The Battle of the Neighborhoods Report By Eviatar Shemesh

Introduction & Business Problem

The City of New York is the most populous city in the United States.

It is diverse and is the financial capital of USA.

It is multicultural.

It provides lot of business opportunities and business friendly environment.

It has attracted many different players into the market. It is a global hub of business and commerce.

The city is a major centre for banking and finance, retailing, world trade, transportation, tourism, real estate, new media, traditional media, advertising, legal services, accountancy, insurance, theatre, fashion, and the arts in the United States.

This also means that the market is highly competitive.

As it is highly developed city so cost of doing business is also one of the highest.

I was hired by a coffee shop named Devocion, a small company that makes the best and freshest coffee in New York.

We have 4 shops and we open a new one in Brooklyn, Cause the shop we opened there at Williamsburg was a hit, despite all the competition we made huge profits.

We want to collect data and get some few neighborhoods that coffee is popular at, but we believe that it'll be like the coffee shop we opened at Williamsburg.

Target Audience

The objective is to locate and recommend to the Devocion which neighborhoods of Brooklyn will be best choice to start a Coffee Shop.

The Management also expects to understand the rationale of the recommendations made.

This would interest anyone who wants to start a new coffee shop in Brooklyn,
in neighborhoods where coffee is very popular, and the competition is intense

Data

Our data will be collected from 2 sources:

1.JSON File with New York Neighborhoods and borough, where we will extract only the relevant data, of Brooklyn.

The JSON file will be collected at: https://cocl.us/new_york_dataset .

We will organize it and extract only the Brooklyn data. This File contains 4 columns:
1)Borough
2)Neighbourhood
3)Latitude
4)Longitude

Only Brooklyn Data

```
brooklyn_data = neighborhoods[neighborhoods['Borough'] == 'Brooklyn'].reset_index(drop=True)
brooklyn_data.head()
```

	Borough	Neighborhood	Latitude	Longitude
0	Brooklyn	Bay Ridge	40.625801	-74.030621
1	Brooklyn	Bensonhurst	40.611009	-73.995180
2	Brooklyn	Sunset Park	40.645103	-74.010316
3	Brooklyn	Greenpoint	40.730201	-73.954241
4	Brooklyn	Gravesend	40.595260	-73.973471

```
brooklyn_data.drop(['Borough'], axis = 1, inplace = True)
brooklyn_data.head()
```

	Neighborhood	Latitude	Longitude
0	Bay Ridge	40.625801	-74.030621
1	Bensonhurst	40.611009	-73.995180
2	Sunset Park	40.645103	-74.010316
3	Greenpoint	40.730201	-73.954241
4	Gravesend	40.595260	-73.973471

2.Foursquare API, to search for common venues around each neighbourhood, and cluster them into groups. Using it, we get the top 10 most common venues to each neighbourhood, which looks like that

	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	Pharmacy	Chinese Restaurant	Pizza Place	Gas Station	Bubble Tea Shop	Italian Restaurant	Fast Food Restaurant	Sushi Restaurant	Deli / Bodega	Dessert Shop
1	Bay Ridge	Italian Restaurant	Pizza Place	Spa	American Restaurant	Greek Restaurant	Bar	Bagel Shop	Thai Restaurant	Ice Cream Shop	Playground
2	Bedford Stuyvesant	Coffee Shop	Café	Pizza Place	Bar	Bagel Shop	Fried Chicken Joint	New American Restaurant	Boutique	Gift Shop	Gourmet Shop
3	Bensonhurst	Grocery Store	Chinese Restaurant	Flower Shop	Ice Cream Shop	Pizza Place	Sushi Restaurant	Donut Shop	Italian Restaurant	Noodle House	Liquor Store
4	Bergen Beach	Harbor / Marina	Athletics & Sports	Baseball Field	Playground	Donut Shop	Farmers Market	Fast Food Restaurant	Field	Filipino Restaurant	Fish & Chips Shop

Methodology

In this section we will talk about the data processing and methods to get the wanted result.

First, we collect the New York data using the JSON file.

	Borough	Neighborhood	Latitude	Longitude
0	Bronx	Wakefield	40.894705	-73.847201
1	Bronx	Co-op City	40.874294	-73.829939
2	Bronx	Eastchester	40.887556	-73.827806
3	Bronx	Fieldston	40.895437	-73.905643
4	Bronx	Riverdale	40.890834	-73.912585

After that, we clean it by getting only the data where the Borough is Brooklyn, and drop the borough column cause it's irrelevant, all the boroughs are Brooklyn.

```
brooklyn_data = neighborhoods[neighborhoods['Borough'] == 'Brooklyn'].reset_index(drop=True)
brooklyn_data.head()
```

	Borough	Neighborhood	Latitude	Longitude
0	Brooklyn	Bay Ridge	40.625801	-74.030621
1	Brooklyn	Bensonhurst	40.611009	-73.995180
2	Brooklyn	Sunset Park	40.645103	-74.010316
3	Brooklyn	Greenpoint	40.730201	-73.954241
4	Brooklyn	Gravesend	40.595260	-73.973471

```
brooklyn_data.drop(['Borough'], axis = 1, inplace = True)
brooklyn_data.head()
```

	Neighborhood	Latitude	Longitude
0	Bay Ridge	40.625801	-74.030621
1	Bensonhurst	40.611009	-73.995180
2	Sunset Park	40.645103	-74.010316
3	Greenpoint	40.730201	-73.954241
4	Gravesend	40.595260	-73.973471

Then we move to our second resource, the Foursquare API, we right few functions.

The first one is to extract the category out of each 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']
```

The second one is used to get nearby venues of each location

```
#function that gets the nearby venues
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',
                  'Latitude',
                  'Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']

    return(nearby_venues)
```

We apply the methods on our data of Brooklyn neighborhoods, max 100 per neighborhood, and maximum distance of 500 meters.
The result of this run will be inserted into a new Data Frame

```
LIMIT = 100
radius =500
brooklyn_venues = getNearbyVenues(names = brooklyn_data['Neighborhood'],
                                  latitudes = brooklyn_data['Latitude'],
                                  longitudes = brooklyn_data['Longitude']
                                 )
```

We use the one hot encoding method and inserting into a new Data Frame the top 10 most common category venues for each neighborhood

```
# one hot encoding
brooklyn_onehot = pd.get_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_grouped = brooklyn_onehot.groupby('Neighborhood').mean().reset_index()
```

```
def return_most_common_venues(row, num_top_venues):
    row_categories = row.iloc[1:]
    row_categories_sorted = row_categories.sort_values(ascending=False)

    return row_categories_sorted.index.values[0:num_top_venues]
```

```
num_top_venues = 10
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
Pc_venues_sorted = pd.DataFrame(columns=columns)
Pc_venues_sorted['Neighborhood'] = brooklyn_grouped['Neighborhood']

for ind in np.arange(brooklyn_grouped.shape[0]):
    Pc_venues_sorted.iloc[ind, 1:] = return_most_common_venues(brooklyn_grouped.iloc[ind, :], num_top_venues)
```

```
Pc_venues_sorted.head()
```

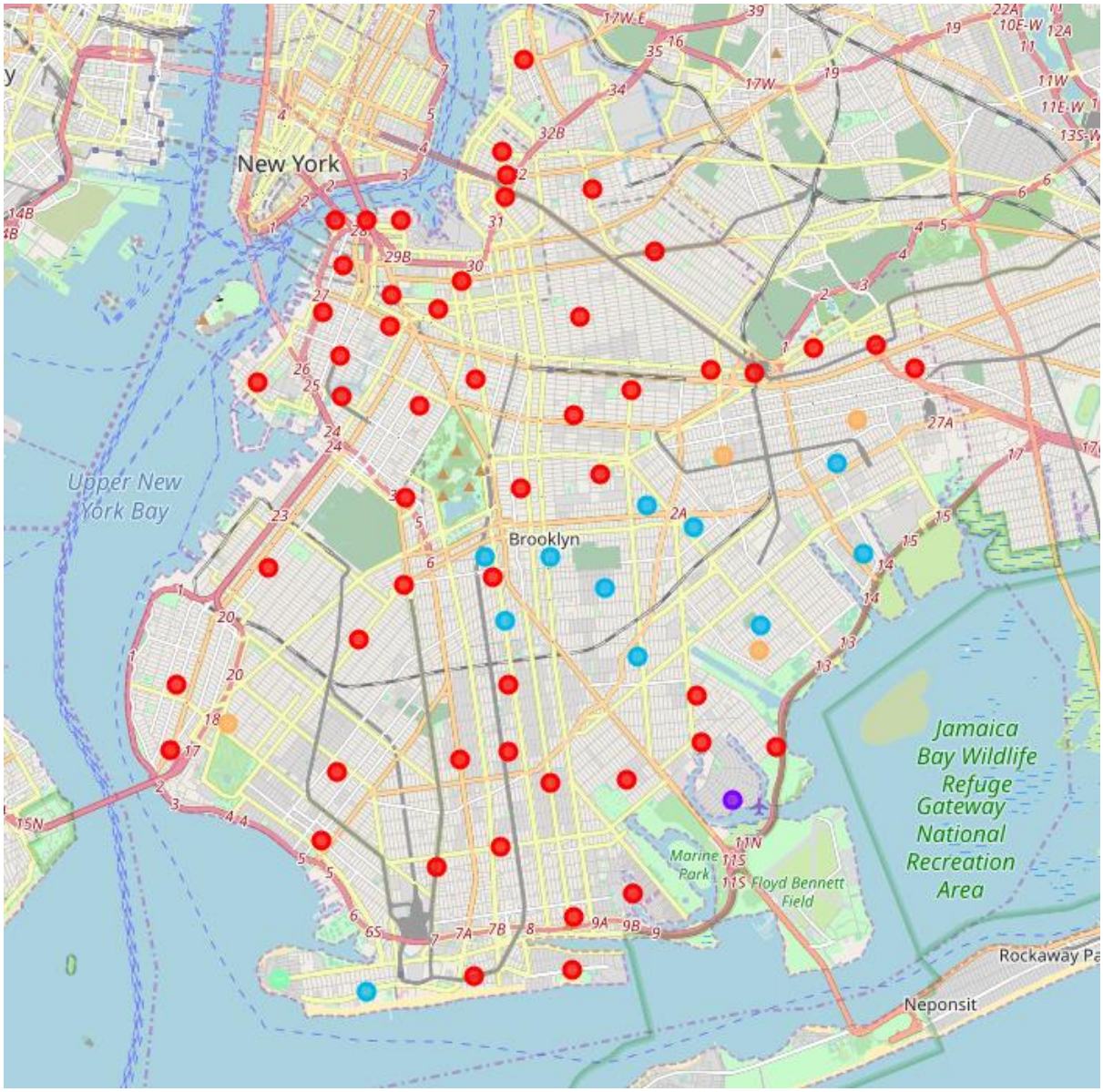
	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0	Bath Beach	Pharmacy	Chinese Restaurant	Pizza Place	Gas Station	Bubble Tea Shop	Italian Restaurant	Fast Food Restaurant	Sushi Restaurant	Deli / Bodega	Dessert Shop
1	Bay Ridge	Italian Restaurant	Pizza Place	Spa	American Restaurant	Greek Restaurant	Bar	Bagel Shop	Thai Restaurant	Ice Cream Shop	Playground
2	Bedford Stuyvesant	Coffee Shop	Café	Pizza Place	Bar	Bagel Shop	Fried Chicken Joint	New American Restaurant	Boutique	Gift Shop	Gourmet Shop
3	Bensonhurst	Grocery Store	Chinese Restaurant	Flower Shop	Ice Cream Shop	Pizza Place	Sushi Restaurant	Donut Shop	Italian Restaurant	Noodle House	Liquor Store
4	Bergen Beach	Harbor / Marina	Athletics & Sports	Baseball Field	Playground	Donut Shop	Farmers Market	Fast Food Restaurant	Field	Filipino Restaurant	Fish & Chips Shop

After we have This Data Frame, we use the KNN to cluster all of our Brooklyn neighborhoods into 5 groups, to find which neighborhoods are similar to our best shop neighborhood, Williamsburg

```
kclusters = 5
brooklyn_grouped_clustering = brooklyn_grouped.drop('Neighborhood', 1)
# run k-means clustering
kmeans = KMeans(n_clusters=kclusters, random_state=0).fit(brooklyn_grouped_clustering)
#add cluster labels
Pc_venues_sorted.insert(0, 'Cluster Labels', kmeans.labels_)
brooklyn_merged = brooklyn_data
brooklyn_merged = brooklyn_merged.join(Pc_venues_sorted.set_index('Neighborhood'), on='Neighborhood')
brooklyn_merged.head()
```

	Neighborhood	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	Bay Ridge	40.625801	-74.030621	0	Italian Restaurant	Pizza Place	Spa	American Restaurant	Greek Restaurant	Bar	Bagel Shop	Thai Restaurant	Ice Cream Shop	Playground
1	Bensonhurst	40.611009	-73.995180	0	Grocery Store	Chinese Restaurant	Flower Shop	Ice Cream Shop	Pizza Place	Sushi Restaurant	Donut Shop	Italian Restaurant	Noodle House	Liquor Store
2	Sunset Park	40.645103	-74.010316	0	Pizza Place	Bank	Bakery	Latin American Restaurant	Mexican Restaurant	Mobile Phone Shop	Gym	Fried Chicken Joint	Pharmacy	Café
3	Greenpoint	40.730201	-73.954241	0	Bar	Pizza Place	Coffee Shop	Cocktail Bar	Yoga Studio	Deli / Bodega	French Restaurant	Sushi Restaurant	Restaurant	Furniture / Home Store
4	Gravesend	40.595260	-73.973471	0	Italian Restaurant	Pizza Place	Bus Station	Lounge	Bakery	Chinese Restaurant	Martial Arts Dojo	Men's Store	Metro Station	Furniture / Home Store

We put it on a map to show the cluster output



After that, we check which cluster group our Williamsburg neighborhood is, and inserting it into a new Data Frame.

```
brooklyn_merged.loc[brooklyn_merged['Neighborhood'] == 'Williamsburg']
```

	Neighborhood	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
15	Williamsburg	40.707144	-73.958115	0	Coffee Shop	Bar	Bagel Shop	Yoga Studio	Greek Restaurant	Korean Restaurant	Tapas Restaurant	Taco Place	Event Space	Liquor Store

```
cluster3 = brooklyn_merged[brooklyn_merged['Cluster Labels'] == 0 ].reset_index(drop = True)
cluster3.head()
```

	Neighborhood	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	Bay Ridge	40.625801	-74.030621	0	Italian Restaurant	Pizza Place	Spa	American Restaurant	Greek Restaurant	Bar	Bagel Shop	Thai Restaurant	Ice Cream Shop	Playground
1	Bensonhurst	40.611009	-73.995180	0	Grocery Store	Chinese Restaurant	Flower Shop	Ice Cream Shop	Pizza Place	Sushi Restaurant	Donut Shop	Italian Restaurant	Noodle House	Liquor Store
2	Sunset Park	40.645103	-74.010316	0	Pizza Place	Bank	Bakery	Latin American Restaurant	Mexican Restaurant	Mobile Phone Shop	Gym	Fried Chicken Joint	Pharmacy	Café
3	Greenpoint	40.730201	-73.954241	0	Bar	Pizza Place	Coffee Shop	Cocktail Bar	Yoga Studio	Deli / Bodega	French Restaurant	Sushi Restaurant	Restaurant	Furniture / Home Store
4	Gravesend	40.595260	-73.973471	0	Italian Restaurant	Pizza Place	Bus Station	Lounge	Bakery	Chinese Restaurant	Martial Arts Dojo	Men's Store	Metro Station	Furniture / Home Store

As I mentioned above, our wanted neighborhoods are ones similar and where coffee shops are very popular, so into a new Data Frame we insert only the neighborhoods that are at the same cluster as our Williamsburg neighborhood is, and the most common venue category is coffee shop

```
cl3_coffee = cluster3.loc[cluster3['1st Most Common Venue'] == 'Coffee Shop'].reset_index(drop = True)
cl3_coffee
```

	Neighborhood	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	Williamsburg	40.707144	-73.958115	0	Coffee Shop	Bar	Bagel Shop	Yoga Studio	Greek Restaurant	Korean Restaurant	Tapas Restaurant	Taco Place	Event Space	Liquor Store
1	Bedford Stuyvesant	40.687232	-73.941785	0	Coffee Shop	Café	Pizza Place	Bar	Bagel Shop	Fried Chicken Joint	New American Restaurant	Boutique	Gift Shop	Gourmet Shop
2	Park Slope	40.672321	-73.977050	0	Coffee Shop	Burger Joint	Bagel Shop	Pet Store	Korean Restaurant	Bookstore	Italian Restaurant	Bakery	Pizza Place	American Restaurant
3	North Side	40.714823	-73.958809	0	Coffee Shop	Pizza Place	Yoga Studio	Wine Bar	Bar	Bakery	American Restaurant	Vegetarian / Vegan Restaurant	Jewelry Store	Cocktail Bar
4	Dumbo	40.703176	-73.988753	0	Coffee Shop	Park	Scenic Lookout	Bakery	Café	Boxing Gym	Italian Restaurant	Gym	Pizza Place	Bar

So, those 4 neighborhoods are potential neighborhoods to open new Devocion Coffee Shop.

Result

As I mentioned above, the result is list of 4 neighborhoods that are potential locations to open new Devocion Coffee Shop.

Bedford Stuyvesant	40.687232	-73.941785
Park Slope	40.672321	-73.977050
North Side	40.714823	-73.958809
Dumbo	40.703176	-73.988753

Discussion

Based on the results, I'm recommending our company to open a new coffee shop at Bedford Stuyvesant, and I'll explain why.

Our Goals where to find similar neighbourhood to Williamsburg, where coffee is very popular.

As you can see in the results, in this neighborhood, the top 2 venues categories out there are connected to coffee, so this will be my recommendation.

Conclusion

To conclude, I'm very happy with the results.

They came after a lot of work, clean data and the most important thing, a lot of data.

The list of neighborhoods is very small(Only 4 neighborhoods), so no much research will be needed to select the new location for our Devocion Coffee Shop.