Analysis of Restaurant distribution in the city of Chennai

1) *Introduction to the Business problem*

The restaurant industry is one that involves a large amount of manpower, labour and investment to pull off successfully. Considering the fact that it is such a vast industry, it warrants a great deal of thought and analysis. One cannot start a restaurant willy nilly and hope for success of his/her franchise.

With this in mind, my project aims at performing an analysis on the city of Chennai in the country of India. Chennai is one of the major metropolitan cities in the country of India and considering the fact that India is a developing nation with an ever-growing middle class, it makes sense that a large number of stakeholders could benefit from this analysis.

Due to the aforementioned reasons, there is a great deal of potential in analysing the distribution of restaurants in different neighbourhoods of the city.

The goal of the project would be to answer the following question:-

- What are the various restaurant categories in the city of Chennai?
- What is the quantitative and area based distributions of these restaurants?
- How would you analyse these areas based on their restaurant distribution?

Due to the vastness in coverage of information, any individual related to the restaurant field can derive benefit from this model by understanding the various trends and distributions.

This model is especially useful for stakeholders who are opening up new venues and are interest in analysing the areas with the greatest potential to turn a profit.

2) *Data*

To begin, we need to figure out the different types of data that we require to solve the problem and identify the sources that provide the required data.

The different information we require are:-

- The different neighbourhoods in the city of Chennai
- Their respective latitudes and longitudes
- Venue data for each location which can be obtained from the foursquare API
- The latitude and longitude of the city's centre

A) GETTING THE LATITUDE AND LONGITUDE OF THE DIFFERENT NEIGHBOURHOODS

Upon surfing through the internet, a website was found that contained the names of the various neighbourhoods in the city of Chennai and their respective latitudes and longitude details.

The website address :- https://chennaiiq.com/chennai/latitude_longitude_areas.asp

Upon analysing the websites html, it was found that the data was stored under the stable tag and had a specific class name. By using the BeautifulSoup module in python, the specific table under consideration was imported as a html file.

```
In [4]: url='https://chennaiiq.com/chennai/latitude_longitude_areas.asp'
page=requests.get(url)
soup=BeautifulSoup(page.text,'html.parser')
tables=soup.find_all('table',{'class':'TBox'})
tables
```

```
<tr class="tabH1">
<td>S.No.</td>
<td>Location</td>
<td>Latitude</td>
<td>Longitude</td>
</tr>
<tr class="tab"><td align="right">1</td><td> Adyar Bus Debot.</td><td> 12°59'50" N</td><td> 80°15'25" E</td></tr><tr class
="tab"><td align="right">2</td><td> Adyar Signal</td><td> 13°00'23" N</td><td> 80°15'27" E</td></tr><tr class="tab"><td align
="right">3</td><td> Alandur</td><td> 13°00'28" N</td><td> 80°12'35" E</td></tr><tr class="tab"><td align="right">4</td><td> A
mbattur</td><td> 13°06'36" N</td><td> 80°10'12" E</td></tr><tr class="tab"><td align="right">5</td><td> Anna Arch</td><td> 1
3°04'28" N</td><td> 80°13'06" E</td></tr><tr class="tab"><td align="right">6</td><td> Anna Nagar Roundana</td><td> 13°05'0
4" N</td><td> 80°13'05" E</td></tr><tr class="tab"><td align="right">7</td><td> Anna Nagar West Terminus</td><td> 13°05'35" N
</td><td> 80°11'55" E</td></tr><tr class="tab"><td align="right">8</td><td> Anna Statue</td><td> 13°04'05" N</td><td> 80°16'1
9" E</td></tr><tr class="tab"><td align="right">9</td><td> Anna University Entrance</td><td> 13°00'29" N</td><td> 80°14'06" E
</td></tr><tr class="tab"><td align="right">10</td><td> Avadi</td><td> 13°07'13" N</td><td> 80°06'36" E</td></tr><tr class="t
```

Upon further analysis, it is found that a new row is demarcated by the <tr><tr> tag in html and each individual data element in the row was demarcated by the <td><td> tag in html. By using the find_all method in a nested for loop, we can extract the individual columns into respective lists and these lists can be combined to form a pandas dataframe as shown below.

	Neighborhood	latitude	longitude
1	Adyar Bus Debot.	12°59'50" N	80°15'25" E
2	Adyar Signal	13°00'23" N	80°15'27" E
3	Alandur	13°00'28" N	80°12'35" E
4	Ambattur	13°06'36" N	80°10'12" E
5	Anna Arch	13°04'28" N	80°13'06" E

Upon close inspection, it can be seen that the latitude and longitude value is returned as a string and is represented in the degrees minutes seconds (DMS) system of angle representation, along with the direction mention as N/S/E/W. While this is a perfectly accurate representation of a location, it is not very easy to feed into folium, and in general, a string representation of a numerical data that contains an alphabet can be very difficult to utilize.

As a result, a function was written to convert the location details from DMS system of representation to the Decimal system of representation.

```
def dms_to_dec_latlng(lat,long):
    min_pat=re.compile(r'[°](\d{2})[\']')
    sec_pat=re.compile(r'[\'](\d{2})[\"]')
    minutes_lat=(float(min_pat.search(lat).group(1)))/60
    seconds_lat=(float(sec_pat.search(lat).group(1)))/3600
    minutes_long=(float(min_pat.search(long).group(1)))/60
    seconds_long=(float(sec_pat.search(long).group(1)))/3600
    dec_lat=minutes_lat+seconds_lat
    dec_long=minutes_long+seconds_long
    pattern_south=re.compile(r'S$')
    pattern_west=re.compile(r'W$')
    south=pattern_south.search(lat)
    west=pattern_west.search(long)
    if (south==None):
        if (west==None):
            return [(float(lat[0:2])+dec_lat),(float(long[0:2])+dec_long)]
        else:
            return [(float(lat[0:2])+dec_lat),((float(long[0:2])+dec_long))*(-1)]
    else:
        if(west==None):
            return [((float(lat[0:2])+dec_lat))*(-1),(float(long[0:2])+dec_long)]
        else:
            return [((float(lat[0:2])+dec_lat))*(-1),((float(long[0:2])+dec_long))*(-1)]
```

```
for index in range(1,(df.shape[0]+1)):
    lat=df.loc[index,'latitude']
    lng=df.loc[index,'longitude']
    fixed_loc=dms_to_dec_latlng(lat,lng)
    df.at[index,'latitude']=fixed_loc[0]
    df.at[index,'longitude']=fixed_loc[1]
```

Upon running the above code block, the dataframe is converted to a form as shown below.

	Neighborhood	latitude	longitude
1	Adyar Bus Debot.	12.9972	80.2569
2	Adyar Signal	13.0064	80.2575
3	Alandur	13.0078	80.2097
4	Ambattur	13.11	80.17
5	Anna Arch	13.0744	80.2183
...	...	...	...
101	Velachery Bus Terminus	12.9758	80.2208
102	Villivakkam	13.11	80.2033
103	Vyasar Padi	13.1092	80.2625
104	Washermanpet	13.1086	80.2811
105	Woodlands Drive In	13.05	80.2511

105 rows × 3 columns

This is a format that is much easier to work with and makes handling the location data much cleaner.

Now that we have obtained the location data, we can feed this into Foursquare using API calls and retrieve the venue information from their website.

B) FOURSQUARE API CALLING TO RETRIEVE VENUE INFORMATION

The foursquare database can only be accessed using API calls that contain the user authentication information as well as the respective venue location and the endpoint call.

In our case the information we provide through an API call are:-

- Client ID and client Secret
- Version of foursquare
- The endpoint – explore? in our case
- The latitude and longitude of that location
- The radius (or distance about specified point up to which venues are to be returned)
- The limit value which limits the number of datapoints sent back

An API call would look like this

```
https://api.foursquare.com/v2/venues/categories?&client_id=GIL5KFTBUZBWMUCEWGMC1NISL2E1BFTEWROCHKPEJDBQVU2M&client_secret=TBEJZ
QQAZ4SCSEZ2WZIJ2GPTOMO4C1ZQ4FI5OC5YBSJKT450&v=20200404&ll=13.0801721,80.2838331&radius=1000&limit=100
```

By running a loop that creates and sends an API call for each neighbourhood is made and the data that is returned is placed into a pandas dataframe that looks like this.

	Neighborhood	Area Latitude	Area Longitude	Venue Name	Venue Latitude	Venue Longitude	Venue Category
0	Adyar Bus Debot.	12.997222	80.256944	Zaitoon Restaurant	12.996861	80.256178	Middle Eastern Restaurant
1	Adyar Bus Debot.	12.997222	80.256944	Kuttanadu Restaurant	12.997010	80.257799	Asian Restaurant
2	Adyar Bus Debot.	12.997222	80.256944	Zha Cafe	12.999730	80.254806	Café
3	Adyar Bus Debot.	12.997222	80.256944	Kovai Pazhamudir Nilayam	12.996522	80.259776	Fruit & Vegetable Store
4	Adyar Bus Debot.	12.997222	80.256944	Domino's Pizza	13.002000	80.254000	Pizza Place

With this all the necessary data has been retrieved from the interweb and put into easily accessible dataframes and csv files. Now we can move on to the methodology section of the project.

3) ***METHODOLOGY AND ANALYSIS***

The data now has to be analysed and different inferences have to be derived from it. Furthermore, the data has to be grouped into meaningful sets of data to make it easy for the stakeholder to identify regions based on his requirements.

A) DISTRIBUTION ANALYSIS OF THE NEIGHBOURHOODS

The dataframe obtained from the Foursquare API has information regarding every venue surrounding the given location coordinates. However, our analysis requires only the restaurants in the neighbourhood. The initial dataframe has the following 182 unique columns.

```
venue_list=chennai_venues['Venue Category'].unique()
print(venue_list.shape[0])
venue_list
```

```
182

array(['Middle Eastern Restaurant', 'Asian Restaurant', 'Café',
       'Fruit & Vegetable Store', 'Pizza Place', 'Fast Food Restaurant',
       "Women's Store", 'Indian Restaurant', 'Chinese Restaurant',
       'Diner', 'Sandwich Place', 'Gym', 'BBQ Joint', 'Hotel',
       'Multiplex', 'Movie Theater', 'Arcade', 'Breakfast Spot',
       'Electronics Store', 'Hotel Bar', 'Juice Bar', 'Bakery',
       'Ice Cream Shop', 'Pharmacy', 'Shopping Mall',
       'North Indian Restaurant', 'Vegetarian / Vegan Restaurant',
       'Grocery Store', 'Bookstore', 'Dessert Shop', 'Rock Club',
       'Lounge', 'Snack Place', 'Mediterranean Restaurant',
       'Italian Restaurant', 'Coffee Shop', 'Department Store',
       'Comfort Food Restaurant', 'Furniture / Home Store',
       'Gourmet Shop', 'Bus Station', 'Bar', 'Pool Hall', 'Airport',
       'Golf Course', 'Cafeteria', 'Supermarket', 'Gas Station',
       'Clothing Store', 'South Indian Restaurant', 'Scenic Lookout',
       'Food', 'Sporting Goods Shop', 'Pub', 'Event Space', 'Park',
       'Restaurant', 'Burger Joint', 'Paper / Office Supplies Store',
       'Shoe Store', 'Farmers Market', 'Bistro', 'Bank', 'Jewelry Store',
```

By analysing the different category labels of each of the venues, we can identify the ones that correspond to a restaurant or food establishment. By writing a string matching code which checks for the keywords corresponding to restaurants in the venues list, we get a dataframe with the following 56 categories which all correspond to restaurants.

```
56

array(['Middle Eastern Restaurant', 'Asian Restaurant', 'Pizza Place',
       'Fast Food Restaurant', 'Indian Restaurant', 'Chinese Restaurant',
       'Diner', 'Sandwich Place', 'BBQ Joint', 'Breakfast Spot',
       'Juice Bar', 'North Indian Restaurant',
       'Vegetarian / Vegan Restaurant', 'Dessert Shop', 'Snack Place',
       'Mediterranean Restaurant', 'Italian Restaurant', 'Coffee Shop',
       'Comfort Food Restaurant', 'Cafeteria', 'South Indian Restaurant',
       'Food', 'Restaurant', 'Burger Joint', 'Bistro', 'Food Court',
       'Multicuisine Indian Restaurant', 'Korean Restaurant',
       'Rajasthani Restaurant', 'Steakhouse', 'Mexican Restaurant',
       'American Restaurant', 'Halal Restaurant', 'Bed & Breakfast',
       'African Restaurant', 'Kerala Restaurant', 'Thai Restaurant',
       'Buffet', 'Kebab Restaurant', 'Bengali Restaurant',
       'Japanese Curry Restaurant', 'Portuguese Restaurant',
       'Seafood Restaurant', 'Japanese Restaurant',
       'Molecular Gastronomy Restaurant', 'Modern European Restaurant',
       'Russian Restaurant', 'Hyderabadi Restaurant', 'Food Truck',
       'Andhra Restaurant', 'Food & Drink Shop', 'Fried Chicken Joint',
       'Burrito Place', 'Malay Restaurant', 'New American Restaurant',
       'Falafel Restaurant'], dtype=object)
```

Now that we have a dataframe of around 1850 restaurants belonging to 56 different categories, it would be very useful to know how many restaurants of each category are present. This can be easily obtained by using the pandas groupby method.

This is the corresponding dataframe that is returned and conveys some key insights.

Venue Category	
Indian Restaurant	625
Chinese Restaurant	115
Pizza Place	92
Fast Food Restaurant	84
Coffee Shop	83
Restaurant	78
Sandwich Place	70
Vegetarian / Vegan Restaurant	68
Italian Restaurant	64
Asian Restaurant	63
Juice Bar	63
Dessert Shop	50
South Indian Restaurant	47
Middle Eastern Restaurant	46
BBQ Joint	33
Snack Place	28
Thai Restaurant	20
Breakfast Spot	15
Food Court	14

There appears to be an overwhelming excess of Indian restaurants in the city of Chennai which would lead us to believe that any stakeholder opening a restaurant in Chennai should steer clear of opening Indian restaurants in the city. However upon clustering a few interesting observations do come up.

Similarly it can be seen that there are a great deal of Chinese and chain pizza restaurants and it would be beneficial for stakeholders to avoid them or choose clusters that have a clear lack in them.

Now we separate the restaurants into a high frequency and low frequency dataframe to cluster separately to prevent any one category skewing the other due to the overwhelming difference in their magnitude.

This is done using the piece of code below:-

```
chennai_rest_lowfreq=chennai_restaurants
for i in range(0,chennai_restaurants.shape[0]):
    cell=chennai_restaurants.loc[i,'Venue Category']
    for ele in hfr:
        if cell==ele:
            chennai_rest_lowfreq=chennai_rest_lowfreq.drop(i,axis=0)
            break
```

B) CLUSTERING THE RESTAURANTS DATA INTO SUITABLE CLUSTERS USING KMEANS

Now, to create useful clusters based on the data we use the KMEANS clustering method to generate the clusters. It works by assuming optimum centroids and constants moving them to the cluster means until two consecutive loops have no change in centroid positions. It only converges to the local optimum, not the global optimum and itll require multiple tests to get the optimum cluster.

To do this, we initially use the get dummies on the dataframe and find the mean of it grouped by the neighbourhoods to get the clustering table. It would look like the one shown below.

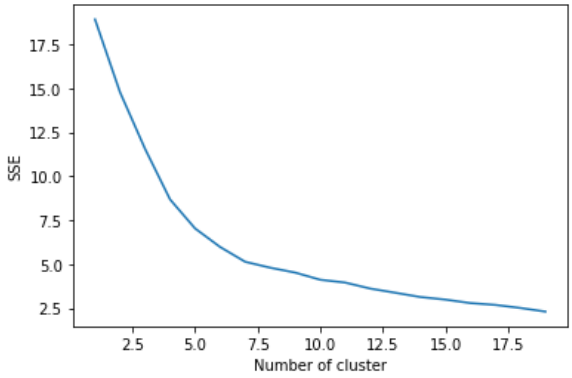
Neighborhood	Asian Restaurant	BBQ Joint	Bistro	Breakfast Spot	Buffet	Burger Joint	Chinese Restaurant	Coffee Shop	Dessert Shop	Fast Food Restaurant	...	Juice Bar	Kebab Restaurant	Middle Eastern Restaurant
AVM Studio	0.095238	0.047619	0.0	0.000000	0.0	0.0	0.000000	0.000000	0.000000	0.095238	...	0.000000	0.0	0.000000
Adyar Bus Debot.	0.090909	0.045455	0.0	0.045455	0.0	0.0	0.045455	0.000000	0.000000	0.090909	...	0.045455	0.0	0.045455
Adyar Signal	0.029412	0.000000	0.0	0.000000	0.0	0.0	0.029412	0.029412	0.058824	0.058824	...	0.058824	0.0	0.000000
Alandur	0.000000	0.000000	0.0	0.000000	0.0	0.0	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.0	0.000000
Ambattur	0.000000	0.000000	0.0	0.000000	0.0	0.0	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.0	0.000000

Now we can create a cluster model on this table and analyse the clusters. To decide how many clusters to use, we analyse the sum of squares error for the various values of k as shown below and identify the elbow point which will act as our k value.

```
sse_hf={}
sse_lf={}
for k in range(1,20):
    kmeans_hf=KMeans(n_clusters=k).fit(hf_clustering)
    kmeans_lf=KMeans(n_clusters=k).fit(lf_clustering)
    sse_hf[k]=kmeans_hf.inertia_
    sse_lf[k]=kmeans_lf.inertia_
```

k=7 is the elbow point for the high frequency data

```
plt.figure()
plt.plot(list(sse_hf.keys()), list(sse_hf.values()))
plt.xlabel("Number of cluster")
plt.ylabel("SSE")
plt.show()
```



Now we can cluster the data and add the cluster labels to the initial dataframe for easy access.

For example, cluster 6 will be represented by the dataframe shown below. It is clear that restaurants in this cluster have an abundance of vegan, Thai and BBQ restaurants and a clear lack of pizza places, snacking places and exotic cuisines.

Cluster 6 - A very high incidence of **Vegetarian and Thai restaurants** and a high incidence of **BBQ and bistro joints**

Low incidence of **Pizza places**, **snacking places**, and **exotic cuisines(apart from thai)**

```
hf_rest_sorted.loc[hf_rest_sorted['Cluster Labels']==6,:]
```

	Neighborhood	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	11th Most Common Venue
42	Korukkupet	6	Vegetarian / Vegan Restaurant	Thai Restaurant	BBQ Joint	Bistro	Breakfast Spot	Buffet	Burger Joint	Chinese Restaurant	Coffee Shop	Dessert Shop	Fast Food Restaurant
68	Poonamallee	6	Vegetarian / Vegan Restaurant	Thai Restaurant	BBQ Joint	Bistro	Breakfast Spot	Buffet	Burger Joint	Chinese Restaurant	Coffee Shop	Dessert Shop	Fast Food Restaurant
78	TVS Signal	6	Vegetarian / Vegan Restaurant	Thai Restaurant	BBQ Joint	Bistro	Breakfast Spot	Buffet	Burger Joint	Chinese Restaurant	Coffee Shop	Dessert Shop	Fast Food Restaurant
90	Tondiarpet	6	Vegetarian / Vegan Restaurant	Thai Restaurant	BBQ Joint	Bistro	Breakfast Spot	Buffet	Burger Joint	Chinese Restaurant	Coffee Shop	Dessert Shop	Fast Food Restaurant

This type of analysis is done for all the other clusters and is made into a dataframe itself.

It would look something like this and would provide very important information to the stakeholders.

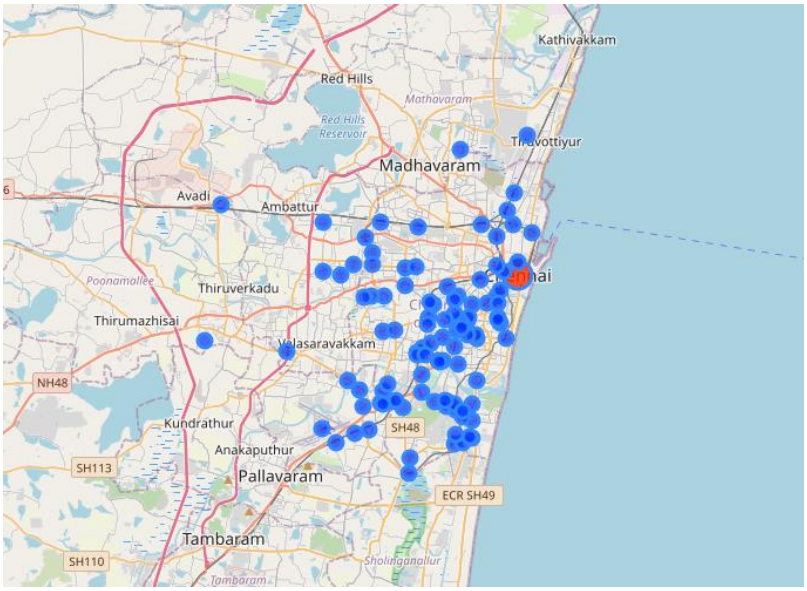
	Cluster Labels	High Frequency locations	Low Frequency Locations
0	0	Indian followed by Pizza & Chinese	BBQ, asian, Cafe & coffe shops
1	1	Fast food restaurants followed by vegan restaurants	Indian restaurants followed by exotic cuisines
2	2	Indian followed by Dessert & Breakfast shops	South Indian, Snacking & Sandwich places, Pizza places, fast food restaurants & Italian & thai restaurants
3	3	Pizza place followed by vegetarian & BBQ joints	Exotic restaurants
4	4	Not statistically significant	Not statistically significant
5	5	Indian followed by Chinese and BBQ joints	Non chinese asian, burger, buffet, snacking
6	6	Vegetarian & Thai followed by Vegan & BBQ joints	Pizza places, snacking places & exotic cuisines

For example, we saw that there is an overwhelmingly high number of Indian restaurants. However, by clustering, it is clear that all the neighbourhoods that fall into cluster number 1 have a major lack in the number of Indian restaurants. As a result, if a stakeholder is fixated on opening an Indian restaurant, he/she can initially consider cluster 1 neighbourhoods as the ideal location for opening Indian restaurants.

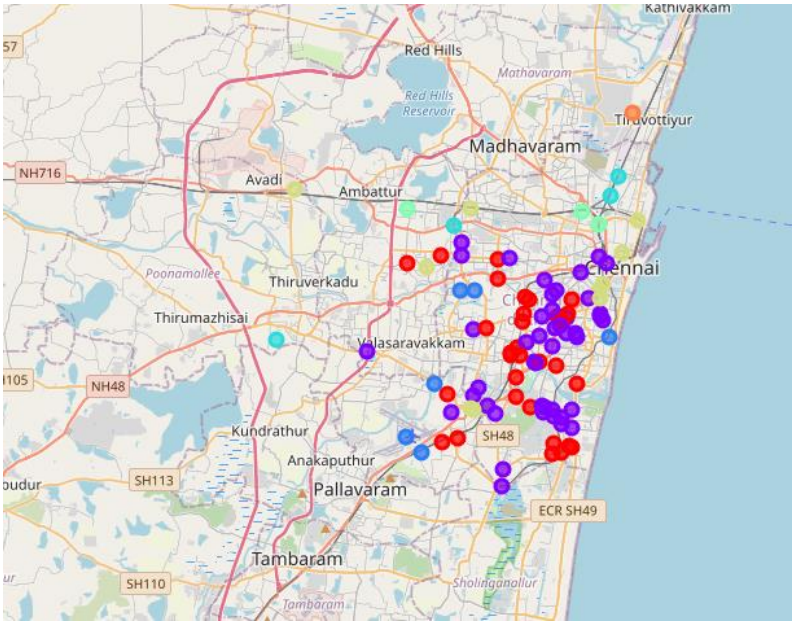
Similar analysis has been carried out for various other categories and has been placed into the dataframe shown above.

C) MAP BASED REPRESENTATION OF THE CLUSTERS

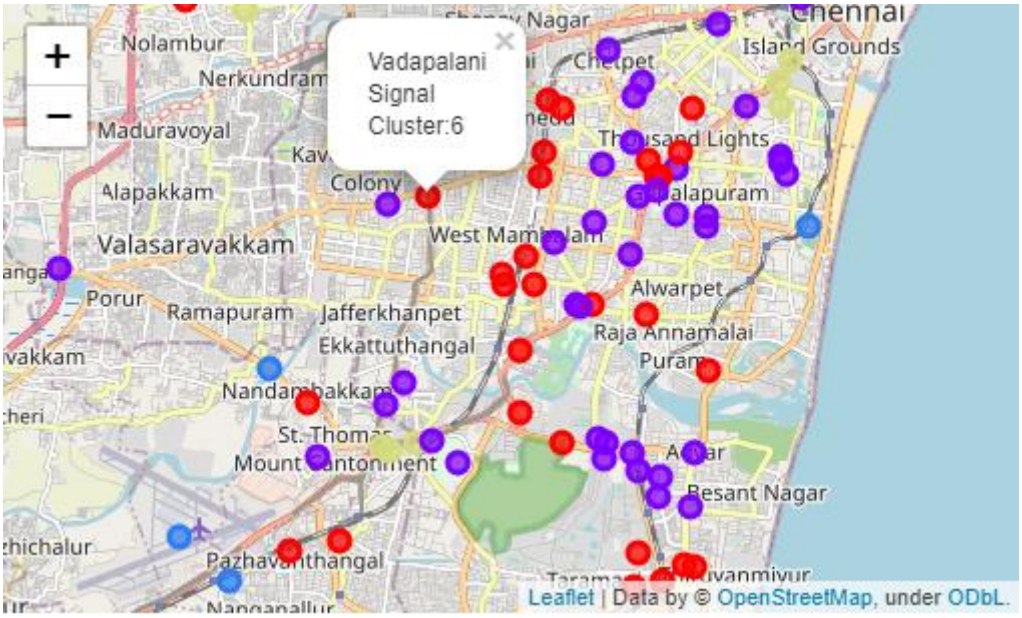
The different neighbourhoods in the city of Chennai are represented with blue circle markers while the city centre is represented by the Red marker.



When we perform unsupervised clustering on the dataset using Kmeans clustering, the neighbourhoods are plotted on the folium map with different colors for the different cluster labels. While not visible in the screenshot below, each circle marker has a popup that indicated the neighbourhood name as well as the cluster number.



When we click on any of the circle marker, we get a popup with the neighbourhood name and the cluster labels.



Furthermore, if the distance between any two neighbourhoods or the distance from the city centre is to be found, I've written a function to obtain the distance between two locations with their latitude and longitude data using the haversine formula.

```
from math import sin,cos,atan2,pi,sqrt
def get_distance(lat1,lng1,lat2,lng2):
    p=pi/180
    del_lat=(lat2-lat1)*p
    del_lng=(lng2-lng1)*p
    lat1*=p
    lat2*=p
    R=6371 ##in kms

    a=(((sin(del_lat/2))**2)+(cos(lat1)*cos(lat2))*((sin(del_lng/2))**2))
    c=2*(atan2(sqrt(a),sqrt(1-a)))
    d=R*c
    return d
```

```
print("{} kms".format(get_distance(12.997222,80.256944,13.0801721,80.2838331)))
```

```
9.6726428077056 kms
```

The function returns the information in kilometres.

4) ***RESULTS AND DISCUSSION***

The entire purpose behind this project was to provide an insight into the distribution of restaurants in the city of Chennai in Tamil Nadu, India. The analysis focuses on retrieving all the venues in Chennai, accurately filtering out the restaurants and clustering them to analyse the neighbourhoods. On our analysis, we can clearly see that certain locations clearly have a greater incidence of certain restaurants and a clear lack of others.

The analysis showed an exceptionally high number of Indian restaurants and a reasonably high number of Chinese restaurants. Some very key insights were derived in the analysis. For example, the few major areas that had a major lack of Indian restaurants were identified which could be crucial for stakeholders interested in entering the Indian restaurant game. Similar insights have shown certain key areas that are definitely not suitable for opening massive chain categories like pizza places, vegetarian restaurants etc. For a future design, the data could be fed into an app or platform where based on the type of restaurant or choice of area, the cluster data, similar neighbourhoods, distribution of restaurants etc can be easily obtained.

5) ***CONCLUSION***

The purpose of this project was the identify the various restaurant categories and their quantitative distribution in the city of Chennai. This is followed up by clustering neighbourhoods based on their distribution of restaurants. This will enable stakeholders to identify neighbourhoods that have a lack of a certain type of restaurant and will also help in preventing the establishment of a restaurant in a neighbourhood that has an overwhelming excess of that specific restaurant type.

Final decision on optimal restaurant location will be made by stakeholders based on specific characteristics of neighbourhoods and locations in every recommended zone, taking into consideration additional factors like attractiveness of each location ,real estate availability, prices, social and economic dynamics of every neighbourhood etc.