Coursera Capstone Project : Applied Data Science

Jiajun Zhou
zhouthomas177@gmail.com

1 Introduction

Nowadays, times are exceptionally difficult for our city's restaurants right now, with the spread of the coronavirus, and the unfortunate but necessary social distancing requirements, currently providing many obstacles for Hong Kong's stellar eateries. In light of the current situation, I think that recognition and support of our restaurant scene is particularly important right now. With nearly 7 million inhabitants and the highest metropolitan GDP in the world, an estimated 60 million overseas or mainland visitors are expected to flock to the Hong Kong.

Hong Kong is well-known as one of restaurant capitals of the world. Hong Kong has hit the record high of having 15,000 licensed restaurants, cafes and bars by year end of 2017, with 20.4 restaurants per 10,000 people. It is one of the highest density city of restaurants in the world. Hong Kong is the first batch of Asia's cities to be included in the Michelin's Guide Hong Kong.

I believe it's difficult for a travelers, especially restaurant-goers, to make a choice from among many options since there is also too much information on the web because everybody's got their own take of where to go and it's all so fragmented that you have to assemble it yourself especially if you're wanting non-touristy recommendations.

2 Business Problem

I have done that by updating the list districts around the city that have been doing great things, even if current events have led to a temporary halt in operations. Read on, then as we present our pick of Hong Kong's 10 most common restaurants and get some inspiration for where your next meal could be. Thus, the main objective of the project will be to find ideal spots in different districts of the city. The Foursquare API will used to get location data and clustering methods to group their restaurant venues

information.

3 Data

The data for this project has been retrieved and processed through multiple sources, giving careful considerations to the accuracy of the methods used. For this project we need following data:

- **Hongkong data that contains list districts (Wards) along with their latitude and longitude.**
  Datasource : https://en.wikipedia.org/wiki/Districts_of_Hong_Kong

  Description: We will Scrap HK districts (Wards) Table from Wikipedia and get the coordinates of these major districts using geocoder class of Geopy client.

- **Restaurants in each neighborhood of Hong Kong:**
  Data source: Foursquare APIs
  Description : By using this API we will get all the venues in each neighborhood. We can filter these venues to get only restaurants.

3.1 Neighborhoods

The data of the neighborhoods in HK can be extracted out by web scraping using Pandas for Python. The neighborhood data is scraped from a Wikipedia webpage.

1. Use pandas to tranform the wiki's data into a dataframe.

```
[49]: df = pd.read_html('https://en.wikipedia.org/wiki/Districts_of_Hong_Kong')[5]

[50]: df
```

[50]:

	District	Chinese	Population[when?] [6]	Area(km²)	Density(/km²)	Region
0	Central and Western	中西區	244600	12.44	19983.92	Hong Kong Island
1	Eastern	東區	574500	18.56	31217.67	Hong Kong Island
2	Southern	南區	269200	38.85	6962.68	Hong Kong Island
3	Wan Chai	灣仔區	150900	9.83	15300.10	Hong Kong Island
4	Sham Shui Po	深水埗區	390600	9.35	41529.41	Kowloon
5	Kowloon City	九龍城區	405400	10.02	40194.70	Kowloon
6	Kwun Tong	觀塘區	641100	11.27	56779.05	Kowloon
7	Wong Tai Sin	黃大仙區	426200	9.30	45645.16	Kowloon
8	Yau Tsim Mong	油尖旺區	318100	6.99	44864.09	Kowloon
9	Islands	離島區	146900	175.12	825.14	New Territories
10	Kwai Tsing	葵青區	507100	23.34	21503.86	New Territories
11	North	北區	310800	136.61	2220.19	New Territories
12	Sai Kung	西貢區	448600	129.65	3460.08	New Territories
13	Sha Tin	沙田區	648200	68.71	9433.85	New Territories
14	Tai Po	大埔區	307100	136.15	2220.35	New Territories
15	Tsuen Wan	荃灣區	303600	61.71	4887.38	New Territories
16	Tuen Mun	屯門區	495900	82.89	5889.38	New Territories
17	Yuen Long	元朗區	607200	138.46	4297.99	New Territories

After pre-processing, the data frame is obtained like this:

	District	Chinese	Population	Area	Density	Region
0	Central and Western	中西區	244600	12.44	19983.92	Hong Kong Island
1	Eastern	東區	574500	18.56	31217.67	Hong Kong Island
2	Southern	南區	269200	38.85	6962.68	Hong Kong Island
3	Wan Chai	灣仔區	150900	9.83	15300.10	Hong Kong Island
4	Sham Shui Po	深水埗區	390600	9.35	41529.41	Kowloon
5	Kowloon City	九龍城區	405400	10.02	40194.70	Kowloon
6	Kwun Tong	觀塘區	641100	11.27	56779.05	Kowloon
7	Wong Tai Sin	黃大仙區	426200	9.30	45645.16	Kowloon
8	Yau Tsim Mong	油尖旺區	318100	6.99	44864.09	Kowloon
9	Islands	離島區	146900	175.12	825.14	New Territories
10	Kwai Tsing	葵青區	507100	23.34	21503.86	New Territories
11	North	北區	310800	136.61	2220.19	New Territories
12	Sai Kung	西貢區	448600	129.65	3460.08	New Territories
13	Sha Tin	沙田區	648200	68.71	9433.85	New Territories
14	Tai Po	大埔區	307100	136.15	2220.35	New Territories
15	Tsuen Wan	荃灣區	303600	61.71	4887.38	New Territories
16	Tuen Mun	屯門區	495900	82.89	5889.38	New Territories
17	Yuen Long	元朗區	607200	138.46	4297.99	New Territories

3.2 Geocoding

The latitude and longitude of the neighborhoods are retrieved using Google Maps Geocoding API. The geometric location values are then stored into the initial data frame.

```
from geopy.geocoders import Nominatim # module to convert an address into latitude and longitude values
geolocator = Nominatim(user_agent="HK_explorer")

df['Major_Dist_Coord']= df['Chinese'].apply(geolocator.geocode).apply(lambda x: (x.latitude, x.longitude))
df[['Latitude', 'Longitude']] = df['Major_Dist_Coord'].apply(pd.Series)

df.drop(['Major_Dist_Coord'], axis=1, inplace=True)
df
```

	District	Chinese	Population	Area	Density	Region	Latitude	Longitude
0	Central and Western	中西區	244600	12.44	19983.92	Hong Kong Island	22.274848	114.148725
1	Eastern	東區	574500	18.56	31217.67	Hong Kong Island	22.273078	114.233594
2	Southern	南區	269200	38.85	6962.68	Hong Kong Island	22.219263	114.225230
3	Wan Chai	灣仔區	150900	9.83	15300.10	Hong Kong Island	22.273947	114.181749
4	Sham Shui Po	深水埗區	390600	9.35	41529.41	Kowloon	22.331254	114.159321
5	Kowloon City	九龍城區	405400	10.02	40194.70	Kowloon	22.321800	114.188594
6	Kwun Tong	觀塘區	641100	11.27	56779.05	Kowloon	22.308649	114.227661
7	Wong Tai Sin	黃大仙區	426200	9.30	45645.16	Kowloon	22.344322	114.202150
8	Yau Tsim Mong	油尖旺區	318100	6.99	44864.09	Kowloon	22.307404	114.165526
9	Islands	離島區	146900	175.12	825.14	New Territories	35.736156	139.714222
10	Kwai Tsing	葵青區	507100	23.34	21503.86	New Territories	22.341007	114.104285
11	North	北區	310800	136.61	2220.19	New Territories	35.755838	139.736687
12	Sai Kung	西貢區	448600	129.65	3460.08	New Territories	22.307010	114.371345
13	Sha Tin	沙田區	648200	68.71	9433.85	New Territories	22.391573	114.208098
14	Tai Po	大埔區	307100	136.15	2220.35	New Territories	22.480971	114.304103
15	Tsuen Wan	荃灣區	303600	61.71	4887.38	New Territories	22.364987	114.077688
16	Tuen Mun	屯門區	495900	82.89	5889.38	New Territories	22.378840	113.952830
17	Yuen Long	元朗區	607200	138.46	4297.99	New Territories	22.457296	114.021319

3.3 Venue Data

From the location data obtained after Web Scraping and Geocoding, the venue data is found out by passing in the required parameters to the FourSquare API, and creating another Data Frame to contain all the venue details along with the respective neighborhoods.

```
results = requests.get(url).json()

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']
```

```
venues = results['response']['groups'][0]['items']
nearby_venues = json_normalize(venues) # flatten JSON

# filter columns
filtered_columns = ['venue.name', 'venue.categories', 'venue.location.lat', 'venu
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.head()
```

```
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages/ipykernel_launche
r.py:3: FutureWarning: pandas.io.json.json_normalize is deprecated, use pandas.js
on_normalize instead
  This is separate from the ipykernel package so we can avoid doing imports until
```

	name	categories	lat	lng
0	Victoria Peak (太平山)	Scenic Lookout	22.271280	114.149976
1	Morning Trail, The Peak (山頂晨運徑)	Trail	22.278008	114.144432
2	Victoria Peak Garden (山頂公園)	Garden	22.273937	114.143373
3	Hong Kong Trail (Section 1) (港島徑（第一段）)	Trail	22.272874	114.145895
4	New Punjab Club	Pakistani Restaurant	22.280250	114.155475

4 Methodology

A thorough analysis of the principles of methods have been made in order to ensure the inferences to be made are as accurate as possible. The exploratory data analysis(EDA) is used to uncover hidden properties of data.

4.1 Accuracy of the Foursquare API

In the initial development phase with Foursquqre API, the number of erroneous results were of an appreciable amount, which led to the development of an algorithm to analyze the top 100 venues that in Central and Western. We found that 50 unique venue categories and Japanese restaurants in the top of the list.

```
print('{} venues were returned by Foursquare.'.format(nearby_venues.shape[0]))
```

```
100 venues were returned by Foursquare.
```

```
print ('{} unique categories in Central'.format(nearby_venues['categories'].value
```

```
60 unique categories in Central
```

```
print (nearby_venues['categories'].value_counts()[0:10])
```

```
Japanese Restaurant     7
Hotel                   6
French Restaurant       5
Café                    5
Coffee Shop             4
Italian Restaurant      4
Gym / Fitness Center    3
Park                    3
Trail                   2
Cocktail Bar            2
Name: categories, dtype: int64
```

Then, I will analysis the restaurant category only in all 18 districts in the following, and find out top 10 venues categories in the chart plot.

```
hk_venues = getNearbyVenues(names=df['District'],
                            latitudes=df['Latitude'],
                            longitudes=df['Longitude']
                           )
```

```
Central and Western
Eastern
Southern
Wan Chai
Sham Shui Po
Kowloon City
Kwun Tong
Wong Tai Sin
Yau Tsim Mong
Islands
Kwai Tsing
North
Sai Kung
Sha Tin
Tai Po
Tsuen Wan
Tuen Mun
Yuen Long
```

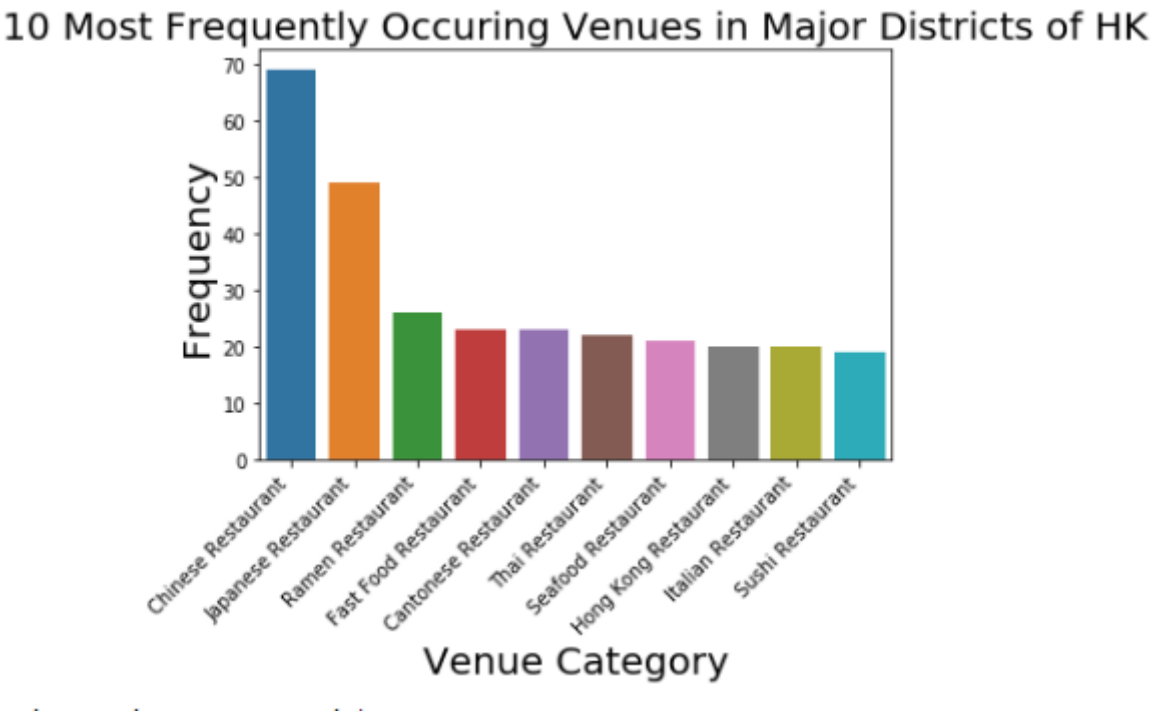


Figure 1: Venue category of HK's restaurants.

Chinese restaurants are the most common venue in HK. However, is it Chinese restaurant is the most common in every district? Let's go back to exploring the data a little more.

4.2 Folium

Folium builds on the data wrangling strengths of the Python ecosystem and the mapping strengths of the leaflet.js library. All cluster visualization are done with help of Folium which in turn generates a Leaflet map made using OpenStreetMap technology. I utilized the folium library to visualize geographic details of Hongkong and its latitude and longitude of 18 major districts.

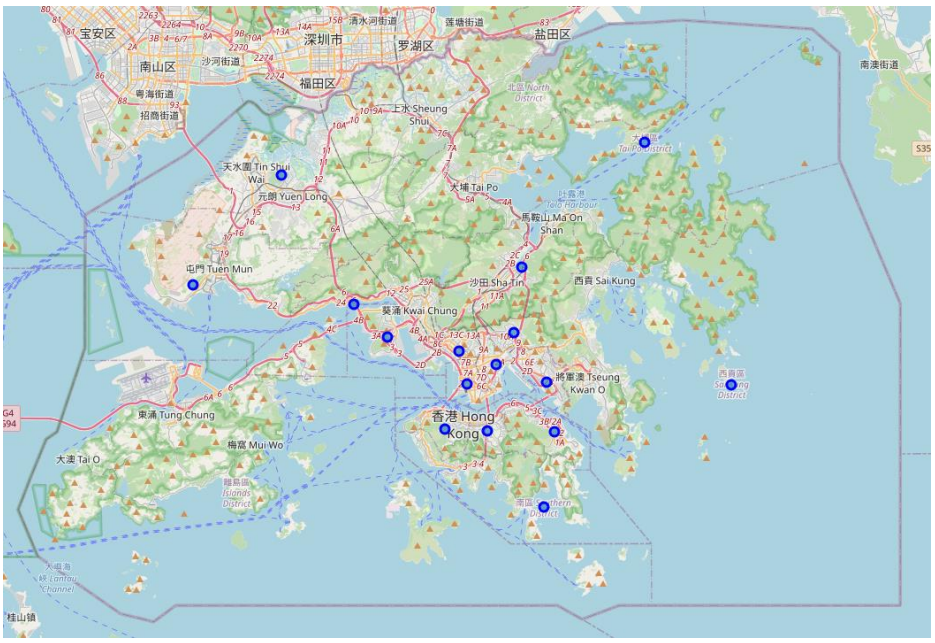


Figure 2: Neighbourhoods of HK.

4.3 One hot encoding

One hot encoding is a process by which categorical variables are converted into a form that could be provided to ML algorithms to do a better job in prediction. For the K-means Clustering Algorithm, all unique items under Venue Category are one-hot encoded.

7. One hot encoding to predict all items.

```
# one hot encoding
HK_onehot = pd.get_dummies(HK_Venues_only_restaurant[['Venue Category']], prefix=

# add neighborhood column back to dataframe
HK_onehot['Neighborhood'] = HK_Venues_only_restaurant['Neighborhood']


HK_onehot.head()
```

Secondly, use pandas groupby on the neighborhood column and obtain the mean of the frequency of occurrence of every category.

8. Group rows by neighborhood and taking the mean of frequency of each category.

```
HK_grouped = HK_onehot.groupby('Neighborhood').mean().reset_index()
HK_grouped
```

	Neighborhood	American Restaurant	Argentinian Restaurant	Asian Restaurant	Australian Restaurant	Cantonese Restaurant	Chinese Restaurant	Re
0	Central and Western	0.000000	0.029412	0.029412	0.000000	0.029412	0.029412	
1	Eastern	0.000000	0.000000	0.000000	0.000000	0.000000	0.222222	
2	Islands	0.000000	0.000000	0.025641	0.000000	0.000000	0.076923	
3	Kowloon City	0.000000	0.000000	0.045455	0.000000	0.090909	0.090909	
4	Kwai Tsing	0.000000	0.000000	0.029412	0.029412	0.088235	0.176471	
5	Kwun Tong	0.000000	0.000000	0.035714	0.000000	0.035714	0.107143	
6	North	0.000000	0.000000	0.047619	0.000000	0.000000	0.095238	
7	Sai Kung	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
8	Sha Tin	0.000000	0.000000	0.033333	0.000000	0.166667	0.300000	
9	Sham Shui Po	0.000000	0.000000	0.000000	0.000000	0.000000	0.120000	
10	Southern	0.043478	0.000000	0.043478	0.043478	0.000000	0.130435	
11	Tsuen Wan	0.000000	0.000000	0.000000	0.000000	0.071429	0.321429	
12	Tuen Mun	0.000000	0.000000	0.000000	0.000000	0.107143	0.000000	
13	Wan Chai	0.000000	0.000000	0.035714	0.000000	0.035714	0.107143	
14	Wong Tai Sin	0.000000	0.000000	0.033333	0.000000	0.133333	0.100000	
15	Yau Tsim Mong	0.000000	0.000000	0.000000	0.000000	0.000000	0.083333	
16	Yuen Long	0.021739	0.000000	0.000000	0.000000	0.021739	0.260870	

4.4 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.

9. Top 10 common venues are selected and used to tranin the K-means Clustering Algorithm.

```
num_top_venues = 10

for hood in HK_grouped['Neighborhood']:
    print("----"+hood+"----")
    temp = HK_grouped[HK_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
    print('\n')
```

```
----Central and Western----
                          venue  freq
0            Japanese Restaurant  0.21
1              French Restaurant  0.15
2             Italian Restaurant  0.12
3  Vegetarian / Vegan Restaurant  0.06
4                Thai Restaurant  0.06
5             Mexican Restaurant  0.03
6           Hong Kong Restaurant  0.03
7        Scandinavian Restaurant  0.03
8           Pakistani Restaurant  0.03
9         Argentinian Restaurant  0.03


----Eastern----
                  venue  freq
0   Japanese Restaurant  0.22
1    Chinese Restaurant  0.22
2    Seafood Restaurant  0.11
3      Sushi Restaurant  0.07
4  Taiwanese Restaurant  0.07
5     Indian Restaurant  0.04
6  Hong Kong Restaurant  0.04
7       Thai Restaurant  0.04
8     Hotpot Restaurant  0.04
9   Dumpling Restaurant  0.04
```

4.5 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). The silhouette ranges from -1 to +1, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters. Based on the Silhouette Score of various clusters below 20, the optimal cluster size is determined

10. Find optimal number to clustering data.

[227]:
```
def plot(x, y, xlable, ylabel):
    plt.figure(figsize=(20,10))
    plt.plot(np.arange(3, x), y, 'o-')
    plt.xlabel(xlabel)
    plt.ylabel(ylabel)
    plt.xticks(np.arrge(3, x))
    plt.show()
```

[177]:
```
max_range = 17
```

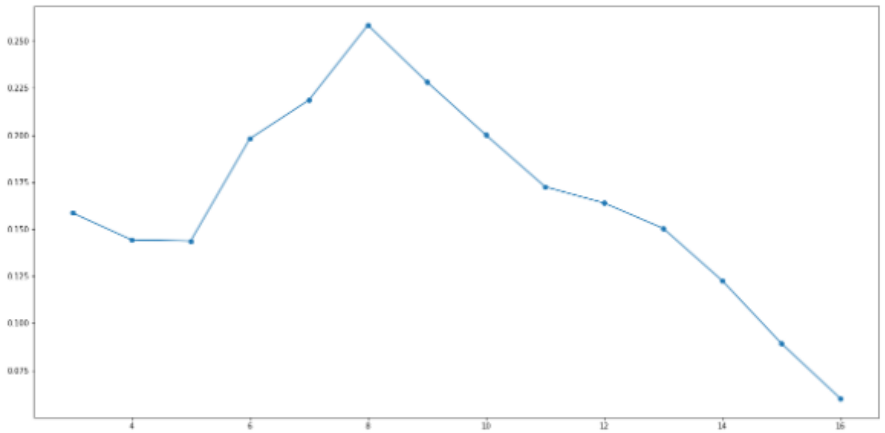
[228]:
```
from sklearn.metrics import silhouette_samples, silhouette_score

indices = []
scores = []
hk_grouped_clustering = HK_grouped.drop('Neighborhood', 1)
for kclusters in range(3, max_range):
    kgc = hk_grouped_clustering
    kmeans = KMeans(n_clusters = kclusters, init = 'k-means++',
                    random_state=0).fit_predict(kgc)

    score = silhouette_score(kgc, kmeans)

    indices.append(kclusters)
    scores.append(score)
```

[229]:
```
plot(max_range, scores, "No. of clusters", "Silhouette Score")
```



[230]:
```
optimal_value = np.argmax(scores) + 2
optimal_value
```

[230]: 7

4.6 K-means clustering

The venue data is then trained using K-means Clustering Algorithm to get the desired clusters to base the analysis on. K-means was chosen as the variables (Venue Categories) are huge, and in such situations K-means will be computationally faster than other clustering algorithms.

```
# set number of clusters
kclusters = optimal_value
#neighborhoods_venues_sorted.insert(0, 'Cluster Labels', kmeans.labels_)
hk_grouped_clustering = HK_grouped.drop('Neighborhood', 1)

# run k-means clustering
kmeans = KMeans(n_clusters=kclusters, random_state=0).fit(hk_grouped_clustering)
neighborhoods_venues_sorted.insert(0, 'Cluster Labels', kmeans.labels_)
# check cluster labels generated for each row in the dataframe
kmeans.labels_[0:10]
```

5 Results

The neighborhoods are divided into 7 clusters by using the optimal approach. The clustered neighborhoods are visualized using different colors so as to make them distinguishable.

```
# create map
map_restaurants10 = folium.Map(location=[latitude,longitude], tiles='cartodbposi
                               attr="<a href=https://github.com/python-visualiza

# 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]

for lat, lon, poi, cluster in zip(hk_merged['Latitude'],
                                  hk_merged['Longitude'],
                                  hk_merged['Neighborhood'],
                                  hk_merged['Cluster Labels'].astype(int)):
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    folium.CircleMarker(
        [lat, lon],
        radius=list_rest_no[list_dist.index(poi)]*0.5,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7).add_to(map_restaurants10)

map_restaurants10
```

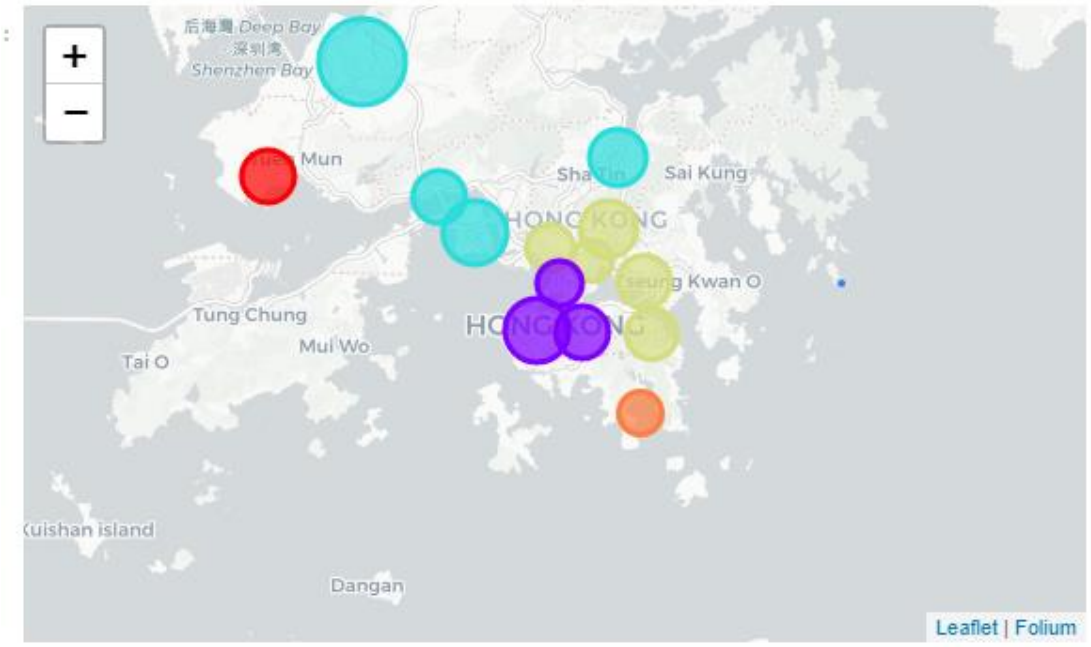


Figure 3: Neighborhoods of Hong Kong (Clustered).

6 Discussion

After analyzing the various clusters produced by the Machine learning algorithm, cluster no.7, is a prime fit to solving the problem of finding a cluster with common venue as a train station mentioned before.

Cluster 2

```
hk_merged.loc[hk_merged['Cluster Labels'] == 1, hk_merged.columns[[1] + list(range(5, hk_merged.shape[1]))]]
```

	Chinese	Region	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	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0	中西區	Hong Kong Island	22.274848	114.148725	1.0	Japanese Restaurant	French Restaurant	Italian Restaurant	Vegetarian / Vegan Restaurant	Thai Restaurant	Dim Sum Restaurant	Pakistani Restaurant	Hong Kong Restaurant	Restaurant	Scandinavian Restaurant
3	灣仔區	Hong Kong Island	22.273947	114.181749	1.0	Italian Restaurant	Dumpling Restaurant	Thai Restaurant	Chinese Restaurant	Japanese Restaurant	Greek Restaurant	Middle Eastern Restaurant	Seafood Restaurant	French Restaurant	Shanghai Restaurant
8	油尖旺區	Kowloon	22.307404	114.165526	1.0	Japanese Restaurant	Dumpling Restaurant	Italian Restaurant	Thai Restaurant	Chinese Restaurant	Pakistani Restaurant	Ramen Restaurant	Restaurant	Scandinavian Restaurant	Shaanxi Restaurant

Figure 4: the most common venue of cluster 2

The three places namely central and western, Wan chai and Yau Tsim Mong in the core area of the city of HK, hence the demographic of the population in these areas group in the same cluster. It is reasonable. Let's summarize our findings:

- Chinese restaurants top the charts of most common venues in the 4 districts namely Sha Tin, Yuen Long, Tsuen Wan and Kwai Tsing. Western food are popular in the 3 districts of cluster 2
- Hong Kong islands and Yuen Long has maximum number of restaurants.
- Since the clustering was based only on the category of restaurants on each district, HK's central 3 wards all fall in the same cluster, which indicate that each of those districts presents a similar experience to the traveler in terms of category of food.
- Sai Kung has the least number of restaurants.

The clustering is completely based on the most common venues obtained from Foursquare data. However, in our analysis, we have ignored other factors like distance of the venues from closest transportation stations, range of price, Michelin Restaurants and so on. Such kind of data and it would be difficult to farm it for a small exploratory study. Hence, our analysis only helps visitors to get an overview of Restaurants distribution by categories in the 18 major districts of HK.

HK$ million

Period		Restaurant receipts by type of restaurant					**Total restaurant receipts**	**Total restaurant purchases**
		Chinese restaurants	Non-Chinese restaurants	Fast food shops	Bars	Miscellaneous eating and drinking places		
2019	Q1	13,385	9,571	5,902	398	2,229	**31,485**	**9,937**
	Q2*	11,977	8,368	5,786	451	2,089	**28,670**	**9,304**
2019	Jan	4,684	3,263	2,017	134	769	**10,867**	**3,503**
	Feb	4,389	2,987	1,848	119	686	**10,029**	**3,104**
	Mar	4,312	3,321	2,037	145	775	**10,589**	**3,330**
	Apr*	3,831	2,765	1,879	157	700	**9,332**	**3,045**
	May*	4,112	2,832	1,985	151	695	**9,774**	**3,186**
	Jun*	4,034	2,771	1,921	143	695	**9,565**	**3,072**

Figure 5: Value of restaurant receipts and purchases

7 Conclusion

As the city will grow at a rapid rate in the next upcoming years, opening food outlets catered for that section of the society will see a massive increase, which would lead to a further increase in business. Like this project, data was used to cluster neighborhood in HK. The most common food venues in 18 major districts can be obtained. The results help visitors to decide which district that fit the most his food needs.

If the food outlets have an average rate of US $0.5 equivalent to 15 percent of the per capita household expenditure, for their items, then profits can be expected to be high as the food rates are neither too low or too high for a person of the concerned demographic to spend. It will guide the officer and businessmen to develop different kind of restaurants in the right district in the future.