Coursera Capstone Project

Clustering Neighbourhoods of Greater Manchester

Vishwajeet Vijay Patil
svishpatil@gmail.com

1. Introduction

- For the last decade, the United Kingdom's grocery landscape has been dominated by the 'big four' supermarket chains: Tesco, Asda, Sainsbury's and Morrisons. However, on the back of the economic recession, rising food prices and tightened belts, the market has been shaken by British consumers' search for value.
- Although rising food prices have not caused the quantity of goods purchased to fall, consumers seem more likely than ever to search for cheaper alternatives to the 'big four.' Discount supermarkets are enjoying a surge in popularity among food shoppers. According to figures from Kantar Worldpanel, all of the leading four supermarket brands have lost market share in the three months to August 2016.

2. Business Problem :-

- Consumer surveys indicate a shift in thinking among shoppers. According to recent evaluations.
- They are chosen over supermarkets because of their public perception as cheaper and, ironically, to avoid the complexity of overpromotion. Meanwhile, online grocery shopping could further revolutionize the market as e-commerce gains popularity among shoppers in the United Kingdom.
- Thus in hope of the supermarkets to keep up their revenue , the main objective of this project is to find the best locations to situate the supermarkets so that it has greater accessibility of buyers and draws more attention along with keeping in mind the surrounding neighbourhoods.

3. Data :-

- The data for this project has been collected from multiple sources.

3.1 Neighbourhoods :-

- The data of the neighbourhoods in Greater Manchester is extracted out by web scraping using BeautifulSoup library for Python. The neighbourhood data is scraped from Wikipedia

Geeting Areas of Greater Manchester from wikipedia

In [22]:
```
source = requests.get('https://en.wikipedia.org/wiki/Category:Areas_of_Greater_Manchester').text
soup = BeautifulSoup(source, 'lxml')
dummy = soup.find_all(class_ = 'mw-category-group')

manchester = []
for i in range(len(dummy)):
    lists = dummy[i].find_all('a')
    for lis in lists:
        temp = lis.get('title')
        manchester.append(temp)

for i in range(len(manchester)):
    if i<=20:
        manchester.remove(manchester[i])

manchester.remove('Category:Areas of Bolton')
manchester.remove('Category:Areas of Manchester')
manchester.remove('Category:Areas of the Metropolitan Borough of Rochdale')
manchester.remove('Category:Areas of Stockport')

print(manchester)
```

```
['Agecroft', 'Ashton upon Mersey', "Besses o' th' Barn", 'Boothstown', 'Bradshaw, Greater Manchester', 'Brandlesholme', 'Bromle
y Cross', 'Carrbrook', 'Copley, Greater Manchester', 'Davyhulme', 'Fairfield, Tameside', 'Fishpool', 'Four Heatons', 'Gigg, Gre
ater Manchester', 'Hart Common', 'Heyheads', 'Higher End', 'Hindsford', 'Houldsworth Model Village', 'Howe Bridge', 'Ladybrook
Valley', 'Langley, Greater Manchester', 'Lees, Greater Manchester', 'Littlemoss', 'Longshaw', 'Makerfield', 'Matley', 'Middlebr
ook, Greater Manchester', 'Monton', 'Moses Gate', 'Mosley Common', 'Norbury, Greater Manchester', 'North Reddish', 'Old Traffor
d (area)', 'Orrell, Greater Manchester', 'Patricroft', 'Peel Green', 'Pennington, Greater Manchester', 'Pilsworth', 'Reddish',
'Redvales', 'Romiley', 'Rumworth', 'Rusholme', 'Salford Quays', 'Sedgley Park', 'Shakerley', 'South Reddish', 'South Turton',
'Tonge, Middleton', 'Torkington', 'Trafford Park', 'Unsworth', 'Urmston', 'Wallsuches', 'Walmersley', 'Warburton Green', 'Wardl
ey, Greater Manchester', 'Wingates', 'Woodley, Greater Manchester']
```

3.2 Geocoding :-

- The file contents are retrieved into a Pandas DataFrame. The latitude and longitude of the neighbourhoods are retrieved using OpenCage Geocoding API. The geometric location values are then stored into the initial dataframe.

Geeting latitude and longitude

In [86]:
```
from opencage.geocoder import OpenCageGeocode

key = 'ab247b6526314f699593f70221a57167'
geocoder = OpenCageGeocode(key)

end = ' , Greater Manchester'

lat = []
lon = []

for name in df['Area of Manchester']:
    query = str(name) + end
    result = geocoder.geocode(query)
    lat.append(result[0]['geometry']['lat'])
    lon.append(result[0]['geometry']['lng'])
```

In [87]:
```
df['Latitudes'] = lat
df['Longitudes'] = lon

df.rename(columns={'Area of Manchester' : 'Neighborhoods'} , inplace=True)
df.head()
```

Out[87]:

	Neighborhoods	Latitudes	Longitudes
0	Agecroft	53.505212	-2.299912
1	Ashton upon Mersey	53.429835	-2.343177
2	Besses o' th' Barn	53.541943	-2.286099
3	Boothstown	53.500480	-2.431061
4	Bradshaw, Greater Manchester	53.606603	-2.398959

3.3 Venue Data :-

- FourSquare API is used to find the venue data and new dataframe is created along with the respective neighbourhoods.

Using the FourSquare API

In [99]:

```
explore_df_list = []

for i, nbd_name in enumerate(df['Neighborhoods']):
    try :
        # Getting the data of neighbourhood
        nbd_name = df.loc[i, 'Neighborhoods']
        nbd_lat = df.loc[i, 'Latitudes']
        nbd_lng = df.loc[i, 'Longitudes']

        radius = 1000 # Setting the radius as 1000 metres
        LIMIT = 30 # Getting the top 30 venues

        url = 'https://api.foursquare.com/v2/venues/explore?client_id={} \
        &client_secret={}&ll={},{}&v={}&radius={}&limit={}'\
        .format(CLIENT_ID, CLIENT_SECRET, nbd_lat, nbd_lng, VERSION, radius, LIMIT)

        results = json.loads(requests.get(url).text)
        results = results['response']['groups'][0]['items']

        nearby = json_normalize(results) # Flattens JSON

         # Filtering the columns
        filtered_columns = ['venue.name', 'venue.categories', 'venue.location.lat', 'venue.location.lng']
        nearby = nearby.loc[:, filtered_columns]

        # Renaming the columns
        columns = ['Name', 'Category', 'Latitude', 'Longitude']
        nearby.columns = columns

        # Gets the categories
        nearby['Category'] = nearby.apply(get_category_type, axis=1)

        # Gets the data required
        for i, name in enumerate(nearby['Name']):
            s_list = nearby.loc[i, :].values.tolist()  # Converts the numpy array to a python list
            f_list = [nbd_name, nbd_lat, nbd_lng] + s_list
```

4. Methodology :-

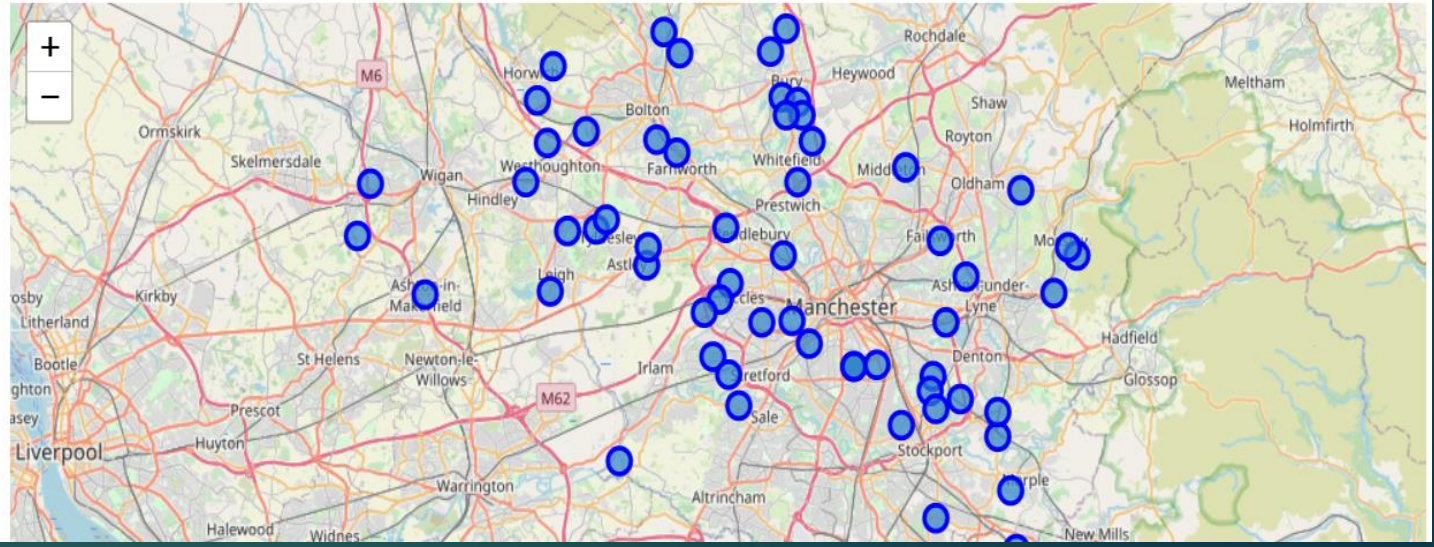
- Different methods and algorithms are use to check the data and make sure the prediction are accurate and worthy.

4.1 Folium :-

- Folium builds on the data wrangling strengths of the Python ecosystem and the mapping strengths of the leaflet.js library.
- All cluster visualizations are done with help of Folium which in turn generates a Leaflet map made using OpenStreetMap technology.

```
# Creates map of Greater Manchester using latitude and longitude values
map_man = folium.Map(location=[gman_lat, gman_lng], zoom_start=10)

# Add markers to map
for lat, lng, neighborhood in zip(df['Latitudes'], df['Longitudes'], df['Neighborhoods']):
    label = '{}'.format(neighborhood)
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=8,
        popup=label,
        color='blue',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.7,
        parse_html=False).add_to(map_man)

map_man
```

Out[90]:



4.2 Top 10 most common venues :-

- Due to high variety in the venues, only the top 10 common venues are selected and a new DataFrame is made, which is used to train the K-means Clustering Algorithm.

```
num_top_venues = 10
indicators = ['st', 'nd', 'rd']

# Create columns according to number of top venues
columns = ['Neighbourhood']
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
neighbourhoods_venues_sorted = pd.DataFrame(columns=columns)
neighbourhoods_venues_sorted['Neighbourhood'] = man_grouped['Neighbourhood']

for ind in np.arange(man_grouped.shape[0]):
    neighbourhoods_venues_sorted.iloc[ind, 1:] = return_most_common_venues(man_grouped.iloc[ind, :], num_top_venues)

neighbourhoods_venues_sorted.head()
```

Out[103]:

	Neighbourhood	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	Agecroft	Pizza Place	Café	Sandwich Place	Pet Store	Chinese Restaurant	Food & Drink Shop	Hotel	Supermarket	Grocery Store	Construction & Landscaping
1	Ashton upon Mersey	Grocery Store	Pub	Hotel	Convenience Store	Event Service	Film Studio	Fast Food Restaurant	Farm	Falafel Restaurant	Event Space
2	Besses o' th' Barn	Indian Restaurant	Coffee Shop	Bakery	Pharmacy	Soccer Stadium	Hotel	Latin American Restaurant	Pizza Place	Pub	Seafood Restaurant
3	Boothstown	Indian Restaurant	Pub	Flower Shop	Wine Bar	Food & Drink Shop	Doctor's Office	Dog Run	Electronics Store	English Restaurant	Event Service
4	Bradshaw, Greater Manchester	Pub	Supermarket	Cosmetics Shop	Gastropub	Construction & Landscaping	Chinese Restaurant	Home Service	Event Service	Event Space	Flower Shop

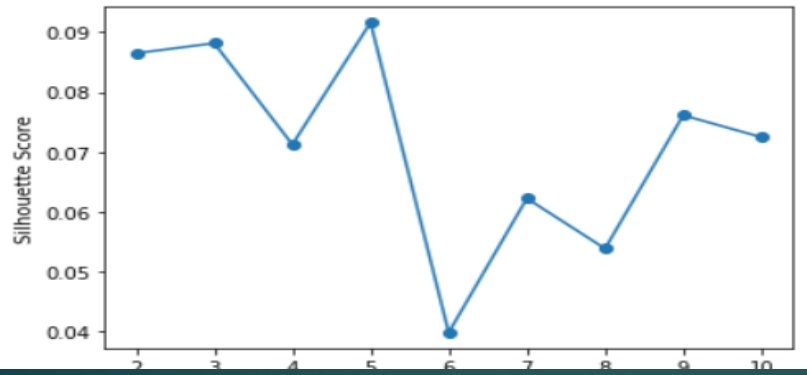
4.3 Optimal number of clusters :-

- Silhouette Score is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation).

Run Markdown

In [137]:
```
from sklearn.metrics import silhouette_score

indices = []
scores = []

for kclusters in range(2, 11) :

    # Run k-means clustering
    kgc = man_grouped_clustering
    kmeans = KMeans(n_clusters = kclusters, init = 'k-means++', random_state = 0).fit_predict(kgc)

    # Gets the score for the clustering operation performed
    score = silhouette_score(kgc, kmeans)

    # Appending the index and score to the respective lists
    indices.append(kclusters)
    scores.append(score)
```

In [150]:
```
plt.plot(range(2,11), scores, 'o-')
plt.xlabel("No of clusters")
plt.ylabel("Silhouette Score")
plt.show()
```



4.4 K-means clustering :-

- Silhouette Score is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation).

5. Result :-

- The neighbourhoods are divided into n (5 here) clusters where n is the number of clusters found using the optimal approach.

selecting 5 cluster

```
In [142]: opt = 5

          kgc = man_grouped_clustering
          kmeans = KMeans(n_clusters = opt, init = 'k-means++', random_state = 0).fit(kgc)
```

```
In [144]: neighbourhoods_venues_sorted.insert(0, 'Cluster Labels', kmeans.labels_)
```

```
In [145]: man_merged = df
          man_merged = man_merged.join(neighbourhoods_venues_sorted.set_index('Neighbourhood'), on='Neighborhoods')
          man_merged.dropna(inplace = True)
          man_merged['Cluster Labels'] = man_merged['Cluster Labels'].astype(int)
          man_merged
```

Out[145]:

	Neighborhoods	Latitudes	Longitudes	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
0	Agecroft	53.505212	-2.299912	4	Pizza Place	Café	Sandwich Place	Pet Store	Chinese Restaurant	Food & Drink Shop	Hotel	Supermarket
1	Ashton upon Mersey	53.429835	-2.343177	1	Grocery Store	Pub	Hotel	Convenience Store	Event Service	Film Studio	Fast Food Restaurant	Farm
2	Besses o' th' Barn	53.541943	-2.286099	4	Indian Restaurant	Coffee Shop	Bakery	Pharmacy	Soccer Stadium	Hotel	Latin American Restaurant	Pizza Place
3	Boothstown	53.500480	-2.431061	4	Indian Restaurant	Pub	Flower Shop	Wine Bar	Food & Drink Shop	Doctor's Office	Dog Run	Electronics Store
4	Bradshaw, Greater Manchester	53.606603	-2.398959	1	Pub	Supermarket	Cosmetics Shop	Gastropub	Construction & Landscaping	Chinese Restaurant	Home Service	Event Service

6. Discussion :-

- After analyzing the various clusters produced by the Machine learning algorithm, cluster no. 0 , is a prime fit to solving the problem of finding a cluster with a common venue as a train station mentioned before.

7. Conclusion :-

- These four places Hindsford , Romiley, Pilsworth , Shakerley fall in the heart of the Greater Manchester area.
- These four areas have a large number of small localized markets of different aspects like movie-theaters , gyms , hotels and restaurants which can profit indirectly due to the increased footfall in these areas if supermarkets are localised in these places