLINGS	1	390	61		A4	189 EAST 7TH STREET		...	1	0	1	987	2183	1860	1	A4
2	1	ALPHABET CITY	01 ONE FAMILY DWELLINGS	1	400	19		A4	526 EAST 5TH STREET		...	1	0	1	1883	5200	1900	1	A4

Obtaining Longitude and Latitude information:

As we can see from the snapshot below that we need Longitude and latitude information in the raw Data Source (dataset), geopy Geocoding library for python has been used to generate Coordinates for Boroughs and its corresponding Neighborhood's.

```
print(df_data.dtypes,'/n')
```

```
(79626, 21) /n
BOROUGH                              int64
NEIGHBORHOOD                        object
BUILDING CLASS CATEGORY             object
TAX CLASS AT PRESENT                object
BLOCK                                int64
LOT                                  int64
EASE-MENT                           object
BUILDING CLASS AT PRESENT           object
ADDRESS                             object
APARTMENT NUMBER                    object
ZIP CODE                             int64
RESIDENTIAL UNITS                    int64
COMMERCIAL UNITS                     int64
TOTAL UNITS                          int64
LAND SQUARE FEET                     int64
GROSS SQUARE FEET                    int64
YEAR BUILT                           int64
TAX CLASS AT TIME OF SALE            int64
BUILDING CLASS AT TIME OF SALE      object
SALE PRICE                           int64
SALE DATE                   datetime64[ns]
```

Extracting Borough and Neighborhood Information:

Extracting Neighborhood , Borough information and calculating L

```
l1=df_data.groupby(['BOROUGH'])

df3=l1.apply(lambda x: x['NEIGHBORHOOD'].unique())

df3
```

```
227]:  BOROUGH
       Bronx            [BATHGATE, BAYCHESTER, BEDFORD PARK/NORWOOD, B...
       Brooklyn         [BATH BEACH, BAY RIDGE, BEDFORD STUYVESANT, BE...
       Manhattan        [ALPHABET CITY, CHELSEA, CHINATOWN, CIVIC CENT...
       Queens           [AIRPORT LA GUARDIA, ARVERNE, ASTORIA, BAYSIDE...
       Staten Island    [ANNADALE, ARDEN HEIGHTS, ARROCHAR, ARROCHAR-S...
       dtype: object
```

converting list into dataframe

```
df4=pd.DataFrame(df3)
df4
```

228]:

	0
BOROUGH	
Bronx	[BATHGATE, BAYCHESTER, BEDFORD PARK/NORWOOD, B...

Splitting the Neighborhood data:

```
s = df4.apply(lambda x: pd.Series(x[0]), axis=1).stack(
s.name = 0
dfnew = df4.drop(0, axis=1).join(s)
dfnew[0] = pd.Series(dfnew[0], dtype=object)

dfnew.rename(columns={0: 'NEIGHBORHOOD'},inplace=True)
dfnew.reset_index(inplace=True)
```

```
dfnew
```

	BOROUGH	NEIGHBORHOOD
0	Bronx	BATHGATE
1	Bronx	BAYCHESTER
2	Bronx	BEDFORD PARK/NORWOOD
3	Bronx	BELMONT
4	Bronx	BRONX PARK
5	Bronx	BRONXDALE
6	Bronx	CASTLE HILL/UNIONPORT

Now that the new data set has Borough/ Neighborhood data

Library gepoy is used to generate Latitude and longitude information

```
import geopy
from geopy.geocoders import Nominatim

nom=Nominatim()
#n=nom.geopycode

dfnew['Address']=dfnew['NEIGHBORHOOD'] + ','+'NY'
```

```
dfnew['coordinates']=dfnew["Address"].apply(nom.geocode)  # sending string to gecode method
```

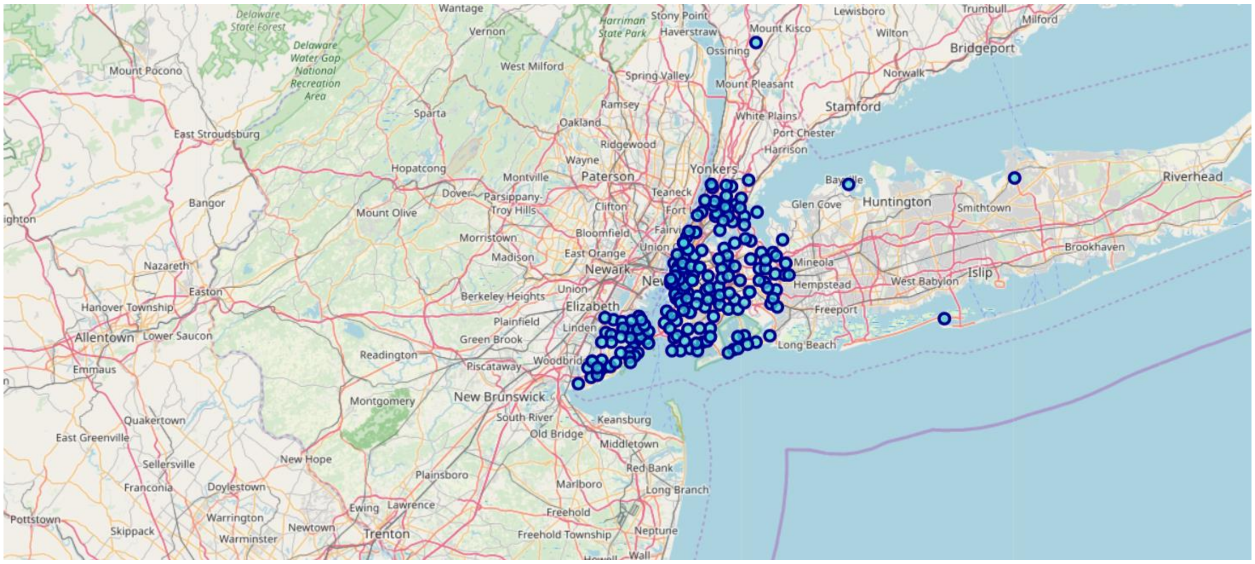
```
dfnew['Latitude']=dfnew['coordinates'].apply(lambda x: x.latitude if x!= None else None) # latitude
```

```
dfnew['Longitude']=dfnew['coordinates'].apply(lambda x: x.longitude if x!= None else None)# longitude
```

```
dfnew
```

	BOROUGH	NEIGHBORHOOD	Address	coordinates	Latitude	Longitude
0	Bronx	BATHGATE	BATHGATE,NY	(Bath, Steuben County, New York, 14810, USA, (...	42.337	-77.318
1	Bronx	BAYCHESTER	BAYCHESTER,NY	(Baychester, The Bronx, Bronx County, NYC, New...	40.861	-73.841
2	Bronx	BEDFORD PARK/NORWOOD	BEDFORD PARK/NORWOOD,NY	None	nan	nan
3	Bronx	BELMONT	BELMONT,NY	(Belmont, Allegany County, New York, USA, (42....	42.223	-78.034

Next, Creating a NYC map using folium with Boroughs and Neighborhood superimposed on the map.



DATA CLEANSING:

1. Does the data contain duplicates?

```
#checking for duplicate records
sum(df_data.duplicated(df_data.columns))
```

```
6]:  570
```

```
# remove duplicate records
df_data = df_data.drop_duplicates(df_data.columns, keep='last')
```

Checking and removing duplicate values

2. Checking for Null (NaN) values?

```
df_data.isnull().sum()
```

```
12]: BOROUGH                          0
     NEIGHBORHOOD                     0
     BUILDING CLASS CATEGORY          0
     TAX CLASS AT PRESENT             0
     BLOCK                            0
     LOT                              0
     EASE-MENT                        0
     BUILDING CLASS AT PRESENT        0
     ADDRESS                          0
```

No Null Values found

3. Checking for any invalid entries?

```
df_data['SALE PRICE'].value_counts().
```

```
0]: 0         23052
    10          663
    650000      427
```

As we can see there are some entries where sales price is equal to 10 / 0 getting rid of invalid entries.

4. Checking if Total units ==0?

```
df_data=df_data[df_data['TOTAL UNITS'] == df_data['COMMERCIAL UNITS'] + df_data['RESIDENTIAL UNITS']]

df_data.shape
```

```
8]: (40082, 21)
```

Rows where Total units = sum (commercial units, Residential units) are taken into account.

```
print(df_data.dtypes,'/n')

(79626, 21) /n
BOROUGH                              int64
NEIGHBORHOOD                        object
BUILDING CLASS CATEGORY             object
TAX CLASS AT PRESENT                object
BLOCK                                int64
LOT                                  int64
EASE-MENT                           object
BUILDING CLASS AT PRESENT           object
ADDRESS                             object
APARTMENT NUMBER                    object
ZIP CODE                             int64
RESIDENTIAL UNITS                    int64
COMMERCIAL UNITS                     int64
TOTAL UNITS                          int64
LAND SQUARE FEET                     int64
GROSS SQUARE FEET                    int64
YEAR BUILT                           int64
TAX CLASS AT TIME OF SALE            int64
BUILDING CLASS AT TIME OF SALE      object
SALE PRICE                           int64
SALE DATE                   datetime64[ns]
```

5. Changing the Numerical representation of Boroughs to their actual Names.

Borough is represented in terms of value 1 (Manhattan),2(Bronx),3(Brooklyn),4(Queens),5(Staten Island) Converting the numerical values for Borough Column to their Corresponding names

```
df_data['BOROUGH'].loc[df_data['BOROUGH'] == 1] = 'Manhattan'
df_data['BOROUGH'].loc[df_data['BOROUGH'] == 2] = 'Bronx'
df_data['BOROUGH'].loc[df_data['BOROUGH'] == 3] = 'Brooklyn'
df_data['BOROUGH'].loc[df_data['BOROUGH'] == 4] = 'Queens'
df_data['BOROUGH'].loc[df_data['BOROUGH'] == 5] ='Staten Island'
```

Methodology

Introduction: Foursquare API

FourSquare API is used to for exploring / obtaining venue information, Foursquare user information, explore geographical information and to get trending venues around a location.

Foursquare API can be used by Constructing an URL with credentials obtained by signing up into Foursquare and sending a request to the API for search of a specific venue, to explore the geographical locations around a venue etc.

Note: The snap shots are taken to show as results, are from Borough Manhattan.

The above process is carried out for Each Borough and their neighborhoods (Brooklyn, Bronx, Staten Island, Queens, Manhattan).

1. Credentials for generating URL request to

```
CLIENT_ID = 'D5P5G0EQI0J54NBGZGDC3YPBDY
CLIENT_SECRET = 'PQ4FRAFCPMISSULNC1YBSN
VERSION = '20190117' # Foursquare API v

print('Your credentails:')
print('CLIENT_ID: ' + CLIENT_ID)
print('CLIENT_SECRET:' + CLIENT_SECRET)
```

2. Exploring the Manhattan Borough

A URL request is the outcome here

```
address = 'Manhattan, NY'

geolocator = Nominatim(user_agent="ny_explorer")
location = geolocator.geocode(address)
man_latitude = location.latitude
man_longitude = location.longitude
print('The geograpical coordinate of Manhattan are {}, {}.'.format(man_latitude,
```

```
The geograpical coordinate of Manhattan are 40.7900869, -73.9598295.
```

Exploring first 100 venues in Mahattan Borough with in 500 meters radius

The following code generates a URL

```
LIMIT=100

radius=500
url_man = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secr
    CLIENT_ID,  #client id generated from foursqaure
    CLIENT_SECRET,
    VERSION,
 man_latitude,
  man_longitude,
    radius,
    LIMIT)
url_man # display URL
```

```
52]: 'https://api.foursquare.com/v2/venues/explore?&client_id=D5P5G0EQI0J54NBGZGDC
     0GJZU4&v=20190117&ll=40.7900869,-73.9598295&radius=500&limit=100'
```

3. Processing the URL obtained

```
results_man = requests.get(url_man).json()
results_man
```


```
3]: {'meta': {'code': 200, 'requestId': '5d1a4630e7065500250f981c'},
     'response': {'groups': [{'items': [{'reasons': {'count': 0,
          'items': [{'reasonName': 'globalInteractionReason',
            'summary': 'This spot is popular',
            'type': 'general'}]},
         'referralId': 'e-0-4a5a4eb2f964a52021ba1fe3-0',
         'venue': {'categories': [{'icon': {'prefix': 'https://ss3.4sqi.net/img/categories_v2/parks_outd
              'suffix': '.png'},
            'id': '4bf58dd8d48988d163941735',
            'name': 'Park',
            'pluralName': 'Parks',
            'primary': True,
            'shortName': 'Park'}],
          'id': '4a5a4eb2f964a52021ba1fe3',
          'location': {'address': 'Central Park',
           'cc': 'US',
           'city': 'New York',
           'country': 'United States',
           'crossStreet': 'at 97th St',
```


The result is a Json file.

4. Function to retrieve venues across each Neighborhood in Manhattan and using this function and making calls to Foursquare API lopping through each Neighborhood in NYC Master Data set.

```
def getNearbyVenues(names, latitudes, longitudes, radius=500):

    venues_list=[]
    for name, lat, lng in zip(names, latitudes, longitudes):
        print(name)

        # create the API request URL
        url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={}&v={}&ll={},{}&radius={}&limit={}'.format(
            CLIENT_ID,
            CLIENT_SECRET,
            VERSION,
            lat,
            lng,
            radius,
            LIMIT)

        # make the GET request
        results = requests.get(url).json()["response"]['groups'][0]['items']

        # return only relevant information for each nearby venue
        venues_list.append([(
            name,
            lat,
            lng,
            v['venue']['name'],
            v['venue']['location']['lat'],
            v['venue']['location']['lng'],
            v['venue']['categories'][0]['name']) for v in results])

    nearby_venues = pd.DataFrame([item for venue_list in venues_list for item in venue_list])
    nearby_venues.columns = ['Neighborhood', 'Neighborhood Latitude','Neighborhood Longitude',
                  'Venue', 'Venue Latitude', 'Venue Longitude','Venue Category']
return(nearby_venues)
```

The result of this Function is collection of all venues corresponding to each Neighborhood resulting into a data frame containing Latitude, longitude, Venue, Venue category, Neighborhood information.

Now we use this Function to write the code to run the above function on each neighborhood and create a new data frame called

manhattan_venues (For Borough Manhattan)

Brooklyn_venues(For Borough Brooklyn)

Queens_venues(For Borough Queens)

SI__venues(For Borough Staten Island)

BR_data.shape(For Borough Bronx)

For Example: manhattan_venues

```
print(manhattan_venues.shape)
manhattan_venues.head()
```

```
(3142, 7)
```

8]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	ALPHABET CITY	40.725	-73.980	Sunny & Annie Gourmet Deli	40.725	-73.982	Deli / Bodega
1	ALPHABET CITY	40.725	-73.980	Alphabet City Beer Co.	40.724	-73.979	Beer Bar
2	ALPHABET CITY	40.725	-73.980	Bobwhite Counter	40.724	-73.979	Fried Chicken Joint
3	ALPHABET CITY	40.725	-73.980	Sake Bar Satsko	40.725	-73.980	Sake Bar
4	ALPHABET CITY	40.725	-73.980	The Wayland	40.725	-73.978	Cocktail Bar

And we can also see number of unique venue categories returned by Manhattan Venues.

```
print('There are {} uniques categories.'.format(len(manhattan_venues['Venue Category'].unique())))
```

```
There are 283 uniques categories.
```

5. One -hot coding -Analyzing each Neighborhood

One-Hot encoding helps analyze frequency of each category (Venues) in a Neighborhood.

For Example: Manhattan

```
manhattan_onehot.head()
```

1]:

	Neighborhood	Accessories Store	African Restaurant	American Restaurant	Amphitheater	Animal Shelter	Antique Shop	Arcade	Arepa Restaurant	Argentinian Restaurant	...	Vietnamese Restaurant
0	ALPHABET CITY	0	0	0	0	0	0	0	0	0	...	0
1	ALPHABET CITY	0	0	0	0	0	0	0	0	0	...	0
2	ALPHABET CITY	0	0	0	0	0	0	0	0	0	...	0
3	ALPHABET CITY	0	0	0	0	0	0	0	0	0	...	0
4	ALPHABET CITY	0	0	0	0	0	0	0	0	0	...	0

Group by rows by each neighborhood and by taking the mean of the frequency of occurrence of each category.

```
manhattan_grouped = manhattan_onehot.groupby('Neighborhood').mean().reset_inde
manhattan_grouped
```

2]:

	Neighborhood	Accessories Store	African Restaurant	American Restaurant	Amphitheater	Animal Shelter	Antique Shop
0	ALPHABET CITY	0.000	0.000	0.020	0.000	0.000	0.000
1	CHELSEA	0.000	0.000	0.020	0.000	0.000	0.000
2	CHINATOWN	0.000	0.000	0.000	0.000	0.010	0.000

6. Displaying top 10 most common venue categories for Each Neighborhood.

For Example: in Borough Manhattan

```
neighborhoods_venues_sorted_man.head()
```

4]:

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0	ALPHABET CITY	Cocktail Bar	Bar	Coffee Shop	Wine Bar	Italian Restaurant	Garden	Latin American Restaurant	Dessert Shop	Eastern European Restaurant	Nightclub
1	CHELSEA	Art Gallery	Italian Restaurant	Coffee Shop	Ice Cream Shop	Health & Beauty Service	Theater	Bagel Shop	Bakery	French Restaurant	Café
2	CHINATOWN	Chinese Restaurant	Bakery	Vietnamese Restaurant	Bubble Tea Shop	Salon / Barbershop	Italian Restaurant	Dessert Shop	Spa	Malay Restaurant	Noodle House
3	CIVIC CENTER	Chinese Restaurant	Sandwich Place	Dim Sum Restaurant	Coffee Shop	Vietnamese Restaurant	Bakery	Park	Optical Shop	Dessert Shop	Bubble Tea Shop
4	EAST VILLAGE	Ice Cream Shop	Coffee Shop	Chinese Restaurant	Japanese Restaurant	Seafood Restaurant	Ramen Restaurant	Sushi Restaurant	Dessert Shop	Pizza Place	Pet Store

Note: The snap shots are taken to show as results are from Borough Manhattan.

The above process is carried out for Each Borough and their neighborhoods (Brooklyn, Bronx, Staten Island, Queens, Manhattan).

The result sets for Boroughs Brooklyn, Bronx, Staten Island, Queens:

Brooklyn: Top 10 most common Venues

```
neighborhoods_venues_sorted_brook.head()
```

]:

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0	BATH BEACH	Pizza Place	Supplement Shop	Cantonese Restaurant	Japanese Restaurant	Bank	Pharmacy	Italian Restaurant	Restaurant	Rental Car Location	Tea Room
1	BAY RIDGE	Spa	Pizza Place	Bar	Coffee Shop	Grocery Store	Bakery	Bagel Shop	Mexican Restaurant	Italian Restaurant	American Restaurant
2	BEDFORD STUYVESANT	Café	Pizza Place	Coffee Shop	Bar	Caribbean Restaurant	Boutique	Wine Shop	Nightclub	Sandwich Place	Fried Chicken Joint

Queens: Top 10 most common Venues

```
neighborhoods_venues_sorted_Q.head()
```

3]:

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0	AIRPORT LA GUARDIA	Airport Lounge	Pizza Place	Burger Joint	Bakery	Bagel Shop	Coffee Shop	Electronics Store	Bar	Sandwich Place	Pub
1	ARVERNE	Beach	Deli / Bodega	Playground	Boat or Ferry	Café	Grocery Store	Gas Station	Women's Store	Event Space	Fish Market
2	ASTORIA	Pizza Place	Italian Restaurant	Thrift / Vintage Store	Library	Event Space	Mexican Restaurant	Residential Building (Apartment / Condo)	Bar	Convenience Store	Grocery Store

Bronx: Top 10 most common Venues

```
neighborhoods_venues_sorted_BR.head()
```

6]:

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0	BATHGATE	Bar	Breakfast Spot	Liquor Store	American Restaurant	Pharmacy	Restaurant	Other Repair Shop	Bakery	Market	Food
1	BAYCHESTER	Pharmacy	Donut Shop	Bus Station	Historic Site	Bike Trail	Pizza Place	Sandwich Place	Café	Bus Line	Deli / Bodega
2	BELMONT	Bowling Alley	Bar	Diner	Pharmacy	Flower Shop	Moving Target	Food Truck	Food & Drink Shop	Food	Fast Food Restaurant

Staten Island: Top 10 most common Venues

```
neighborhoods_venues_sorted_SI.head()
```

1]:

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0	ANNADALE	Pizza Place	Restaurant	Bakery	Construction & Landscaping	Cosmetics Shop	Dance Studio	Deli / Bodega	Pub	Diner	Bus Stop
1	ARROCHAR	Pizza Place	Bus Stop	Cosmetics Shop	Park	Deli / Bodega	Bagel Shop	Bakery	Discount Store	Event Space	Elementary School

K-means Clustering and Elbow Method

K-means is an unsupervised learning methods of clustering unlabeled data into k clusters.

K-means is used in this project to cluster Neighborhoods of NYC and their Boroughs.

```
X = df_Brough_avgsales[['Latitude', 'Longitude']].values
```

k-clusters

```
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt

k_c=range(1,6)
kmean = [KMeans(n_clusters=i).fit(X) for i in k_c]
kmean
```

```
[131]: [KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
            n_clusters=1, n_init=10, n_jobs=1, precompute_distances='auto',
            random_state=None, tol=0.0001, verbose=0),
        KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
            n_clusters=2, n_init=10, n_jobs=1, precompute_distances='auto',
            random_state=None, tol=0.0001, verbose=0),
        KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
            n_clusters=3, n_init=10, n_jobs=1, precompute_distances='auto',
            random_state=None, tol=0.0001, verbose=0),
        KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
            n_clusters=4, n_init=10, n_jobs=1, precompute_distances='auto',
            random_state=None, tol=0.0001, verbose=0),
        KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
            n_clusters=5, n_init=10, n_jobs=1, precompute_distances='auto',
            random_state=None, tol=0.0001, verbose=0)]
```

- Feature Selection

I have used Longitude and latitude of 5 Borough to calculate clusters

The dataset of neighborhood venues for all 5 boroughs in New York City is consolidated to one dataset. For each venue category, the mean of frequency of venues across each neighborhood was calculated. This information would then be used to fit a K-Means clustering algorithm to the data to determine neighborhoods of similar venue profile.

First, the total number of venues for each category was determined:

For Example:

The result of One -hot Encoding was taken and Group by was applied to rows by each neighborhood and by taking the mean of the frequency of occurrence of each category.

```
manhattan_grouped = manhattan_onehot.groupby('Neighborhood').mean().reset_inde
manhattan_grouped
```

2]:

	Neighborhood	Accessories Store	African Restaurant	American Restaurant	Amphitheater	Animal Shelter	Antique Shop
0	ALPHABET CITY	0.000	0.000	0.020	0.000	0.000	0.000
1	CHELSEA	0.000	0.000	0.020	0.000	0.000	0.000
2	CHINATOWN	0.000	0.000	0.000	0.000	0.010	0.000

Note: I m just showing the results of Borough Manhattan

- I have applied One – Hot encoding to all 5 Boroughs.

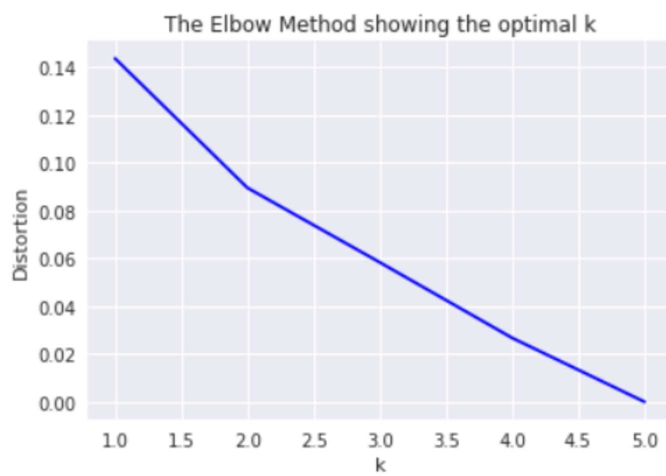
Elbow Method

Determining optimal k

The technique to determine K, the number of clusters, is called the elbow method.

Values for k on horizontal axis and Distortion (% of variance) Vertical axis

1.When K increases, the centroids are closer to the cluster's centroids. The improvements will decline, creating the elbow shape.



From the figure we can say that optimal Value of K=2

- Creating cluster labels for all 5 Boroughs

Since I have used Foursquare API to Analyze each Borough and its neighborhoods,

Cluster labels for all 5 Boroughs have been created and the added to their corresponding Datasets.

The snap shots should Explain the process more clearly:

1. Manhattan Clusters

```
kclusters = 2
# run k-means clustering
kmeans_man = KMeans(n_clusters=kclusters, random_state=0).fit(manhattan_grouped[manhattan_grouped.columns[1:284]])

# check cluster labels generated for each row in the dataframe
kmeans_man.labels_[0:10]
```

```
2]: array([1, 1, 1, 1, 1, 0, 1, 1, 1, 1], dtype=int32)
```

2.Brooklyn-Cluster lables

```
#brook_grouped.shape
kclusters = 2
# run k-means clustering
kmeans_brook = KMeans(n_clusters=kclusters, random_state=0).fit(brook_grouped[brook_grouped.columns[1:256]])

# check cluster labels generated for each row in the dataframe
kmeans_brook.labels_[0:10]
```

```
3]: array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0], dtype=int32)
```

3.Queens - cluster lables

```
#Q_grouped.shape
kclusters = 2
# run k-means clustering
kmeans_Q = KMeans(n_clusters=kclusters, random_state=0).fit(Q_grouped[Q_grouped.columns[1:211]])

# check cluster labels generated for each row in the dataframe
kmeans_Q.labels_[0:10]
```

```
4]: array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0], dtype=int32)
```

5.Staten Island - Cluster Lables

```
kclusters = 2

# run k-means clustering
kmeans_SI = KMeans(n_clusters=kclusters, random_state=0).fit(SI_grouped[SI_grouped.columns[1:153]])

# check cluster labels generated for each row in the dataframe
kmeans_SI.labels_[0:10]
```

```
t[146]: array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0], dtype=int32)
```


5.Staten Island - Cluster Lables

```
kclusters = 2

# run k-means clustering
kmeans_SI = KMeans(n_clusters=kclusters, random_state=0).fit(SI_grouped[SI_grouped.columns[1:153]])

# check cluster labels generated for each row in the dataframe
kmeans_SI.labels_[0:10]
```

```
t[146]: array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0], dtype=int32)
```


Adding the labels created above to Corresponding Boroughs:

```
## Adding Cluster Lables --Manhattan
neighborhoods_venues_sorted_man.insert(0, 'Cluster Labels', kmeans_man.labels_)

## Adding Cluster Lables --Brooklyn
neighborhoods_venues_sorted_brook.insert(0, 'Cluster Labels', kmeans_brook.labels_)

## Adding Cluster Lables --Qunees
neighborhoods_venues_sorted_Q.insert(0, 'Cluster Labels', kmeans_Q.labels_)

## Adding Cluster Lables --Bronx
neighborhoods_venues_sorted_BR.insert(0, 'Cluster Labels', kmeans_BR.labels_)

## Adding Cluster Lables --Staten Island
neighborhoods_venues_sorted_SI.insert(0, 'Cluster Labels', kmeans_SI.labels_)
```

Let's create a new data frame that includes the cluster as well as the top 10 venues for each neighborhood for Each Borough.

The Data Frames contain top 10 common venues for all 5 boroughs and Neighborhoods with Coordinates information and cluster labels.

```
# merge to add latitude/longitude for each neighborhood

#1 Manhattan
Borough_merged1 = manhattan_data.join(neighborhoods_venues_sorted_man.set_index('Neighborhood'), on='NEIGHBORHOOD')

#2 Brooklyn
Borough_merged2 = Brooklyn_data.join(neighborhoods_venues_sorted_brook.set_index('Neighborhood'), on='NEIGHBORHOOD')

#3 Queens
Borough_merged3 = Queens_data.join(neighborhoods_venues_sorted_Q.set_index('Neighborhood'), on='NEIGHBORHOOD')

#4 Bronx
Borough_merged4 = BR_data.join(neighborhoods_venues_sorted_BR.set_index('Neighborhood'), on='NEIGHBORHOOD')

#5 Staten Island
Borough_merged5 = SI_data.join(neighborhoods_venues_sorted_SI.set_index('Neighborhood'), on='NEIGHBORHOOD')
```

Combining the data showed in the above figure to New_Merge data set with clusters labels and top 10 most common venues for all 5 Boroughs and its Neighborhoods.

The snapshots of the data set New Merge:

```
New_Merge=pd.concat([Borough_merged1,Borough_merged2, Borough_merged3,Borough_merged4,Borough_merged5],ignore_index=True)
```

```
New_Merge.head()
```

7]:

	BOROUGH	NEIGHBORHOOD	Address	coordinates	Latitude	Longitude	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0	Manhattan	ALPHABET CITY	ALPHABET CITY,NY	(Alphabet City, Manhattan Community Board 3, M...	40.725	-73.980	1	Cocktail Bar	Bar	Coffee Shop	Wine Bar	Italian Restaurant	Garden	Latin American Restaurant	Dessert Shop	Eastern European Restaurant	Nightclub
1	Manhattan	CHELSEA	CHELSEA,NY	(Chelsea, Manhattan Community Board 4, Manhatt	40.746	-74.002	1	Art Gallery	Italian Restaurant	Coffee Shop	Ice Cream Shop	Health & Beauty Service	Theater	Bagel Shop	Bakery	French Restaurant	Café

```
New_Merge['BOROUGH'].value_counts()
```

```
74]: Queens           54
     Brooklyn         54
     Staten Island    48
     Manhattan        37
     Bronx            28
     Name: BOROUGH, dtype: int64
```

Clusters:

Cluster 0

```
162]: New_Merge.loc[New_Merge['Cluster Labels'] == 0, New_Merge.columns[[1] + list(range(5, New_Merge.shape[1]))]]
```

Out[162]:

	NEIGHBORHOOD	Longitude	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
6	FASHION	19.132	0	Market	Train Station	Motel	Shopping Mall	Filipino Restaurant	Event Space	Exhibit	Falafel Restaurant	Farmers Market	Fast Food Restaurant
38	BATH BEACH	-74.001	0	Pizza Place	Supplement Shop	Cantonese Restaurant	Japanese Restaurant	Bank	Pharmacy	Italian Restaurant	Restaurant	Rental Car Location	Tea Room
39	BAY RIDGE	-74.027	0	Spa	Pizza Place	Bar	Coffee Shop	Grocery Store	Bakery	Bagel Shop	Mexican Restaurant	Italian Restaurant	American Restaurant
40	BEDFORD STUYVESANT	-73.941	0	Café	Pizza Place	Coffee Shop	Bar	Caribbean Restaurant	Boutique	Wine Shop	Nightclub	Sandwich Place	Fried Chicken Joint
41	BENSONHURST	-73.993	0	Chinese Restaurant	Pizza Place	Mobile Phone Shop	Bubble Tea Shop	Japanese Restaurant	Cantonese Restaurant	Bank	Bakery	Gift Shop	Gourmet Shop
42	BERGEN BEACH	-73.907	0	Deli / Bodega	Peruvian Restaurant	Playground	Italian Restaurant	Pizza Place	Supermarket	Chinese Restaurant	Donut Shop	Sushi Restaurant	Fish Market
43	BOERUM HILL	-73.984	0	Spa	Coffee Shop	Sandwich Place	Bar	Yoga Studio	Hotel	Dance Studio	Middle Eastern Restaurant	Cosmetics Shop	Cocktail Bar

Cluster 1

Cluster 1

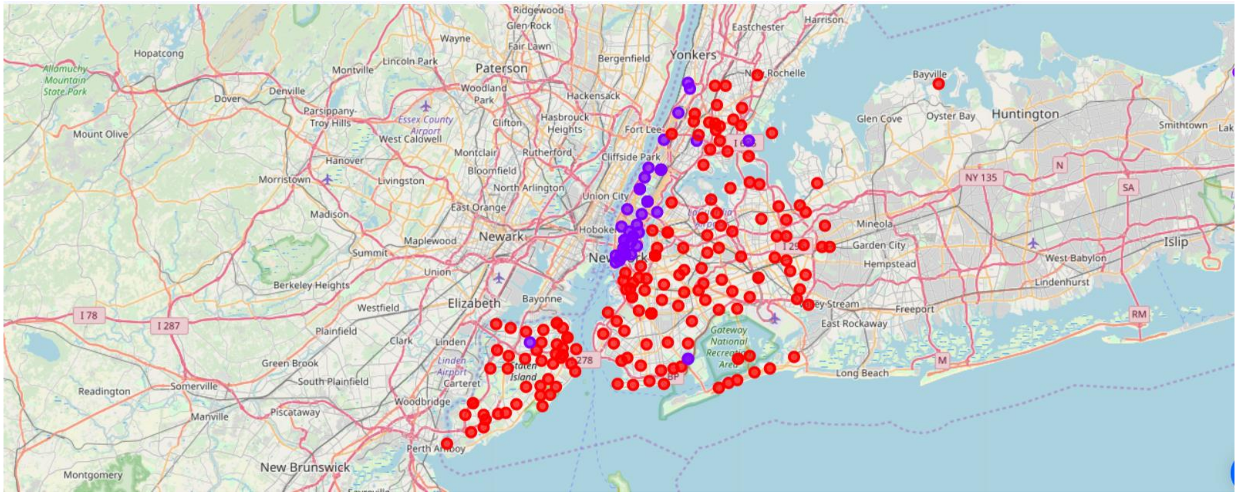
```
New_Merge.loc[New_Merge['Cluster Labels'] == 1, New_Merge.columns[[1] + list(range(5, New_Merge.shape[1]))]]
```

[163]:

	NEIGHBORHOOD	Longitude	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0	ALPHABET CITY	-73.980	1	Cocktail Bar	Bar	Coffee Shop	Wine Bar	Italian Restaurant	Garden	Latin American Restaurant	Dessert Shop	Eastern European Restaurant	Nightclub
1	CHELSEA	-74.002	1	Art Gallery	Italian Restaurant	Coffee Shop	Ice Cream Shop	Health & Beauty Service	Theater	Bagel Shop	Bakery	French Restaurant	Café
2	CHINATOWN	-73.996	1	Chinese Restaurant	Bakery	Vietnamese Restaurant	Bubble Tea Shop	Salon / Barbershop	Italian Restaurant	Dessert Shop	Spa	Malay Restaurant	Noodle House
3	CIVIC CENTER	-74.002	1	Chinese Restaurant	Sandwich Place	Dim Sum Restaurant	Coffee Shop	Vietnamese Restaurant	Bakery	Park	Optical Shop	Dessert Shop	Bubble Tea Shop
5	EAST VILLAGE	-73.987	1	Ice Cream Shop	Coffee Shop	Chinese Restaurant	Japanese Restaurant	Seafood Restaurant	Ramen Restaurant	Sushi Restaurant	Dessert Shop	Pizza Place	Pet Store
7	FINANCIAL	-74.009	1	Coffee Shop	American Restaurant	Pizza Place	Hotel	Juice Bar	Steakhouse	Café	Wine Shop	Sandwich Place	Gym

Data Visualization

Map with Cluster Label's:



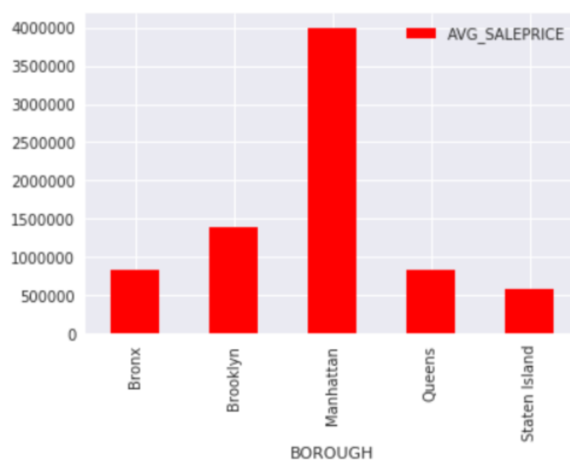
Analyzing Sale Price and other Features

- **Borough with Highest Average Sales Price**

<matplotlib.axes._subplots.AxesSubplot at 0x7ff50a5361d0>

[16]:

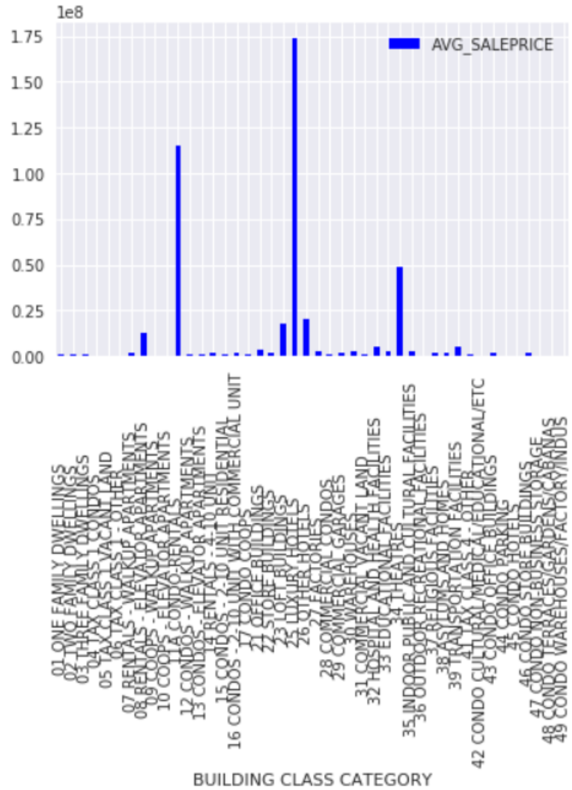
	SALEPRICE
BOROUGH	
Manhattan	3992252.023
Brooklyn	1389220.312
Queens	829326.043
Bronx	825982.163
Staten Island	577052.335



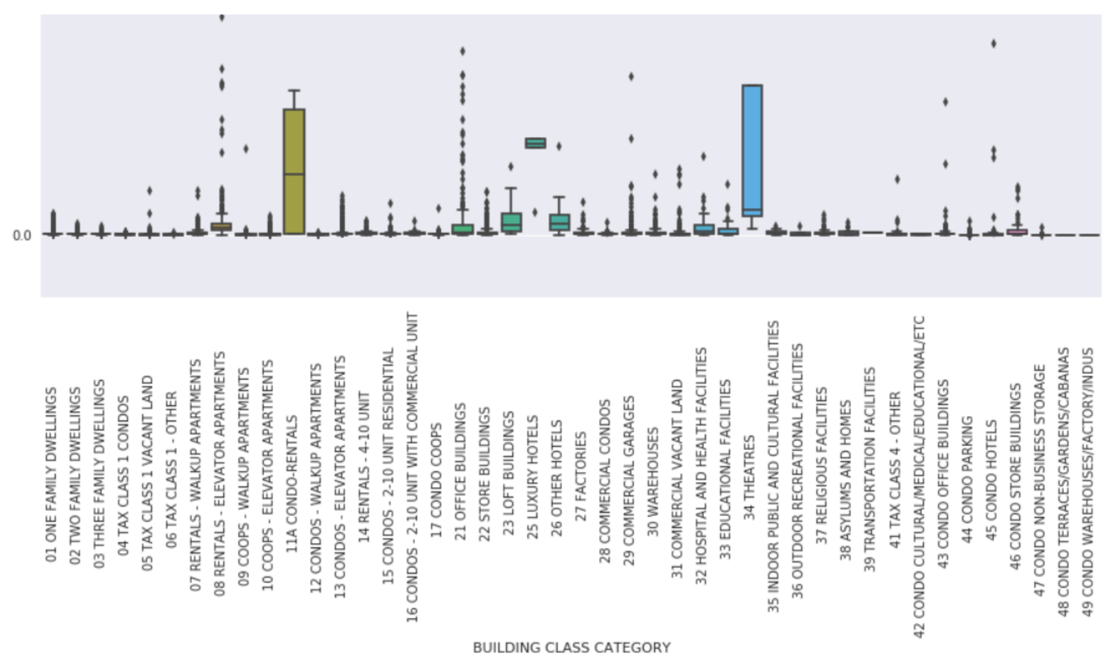
The figure clearly shows that Borough Manhattan has the highest Averaged sale Priced Properties.

- **Building Class Category**

BUILDING CLASS CATEGORY	AVG_SALEPRICE
25 LUXURY HOTELS	173926237.000
11A CONDO-RENTALS	115175000.000
34 THEATRES	48490000.000
26 OTHER HOTELS	20500000.000
23 LOFT BUILDINGS	17705951.000
08 RENTALS - ELEVATOR APARTMENTS	12900000.000
32 HOSPITAL AND HEALTH FACILITIES	5300486.500
39 TRANSPORTATION FACILITIES	5000000.000
21 OFFICE BUILDINGS	3700000.000
35 INDOOR PUBLIC AND CULTURAL FACILITIES	2808000.000
33 EDUCATIONAL FACILITIES	2600000.000
27 FACTORIES	2575000.000
30 WAREHOUSES	2400000.000
38 ASYLUMS AND HOMES	2008987.000
46 CONDO STORE BUILDINGS	1929831.000
22 STORE BUILDINGS	1875000.000
29 COMMERCIAL GARAGES	1830000.000



- **Sale Price Distribution over Building class category**



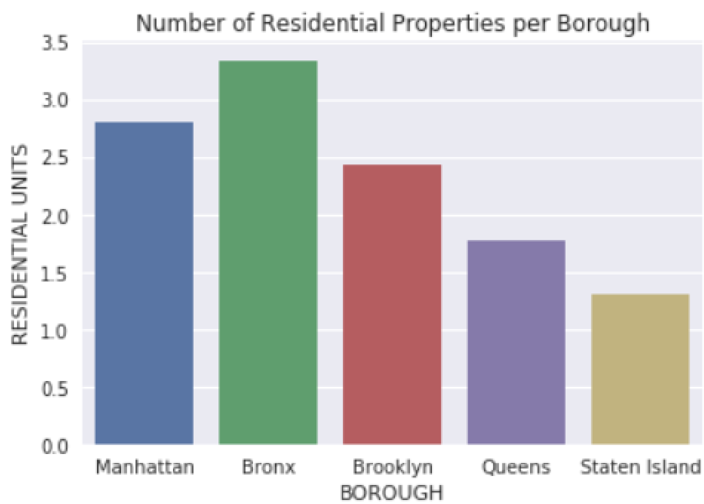
From the figure

Observation: From the above plot we can state that

25 LUXURY HOTELS, 11A CONDO-RENTALS, 34 THEATRES

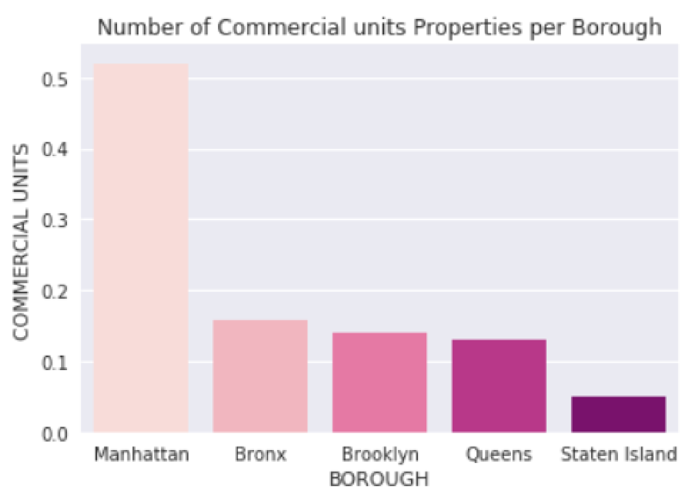
are highly Priced Building Class Categories

- **Residential Properties Per Borough**



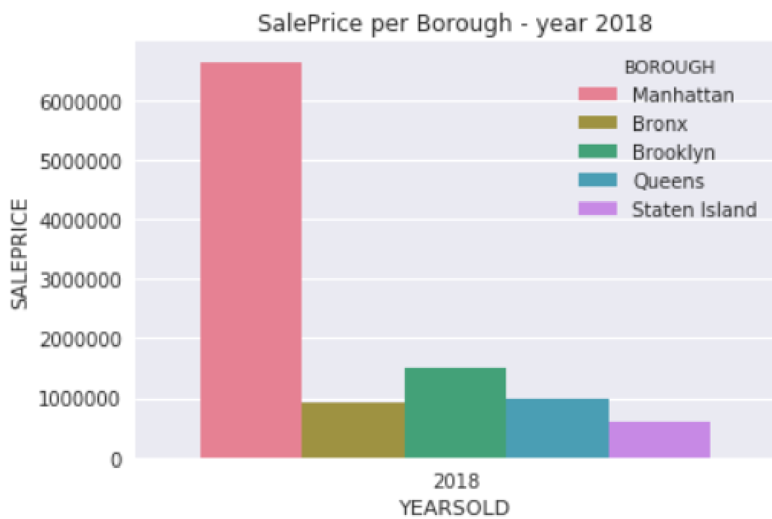
This plot shows that Borough Bronx has more Residential properties than the other Boroughs.

- **Commercial Properties per Borough**



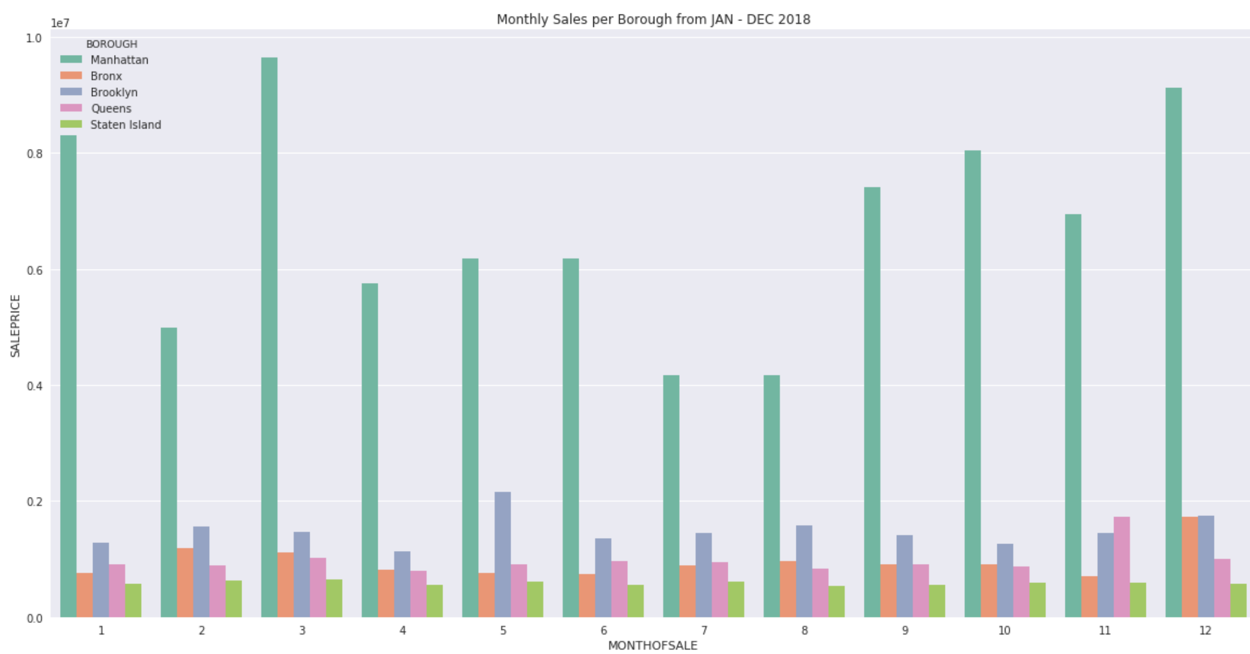
This plot shows That Manhattan houses the highest number of commercials properties.

- **Borough with Highest SalePrice for sale year -2o18**



Borough Manhattan = Highest Sale Priced Properties in Year -2018

- **Monthly Sales Per Borough from Jan -Dec 2018**



From the plots above we can say that Borough Manhattan has highest priced properties.

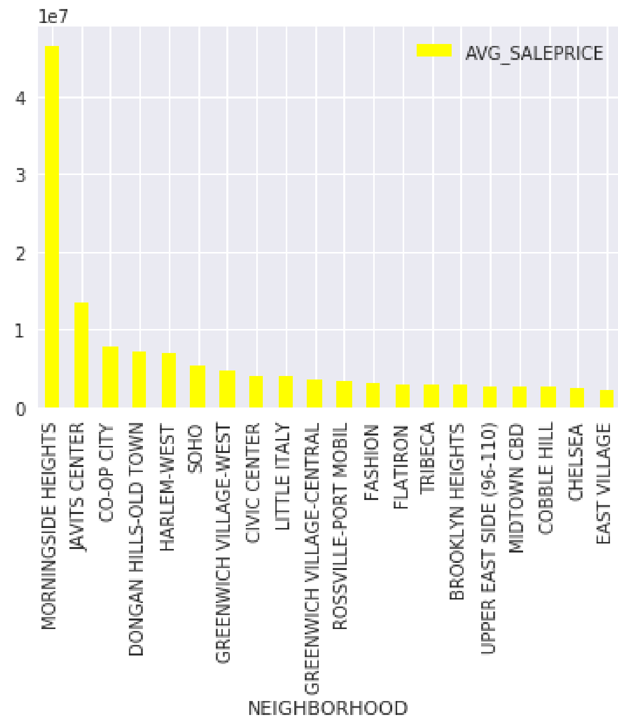
In year 2018 Manhattan has sold highest valued Properties

Borough Manhattan experienced high Property sale Prices in Jan and Dec months of year -2018

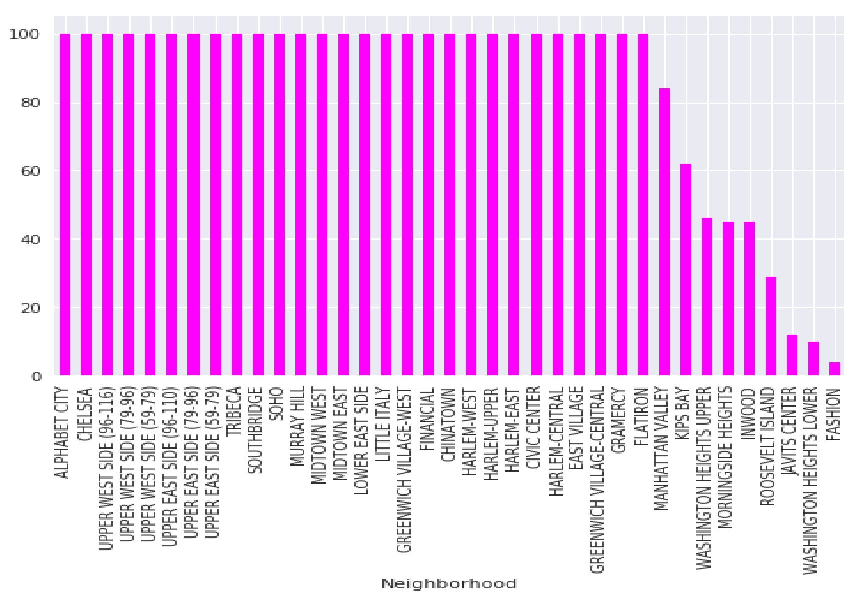
- **Average Sale Price of top 20 Neighborhoods**

As we can see from the figure below Neighborhoods Morning side heights, Javits center

Co-op city are top three Neighborhoods with highest property sale price, and cobble hill, East Village, Chelsea being the bottom three neighborhoods with least priced Properties.

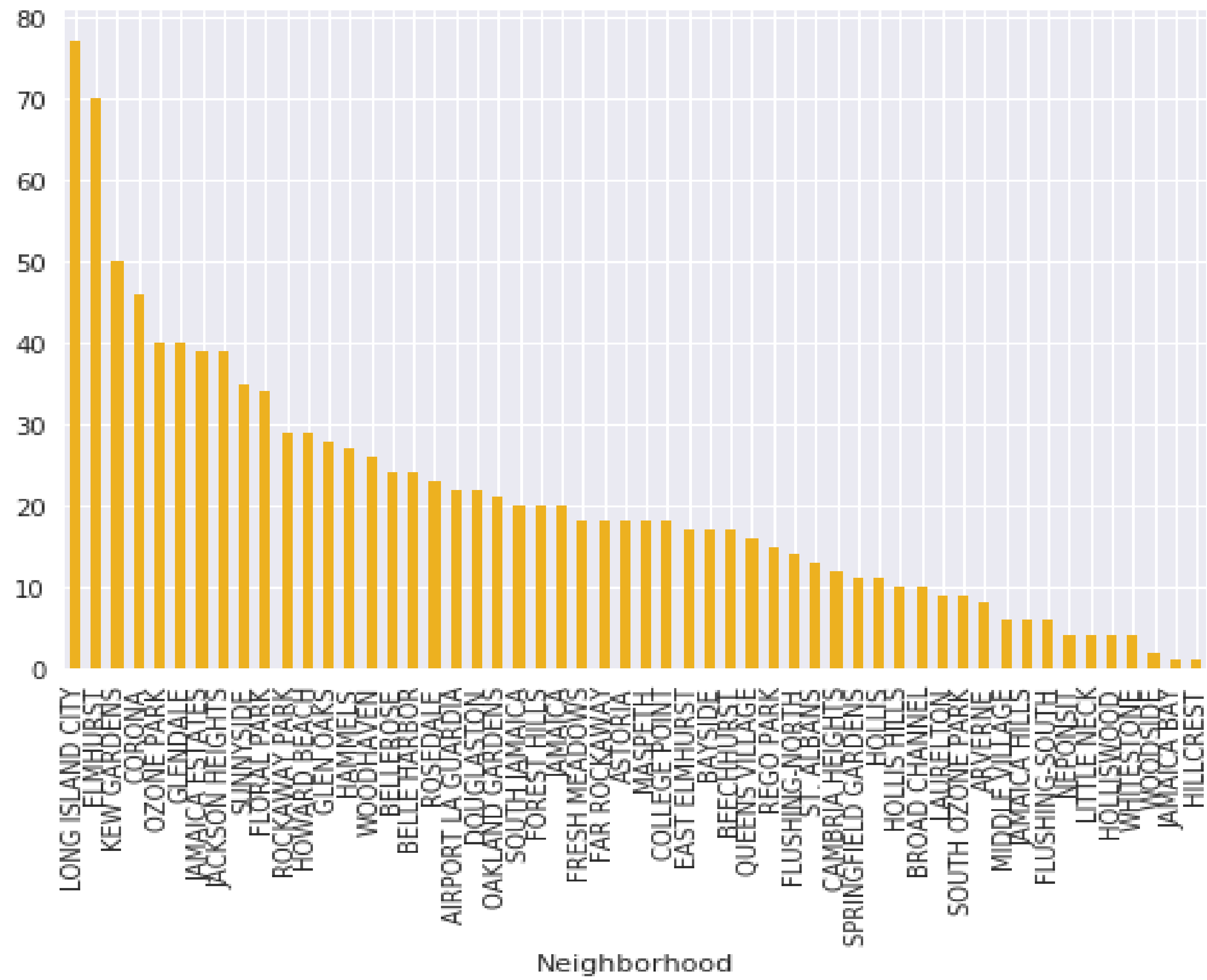


- **Number of venues per neighborhood in Manhattan Borough?**



As the figure shows majority of the Neighborhoods has many venues.

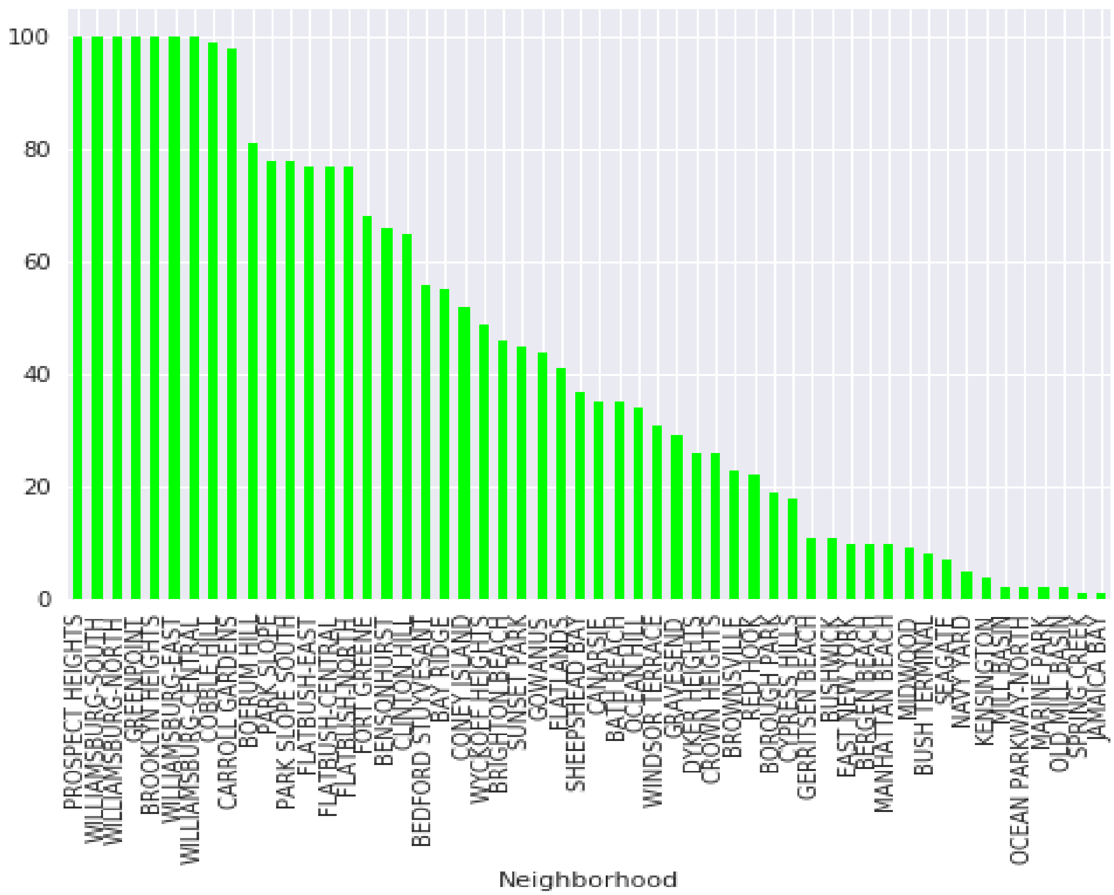
- **Number of venues per neighborhood in Queens Borough?**



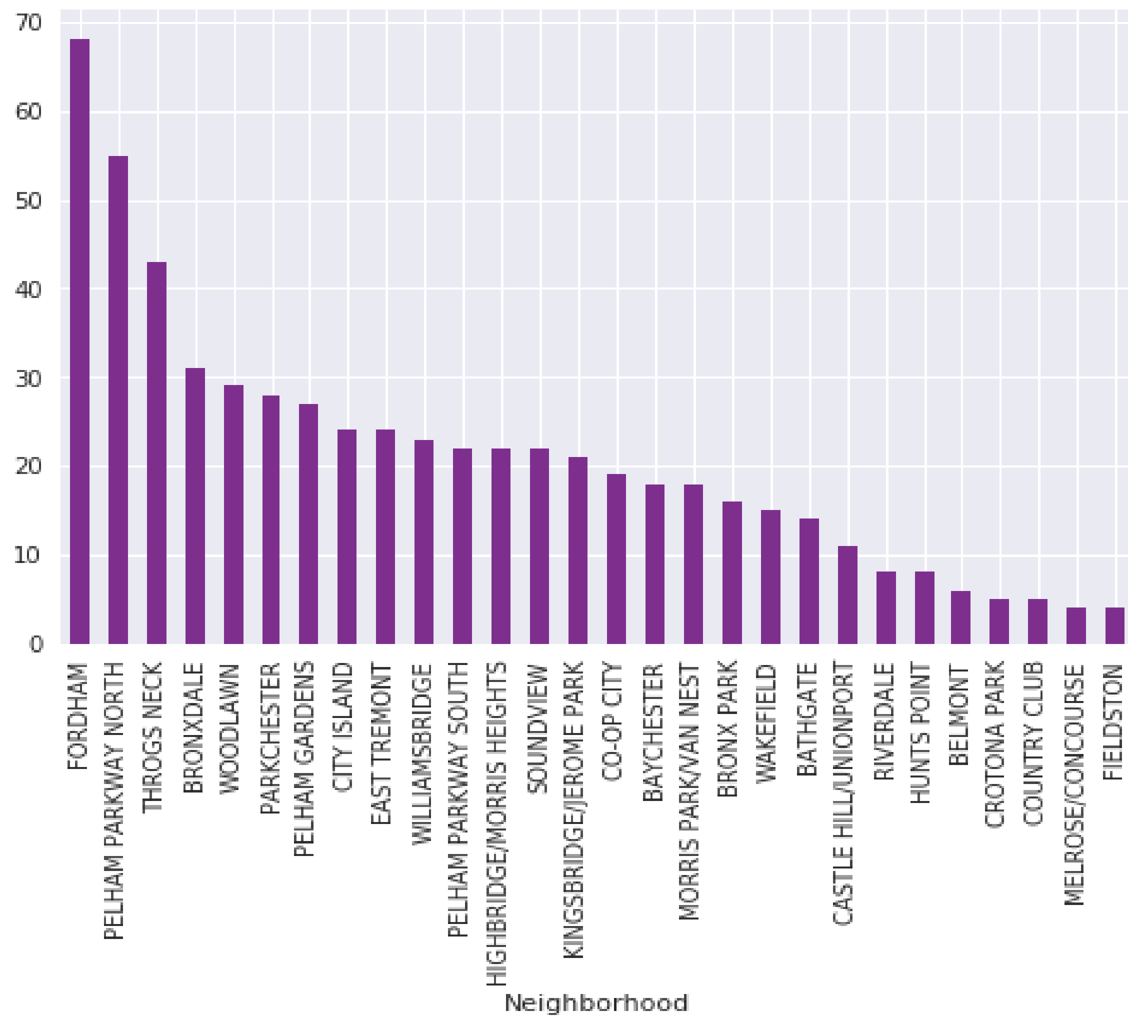
Neighborhoods Long Island City, Elmhurst, Kew Gardens have a greater number of venues then rest of the Neighborhoods in Queens Borough.

- **Number of venues per neighborhood in Brooklyn Borough?**

As we can see from the figure Prospect Heights, Williamsburg-south, Williamsburg-North, Green points and Brooklyn Heights have larger group of venues in Brooklyn Borough. Whereas Jamaica Bay, Spring creek and old mill basin have least number of venues.

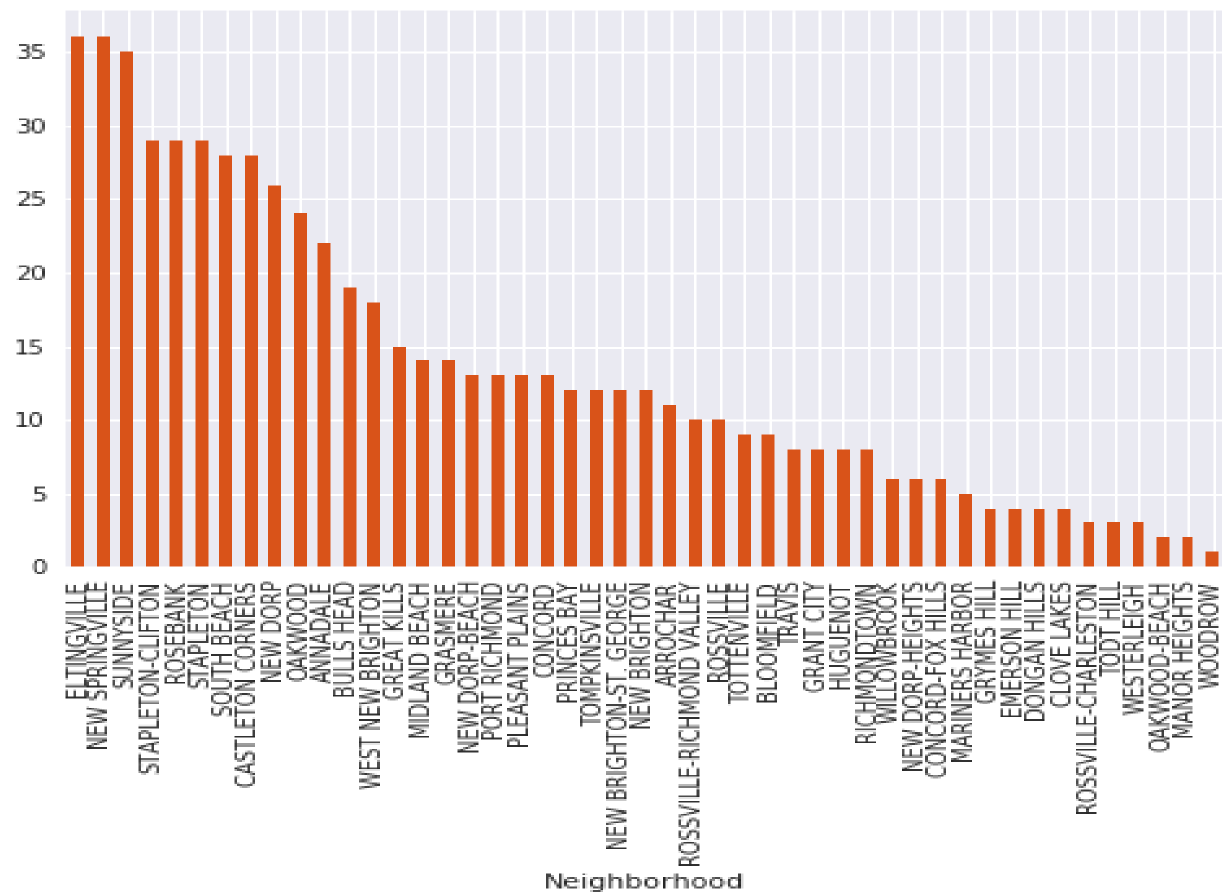


- **Number of venues per neighborhood in Bronx Borough?**



Fordham, Pelham parkway station and Throgs Neck Neighborhood have a greater number of venues than the rest of the neighborhoods in Bronx.

- **Number of venues per neighborhood in Staten Island Borough?**



From the figure above Eltingville , New Springville , Sunside hold maximum number of venues in Staten Island Borough.

- **Top 2 Neighborhoods with Max sale price in Each borough.**

Manhattan: CHELSEA, UPPER WEST SIDE (59-79)

Brooklyn: SPRING CREEK, RED HOOK

Queens: LONG ISLAND CITY, REGO PARK

Bronx: WESTCHESTER, PELHAM GARDENS

Staten Island: ROSEBANK, ROSSVILLE-CHARLESTON

```
Text(0.5,1,'Top 2 Neighbhorhoods with MAX SalePrice across Each Borough')
```