1.Introduction:

1.1 Background:

London is the capital and the largest city of the England and the UK. It is one of the most ethnically diverse cities in the world. For this reason, It is also seen as a world city. According to the 2011 Census, London has a total population of 8 million (approximately) of which 20% belong to Asian ethnic group which is 1.5 million approximately. Even though the Asian community is massive, there is a lack of an high-end Asian Restaurant with multiple cuisines that not only provides food but also provides service with the ambience. The restaurant industry in London is growing exponentially with the increasing demand. This demand has spurred the competition to open restaurants in a nice area of the city.

Data Science helps in identifying the appropriate market trends and evolving consumer preferences so that restaurants can better address them. Using various data analysis techniques, the London areas are explored through segmenting and clustering, to identify a good location to open an Asian restaurant.

1.2 Business Problem:

A successful Asian restaurant chain is looking to expand its operations through London. We were asked to identify and recommend the neighborhoods in London that will be good choice to start an Asian restaurant.

1.3 Target Audience:

This project will primarily help the following categories:

1. Companies that are looking to invest in food service industry of London.
2. Individuals looking to relocate neighborhoods in London with particular venues.

2.Data:

2.1 Data sources:

For this project, we will make use of the following data.

1.London Neighborhood's: I have used web scraping techniques to get the list of areas and boroughs in the London. I've extracted the Location, borough, post town and post codes of the areas in London.

Data source: https://en.wikipedia.org/wiki/List_of_areas_of_London

2. London Demographics: From the following Wikipedia page, I have extracted the demographics of each Borough in London through web scraping.

Data source: https://en.wikipedia.org/wiki/Demography_of_London

3. Geopy library: To get the latitude and longitude of each neighborhood.

4. Foursquare API: I've used the foursquare API to locate various venues in each of the London neighborhoods.

2.2 Data Collection and Cleaning:

The BeautifulSoup package is used to scrape the needed data from Wikipedia. The following data frame was obtained by scraping the list of London areas from Wikipedia page:

Out[67]:

	Neighborhood	Borough	Post_town	Post_code
0	Abbey Wood	Bexley, Greenwich [7]	LONDON	SE2
1	Acton	Ealing, Hammersmith and Fulham[8]	LONDON	W3, W4
2	Addington	Croydon[8]	CROYDON	CR0
3	Addiscombe	Croydon[8]	CROYDON	CR0
4	Albany Park	Bexley	BEXLEY, SIDCUP	DA5, DA14
...	...	...	...	...
528	Woolwich	Greenwich	LONDON	SE18
529	Worcester Park	Sutton, Kingston upon Thames	WORCESTER PARK	KT4
530	Wormwood Scrubs	Hammersmith and Fulham	LONDON	W12
531	Yeading	Hillingdon	HAYES	UB4
532	Yiewsley	Hillingdon	WEST DRAYTON	UB7

533 rows × 4 columns

The data frame needs to be cleaned. The borough column has numbers attached to it's values that should be stripped. After stripping the numbers, the following data frame is obtained:

In [68]:
```
df_1['Borough'] = df_1['Borough'].map(lambda x: x.rstrip(']').rstrip('0123456789').rstrip('['))
df_1
```

Out[68]:

	Neighborhood	Borough	Post_town	Post_code
0	Abbey Wood	Bexley, Greenwich	LONDON	SE2
1	Acton	Ealing, Hammersmith and Fulham	LONDON	W3, W4
2	Addington	Croydon	CROYDON	CR0
3	Addiscombe	Croydon	CROYDON	CR0
4	Albany Park	Bexley	BEXLEY, SIDCUP	DA5, DA14
...	...	...	...	...
528	Woolwich	Greenwich	LONDON	SE18
529	Worcester Park	Sutton, Kingston upon Thames	WORCESTER PARK	KT4
530	Wormwood Scrubs	Hammersmith and Fulham	LONDON	W12
531	Yeading	Hillingdon	HAYES	UB4
532	Yiewsley	Hillingdon	WEST DRAYTON	UB7

533 rows × 4 columns

The demographics of the all the London boroughs is obtained from the 'Demography of London' Wikipedia page. After cleaning and parsing the html accordingly, the resulting data frame is below:

```
In [74]: df_demographics = pd.DataFrame({'Local_authority':local_authority,'White': white,'Mixed': mixed,'Asian':asian,'Blacl
         df_demographics
```

Out[74]:

	Local_authority	White	Mixed	Asian	Black	Others
0	Barnet	64.1	4.8	18.5	7.7	4.8
1	Barking and Dagenham	58.3	4.2	15.9	20	1.6
2	Bexley	81.9	2.3	6.6	8.5	0.8
3	Brent	36.3	5.1	34.1	18.8	5.8
4	Bromley	84.3	3.5	5.2	6	0.9
5	Camden	66.3	5.6	16.1	8.2	3.8
6	City of London	78.6	3.9	12.7	2.6	2.1
7	Croydon	55.1	6.6	16.4	20.2	1.8
8	Ealing	49	4.5	29.7	10.9	6
9	Enfield	61	5.5	11.2	17.2	5.1
10	Greenwich	62.5	4.8	11.7	19.1	1.9
11	Hackney	54.7	6.4	10.5	23.1	5.3
12	Haringey	60.5	6.5	9.5	18.8	4.7
13	Harrow	42.2	4	42.6	8.2	2.9

Since, the business focusses on Asian market the data frame is sorted in descending order of the column ‘Asian’ that represents the percentage of Asian population. Sorted data frame can be observed below:

```
In [13]: df_demographics.sort_values(by='Asian', inplace=True, ascending=False)
         df_demographics.reset_index(inplace=True)
         df_demographics.drop('index', axis=1, inplace=True)
         df_demographics.head()
```

Out[13]:

	Local_authority	White	Mixed	Asian	Black	Others
0	Newham	29.0	4.5	43.5	19.6	3.5
1	Harrow	42.2	4.0	42.6	8.2	2.9
2	Redbridge	42.5	4.1	41.8	8.9	2.7
3	Tower Hamlets	45.2	4.1	41.1	7.3	2.3
4	Hounslow	51.4	4.1	34.4	6.6	3.6

The top eight boroughs with highest Asian population are observed. We limit the London neighborhoods that we initially obtained to these eight boroughs. The part of the data frame can be observed below:

```
In [78]: top_8 = ['Newham', 'Redbridge','Tower Hamlets','Hounslow','Brent', 'Ealing','Hillingdon','Waltham Forest']

In [88]: df_final = df_1[df_1['Borough'].isin(top_8)].reset_index(drop=True)
         df_final.loc[10:10,'Neighborhood':'Neighborhood'] = 'Bromley'
         df_final.head(11)
```

Out[88]:

	Neighborhood	Borough	Post_town	Post_code
0	Aldborough Hatch	Redbridge	ILFORD	IG2
1	Alperton	Brent	WEMBLEY	HA0
2	Barkingside	Redbridge	ILFORD	IG6
3	Beckton	Newham	LONDON, BARKING	E6, E16, IG11
4	Bedford Park	Ealing	LONDON	W4
5	Bethnal Green	Tower Hamlets	LONDON	E2
6	Blackwall	Tower Hamlets	LONDON	E14
7	Bow	Tower Hamlets	LONDON	E3
8	Brentford	Hounslow	BRENTFORD	TW8
9	Brent Park	Brent	LONDON	NW10
10	Bromley	Tower Hamlets	LONDON	E3

The Geopy library is used to get the latitude and longitude of each neighborhood and are added as columns to the data frame. The head of the final data frame is as follows:

Out[95]:

	Neighborhood	Borough	Post_town	Post_code	Latitude	Longitude
0	Aldborough Hatch	Redbridge	ILFORD	IG2	51.585590	0.098750
1	Alperton	Brent	WEMBLEY	HA0	51.540804	-0.300096
2	Barkingside	Redbridge	ILFORD	IG6	51.585818	0.088624
3	Beckton	Newham	LONDON, BARKING	E6, E16, IG11	51.516080	0.059426
4	Bedford Park	Ealing	LONDON	W4	51.498020	-0.255647

3.Methodology:

Map Visualization:

The Geopy library is used to get the latitude and longitude of London.

```
# Getting London Coordinates

address = 'London, UK'
geolocator = Nominatim(user_agent="ny_explorer")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print('The geograpical coordinate of the City of London are {}, {}.'.format(latitude, longitude))
```

```
The geograpical coordinate of the City of London are 51.5073219, -0.1276474.
```

Using the Folium library, we generate the map of the London by passing its coordinates as arguments.

```
# Creating a map
map_london = folium.Map(location=[latitude, longitude], zoom_start=11)
map_london
```

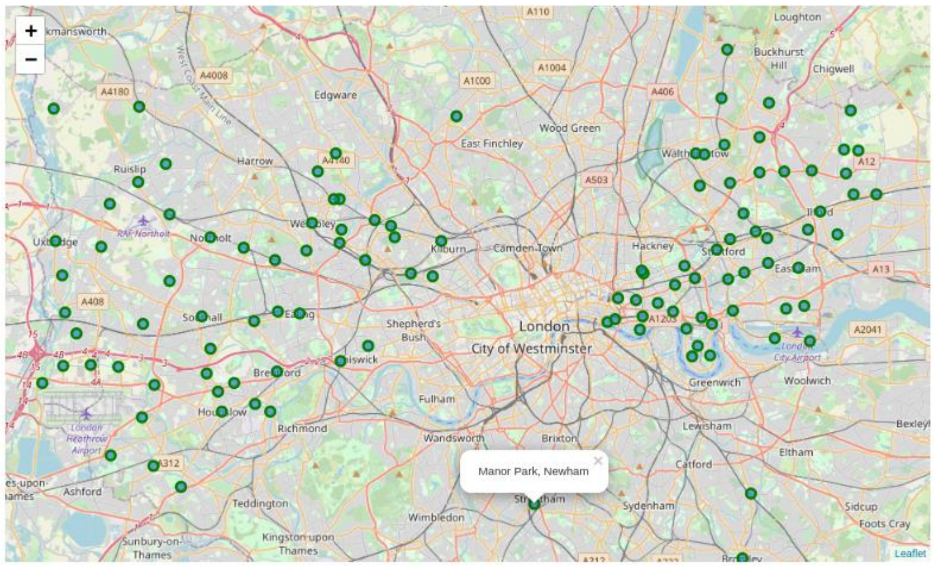
Now, to visualize Neighborhoods on top of the London map, we create markers and add them to the London map.

```python
# create map of London using latitude and longitude values
map_london = folium.Map(location=[latitude, longitude], zoom_start=10)

# add markers to map
for lat, lng, borough, neighborhood in zip(df['Latitude'], df['Longitude'], df['Borough'], df['Neighborhood']):
    label = '{}, {}'.format(neighborhood, borough)
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=5,
        popup=label,
        color='green',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.7,
        parse_html=False).add_to(map_london)

map_london
```

The generated map with London neighborhoods is shown below:



Exploring Neighborhoods:

Using the Foursquare API, we explore top 100 venues within 1000 meters radius of each neighborhood. Let's start with exploring the first neighborhood. We create a URL for the API call.

```python
In [45]: radius = 1000
         LIMIT = 100
         url = 'https://api.foursquare.com/v2/venues/explore?client_id={}&client_secret={}&ll={},{}&v={}&radius={}&limit={}'
         url
```

The URL defined above is used to create a GET request and the resulting json file is saved.

```
In [46]: results = requests.get(url).json()
         results
```


```
Out[46]: {'meta': {'code': 200, 'requestId': '5eda6a9229ce6a001bdea651'},
          'response': {'suggestedFilters': {'header': 'Tap to show:',
            'filters': [{'name': 'Open now', 'key': 'openNow'}]},
           'headerLocation': 'Wembley Central',
           'headerFullLocation': 'Wembley Central, London',
           'headerLocationGranularity': 'neighborhood',
           'totalResults': 9,
           'suggestedBounds': {'ne': {'lat': 51.5453036045, 'lng': -0.2928745765139981},
            'sw': {'lat': 51.536303595499994, 'lng': -0.30731802348600185}},
           'groups': [{'type': 'Recommended Places',
             'name': 'recommended',
             'items': [{'reasons': {'count': 0,
                'items': [{'summary': 'This spot is popular',
                  'type': 'general',
                  'reasonName': 'globalInteractionReason'}]},
               'venue': {'id': '4bd1c71f77b29c7424318d82',
                'name': 'The Gym',
                'location': {'address': '197 Ealing Rd, The Atlip Centre',
                 'lat': 51.54081869411914,
```


To extract the Category type from the results, get_category_type function is defined.

```
In [47]: # 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']
```

The json file is then cleaned and structured into pandas dataframe as shown below:

```
In [48]: venues = results['response']['groups'][0]['items']

         nearby_venues = json_normalize(venues) # flatten JSON

         # filter columns
         filtered_columns = ['venue.name', 'venue.categories', 'venue.location.lat', 'venue.location.lng']
         nearby_venues =nearby_venues.loc[:, filtered_columns]

         # filter the category for each row
         nearby_venues['venue.categories'] = nearby_venues.apply(get_category_type, axis=1)

         # clean columns
         nearby_venues.columns = [col.split(".")[-1] for col in nearby_venues.columns]

         nearby_venues
```

Out[48]:

	name	categories	lat	lng
0	The Gym	Gym / Fitness Center	51.540819	-0.298715
1	Sainsbury's	Supermarket	51.538740	-0.303272
2	Maru Bhajias	Indian Restaurant	51.543873	-0.297200
3	Subway	Sandwich Place	51.541707	-0.297996
4	East Pan Asian Restaurant	Asian Restaurant	51.537700	-0.301996
5	Loon Fung	Supermarket	51.537559	-0.301984
6	Alperton Depot	Train Station	51.540792	-0.299769
7	The Apple Tree Cakes	Café	51.540744	-0.299267
8	Genesis Gym	Gym / Fitness Center	51.537300	-0.303738

It is observed that among the 8 venues reported by Foursquare API, there were 4 venues that belong to food service industry among which two are Asian Restaurants. It should be noted that since we are limited by availability of data, we focus on what we have.

Now, we apply the same methodology to all the neighborhoods to obtain the top 100 venues in each neighborhood.

```
In [49]: def getNearbyVenues(names, latitudes, longitudes, radius=1000):

    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={}&
            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 getNearbyVenues function is used to create a dataframe called 'London_venues' that contains venues and their details for each neighborhood.

```
: london_venues = getNearbyVenues(names=df['Neighborhood'],
                                   latitudes=df['Latitude'],
                                   longitudes=df['Longitude']
                                  )
```

The resulted data frame has 4711 rows and 7 features that include Neighborhood, Neighborhood Latitude, Neighborhood Longitude, Venue, Venue Latitude, Venue Longitude and Venue category. The first few rows of the data frame is shown below:

```
n [51]: print(london_venues.shape)
        london_venues.head()

        (4711, 7)
```

ut[51]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Aldborough Hatch	51.58559	0.09875	Miller & Carter	51.585350	0.102610	Steakhouse
1	Aldborough Hatch	51.58559	0.09875	Redbridge FC	51.585761	0.088673	Soccer Field
2	Aldborough Hatch	51.58559	0.09875	Barkingside London Underground Station	51.585392	0.088590	Metro Station
3	Aldborough Hatch	51.58559	0.09875	Sainsbury's	51.581451	0.087260	Supermarket
4	Aldborough Hatch	51.58559	0.09875	Jalalabad 2 Restaurant	51.577818	0.099732	Indian Restaurant

The number of venues returned for each neighborhood can be observed as follows:

```
In [52]: london_venues.groupby('Neighborhood').count()
```

Out[52]:

Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
Aldborough Hatch	8	8	8	8	8	8
Alperton	17	17	17	17	17	17
Barkingside	22	22	22	22	22	22
Beckton	22	22	22	22	22	22
Bedford Park	100	100	100	100	100	100
Bethnal Green	100	100	100	100	100	100
Blackwall	100	100	100	100	100	100
Bow	46	46	46	46	46	46
Brent Park	76	76	76	76	76	76
Brentford	44	44	44	44	44	44

One Hot Encoding:

One-Hot encoding is applied to ‘Venue Category’ column in london_venues dataframe to convert categorical variables to integers.

```
In [54]: # one hot encoding
london_onehot = pd.get_dummies(london_venues[['Venue Category']], prefix="", prefix_sep="")

# add neighborhood column back to dataframe
london_onehot['Neighborhood'] = london_venues['Neighborhood']

# move neighborhood column to the first column
fixed_columns = [london_onehot.columns[-1]] + list(london_onehot.columns[:-1])
london_onehot = london_onehot[fixed_columns]

london_onehot.head()
```

Let’s check the encoded dataframe for Asian restaurants:

```
london_onehot.loc[london_onehot['Asian Restaurant'] != 0]
```

	Neighborhood	Accessories Store	Afghan Restaurant	Airport	Airport Lounge	Airport Service	Airport Terminal	American Restaurant	Antique Shop	Arepa Restaurant	Argentinian Restaurant	Art Gallery	Arts & Crafts Store	Asian Restaurant
12	Alperton	0	0	0	0	0	0	0	0	0	0	0	0	1
361	Blackwall	0	0	0	0	0	0	0	0	0	0	0	0	1
420	Brentford	0	0	0	0	0	0	0	0	0	0	0	0	1
458	Brentford	0	0	0	0	0	0	0	0	0	0	0	0	1
490	Brent Park	0	0	0	0	0	0	0	0	0	0	0	0	1
550	Bromley	0	0	0	0	0	0	0	0	0	0	0	0	1
772	Canary Wharf	0	0	0	0	0	0	0	0	0	0	0	0	1
1042	Custom House	0	0	0	0	0	0	0	0	0	0	0	0	1
1106	Custom House	0	0	0	0	0	0	0	0	0	0	0	0	1
1229	Ealing	0	0	0	0	0	0	0	0	0	0	0	0	1
1336	Forest Gate	0	0	0	0	0	0	0	0	0	0	0	0	1
1381	Goodmayes	0	0	0	0	0	0	0	0	0	0	0	0	1
1552	Hanwell	0	0	0	0	0	0	0	0	0	0	0	0	1
1773	Hounslow	0	0	0	0	0	0	0	0	0	0	0	0	1

Grouping rows by neighborhood and by taking the mean of the frequency of occurrence of each category will result in the following dataframe:

```
In [56]: london_grouped = london_onehot.groupby('Neighborhood').mean().reset_index()
london_grouped
```

	Neighborhood	Accessories Store	Afghan Restaurant	Airport	Airport Lounge	Airport Service	Airport Terminal	American Restaurant	Antique Shop	Arepa Restaurant	Argentinian Restaurant	Art Gallery	Arts & Crafts Store	Asian Restaurant
0	Aldborough Hatch	0.00	0.00	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00	0.000000	0.000000	0.000000	0.000000
1	Alperton	0.00	0.00	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00	0.000000	0.000000	0.000000	0.058824
2	Barkingside	0.00	0.00	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00	0.000000	0.000000	0.000000	0.000000
3	Beckton	0.00	0.00	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00	0.000000	0.000000	0.000000	0.000000
4	Bedford Park	0.00	0.00	0.000000	0.000000	0.000000	0.000000	0.000000	0.010000	0.00	0.010000	0.000000	0.000000	0.000000
5	Bethnal Green	0.00	0.00	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.01	0.010000	0.010000	0.000000	0.000000
6	Blackwall	0.00	0.00	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00	0.000000	0.000000	0.000000	0.010000
7	Bow	0.00	0.00	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00	0.000000	0.021739	0.000000	0.000000
8	Brent Park	0.00	0.00	0.000000	0.000000	0.000000	0.000000	0.026316	0.000000	0.00	0.000000	0.000000	0.000000	0.013158

Let’s explore the top 5 most common venues for each neighborhood and print them:

```
num_top_venues = 5

for hood in london_grouped['Neighborhood']:
    print("----"+hood+"----")
    temp = london_grouped[london_grouped['Neighborhood'] == hood].T.reset_index()
    temp.columns = ['venue','freq']
    temp = temp.iloc[1:]
    temp['freq'] = temp['freq'].astype(float)
    temp = temp.round({'freq': 2})
    print(temp.sort_values('freq', ascending=False).reset_index(drop=True).head(num_top_venues))
    print('\n')
```

```
----Aldborough Hatch----
                venue  freq
0         Social Club  0.12
1  Sporting Goods Shop  0.12
2    Indian Restaurant  0.12
3         Soccer Field  0.12
4        Metro Station  0.12


----Alperton----
                 venue  freq
0     Indian Restaurant  0.12
1           Supermarket  0.12
2  Gym / Fitness Center  0.12
3        Clothing Store  0.06
4                  Café  0.06
```

We then create a new dataframe with Neighborhoods and their top 10 most common venues resulted from the foursquare API call

```
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
neighborhoods_venues_sorted = pd.DataFrame(columns=columns)
neighborhoods_venues_sorted['Neighborhood'] = london_grouped['Neighborhood']

for ind in np.arange(london_grouped.shape[0]):
    neighborhoods_venues_sorted.iloc[ind, 1:] = return_most_common_venues(london_grouped.iloc[ind, :], num_top_venu

neighborhoods_venues_sorted.head()
```

The part of the resulting dataframe is as follows:

	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	Aldborough Hatch	Metro Station	Sporting Goods Shop	Café	Supermarket	Steakhouse	Indian Restaurant	Social Club	Soccer Field	Field	Fast Food Restaurant
1	Alperton	Supermarket	Gym / Fitness Center	Indian Restaurant	Hookah Bar	Asian Restaurant	Fast Food Restaurant	Café	Sandwich Place	Electronics Store	Train Station
2	Barkingside	Supermarket	Café	Indian Restaurant	Greek Restaurant	Steakhouse	Metro Station	Sandwich Place	Sporting Goods Shop	Pub	Coffee Shop
3	Beckton	Supermarket	Coffee Shop	Furniture / Home Store	Convenience Store	Grocery Store	Light Rail Station	Soccer Field	Moving Target	Bus Stop	Discount Store
4	Bedford Park	Pub	Coffee Shop	Café	Bakery	Italian Restaurant	Park	Grocery Store	Burger Joint	Ice Cream Shop	French Restaurant

Clustering:

The neighborhoods are clustered using K-Means Clustering. Number of clusters was decided using elbow method. The algorithm has global optimum at k=4. Hence, neighborhoods are segmented into 4 clusters.

```
]: # set number of clusters
kclusters = 4

london_grouped_clustering = london_grouped.drop('Neighborhood', 1)

# run k-means clustering
kmeans = KMeans(n_clusters=kclusters, random_state=0).fit(london_grouped_clustering)

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

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

Now, we create a new dataframe that includes Cluster labels, Neighborhoods and its most common venues. The dataframe is as follows:

```
# add clustering labels
neighborhoods_venues_sorted.insert(0, 'Cluster Labels', kmeans.labels_)

london_merged = df

# merge toronto_grouped with toronto_data to add latitude/longitude for each neighborhood
london_merged = london_merged.join(neighborhoods_venues_sorted.set_index('Neighborhood'), on='Neighborhood')

london_merged.head() # check the last columns!
```

	Neighborhood	Borough	Post_town	Post_code	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	7t C
0	Aldborough Hatch	Redbridge	ILFORD	IG2	51.585590	0.098750	0	Metro Station	Sporting Goods Shop	Café	Supermarket	Steakhouse	Indian Restaurant	
1	Alperton	Brent	WEMBLEY	HA0	51.540804	-0.300096	0	Supermarket	Gym / Fitness Center	Indian Restaurant	Hookah Bar	Asian Restaurant	Fast Food Restaurant	
2	Barkingside	Redbridge	ILFORD	IG6	51.585818	0.088624	0	Supermarket	Café	Indian Restaurant	Greek Restaurant	Steakhouse	Metro Station	Sa
3	Beckton	Newham	LONDON, BARKING	E6, E16, IG11	51.516080	0.059426	1	Supermarket	Coffee Shop	Furniture / Home Store	Convenience Store	Grocery Store	Light Rail Station	
4	Bedford Park	Ealing	LONDON	W4	51.498020	-0.255647	2	Pub	Coffee Shop	Café	Bakery	Italian Restaurant	Park	

Examining Clusters:

Now, we examine each cluster and determine the discriminating venue categories that distinguish each cluster. Based on the defining categories, you can then assign a name to each cluster.

Cluster – 0:

```
london_merged.loc[london_merged['Cluster Labels'] == 0, london_merged.columns[[1] + list(range(5, london_merged.shap
```

	Borough	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	10t Cc
0	Redbridge	0.098750	0	Metro Station	Sporting Goods Shop	Café	Supermarket	Steakhouse	Indian Restaurant	Social Club	Soccer Field	Flea Market	Fish
1	Brent	-0.300096	0	Indian Restaurant	Clothing Store	Supermarket	Soccer Field	Bus Stop	Sandwich Place	Electronics Store	Fast Food Restaurant	Café	
2	Redbridge	0.088624	0	Supermarket	Indian Restaurant	Café	Middle Eastern Restaurant	Steakhouse	Sandwich Place	Sporting Goods Shop	Coffee Shop	Pub	Res
14	Waltham Forest	0.015682	0	Restaurant	Grocery Store	Playground	Soccer Field	Garden Center	Beer Garden	Bar	Thai Restaurant	Italian Restaurant	
17	Brent	-0.191890	0	Turkish Restaurant	Supermarket	Indian Restaurant	Pizza Place	Coffee Shop	Pub	Restaurant	Japanese Restaurant	Park	
22	Brent	-0.239021	0	Park	Yoga Studio	Italian Restaurant	Pizza Place	Discount Store	Coffee Shop	Restaurant	Sandwich Place	Fast Food Restaurant	
25	Newham	0.055320	0	Pub	Fast Food Restaurant	Indian Restaurant	Grocery Store	Supermarket	Park	Sandwich Place	Discount Store	Furniture / Home Store	Elec
26	Hillingdon	-0.401653	0	Grocery Store	Metro Station	Indian Restaurant	Sandwich Place	Bakery	Coffee Shop	Café	Supermarket	Bus Stop	
27	Hounslow	-0.410659	0	Pharmacy	Supermarket	Clothing Store	Historic Site	Portuguese Restaurant	Fast Food Restaurant	Movie Theater	Sandwich Place	Bar	
29	Redbridge	0.065308	0	Pizza Place	Grocery Store	Thai Restaurant	Park	Pub	Chinese Restaurant	Nightclub	Museum	Café	Socc

Cluster – 1:

```
london_merged.loc[london_merged['Cluster Labels'] == 1, london_merged.columns[[1] + list(range(5, london_merged.shap
```

	Borough	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 Co
3	Newham	0.059426	1	Supermarket	Furniture / Home Store	Light Rail Station	Coffee Shop	Grocery Store	Shopping Plaza	Soccer Field	Clothing Store	Café	Bu
6	Tower Hamlets	-0.007184	1	Coffee Shop	Sushi Restaurant	Italian Restaurant	Park	Hotel	Burger Joint	Sandwich Place	Hotel Bar	Café	Jui
9	Brent	-0.275760	1	Coffee Shop	Bar	Clothing Store	Hotel	Sandwich Place	Supermarket	Sporting Goods Shop	Warehouse Store	Indian Restaurant	Food
13	Tower Hamlets	-0.025717	1	Park	Coffee Shop	Hotel	Plaza	Gym / Fitness Center	Italian Restaurant	Burger Joint	Café	Indian Restaurant	Ho
15	Newham	0.008299	1	Grocery Store	Café	Hotel	Gym / Fitness Center	Coffee Shop	Fast Food Restaurant	Platform	Park	Science Museum	Sar
19	Hounslow	-0.409677	1	Bus Stop	Hotel	Restaurant	Pizza Place	Park	Garden Center	Smoke Shop	Fast Food Restaurant	Deli / Bodega	E Rest
20	Tower Hamlets	-0.008243	1	Indian Restaurant	Pub	Bus Stop	Park	Business Service	Sandwich Place	Hotel	Bubble Tea Shop	Café	
21	Newham	-0.082644	1	Hotel	Coffee Shop	Gym / Fitness Center	Restaurant	Hotel Bar	Garden	Scenic Lookout	Asian Restaurant	Italian Restaurant	Steak
39	Hillingdon	-0.436303	1	Hotel	Bus Stop	Hotel Bar	Pub	Indian Restaurant	Gym	Coffee Shop	Rental Car Location	Restaurant	

Cluster – 2:

```
london_merged.loc[london_merged['Cluster Labels'] == 2, london_merged.columns[[1] + list(range(5, london_merged.shap
```

	Borough	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 Mo Commo Venu
4	Ealing	-0.255647	2	Pub	Coffee Shop	Café	Bakery	Park	Italian Restaurant	Burger Joint	Grocery Store	Ice Cream Shop	Frenc Restaura
5	Tower Hamlets	-0.056163	2	Coffee Shop	Pub	Café	Cocktail Bar	Hotel	Restaurant	Bookstore	Flower Shop	Bakery	Convenienc Sto
8	Hounslow	-0.321662	2	Coffee Shop	Pub	Café	Hotel	Gym	Italian Restaurant	Gastropub	Sandwich Place	Canal Lock	Asia Restaura
10	Tower Hamlets	0.014814	2	Clothing Store	Pub	Coffee Shop	Café	Bar	Indian Restaurant	Supermarket	Burger Joint	Park	Gym Fitne Cent
11	Brent	-0.202596	2	Pub	Coffee Shop	Farmers Market	Bakery	Italian Restaurant	Brazilian Restaurant	Middle Eastern Restaurant	Pizza Place	Mediterranean Restaurant	Cocktail B
12	Tower Hamlets	-0.057422	2	Coffee Shop	Pub	Café	Cocktail Bar	Hotel	Bakery	Bookstore	Wine Bar	Flower Shop	Restaura
18	Hillingdon	-0.476384	2	Pub	Restaurant	Coffee Shop	Hotel	Sandwich Place	Canal Lock	Supermarket	Café	Nightclub	Gy
23	Ealing	-0.305195	2	Coffee Shop	Pub	Hotel	Park	Café	Thai Restaurant	Italian Restaurant	Pizza Place	Burger Joint	Bake
24	Hounslow	-0.441359	2	Coffee Shop	Café	Burger Joint	IT Services	Indian Restaurant	Restaurant	Pub	Grocery Store	Fast Food Restaurant	Yoga Stud
31	Ealing	-0.345351	2	Clothing Store	Pub	Hotel	Bakery	Portuguese Restaurant	Coffee Shop	Chinese Restaurant	Sandwich Place	Café	Fast Foo Restaura

Cluster – 3:

```
london_merged.loc[london_merged['Cluster Labels'] == 3, london_merged.columns[[1] + list(range(5, london_merged.shap
```

	Borough	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
7	Tower Hamlets	-0.019482	3	Pub	Grocery Store	Café	Coffee Shop	Bar	Hotel	Pharmacy	Bike Rental / Bike Share	Burger Joint	Bus Stop
16	Waltham Forest	0.003996	3	Pub	Italian Restaurant	Coffee Shop	Fast Food Restaurant	History Museum	Nature Preserve	Convenience Store	Bakery	Park	Grocery Store
28	Newham	0.024925	3	Grocery Store	Hotel	Bus Stop	Pub	Café	Market	Playground	Bar	Ice Cream Shop	Comfort Food Restaurant
37	Hillingdon	-0.482562	3	Café	Canal Lock	Grocery Store	Gastropub	Pub	Sports Club	Yoga Studio	Farmers Market	Factory	Falafel Restaurant
45	Hillingdon	-0.448335	3	Chinese Restaurant	Fast Food Restaurant	Pub	Grocery Store	Park	Pharmacy	Plaza	Construction & Landscaping	Farmers Market	Ethiopian Restaurant
47	Hillingdon	-0.442351	3	Pub	Grocery Store	Chinese Restaurant	Train Station	Bakery	Park	Bus Stop	Indian Restaurant	Fish & Chips Shop	Metro Station
50	Hounslow	-0.326311	3	Pub	Bus Stop	Grocery Store	Pharmacy	Fast Food Restaurant	River	Restaurant	Coffee Shop	Italian Restaurant	Trail
56	Waltham Forest	0.006424	3	Pub	Grocery Store	Café	Fast Food Restaurant	Coffee Shop	Pizza Place	Hotel	Pharmacy	Stationery Store	Burger Joint
60	Redbridge	0.084051	3	Pizza Place	Park	Betting Shop	Portuguese Restaurant	Gas Station	Ice Cream Shop	Pub	Grocery Store	Exhibit	Factory
70	Ealing	[illegible]	3	Canal Lock	Ice Cream	Grocery	Pub	Gym / Fitness	Yoga	Fast Food	Exhibit	Factory	Falafel

Visualizing the Clusters:

We can use folium library to visualize the clusters on London map using different color markers for each cluster.

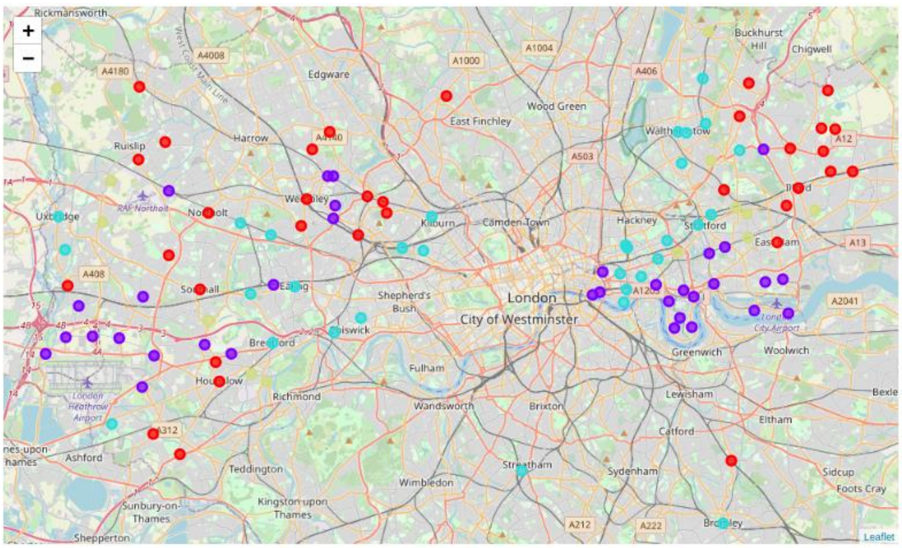
```
# create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=10)

# set color scheme for the clusters
x = np.arange(kclusters)
ys = [i + x + (i*x)**2 for i in range(kclusters)]
colors_array = cm.rainbow(np.linspace(0, 1, len(ys)))
rainbow = [colors.rgb2hex(i) for i in colors_array]

# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(london_merged['Latitude'], london_merged['Longitude'], london_merged['Neighborhood
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    folium.CircleMarker(
        [lat, lon],
        radius=5,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7).add_to(map_clusters)

map_clusters
```

12]:



Results:

The following inferences can be made:

1. In Cluster-0, It is observed that Asian cuisines like Indian, Thai and Turkish restaurants are among the top-10 of most common venues in the almost every neighborhood. Hence, would not be an ideal place to start the business.
2. In Cluster-1, coffee shops and Hotels are among the most common venues.
3. Pubs, Cafes and Parks are the most common venues in the neighborhoods of Cluster-2.
4. Pubs are the most common venue in almost all the neighborhoods in Cluster-3, with Asian restaurants appearing often.

Discussion and Conclusion:

It should be noted that, from the above observations Cluster-1 and Cluster-2 neighborhoods would be viable option to open an Asian Restaurant. Since, both cluster neighborhoods do not have top standard restaurants and the market in these neighborhoods is open for a good restaurant. However, this project is done with a limited scope of data i.e Demographics, Neighborhoods and Venues. And also, since I'm confined to free foursquare developer account, I wasn't able to get current restaurants ratings and reviews.

Hence, this project can be extended further by considering various other factors for segmenting and clustering neighborhoods of London. Much better results can be achieved by considering factors like locations proximity to public transport access and locations visibility.