Segmentation and Clustering Neighborhoods in Chennai

Suraj Eswaran
Fort Collins,Colorado

INTRODUCTION

Chennai is the capital city of **Tamil Nadu** in India. Located on the **Coromandel coast** of the **Bay of Bengal**, Chennai enjoys **commercial**, **cultural**, and **educational** centre in South India. That is why, Chennai is known as the **Cultural Capital of South India**. Over the recent years, Chennai is recognized as the **largest exporter** of services for **IT** and **BPO**. The Chennai Metropolitan Area is one of the largest municipal economies of India. Chennai is nicknamed **"The Detroit of India"**, with more than **one-third** of India's automobile industry being based in the city. Chennai is one of the **100 Indian cities** to be developed as a **smart city** under the **Smart Cities Mission**. Chennai hosts cultural events in its vicinity. The **Madras Music Session** is one of them. The classical dance form of Tamilians is **Bharatanatyam**. **Kollywood** is the Tamil film industry located in the city that has produced great movies made over the years.

Fig 1*: View of Chennai Central Railway Station*

Chennai is located at **13.04°N 80.17°E** on the southeast coast of India and in the northeast corner of Tamil Nadu. It is located on a flat coastal plain known as the **Eastern Coastal Plains**. The city has an average elevation of **6 metres (20 ft)**,its highest point being **60 m (200 ft)**.Chennai features a **tropical wet and dry climate**. Chennai lies on the **thermal equator** and is also coastal, which prevents extreme variation in seasonal temperature. For most of the year, the weather is **hot and humid**. The hottest part of the year is **late May** and **early June**, known locally as **Agni Nakshatram ("fiery star")** or as **Kathiri Veyyil**, with maximum temperatures around **100–108 °F**. The coolest part of the year is **January**, with minimum temperatures around **64–68 °F**. The lowest temperature recorded is **57.0 °F** and highest **113 °F**. Chennai is well connected by road, rail, air, and sea. It has an **international airport** and **seaport**. Within the city a network of **bus services** and **auto-rickshaws** are common modes of transport. The historic town of **Mamallapuram** with its shore temple, about **37 miles (60 km)** south of Chennai, is a popular tourist destination

***Fig 2**: Map of Chennai*

PROBLEM FORMULATION

Chennai has witnessed a dramatic growth over the past couple of years. Even though it has various diversity of choices, it is necessary to know about the diversity of the business problem which we will discuss in the later sections in order to understand the choice of food that chennaites prefer to eat, which can help us open a restaurant in chennai. **Starting a restaurant** deals with various factors before **investors** take part in this business. So,my project is all about **analyzing various restaurants in chennai** and **preferred kind of restaurant to start at chennai**.

MOTIVATION

The main objectives of the project are:

- Analyze **various restaurants** in Chennai
- Find the **preferred restaurant** for the investor to start in Chennai.

METHODOLOGY

I utilized the concept of web scraping from a list of areas of chennai. Along with that,we have used public libraries and API's like **FourSquare API**. **Web Scraping** is a way of exporting data from the website as we don't find all the data in CSV format. When we scrape a web, we write code that sends a request to the server that's hosting the page we specified. Steps involves:

1. Data is collected from **Wikipedia** and performed data cleaning and processed to a dataframe.

2. For finding the **latitude** and **longitude**, we have utilized the Geopy.
3. Perform **one hot encoding** and found **10 common venues** at various areas.
4. Utilized **K-Nearest Cluster** to find the best restaurant to start at chennai.

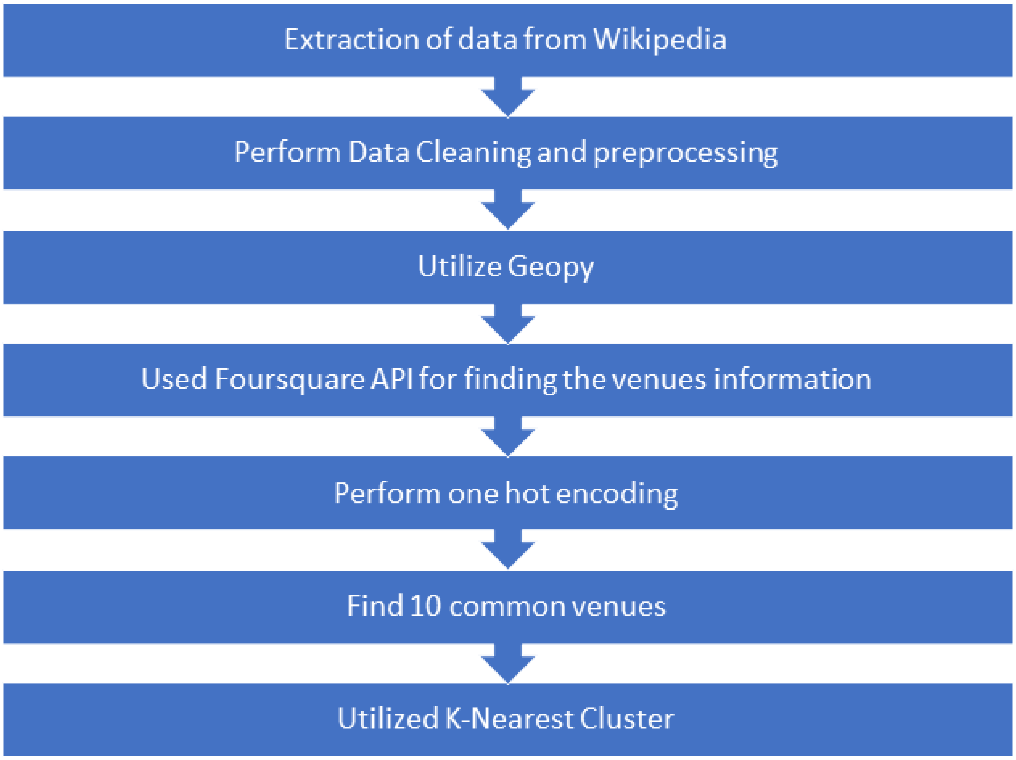

Fig 3: Steps performed for this project

Source of the Dataset

For this project, I have taken a list of **areas of chennai** from Wikipedia which consist of **15 zones** with **200 wards**. For obtaining the venue details, we have utilized the FourSquare API.

Title	Source of the Dataset
Areas Names of Chennai	https://en.wikipedia.org/wiki/Areas_of_Chennai
Venue details of Chennai	https://developer.foursquare.com/
Latitude and Longitude of Chennai Areas	https://geopy.readthedocs.io/en/stable/

Table 1: *Dataset List*

LIST OF LIBRARIES

We have utilized the several python libraries as it contains different kinds of components. Libraries contain the core dependencies of a language with some built-in functions and exceptions with the help of import. It is written in C and takes care of functionality with core modules. Now we will see the list of libraries that we have used for this project:

- **Pandas**: **Pandas** is the most important library in data science. It offers **data structures** and tools for effective **data cleaning**, **manipulation**, and **analysis**. It provides tools to work with different types of data. The primary instrument of Pandas is that it is a **low-dimensional table** consisting of **columns** and **rows**. The table is called a **DataFrame** and is designed to provide easy indexing so you can work with your data. For this project, we have utilized **version 1.2.1** as it supports **beautifulsoup and lxml libraries** and converts the data to pandas DataFrame. Below, we have printed the version of the pandas.
- **Numpy**: It is based on **arrays** as it enables you to apply **mathematical functions** to these arrays. It contains **math functions** and **scientific computing packages**.
- **Requests**: It lets you send **HTTP/1.1 requests** and form data.
- **JSON**: It is a built-in package which can be used to work with **JSON** data.
- **Matplotlib**: It is a numerical plotting library which helps in **data analyzing**.
- **Scikit-learn**: It is a **machine learning based library**.
- **BeautifulSoup**: It is a library which is used for **parsing HTML and XML documents**.
- **Folium**: It is a library used for **visualizing geospatial data**. For this project, we are installing a **0.5.0 folium version**.
- **Lxml**: It is useful for allowing the **easy handling of the HTML and XML files** and can be useful for web scraping.

```
In [2]: import numpy as np # library to handle data in a vectorized manner
pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', None)
import json # library to handle JSON files
from geopy.geocoders import Nominatim # convert an address into latitude and longitude values
import requests # library to handle requests
from pandas.io.json import json_normalize # tranform JSON file into a pandas dataframe
# Matplotlib and associated plotting modules
import matplotlib.cm as cm
import matplotlib.colors as colors
# import k-means from clustering stage
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
```

Installation of Folium Library

We don't have folium version **0.5.0** in our notebook,so we are installing the library for visualizing maps in Chennai.

```
In [3]: !conda install -c conda-forge folium=0.5.0 --yes # uncomment this line if you haven't completed the Foursquare API lab
import folium # map rendering library
print('Libraries imported.')
```

Fig 4: *Usage of Libraries for this project*

EXTRACTION OF DATA

We have to do two processes, **Web Scraping** and **Data Framing** in order to extract data. First, we have to do web scraping from **Wikipedia** for obtaining information about **Chennai Areas**. Some websites provide datasets in CSV or JSON format, most of them don't offer this option.In this case, we use Web Scraping as it is a process of **extracting data** from webpages. It is also known as **Web Harvesting** or **Web Data Extraction**. When we scrape a web, we write the program that sends a request to the server where we have mentioned which will return in **HTML or XML format**. After fetching the data, data extraction is performed where the information being parsed according to our project.

In order to perform web scraping in python, we will use **requests** and **BeautifulSoup** libraries. Requests lets you send HTTP/1.1 requests and form data and BeautifulSoup is a library which is used for parsing HTML and XML documents.

Initially, we have to download the data with the help of requests library which will help us to create a request to a web server for **downloading the HTML contents**. In order to access the data from the web page, requests.get method is utilized. Below, we have shown the command for utilizing the data from Wikipedia for the area list in Chennai.

```
url="https://en.wikipedia.org/wiki/Areas_of_Chennai"
response = requests.get(url)
```

***Fig 5**: Request command to get information from Wikipedia*

After downloading the document from Wikipedia, we will use **BeautifulSoup** library so that we can parse the document.

```
soup = BeautifulSoup(response.text, 'lxml')
#chennai_table = soup.find('table', class_='wikitable')
#chennai_table
#print(soup.title)
chennai_table=str(soup.table)
display_html(chennai_table,raw=True)
```

***Fig 6**: BeautifulSoup command to parse information from Wikipedia*

Next thing that we have to do is to transform the **parsed data** into **pandas DataFrame** so that we can read the data and distinguish it for plotting its latitude and longitude.

```
dfs = pd.read_html(chennai_table)
df=dfs[0]
df
```

***Fig 7**: Convert the parsed document into pandas DataFrame*

DATA PREPROCESSING

We need to link each area with their latitude, longitude, and altitude. For this, we have utilized a geocoder library. We need to use the **Nominatim Geocoding service**, which is constructed on top of **OpenStreetMap** data. Let's find out the latitude and longitude of Chennai City.

```
from geopy.geocoders import Nominatim
address='Chennai, Tamil Nadu'
geolocator = Nominatim(user_agent="chennai_explorer")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print('The geograpical coordinate of Chennai are {}, {}.'.format(latitude, longitude))
```

***Fig 8**: Finding Chennai City Latitude and Longitude*

Now, we have to complete the whole list of latitude and longitude of Chennai City Areas. So, we will use **RateLimiter service** which will provide the zone of each area with their point of reference. We have to divide the point to latitude, longitude and altitude.

```
from geopy.extra.rate_limiter import RateLimiter
geolocator = Nominatim(user_agent="chennai_explorer")
address='Chennai, Tamil Nadu'
location = geolocator.geocode(address)
geocode=RateLimiter(geolocator.geocode,min_delay_seconds=1)
df['locate']=df['Location'].apply(geocode)
df['point']=df['locate'].apply(lambda loc:tuple(loc.point) if loc else None)
df
```

***Fig 9**: Finding Chennai City Areas Latitude and Longitude*

But, it is necessary to know that few areas missed some values, thus they were listed as **NaN**. So, we have to find out those values from the internet and substitute those values manually.

```
df1['altitude'] = df1['altitude'].replace(np.nan, 0)
df1.loc[(df1.Location == 'Ekkaduthangal'),'latitude']='13.0239'
df1.loc[(df1.Location == 'Ekkaduthangal'),'longitude']='80.2001'

df1.loc[(df1.Location == 'Eranavur'),'latitude']='13.1896'
df1.loc[(df1.Location == 'Eranavur'),'longitude']='80.3039'

df1.loc[(df1.Location == 'ICF'),'latitude']='13.0981'
df1.loc[(df1.Location == 'ICF'),'longitude']='80.2195'

df1.loc[(df1.Location == 'Kattivakkam'),'latitude']='13.2161'
df1.loc[(df1.Location == 'Kattivakkam'),'longitude']='80.3182'
```

***Fig 10**: Manually changing those Latitude and Longitude which are NaN*

Finally, we are showing the geographical map of Chennai with the help of **Folium library**. Folium library is used to show various layouts of maps using latitude and longitude.

```
map_chennai= folium.Map(location=[13.0836939,80.270186], zoom_start=10)

for lat, lng, loc in zip(df1['latitude'],df1['longitude'], df1['Location']):
    label = '{}'.format(loc)
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=5,
        popup=label,
        color='green',
        fill=True,
        fill_color='green',
        fill_opacity=0.7,
        parse_html=False).add_to(map_chennai)

map_chennai
```

Fig 11*: Program to print the map of Chennai with some Area markers*

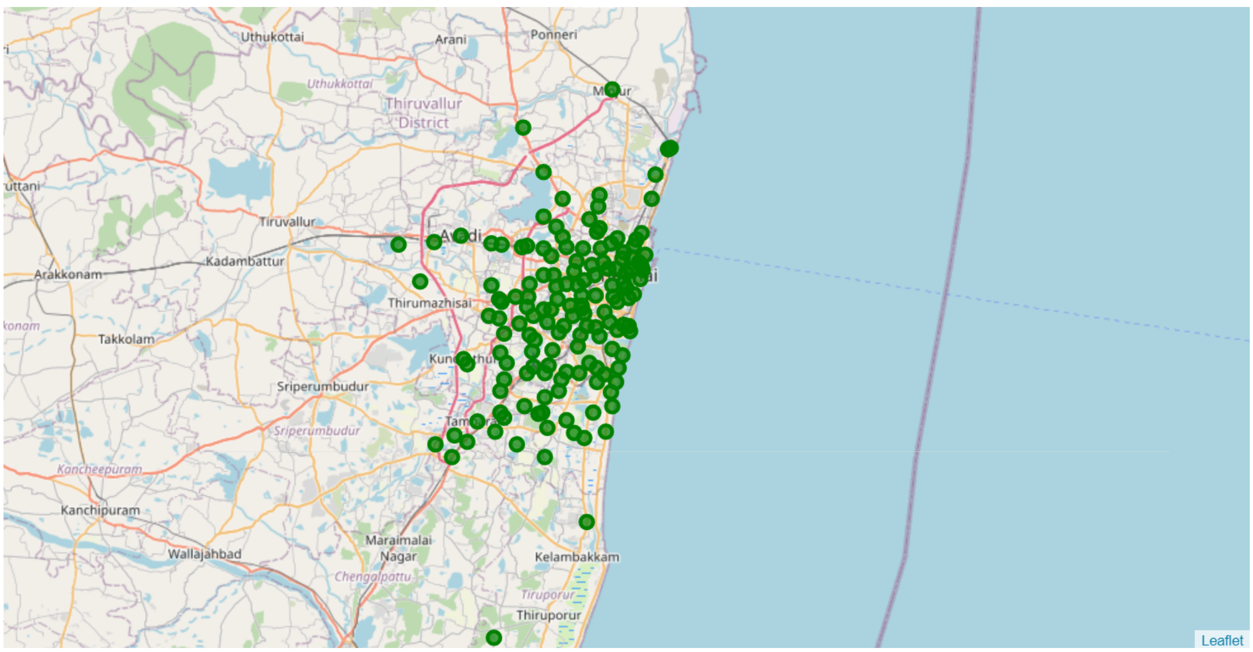

Fig 12*: Map of Chennai with some Area markers*

DATA ANALYSIS

For analysis, we have to utilize **FourSquare API** for retrieving the locations and look for important venues around the location. In order to allow the user's request, we have to specify **Client Key's client ID** and **Client Secret** in the request URL.

```
CLIENT_ID = 'RCI2LJCNCNGYLRIOPU2MSCHC3ZGBSKVPLF5FRG12DA1LNIGL' # your Foursquare ID
CLIENT_SECRET = 'D254LXL03DB0ZGUI3ADD32ELMIX4DBMVYX5Q2V1DH2LCM1Y4' # your Foursquare Secret
ACCESS_TOKEN = 'UXC23PT1T1VQQMHIBYNMKY2YYFO3WJEMCWM1HQ3P5MPK4S3S' # your FourSquare Access Token
VERSION = '20180604'
LIMIT = 100
print('Your credentails:')
print('CLIENT_ID: ' + CLIENT_ID)
print('CLIENT_SECRET:' + CLIENT_SECRET)
```

```
Your credentails:
CLIENT_ID: RCI2LJCNCNGYLRIOPU2MSCHC3ZGBSKVPLF5FRG12DA1LNIGL
CLIENT_SECRET:D254LXL03DB0ZGUI3ADD32ELMIX4DBMVYX5Q2V1DH2LCM1Y4
```

Fig 13*: Defining the FourSquare Credentials and Version*

With the help of FourSquare API, we can find the **top 100 venues** within the radius of **500 meters** with neighborhood latitude and longitude to be the first neighbourhood in Chennai Areas.

```
In [85]: radius = 500
url = 'https://api.foursquare.com/v2/venues/explore?client_id={}&client_secret={}&ll={},{}&v={}&radius={}&limit={}'.format(CLIENT_ID, CLIENT_SECRET, nei_lat, nei_lon, VERSION, radius, LIMIT)
url

Out[85]: 'https://api.foursquare.com/v2/venues/explore?client_id=RCI2LJCNCNGYLRIOPU2MSCHC3ZGBSKVPLF5FRG12DA1LNIGL&client_secret=D254LXL03DB0ZGUI3ADD32ELMIX4DBMVYX5Q2V1DH2LCM1Y4&ll=13.0728321,80.2576906&v=20180604&radius=500&limit=100'
```

Fig 14*: Finding top 100 venues within radius of 500 meters*

Now, we have listed the variety of restaurants which were received from FourSquare API.We have scrutinized the data by analyzing the number of restaurants with the help of one hot encoding. For analyzing the neighborhood, we have used the strategy of one hot encoding to all the venues. So, the number of columns becomes **22**.

```
In [93]: # one hot encoding
chennai_onehot = pd.get_dummies(Chennai_restaurants[['Venue Category']], prefix="", prefix_sep="")

# add neighborhood column back to dataframe
chennai_onehot['Neighbourhood'] = Chennai_restaurants['Neighbourhood']

# move neighborhood column to the first column
fixed_columns = [chennai_onehot.columns[-1]] + list(chennai_onehot.columns[:-1])
chennai_onehot = chennai_onehot[fixed_columns]
chennai_grouped = chennai_onehot.groupby('Neighbourhood').mean().reset_index()
chennai_grouped
```

***Fig 15**: One hot Encoding*

Now, Let’s group the neighbourhood by taking the **mean of frequency** of the number of occurrences that took place in each category.

	Neighbourhood	Afghan Restaurant	Asian Restaurant	Chinese Restaurant	Comfort Food Restaurant	Fast Food Restaurant	Indian Restaurant	Italian Restaurant	Japanese Restaurant	Kebab Restaurant	Korean Restaurant	Mexican Restaurant	Middle Eastern Restaurant
0	Adambakkam	0.000000	0.000000	0.000000	0.0	0.000000	1.000000	0.000000	0.0	0.000000	0.0	0.00	0.000000
1	Adyar	0.000000	0.062500	0.000000	0.0	0.062500	0.562500	0.062500	0.0	0.000000	0.0	0.00	0.000000
2	Alapakkam	0.000000	0.000000	0.000000	0.0	0.500000	0.500000	0.000000	0.0	0.000000	0.0	0.00	0.000000
3	Alwarpet	0.000000	0.000000	0.100000	0.0	0.000000	0.000000	0.100000	0.3	0.000000	0.1	0.00	0.000000
4	Alwarthirunagar	0.000000	0.000000	0.000000	0.0	1.000000	0.000000	0.000000	0.0	0.000000	0.0	0.00	0.000000
5	Aminjikarai	0.000000	0.000000	0.000000	0.0	1.000000	0.000000	0.000000	0.0	0.000000	0.0	0.00	0.000000
6	Anna Nagar	0.000000	0.100000	0.200000	0.0	0.200000	0.300000	0.000000	0.0	0.000000	0.0	0.00	0.100000
7	Arumbakkam	0.000000	0.000000	0.000000	0.0	0.000000	1.000000	0.000000	0.0	0.000000	0.0	0.00	0.000000
8	Ashok Nagar	0.000000	0.000000	0.000000	0.0	0.000000	1.000000	0.000000	0.0	0.000000	0.0	0.00	0.000000
9	Besant Nagar	0.000000	0.000000	0.125000	0.0	0.000000	0.500000	0.125000	0.0	0.000000	0.0	0.00	0.000000

***Fig 16**: Grouping up of categories*

With the help of grouping, we have ranked each neighborhood along with the **top 5** most common kinds of restaurant. For example, the area **Adambakkam** has the highest number of **Indian restaurants** and **Alwarpet** has the most number of **Japanese restaurants**.

```
----Adambakkam----
                             venue  freq
0                Indian Restaurant   1.0
1                Afghan Restaurant   0.0
2  Molecular Gastronomy Restaurant   0.0
3                  Thai Restaurant   0.0
4          South Indian Restaurant   0.0


----Adyar----
                           venue  freq
0              Indian Restaurant  0.56
1  Vegetarian / Vegan Restaurant  0.12
2        North Indian Restaurant  0.12
3           Fast Food Restaurant  0.06
4             Italian Restaurant  0.06
```

***Fig 17**: Most common places in various areas of Chennai*

To cluster these areas with their common restaurants, we need to create a new DataFrame and sort them based on their neighborhood.

```
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))

neighborhoods_venues_sorted = pd.DataFrame(columns=columns)
neighborhoods_venues_sorted['Neighbourhood'] = chennai_grouped['Neighbourhood']


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

neighborhoods_venues_sorted.head()
```

***Fig 18**: Creation of DataFrame based on their ranking*

	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	Adambakkam	Indian Restaurant	Afghan Restaurant	Molecular Gastronomy Restaurant	Thai Restaurant	South Indian Restaurant	Seafood Restaurant	Restaurant	Rajasthani Restaurant	North Indian Restaurant	New American Restaurant
1	Adyar	Indian Restaurant	Vegetarian / Vegan Restaurant	North Indian Restaurant	Fast Food Restaurant	Italian Restaurant	Asian Restaurant	Multicuisine Indian Restaurant	Thai Restaurant	South Indian Restaurant	Seafood Restaurant
2	Alapakkam	Fast Food Restaurant	Indian Restaurant	Afghan Restaurant	Molecular Gastronomy Restaurant	Thai Restaurant	South Indian Restaurant	Seafood Restaurant	Restaurant	Rajasthani Restaurant	North Indian Restaurant
3	Alwarpet	Japanese Restaurant	Restaurant	Chinese Restaurant	Thai Restaurant	Italian Restaurant	Korean Restaurant	Afghan Restaurant	Multicuisine Indian Restaurant	South Indian Restaurant	Seafood Restaurant
4	Alwarthirunagar	Fast Food Restaurant	Afghan Restaurant	Molecular Gastronomy Restaurant	Thai Restaurant	South Indian Restaurant	Seafood Restaurant	Restaurant	Rajasthani Restaurant	North Indian Restaurant	New American Restaurant

***Fig 19**: List of Common Venues*

DATA CLUSTERING

We have to cluster all the neighborhoods with different clusters which help us to understand which neighborhoods have **higher concentrations of venues** with a lesser number of venues. With the help of the highest number of venues will find out which category of restaurant to start in Chennai. For this, we are going to utilize **K-Means Clustering**. K-Means performs **division of objects** into clusters which are **similar** between them and **dissimilar** to the objects belonging to another cluster. Given an **input K** with a set of points, place the **centroids** at random locations and repeat until we find **convergence**. In order to provide the perfect fit, it is necessary to find the **best K value** so that we can cluster the neighborhoods. For this, we have used **Silhouette analysis** which shows how close each point is in the cluster.

```
from matplotlib import pyplot as plt
cost=[]
scores=[]
for i in range(2, 20):
    KM = KMeans(n_clusters = i, random_state=0).fit_predict(chennai_clustering_testing)
    score = silhouette_score(chennai_clustering_testing, KM)
    cost.append(KM)
    scores.append(score)

plt.figure(figsize=(20,10))
plt.plot(range(2,20),scores,'o-',color='g')
plt.xlabel("Value of K")
plt.ylabel("Score")
plt.grid(True)
plt.show()
```

***Fig 19**: Finding the best K value*

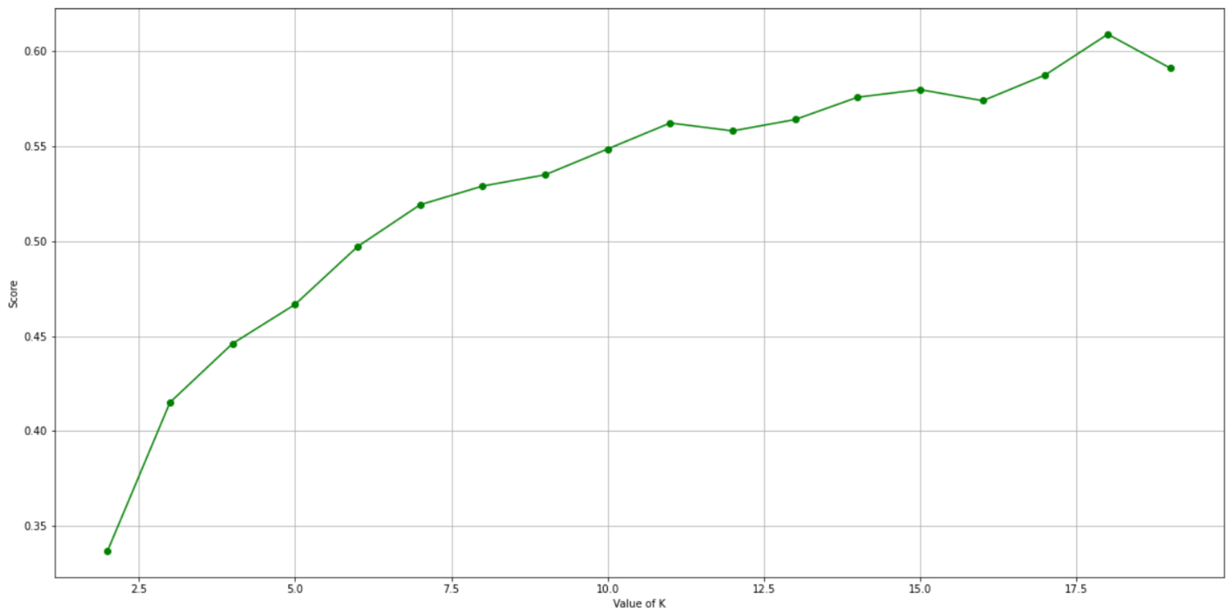

***Fig 19**: Graphical representation of K value vs Score*

From the optimization method, we could find out that the best value of K is said to be **17**. We have used the **argmax function** to find out the best optimal value.

```
In [100]: kclusters = opt

          chennai_clustering_testing = chennai_grouped.drop('Neighbourhood', 1)
          # run k-means clustering
          kmeans = KMeans(n_clusters=kclusters, random_state=0).fit(chennai_clustering_testing)

          # check cluster labels generated for each row in the dataframe
          kmeans.labels_

Out[100]: array([ 4,  2, 15,  8,  1,  1,  8,  4,  4,  2,  4,  4,  1,  2, 11,  8, 12,
                  4,  8, 15,  4,  4, 14,  4,  4, 15,  7,  1, 15,  4,  1,  2,  4,  3,
                 10,  5,  0,  4, 10,  2,  2,  0,  4,  5,  3,  7,  4, 15,  3,  4,  4,
                  4,  9, 11,  1,  2,  2,  3,  4,  6,  2,  5,  1,  4, 13, 15,  4, 15,
                  7,  5,  9,  1,  9,  4,  4,  4,  4,  1,  2,  3,  8,  0,  4,  2, 15,
                  8], dtype=int32)
```

***Fig 20**: K-Means Clustering*

In [101]:
```
neighborhoods_venues_sorted.insert(0, 'Cluster Labels', kmeans.labels_)
Chennai_merged = Chennai_restaurants
```

In [102]:
```
Chennai_merged = Chennai_merged.join(neighborhoods_venues_sorted.set_index('Neighbourhood'), on='Neighbourhood')
Chennai_merged.fillna(0)
Chennai_merged.head()
```

Out[102]:

	Neighbourhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Co
4	Adambakkam	12.982221	80.209121	Bistro	12.983193	80.205020	Indian Restaurant	4	Indian Restaurant	Afghan Restaurant	Molecular Gastronomy Restaurant	Thai Rest
5	Adyar	13.006450	80.257779	Bombay Brassiere	13.006961	80.256419	North Indian Restaurant	2	Indian Restaurant	Vegetarian / Vegan Restaurant	North Indian Restaurant	Fast Rest
6	Adyar	13.006450	80.257779	Adyar Ananda Bhavan	13.005824	80.257368	Indian Restaurant	2	Indian Restaurant	Vegetarian / Vegan Restaurant	North Indian Restaurant	Fast Rest
8	Adyar	13.006450	80.257779	Prems Graama Bhojanam	13.006345	80.253995	Vegetarian / Vegan Restaurant	2	Indian Restaurant	Vegetarian / Vegan Restaurant	North Indian Restaurant	Fast Rest

***Fig 21**: Merging Cluster Labels to the Neighbourhood*

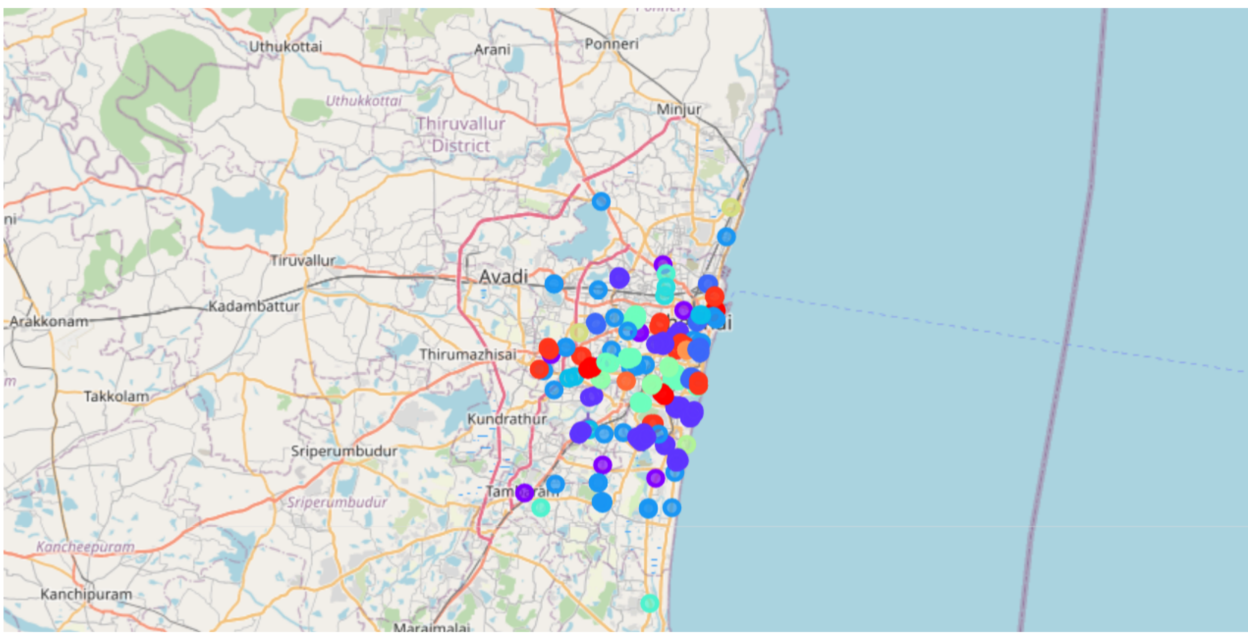

***Fig 21**: Visualizing the resulting clusters*

RESULT AND DISCUSSION:

From our analogy, we have found out the most and least common restaurant in Chennai based on the clusters.

Clusters	***Most Common Restaurant***	***2nd Common Restaurant***	***3rd Common Restaurant***	***Least Common Restaurant***
0	South Indian Restaurant	Vegetarian / Vegan Restaurant	Asian Restaurant	North Indian Restaurant
1	Fast Food Restaurant	Afghan Restaurant	Molecular Gastronomy Restaurant	New American Restaurant
2	Indian Restaurant	Vegetarian / Vegan Restaurant	North Indian Restaurant	Seafood Restaurant
3	Vegetarian / Vegan Restaurant	Asian Restaurant	Thai Restaurant / Molecular Gastronomy Restaurant	New American Restaurant and Multicuisine Indian Restaurant
4	Indian Restaurant	Afghan Restaurant	Molecular Gastronomy Restaurant	New American Restaurant
5	Indian Restaurant and Restaurant	Afghan Restaurant	Thai Restaurant	Multicuisine Indian Restaurant
6	Vegetarian / Vegan Restaurant	Chinese Restaurant	Molecular Gastronomy Restaurant	New American Restaurant
7	Asian Restaurant	Afghan Restaurant	Thai Restaurant	Multicuisine Indian Restaurant
8	Japanese Restaurant and Middle Eastern Restaurant	Chinese Restaurant and Indian Restaurant	Fast Food Restaurant	Korean Restaurant
9	Chinese Restaurant	Indian Restaurant	Italian Restaurant	Rajasthani Restaurant and North Indian

				Restaurant
10	Italian Restaurant	Japanese Restaurant	Afghan Restaurant	North Indian Restaurant
11	Chinese Restaurant	Afghan Restaurant	Molecular Gastronomy Restaurant	New American Restaurant
12	Afghan Restaurant	Asian Restaurant	Thai Restaurant	Multicuisine Indian Restaurant
13	Middle Eastern Restaurant	Rajasthani Restaurant	Asian Restaurant	Multicuisine Indian Restaurant
14	Vegetarian / Vegan Restaurant	Multicuisine Indian Restaurant	Asian Restaurant	New American Restaurant
15	Fast Food Restaurant	Indian Restaurant	Afghan Restaurant	North Indian Restaurant
16	Vegetarian / Vegan Restaurant	Multicuisine Indian Restaurant	American Restaurant	New American Restaurant

Table 2: *List of Clusters with their list of restaurants*

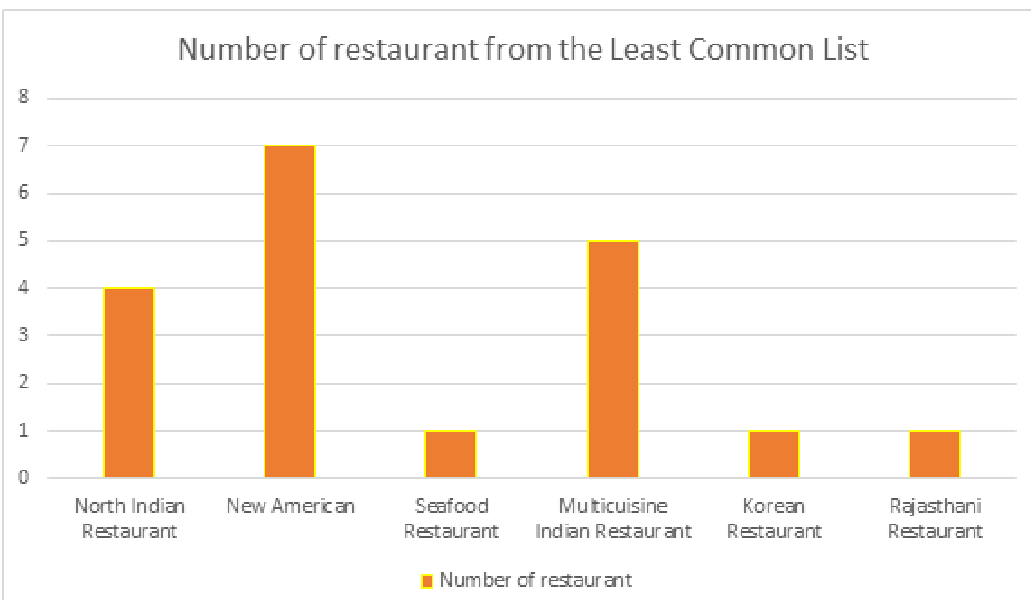

Fig 22: *Visualizing the least number of restaurant*

We can construct the analogy in three ways:

- If we want to start a restaurant by not keeping cuisine in mind, it is better to go for the **least common restaurant** so that it would be different to the customers and also provide a new trend in that area. In this case, We can find out that most of the clusters have **New American Restaurant** to be least, so it can be a trendsetter in a few areas like **Alwarpet** and **Adyar**.
- If we want to start a restaurant by keeping cuisine in mind, then it is better to go for that area which has the **least common restaurant** based on that cuisine so that it would be a change to the customers. For example, if you are **franchise** and looking for which area to start the restaurant, it is better to look over those areas which are least common for that particular cuisine.
- If we want to start a restaurant by keeping cuisine in mind, then they can find out the areas which have **second and third common restaurants** based on that restaurant because there can be cuisine based restaurants which might not be successful, so this can be a **game changer**.

CONCLUSION:

In this project, we have used a **list of areas of Chennai** and its neighborhoods with a variety of restaurants in that neighborhood. For this, we have used **K-Means Clustering** where the neighborhoods are categorized with respect to **top 10 common spots**. Investors can have an idea of what kind of restaurant to be started by analyzing the clustering results. Even though we did not have much information, our future work is to have more features on that type of restaurant in order to have a more optimal model.

REFERENCES:

1. https://en.wikipedia.org/wiki/Areas_of_Chennai
2. https://en.wikipedia.org/wiki/Chennai
3. https://developer.foursquare.com/
4. https://www.britannica.com/place/Chennai
5. https://en.wikipedia.org/wiki/K-means_clustering