

# Coursera Capstone Project

## Applied Data Science

Author : Vishwajeet Patil

### 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 over-promotion. 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.

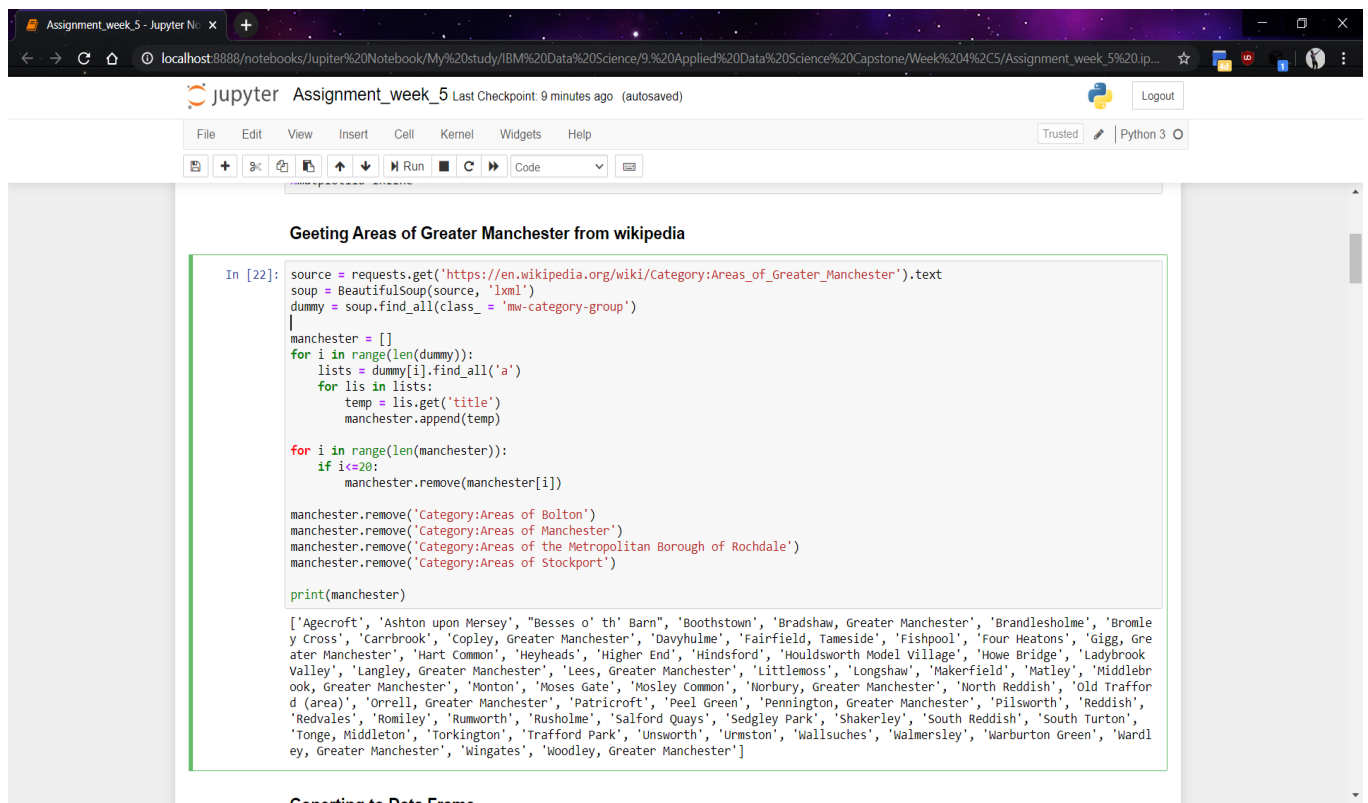
### 3. Data :-

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 a Wikipedia

url=' [https://en.wikipedia.org/wiki/Category:Areas of Greater Manchester](https://en.wikipedia.org/wiki/Category:Areas_of_Greater_Manchester)'



The screenshot shows a Jupyter Notebook interface with a browser window at the top displaying the URL [https://en.wikipedia.org/wiki/Category:Areas\\_of\\_Greater\\_Manchester](https://en.wikipedia.org/wiki/Category:Areas_of_Greater_Manchester). The notebook title is "Assignment\_week\_5" and it shows the last checkpoint was 9 minutes ago. The code cell is titled "Geeting Areas of Greater Manchester from wikipedia" and contains the following Python code:

```
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)
```

The output of the code is a list of 40 neighbourhood names in Greater Manchester, including: ['Agecroft', 'Ashton upon Mersey', 'Besses o' th' Barn', 'Boothstown', 'Bradshaw, Greater Manchester', 'Brandlesholme', 'Bromley Cross', 'Carrbrook', 'Copley, Greater Manchester', 'Davyhulme', 'Fairfield, Tameside', 'Fishpool', 'Four Heaton', 'Gigg, Greater Manchester', 'Hart Common', 'Heyheads', 'Higher End', 'Hindsford', 'Houldsworth Model Village', 'Howe Bridge', 'Ladybrook Valley', 'Langley, Greater Manchester', 'Lees, Greater Manchester', 'Littlemoss', 'Longshaw', 'Makerfield', 'Matley', 'Middlebrook, Greater Manchester', 'Monton', 'Moses Gate', 'Mosley Common', 'Norbury, Greater Manchester', 'North Reddish', 'Old Trafford (area)', 'Orrell, Greater Manchester', 'Patricroft', 'Peel Green', 'Pennington, Greater Manchester', 'Pillsworth', 'Reddish', 'Redvales', 'Romiley', 'Rumworth', 'Rusholme', 'Salford Quays', 'Sedgley Park', 'Shakerley', 'South Reddish', 'South Turton', 'Tonge, Middleton', 'Torkington', 'Trafford Park', 'Unsworth', 'Urmston', 'Wallsuchers', 'Walmersley', 'Warburton Green', 'Wardley, 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.

The screenshot shows a Jupyter Notebook interface with the following code and output:

```
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()
```

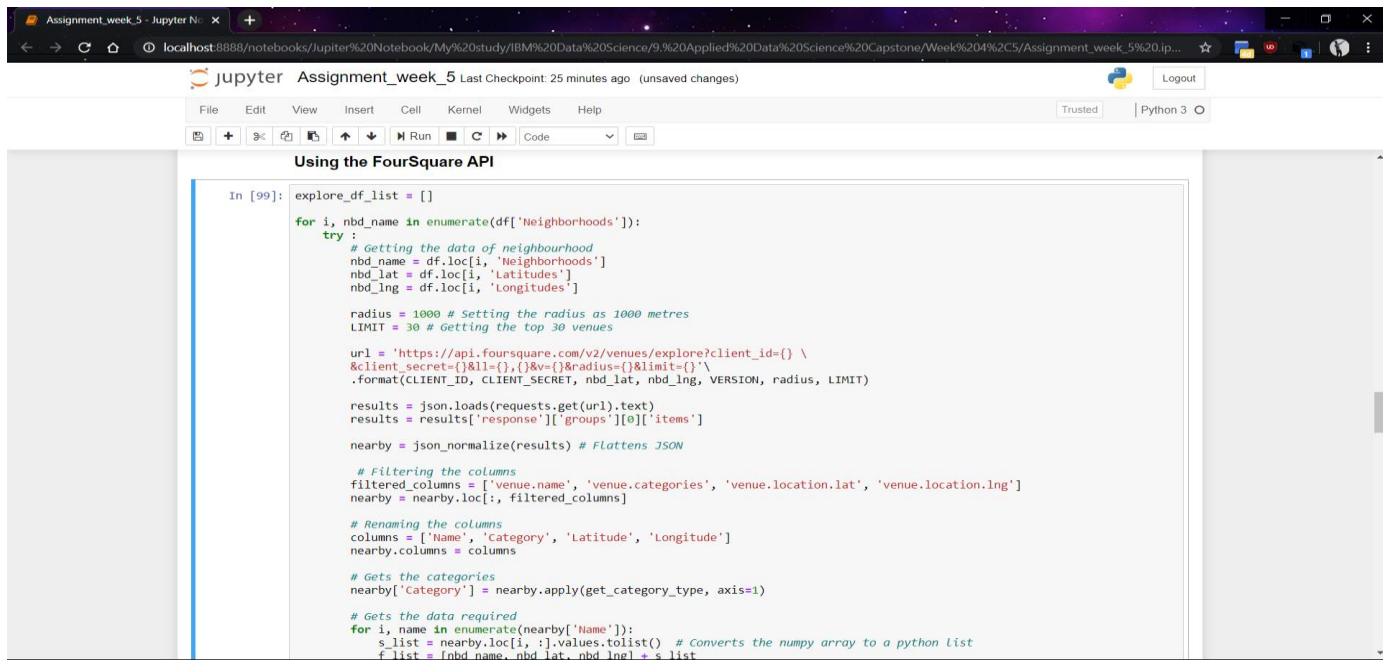
Out[87]:

	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	-2.388959

Dependencies of plotting

### 3.3 Venue Data :-

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



```
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)

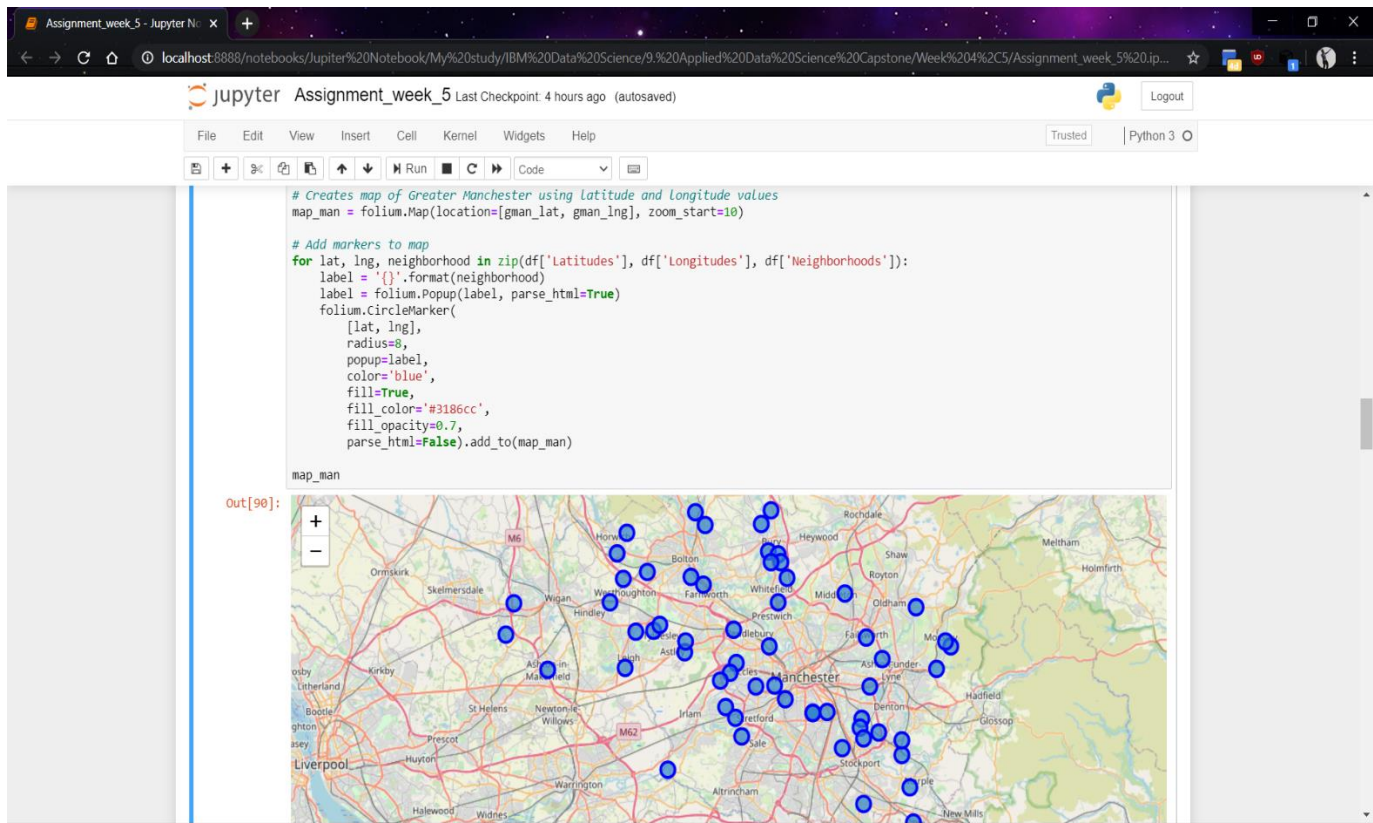
        # Gets the data required
        for i, name in enumerate(nearby['Name']):
            s_list = nearby.loc[i, :].values.tolist() # Converts the numpy array to a python list
            f_list = [nbd_name, nbd_lat, nbd_lng] + s_list
```

## 4. Methodology :-

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

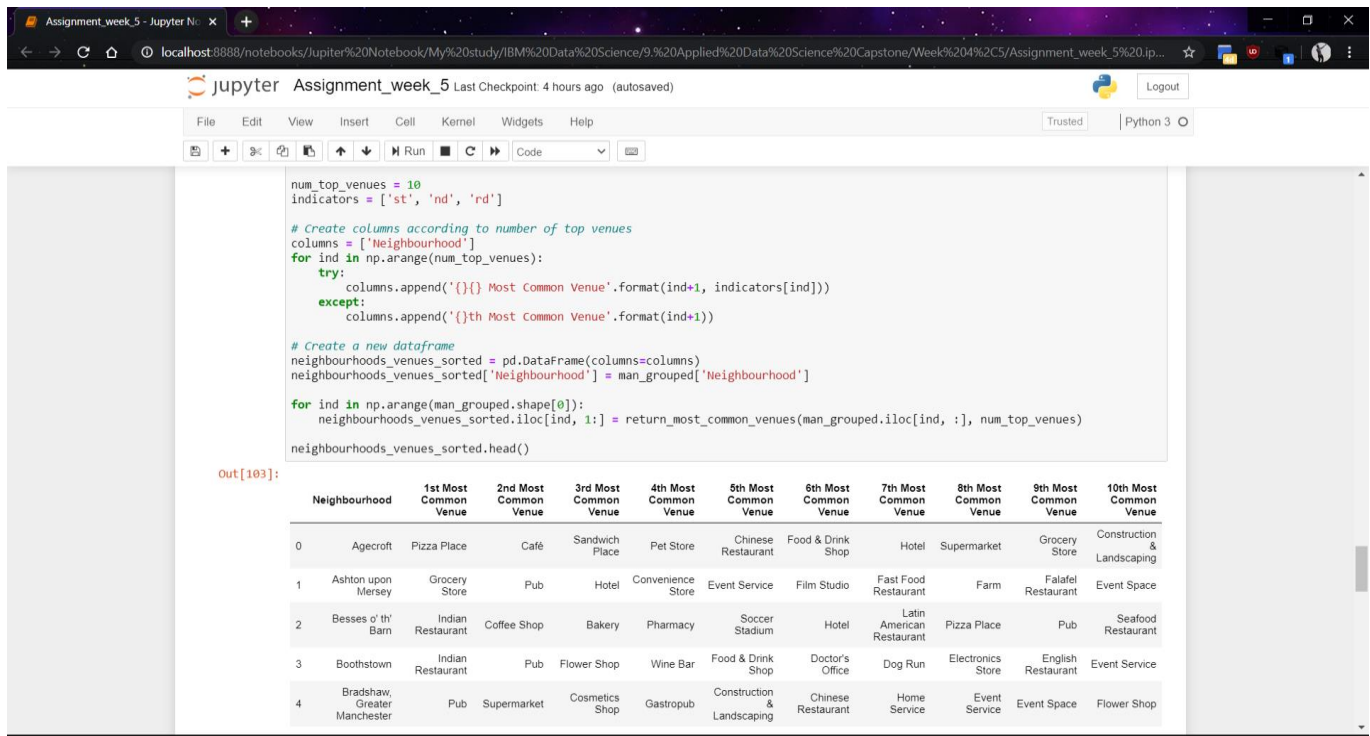
### 4.1 Folium :-

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.



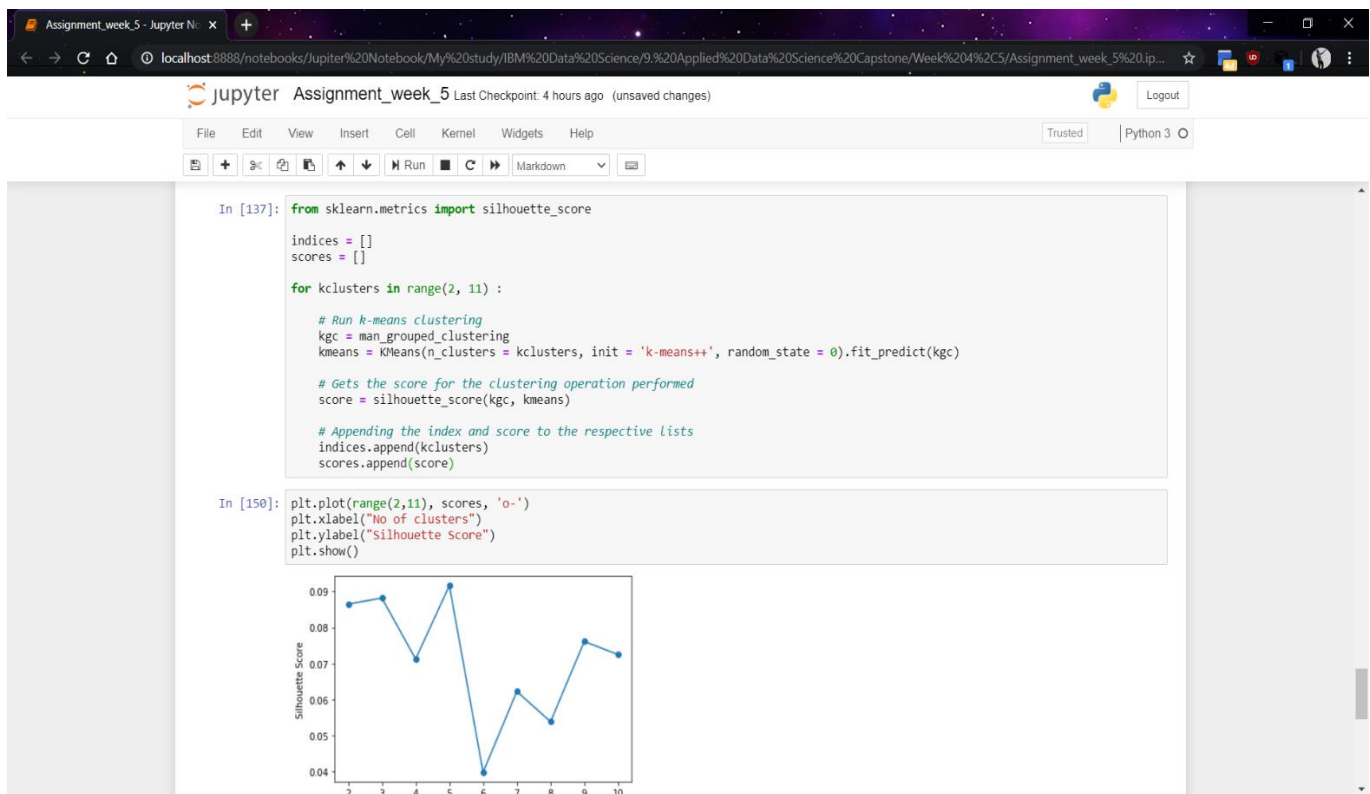
## 4.2 Top 10 most common venues :-

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.



## 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).

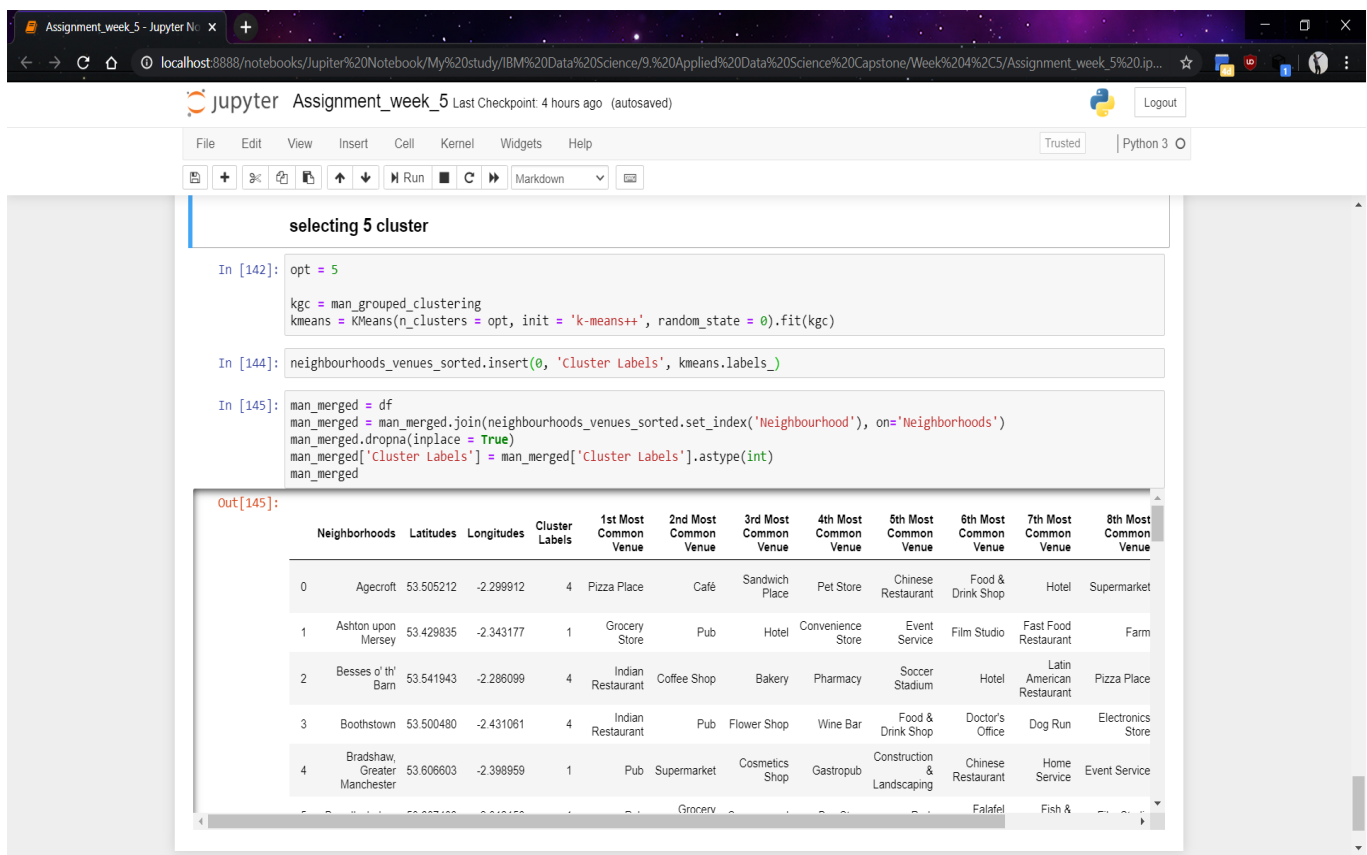


## 4.4 K-means clustering :-

The venue data is then trained using K-means Clustering Algorithm to get the desired clusters to base the analysis on.

## 5. Results :-

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='Neighborhood')
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.



The screenshot shows a Jupyter Notebook titled "Assignment\_week\_5" with a "Last Checkpoint: 4 hours ago (unsaved changes)" status. The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for file operations, cell execution, and code execution. The notebook content is divided into two main sections: a data table and a code cell.

The data table displays the following information:

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	9th Most Common Venue
16 Higher End	53.512547	-2.150836	0	Supermarket	Tram Station	Furniture / Home Store	Fast Food Restaurant	Pub	Event Service	Film Studio	Farm	Falafel Restaurant
17 Hindsford	53.518633	-2.478915	0	Supermarket	Fast Food Restaurant	Roller Rink	Pub	Soccer Field	Wine Bar	Event Space	Film Studio	Farm
38 Pilsworth	53.575140	-2.281866	0	Pub	Hotel	Supermarket	Wine Bar	Event Space	Fish & Chips Shop	Film Studio	Fast Food Restaurant	Farm
41 Romiley	53.414065	-2.096080	0	Supermarket	Pub	Train Station	Gym	Grocery Store	Gastropub	Fast Food Restaurant	Farm	Falafel Restaurant
46 Shakerley	53.523139	-2.469112	0	Supermarket	Train Station	Fast Food Restaurant	Pub	Event Service	Fish & Chips Shop	Film Studio	Farm	Falafel Restaurant

The code cell shows the following execution:

```
In [162]: man_merged[man_merged['cluster Labels'] == 0]
Out[162]:
```

## 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.