

Introduction

An eager venture capitalist is interested in investing into a hot new restaurant in Brooklyn, NY. This will be a new construction and investment. This will initially focus on finding the best neighborhood based on the number of restaurants within a neighborhood.

The data will come from a public New York City source and will focus on the borough of Brooklyn. Foursquare data will also be integrated to get coordinates and venue info to establish a better idea in which what area in Brooklyn would be the best to invest in.

Data

Data used is from a public website for New York City. The data consists of all boroughs in New York City and their corresponding coordinates. In order to help solve the problem, data will be used to determine all data for the borough of Brooklyn and coordinates to use in conjunction with Foursquare data will use for extracting venue information.

Data source:

https://geo.nyu.edu/catalog/nyu_2451_34572

Pre-Processing

The pre-processing of the data is done by pulling data from the original data source into a Python data frame and then integrated with Foursquare data to find venues to corresponding neighborhoods and coordinates

Importing data from source

```
Luckily, New York City dataset exists for free on the web. https://geo.nyu.edu/catalog/nyu_2451_3457
         I downloaded the files and placed it on the server to import easier.
In [2]: []wget -q -0 'newyork_data.json' https://ibm.box.com/shared/static/fbpwbovar71f8p5sgddm06cgipa2rxp
         print('Data downloaded!')
         Data downloaded!
         Load and explore the data
         Next, let's load the data
In [4]: with open('newyork data.ison') as ison data:
             newyork data = json.load(json data)
         Let's take a quick look at the data
In [4]: newyork data
Out[4]: {'type': 'FeatureCollection',
           'totalFeatures': 306.
          'features': [{'type': 'Feature',
            'id': 'nyu 2451 34572.1',
             'geometry': {'type': 'Point'
              coordinates': [-73.84720052054902, 40.89470517661]},
            'geometry name': 'geom',
             'properties': {'name': 'Wakefield',
              'stacked': 1,
              'annolinel': 'Wakefield',
              'annoline2': None,
              'annoline3': None,
              'annoangle': 0.0.
              'borough': 'Bronx'
              'bbox': [-73.84720052054902,
              40.89470517661,
              -73.84720052054902,
              40.89470517661]}},
           {'type': 'Feature',
             id': 'nyu_2451_34572.2',
             geometry': {'type': 'Point'
              coordinates': [-73.82993910812398, 40.87429419303012]},
             'geometry_name': 'geom',
             'properties': {'name': 'Co-op City',
              stacked': 2,
```

Importing data into a python dataframe

The next task is essentially transforming this data of nested Python dictionaries into a pandas dataframe. So let's start by creating an empty dataframe.

```
In [10]: # define the dataframe columns
    column_names = ['Borough', 'Neighborhood', 'Latitude', 'Longitude']

# instantiate the dataframe
    neighborhoods = pd.DataFrame(columns=column_names)
```

Take a look at the empty dataframe to confirm that the columns are as intended.

```
Out[11]: Borough Neighborhood Latitude Longitude
```

Then let's loop through the data and fill the dataframe one row at a time.

Quickly examine the resulting dataframe.

In [13]: neighborhoods.head()

Out[13]: Borough Neighborhood Latitude Longitude o Bronx Wakefield 40.894705 -73.847201 1 Bronx 40.874294 73.829939 Co-op City -73.827806 2 Bronx **Fastchester** 40.887556 40.895437 -73.905643 3 Bronx Fieldston Riverdale 40.890834 -73.912585

Pre-Processing (cont'd)

Foursquare Data is pulled for further processing

Focused on Top 20 Venues in Brooklyn and created new dataframe for the venues

Now, let's get the top 20 yeaues that are in Brooklyn within a radius of 500 meters

First, let's create the GET request URL. Name your URL url.

```
In [21]: # type your answer here
         #<!-- The correct answer is:
         LIMIT = 20 # limit of number of venues returned by Foursquare API
         #<1--
         radius = 500 # define radius
         ##\\ # create URL
         url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={}&v={}&il={},{}&
         radius={}&limit={}'.format(
             CLIENT ID,
             CLIENT SECRET,
             neighborhood latitude,
             neighborhood longitude,
             radius,
             LIMIT)
         url # display URL
Out[21]: 'https://api.foursquare.com/v2/venues/explore?&client id=ORJ4KlW3IW3A50CGBGY3LY4SNONMECQNSZJ2FN2M
         2UHWIUFG&client secret=B4IFZSJU4DRSW4Z2CG5RRF1DYM41NU2B04DFPBFSBHUSNH42&v=20180605&l1=40.62580106
         5010656.-74.03062069353813&radius=500&limit=20
In [22]: results = requests.get(url).json()
Out[22]: {'meta': {'code': 200, 'requestId': '5c1b1860f594df03ec9b713b'},
           'response': {'suggestedFilters': {'header': 'Tap to show:',
            'filters': [{'name': '$-$$$$', 'key': 'price'},
             {'name': 'Open now', 'key': 'openNow'}]},
            'headerLocation': 'Bay Ridge',
            'headerFullLocation': 'Bay Ridge, Brooklyn',
            'headerLocationGranularity': 'neighborhood',
            'totalResults': 92,
            'suggestedBounds': {'ne': {'lat': 40.63030106951066,
              'lng': -74.02470273356597},
             'sw': {'lat': 40.62130106051065, 'lng': -74.03653865351028}},
           'groups': [{'type': 'Recommended Places',
```

Now there's a dataframe with venue name and coordinates

Before we proceed, let's borrow the get_category_type function from the Foursquare lab.

```
In [23]: # function that extracts the category of the venue
def get_category_type(row):
    try:
        categories_list = row['categories']
    except:
        categories_list = row['venue.categories']

if len(categories_list) == 0:
    return None
else:
    return categories_list[0]['name']
```

Now we are ready to clean the json and structure it into a pandas dataframe.

Out[24]:

:		name	categories	lat	Ing
	0	Pilo Arts Day Spa and Salon	Spa	40.624748	-74.030591
	1	Bagel Boy	Bagel Shop	40.627896	-74.029335
	2	Cocoa Grinder	Juice Bar	40.623967	-74.030863
	3	Pegasus Cafe	Breakfast Spot	40.623168	-74.031186
	4	Ho' Brah Taco Joint	Taco Place	40.622960	-74.031371

In [25]: #let's see how many venues returned
print('{} venues were returned by Foursquare.'.format(nearby_venues.shape[0]))

20 venues were returned by Foursquare.

Pre-Processing (cont'd)

Let's assign the corresponding neighborhoods in Brooklyn. Below is the dataframe with neighborhoods, venue, coordinates, venue category. Also, knowing a count of each neighborhood gives an idea of what we are working with.

In [28]: #### Let's check the size of the resulting dataframe
 print(brooklyn_venues.shape)
 brooklyn_venues.head()

(1231, 7)

Out[28]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Bay Ridge	40.625801	-74.030621	Pilo Arts Day Spa and Salon	40.624748	-74.030591	Spa
1	Bay Ridge	40.625801	-74.030621	Bagel Boy	40.627896	-74.029335	Bagel Shop
2	Bay Ridge	40.625801	-74.030621	Cocoa Grinder	40.623967	-74.030863	Juice Bar
3	Bay Ridge	40.625801	-74.030621	Pegasus Cafe	40.623168	-74.031186	Breakfast Spot
4	Bay Ridge	40.625801	-74.030621	Ho' Brah Taco Joint	40.622960	-74.031371	Taco Place

Let's check how many venues were returned for each neighborhood

In [29]: ##Let's check how many venues were returned for each neighborhood brooklyn_venues.groupby('Neighborhood').count()

Out[29]:

	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
Neighborhood						
Bath Beach	20	20	20	20	20	20
Bay Ridge	20	20	20	20	20	20
Bedford Stuyvesant	20	20	20	20	20	20
Bensonhurst	20	20	20	20	20	20
Bergen Beach	6	6	6	6	6	6
Boerum Hill	20	20	20	20	20	20

Pre-Processing (cont'd)

Finally, we get total sum of restaurants for the kmeans ml experiment

```
In [41]: # one hot encoding
          brooklyn onehot = pd.qet dummies(brooklyn venues[['Venue Category']], prefix="", prefix sep="")
          # add neighborhood column back to dataframe
          brooklyn onehot['Neighborhood'] = brooklyn venues['Neighborhood']
          # move neighborhood column to the first column
          fixed columns = [brooklyn onehot.columns[-1]] + list(brooklyn onehot.columns[:-1])
          brooklyn onehot = brooklyn onehot[fixed columns]
          brooklyn_onehot.head()
Out[41]:
                           Sporting
         n/
                 Spanish
                                    Sports
                                                                  Surf
                                                                      Sushi
                                                                                                            Tattoo
                                                                                       Taiwanese
                           Goods
                                           Steakhouse Supermarket
                 Restaurant
                                    Bar
                                                                                 Place
                                                                                       Restaurant Restaurant Parlor
                                                                  Spot Restaurant
                           Shop
                           0
                                                                  0
                                                                                 0
                           0
                                    0
                                           0
                                                     0
                                                                  0
                                                                       0
                                                                                 0
                                                                                       0
                           0
                                           0
                                                     0
                                                                                 0
                                    0
                                                                  0
                                                                       0
                                                     0
                                    0
                                                                  0
                                                                       0
                                                                                 0
          And let's examine the new dataframe size.
In [43]: brooklyn onehot.shape
Out[43]: (1231, 220)
In [44]: #let's group rows by neighborhood and by taking the mean of the frequency of occurrence of each c
          brooklyn_grouped = brooklyn_onehot.groupby('Neighborhood').sum()
          brooklyn grouped
Out[44]:
                                 Vegetarian
                                            Video
                                                  Video
                                                                             Whisky
                                                                                    Wine
                                                                                         Wine
                                                                                               Women's
                       Varenyky
                                                       Vietnamese
                                            Game
                                                                   Waterfront
         rant Bookstore restaurant
                                                  Store
                                                       Restaurant
                                                                                    Bar
                                                                                         Shop Store
                                                                                                        Restaurants
                                 Restaurant
                                           Store
```

Methodology

Now we can run the kmeans ml algorithm and clustering on the dataframe created. We have Total Restaurants, Total Sum. We see that the K5 cluster has the most restaurants.

Out	⊢ T /	71	٠
Ou		: /	٠

	Yoga Studio	American Restaurant	Antique Shop	Arepa Restaurant	Argentinian Restaurant	Art Gallery	Arts & Crafts Store	Asian Restaurant	Athletics & Sports	Auto Worksh
K5	0.375000	0.000000	-6.938894e- 18	1.250000e- 01	-1.734723e- 18	1.387779e- 17	-6.938894e- 18	1.387779e- 17	-6.938894e- 18	-1.73472 18
K1	0.400000	0.533333	6.666667e- 02	6.666667e- 02	6.666667e- 02	1.333333e- 01	6.666667e- 02	1.333333e- 01	6.666667e- 02	-3.46944 18
K4	0.454545	0.272727	9.090909e- 02	-6.938894e- 18	-3.469447e- 18	9.090909e- 02	1.818182e- 01	9.090909e- 02	-1.387779e- 17	-3.46944 18
кз	0.083333	0.083333	-6.938894e- 18	-6.938894e- 18	-3.469447e- 18	8.333333e- 02	-6.938894e- 18	1.666667e- 01	8.333333e- 02	-3.46944 18
K2	0.083333	0.250000	4.166667e- 02	-6.938894e- 18	-3.469447e- 18	1.250000e- 01	1.387779e- 17	8.333333e- 02	8.333333e- 02	4.16666 02

Total Restaurants	Total Sum
7.500000	27.500000
6.533333	26.400000
3.545455	23.545455
4.000000	22.333333
1.458333	15.208333

Results

We can see what Neighborhoods belong to what Group/cluster from the kmeans experiment and also what Neighborhoods are in the Group 5

```
In [76]: #lets find a the group for each neighbourhood
    neigh_summary = pd.DataFrame([brooklyn_grouped.index, 1 + kmeans.labels_]).T
    neigh_summary.columns = ['Neighborhood', 'Group']
    neigh_summary
```

Out[76]:

	Neighborhood	Group
0	Bath Beach	1
1	Bay Ridge	1
2	Bedford Stuyvesant	4
3	Bensonhurst	1
4	Bergen Beach	2
5	Boerum Hill	1
6	Borough Park	3
7	Brighton Beach	1
8	Broadway Junction	2
9	Brooklyn Heights	1
10	Brownsville	3
11	Bushwick	5
12	Canarsie	2
13	Carroll Gardens	4
14	City Line	3
15	Clinton Hill	1
16	Cobble Hill	1
17	Coney Island	2
18	Crown Heights	2
19	Cypress Hills	5
20	Ditmas Park	5

In [77]: neigh_summary[neigh_summary['Group'] == 5]

Out[77]:

	Neighborhood	Group
11	Bushwick	5
19	Cypress Hills	5
20	Ditmas Park	5
28	Flatbush	5
30	Fort Greene	5
56	Prospect Park South	5
62	South Side	5
64	Sunset Park	5

Results (cont'd)

Final results show kmeans, Total Restaurants, and Total Sum. The decision of the investment would likely focus on K5, K1, and K2. Between these three, you can invest on an area that has the least amount of restaurants (K2) or the most (K1, K5)

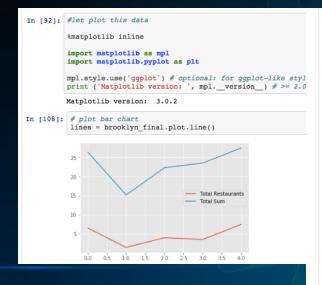
In [106]: brooklyn_final.head()

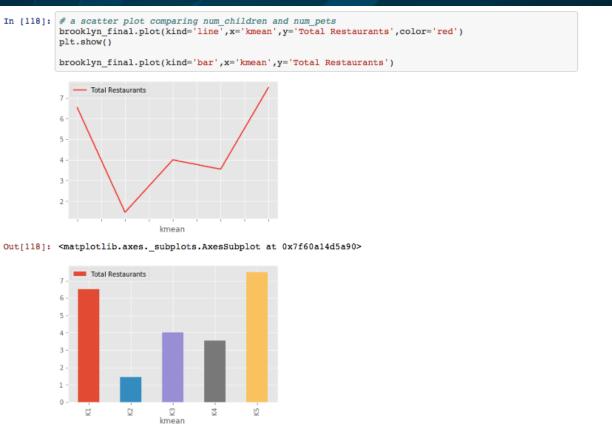
Out[106]:

	kmean	Total Restaurants	Total Sum
0	K1	6.533333	26.400000
1	K2	1.458333	15.208333
2	КЗ	4.000000	22.333333
3	K4	3.545455	23.545455
4	K5	7.500000	27.500000

Discussion

These charts show graphically where the results fall. As mentioned, K1 and K5 have the highest number of restaurants and K2 has the lowest





Conclusion

Final results show that the final decision of the investment would likely focus on K5, K1, and K2. Between these three, you can invest on an area that has the least amount of restaurants (K2) or the most (K1, K5). I would recommend on these neighborhoods in K5.

	Neighborhood	Group
11	Bushwick	5
19	Cypress Hills	5
20	Ditmas Park	5
28	Flatbush	5
30	Fort Greene	5
56	Prospect Park South	5
62	South Side	5
64	Sunset Park	5