Table of Contents

1.Introduction

Ho Chi Minh City, formerly known as Saigon, once “The Pearl of the Far East” is the second largest city in Vietnam (the largest being Hanoi). It is the most crowded city in the country with the official population of over 8 million people on the total area of over 2,095 km2. The city now comprises 19 districts and 5 suburban districts. District 1, along the Saigon River, where downtown Saigon is located, is the commercial center and contains most of the city’s monuments and landmarks. 8 km away in district 5 resides a big market or Cho Lon where Chinese community lives. With favorable weather the whole year round and the convenient access from other countries by air, by road and by sea, Ho Chi Minh city is a busy and dynamic metropolitan.

Brief information about both cities:

District 1 (Vietnamese: Quận 1) is the central urban district of Ho Chi Minh City, the largest city in Vietnam. With a total area of 7.7211 km2 (2.9811 sq mi) the district has a population of 204,899 people as of 2010.The district is divided into 10 small subsets which are called wards (phường). District 1 contains most of the city's administrative offices, consulates, and large buildings. District 1 is the busiest district in the city with the highest living standards. Đồng Khởi street and Nguyễn Huệ boulevard in District 1 are the city's two main commercial centers. Đồng Khởi street is an area in high demand for real estate, hitting a record price of USD50,000 per square meter in 2007. (source: https://en.wikipedia.org/wiki/District_1,_Ho_Chi_Minh_City (https://en.wikipedia.org/wiki/District_1,_Ho_Chi_Minh_City))

District 3 (Vietnamese: Quận 3) is an urban district of Ho Chi Minh City, the largest city in Vietnam. Together with District 1, District 3 is considered the bustling heart of the city, with several businesses, religious sites, historical buildings and tourist attractions. (source: https://en.wikipedia.org/wiki/District_3,_Ho_Chi_Minh_City (https://en.wikipedia.org/wiki/District_3,_Ho_Chi_Minh_City))

2.Objective

In this project, I will study in details the area classification using Foursquare data and machine learning segmentation and clustering. The aim of this project is to segment areas of District 1 and District 3 based on the most common places captured from Foursquare.

I will determine the similarity or dissimilarirty of both district and classification residential/tourism places/others of area located inside District 1 and District 3 by using segmentation and clustering

3. Data

Using the data get from common government website and then translate into English and restructure to csv file for easier manipulation and reading. I uploaded that file to my github for references. Link to the files are:

https://github.com/vhhlinh/Study_data/blob/master/HCMC_District.csv (https://github.com/vhhlinh/Study_data/blob/master/HCMC_District.csv)

Another aspect to consider for this project is the Foursquare data. I believe that the data as good as provided, meaning although we are using Foursquare data for segmentation and clustering, the amount and accuracy of

Reference of data

In [1]:

```
import types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It incl
udes your credentials.
# You might want to remove those credentials before you share the notebook.
client_d2a62246b7af47aba928b58d3432c339 = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='9WdJzzV3kakYC_zt1XQbZp_6DP8nB22nzvE8bZWygqKs',
    ibm_auth_endpoint="https://iam.ng.bluemix.net/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3-api.us-geo.objectstorage.service.networklayer.co
m')

body = client_d2a62246b7af47aba928b58d3432c339.get_object(Bucket='coursera9lin
hvu-donotdelete-pr-spdjuxzcebc3vp',Key='HCMC_District.csv')['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__,
body )

HCMC = pd.read_csv(body)
HCMC.head(5)
```

Out[1]:

	No.	District code	District name	Ward_name	Ward code
0	1	760	District 1	Tan Dinh Ward	26734
1	2	760	District 1	Da Kao Ward	26737
2	3	760	District 1	Ben Nghe Ward	26740
3	4	760	District 1	Ben Thanh Ward	26743
4	5	760	District 1	Nguyen Thai Binh Ward	26746

In [2]:
```
Dist1 = HCMC[HCMC['District name'] == 'District 1']
Dist3 = HCMC[HCMC['District name'] == 'District 3']
```

In [3]:
```
# examine data to find number of District 1
print('District 1 has {} wards.'.format(
        len(Dist1),
        Dist1.shape[0]
    )
)
Dist1.head(15)
```

```
District 1 has 10 wards.
```

Out[3]:

	No.	District code	District name	Ward_name	Ward code
0	1	760	District 1	Tan Dinh Ward	26734
1	2	760	District 1	Da Kao Ward	26737
2	3	760	District 1	Ben Nghe Ward	26740
3	4	760	District 1	Ben Thanh Ward	26743
4	5	760	District 1	Nguyen Thai Binh Ward	26746
5	6	760	District 1	Pham Ngu Lao Ward	26749
6	7	760	District 1	Cau Ong Lanh Ward	26752
7	8	760	District 1	Co Giang Ward	26755
8	9	760	District 1	Nguyen Cu Trinh Ward	26758
9	10	760	District 1	Cau Kho Ward	26761

In [4]:
```
# examine data to find number of District 3
print('District 3 has {} wards.'.format(
        len(Dist3),
        Dist3.shape[0]
    )
)
Dist3.head(15)
```

```
District 3 has 14 wards.
```

Out[4]:

	No.	District code	District name	Ward_name	Ward code
134	135	770	District 3	Ward 8	27121
135	136	770	District 3	Ward 7	27124
136	137	770	District 3	Ward 14	27127
137	138	770	District 3	Ward 12	27130
138	139	770	District 3	Ward 11	27133
139	140	770	District 3	Ward 13	27136
140	141	770	District 3	Ward 6	27139
141	142	770	District 3	Ward 9	27142
142	143	770	District 3	Ward 10	27145
143	144	770	District 3	Ward 4	27148
144	145	770	District 3	Ward 5	27151
145	146	770	District 3	Ward 3	27154
146	147	770	District 3	Ward 2	27157
147	148	770	District 3	Ward 1	27160

In [5]:
```
import numpy as np
import time
import pandas as pd

import json # library to handle JSON files
import requests # library to handle requests
from pandas.io.json import json_normalize # tranform JSON file into a pandas d
ataframe


!conda install -c conda-forge geopy --yes
from geopy.geocoders import Nominatim # convert an address into latitude and l
ongitude values

!conda install -c conda-forge folium=0.5.0 --yes
import folium # map rendering library
import folium # map rendering library
from folium import plugins

# Matplotlib and associated plotting modules
import matplotlib.cm as cm
import matplotlib.colors as colors

import seaborn as sns

# import k-means from clustering stage
from sklearn.cluster import KMeans

print('Libraries imported.')
```

```
Solving environment: done

## Package Plan ##

  environment location: /opt/conda/envs/Python36

  added / updated specs:
    - geopy


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    geographiclib-1.50         |             py_0          34 KB  conda-forge
    geopy-1.20.0               |             py_0          57 KB  conda-forge
    ca-certificates-2019.9.11  |       hecc5488_0         144 KB  conda-forge
    certifi-2019.9.11          |           py36_0         147 KB  conda-forge
    openssl-1.1.1c             |       h516909a_0         2.1 MB  conda-forge
    ------------------------------------------------------------
                                           Total:         2.5 MB

The following NEW packages will be INSTALLED:

    geographiclib:   1.50-py_0         conda-forge
    geopy:           1.20.0-py_0       conda-forge

The following packages will be UPDATED:

    ca-certificates: 2019.8.28-0                   --> 2019.9.11-hecc5488_0 c
onda-forge
    certifi:         2019.9.11-py36_0              --> 2019.9.11-py36_0     c
onda-forge

The following packages will be DOWNGRADED:

    openssl:         1.1.1d-h7b6447c_2             --> 1.1.1c-h516909a_0    c
onda-forge


Downloading and Extracting Packages
geographiclib-1.50   | 34 KB     | ##################################### | 10
0%
geopy-1.20.0         | 57 KB     | ##################################### | 10
0%
ca-certificates-2019 | 144 KB    | ##################################### | 10
0%
certifi-2019.9.11    | 147 KB    | ##################################### | 10
0%
openssl-1.1.1c       | 2.1 MB    | ##################################### | 10
0%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
Solving environment: done

## Package Plan ##
```

```
  environment location: /opt/conda/envs/Python36

  added / updated specs:
    - folium=0.5.0


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    folium-0.5.0               |             py_0          45 KB  conda-forge
    vincent-0.4.4              |             py_1          28 KB  conda-forge
    altair-3.2.0               |           py36_0         770 KB  conda-forge
    branca-0.3.1               |             py_0          25 KB  conda-forge
    ------------------------------------------------------------
                                           Total:         868 KB

The following NEW packages will be INSTALLED:

    altair:  3.2.0-py36_0 conda-forge
    branca:  0.3.1-py_0   conda-forge
    folium:  0.5.0-py_0   conda-forge
    vincent: 0.4.4-py_1   conda-forge


Downloading and Extracting Packages
folium-0.5.0         | 45 KB     | ##################################### | 10
0%
vincent-0.4.4        | 28 KB     | ##################################### | 10
0%
altair-3.2.0         | 770 KB    | ##################################### | 10
0%
branca-0.3.1         | 25 KB     | ##################################### | 10
0%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
Libraries imported.
```

In [6]:
```
# using Geocoder to get the Latitude and Longitude of District 1
i = 1
for i in range (1,11):
    address = (Dist1['Ward_name'].head(i).values)[i-1] + ' District 1, Ho Chi Minh City'
    geolocator = Nominatim(user_agent="my-application")
    location = geolocator.geocode(address)
    latitude = location.latitude
    longitude = location.longitude
    print (Dist1['Ward_name'].head(i).values[i-1], latitude, longitude)
```

```
Tan Dinh Ward 10.7932029 106.6902032
Da Kao Ward 10.7884755 106.698318
Ben Nghe Ward 10.7812343 106.7026503
Ben Thanh Ward 10.7728665 106.6943
Nguyen Thai Binh Ward 10.7688283 106.6987969
Pham Ngu Lao Ward 10.7667074 106.6920012
Cau Ong Lanh Ward 10.7654593 106.696587971109
Co Giang Ward 10.7620099 106.6933648
Nguyen Cu Trinh Ward 10.76239705 106.686651362325
Cau Kho Ward 10.75752515 106.688900225853
```

In [8]:
```
# using Geocoder to get the Latitude and Longitude of District 3
i = 1
for i in range (1,15):
    address = (Dist3['Ward_name'].head(i).values)[i-1] + ' District 3, Ho Chi Minh City'
    geolocator = Nominatim(user_agent="my-application")
    location = geolocator.geocode(address)
    latitude = location.latitude
    longitude = location.longitude
    print (Dist3['Ward_name'].head(i).values[i-1], latitude, longitude)
```

```
Ward 8 10.7979139 106.6748516
Ward 7 10.7806202 106.6863996
Ward 14 10.7895656 106.6786342
Ward 12 10.7882328 106.673673806804
Ward 11 10.7929518 106.6749979
Ward 13 10.7863145 106.6778478
Ward 6 10.779185 106.6911478
Ward 9 10.7816101 106.680207279703
Ward 10 10.7814723 106.6760941
Ward 4 10.7737617 106.6832254
Ward 5 10.7904337 106.6623708999
Ward 3 10.7701933 106.6790566
Ward 2 10.7973984 106.6876578
Ward 1 10.7988373 106.6826542
```

In [10]:
```
body = client_d2a62246b7af47aba928b58d3432c339.get_object(Bucket='coursera9lin
hvu-donotdelete-pr-spdjuxzcebc3vp',Key='Dist1_latlong.csv')['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__,
body )

Dist1_latlong = pd.read_csv(body)
Dist1_latlong.head()
```

Out[10]:

	No.	District code	District name	Ward_name	Ward code	Latitude	Longitude
0	1	760	District 1	Tan Dinh Ward	26734	10.793203	106.690203
1	2	760	District 1	Da Kao Ward	26737	10.788476	106.698318
2	3	760	District 1	Ben Nghe Ward	26740	10.781234	106.702650
3	4	760	District 1	Ben Thanh Ward	26743	10.772866	106.694300
4	5	760	District 1	Nguyen Thai Binh Ward	26746	10.768828	106.698797

In [11]:
```
body = client_d2a62246b7af47aba928b58d3432c339.get_object(Bucket='coursera9lin
hvu-donotdelete-pr-spdjuxzcebc3vp',Key='Dist3_latlong.csv')['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__,
body )

Dist3_latlong = pd.read_csv(body)
Dist3_latlong.head()
```

Out[11]:

	No.	District code	District name	Ward_name	Ward code	Latitude	Longitude
0	135	770	District 3	Ward 8	27121	10.797914	106.674852
1	136	770	District 3	Ward 7	27124	10.780620	106.686400
2	137	770	District 3	Ward 14	27127	10.789566	106.678634
3	138	770	District 3	Ward 12	27130	10.788233	106.673674
4	139	770	District 3	Ward 11	27133	10.792952	106.674998

Based on the data of Latitude and Longitude fo both District, create map with pointed area in it.

In [12]:

```
address = 'District 1, Hochiminh city'
geolocator = Nominatim(user_agent="my-application")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude

# create map of New York using latitude and longitude values
map_dist1 = folium.Map(location=[latitude, longitude], zoom_start=10)

# add markers to map
for lat, lng, borough, neighborhood in zip(Dist1_latlong['Latitude'], Dist1_latlong['Longitude'], Dist1_latlong['Ward_name'], Dist1_latlong['District name']):
    label = '{}, {}'.format(neighborhood, borough)
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=5,
        popup=label,
        color='blue',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.7,
        parse_html=False).add_to(map_dist1)

map_dist1
```

Out[12]:

In [13]:

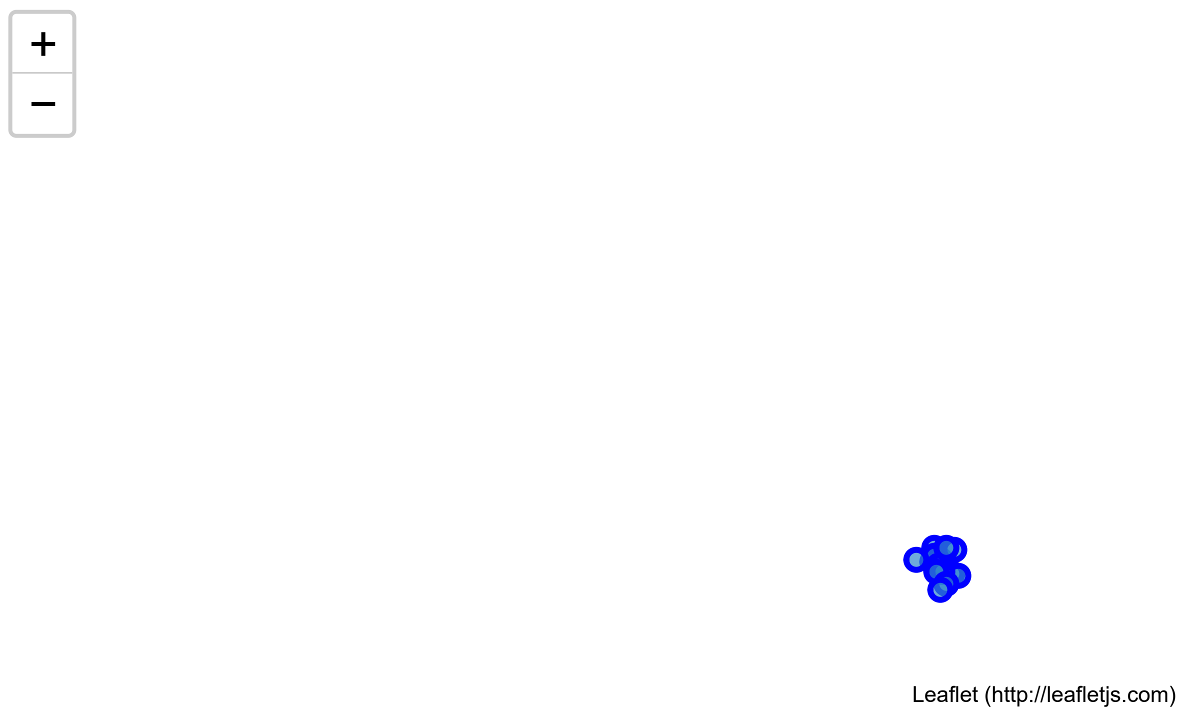
```python
address = 'District 3, Hochiminh city'
geolocator = Nominatim(user_agent="my-application")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude

# create map of New York using latitude and longitude values
map_dist3 = folium.Map(location=[latitude, longitude], zoom_start=10)

# add markers to map
for lat, lng, borough, neighborhood in zip(Dist3_latlong['Latitude'], Dist3_latlong['Longitude'], Dist3_latlong['Ward_name'], Dist3_latlong['District name']):
    label = '{}, {}'.format(neighborhood, borough)
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=5,
        popup=label,
        color='blue',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.7,
        parse_html=False).add_to(map_dist3)

map_dist3
```

Out[13]:



4. Methodology

After converting addresses into their equivalent latitude and longitude values. I will use the Foursquare API to explore neighborhoods in both District 1 and District 3 with steps as below:

- Explore function to get the most common venue categories in each neighborhood
- Use the Folium library to visualize the neighborhoods in District 1 and District 3 and their emerging clusters
- Use feature to group the neighborhoods into clusters
- Use K-means clustering algorithm to analyst. And also, the Folium library to visualize the neighborhoods in Kuala Lumpur and Johor Bahru and their emerging clusters.

Based on dataframe analysis above, I can display the highest number of area within both district.

Using Foursquare API to get venues at surounding area of both District 1 and District 3

In [14]:
```
#Define Foursquare Credentials and Version
CLIENT_ID = 'XJ0QXHGZI4SQBVNG2MI5IP0RQ5DHGJDZA30CKIWNAKDIVBHJ'
CLIENT_SECRET = 'SRRGHAZYV3WLY4JJZ2GQF0NNTSWNYL2HXLMTMMCJOGDDIJ4G'
VERSION = '20180604'

#explore the first neighborhood in our dataframe
#Get the neighborhood's latitude and longitude values.
neighborhood_latitude = Dist1_latlong.loc[0, 'Latitude'] # neighborhood latitu
de value
neighborhood_longitude = Dist1_latlong.loc[0, 'Longitude'] # neighborhood long
itude value
neighborhood_name = Dist1_latlong.loc[0, 'Ward_name'] # neighborhood name

#get the top 100 venues that are in District 1 within a radius of 500 meters
LIMIT = 100 # limit of number of venues returned by Foursquare API
radius = 500 # define radius
url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secre
t={}&v={}&ll={},{}&radius={}&limit={}'.format(
    CLIENT_ID,
    CLIENT_SECRET,
    VERSION,
    neighborhood_latitude,
    neighborhood_longitude,
    radius,
    LIMIT)

#Send the GET request and examine the resutls
results = requests.get(url).json()

#borrow the get_category_type function from the Foursquare lab.
# 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']

#clean the json and structure it into a pandas dataframe
venues = results['response']['groups'][0]['items']
nearby_venues = json_normalize(venues) # flatten JSON

# filter columns
filtered_columns = ['venue.name', 'venue.categories', 'venue.location.lat', 'v
enue.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, axi
s=1)

# clean columns
```

```
nearby_venues.columns = [col.split(".")[-1] for col in nearby_venues.columns]
print('{} venues were returned by Foursquare for District 1, HCMC.'.format(nea
rby_venues.shape[0]))
nearby_venues.head()
```

```
34 venues were returned by Foursquare for District 1, HCMC.
```

Out[14]:

	name	categories	lat	lng
0	Cục Gạch	Vietnamese Restaurant	10.792957	106.689020
1	Cuc Gach Quan	Vietnamese Restaurant	10.790773	106.691795
2	Buddha Chay	Vegetarian / Vegan Restaurant	10.792762	106.688252
3	Bánh canh cua 87	Vietnamese Restaurant	10.794697	106.690917
4	Cơm Tấm Nguyễn Phi Khanh	Breakfast Spot	10.791676	106.692159

In [21]:
```
#explore the first neighborhood in our dataframe
#Get the neighborhood's latitude and longitude values.
neighborhood_latitude = Dist3_latlong.loc[0, 'Latitude'] # neighborhood latitu
de value
neighborhood_longitude = Dist3_latlong.loc[0, 'Longitude'] # neighborhood long
itude value
neighborhood_name = Dist3_latlong.loc[0, 'Ward_name'] # neighborhood name

#get the top 100 venues that are in District 1 within a radius of 500 meters
LIMIT = 100 # limit of number of venues returned by Foursquare API
radius = 500 # define radius
url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secre
t={}&v={}&ll={},{}&radius={}&limit={}'.format(
    CLIENT_ID,
    CLIENT_SECRET,
    VERSION,
    neighborhood_latitude,
    neighborhood_longitude,
    radius,
    LIMIT)

#Send the GET request and examine the resutls
results = requests.get(url).json()

#borrow the get_category_type function from the Foursquare lab.
# 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']

#clean the json and structure it into a pandas dataframe
venues = results['response']['groups'][0]['items']
nearby_venues = json_normalize(venues) # flatten JSON

# filter columns
filtered_columns = ['venue.name', 'venue.categories', 'venue.location.lat', 'v
enue.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, axi
s=1)

# clean columns
nearby_venues.columns = [col.split(".")[-1] for col in nearby_venues.columns]
print('{} venues were returned by Foursquare for District 3, HCMC.'.format(nea
rby_venues.shape[0]))
nearby_venues.head()
```

```
32 venues were returned by Foursquare for District 3, HCMC.
```

Out[21]:

	name	categories	lat	lng
0	EASTIN Grand Hotel Saigon	Hotel	10.796807	106.673363
1	Tung Garden	Cantonese Restaurant	10.796832	106.673386
2	The Open Space	Coffee Shop	10.796959	106.674827
3	The Fig	Café	10.797200	106.676869
4	Yeebo	Chinese Restaurant	10.795833	106.675274

In [22]:
```
#function to repeat the same process to all area
def getNearbyVenues(names, latitudes, longitudes, radius=500):

    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={}&limit={}'.format(
            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 = ['Ward_name',
                  'Ward Latitude',
                  'Ward Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']

    return(nearby_venues)

#run the above function on each neighborhood and create a new dataframe
Dist1_venues = getNearbyVenues(names=Dist1_latlong['Ward_name'],
                                   latitudes=Dist1_latlong['Latitude'],
                                   longitudes=Dist1_latlong['Longitude']
                                  )

#check the size of the resulting dataframe
print(Dist1_venues.shape)
Dist1_venues.head()
```

```
Tan Dinh Ward
Da Kao Ward
Ben Nghe Ward
Ben Thanh Ward
Nguyen Thai Binh Ward
Pham Ngu Lao Ward
Cau Ong Lanh Ward
Co Giang Ward
Nguyen Cu Trinh Ward
Cau Kho Ward
(532, 7)
```

Out[22]:

	Ward_name	Ward Latitude	Ward Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Tan Dinh Ward	10.793203	106.690203	Cục Gạch	10.792957	106.689020	Vietnamese Restaurant
1	Tan Dinh Ward	10.793203	106.690203	Cuc Gach Quan	10.790773	106.691795	Vietnamese Restaurant
2	Tan Dinh Ward	10.793203	106.690203	Buddha Chay	10.792762	106.688252	Vegetarian / Vegan Restaurant
3	Tan Dinh Ward	10.793203	106.690203	Bánh canh cua 87	10.794697	106.690917	Vietnamese Restaurant
4	Tan Dinh Ward	10.793203	106.690203	Cơm Tấm Nguyễn Phi Khanh	10.791676	106.692159	Breakfast Spot

In [23]:
```
Dist3_venues = getNearbyVenues(names=Dist3_latlong['Ward_name'],
                                    latitudes=Dist3_latlong['Latitude'],
                                    longitudes=Dist3_latlong['Longitude']
                                   )

#check the size of the resulting dataframe
print(Dist3_venues.shape)
Dist3_venues.head()
```

```
Ward 8
Ward 7
Ward 14
Ward 12
Ward 11
Ward 13
Ward 6
Ward 9
Ward 10
Ward 4
Ward 5
Ward 3
Ward 2
Ward 1
(468, 7)
```

Out[23]:

	Ward_name	Ward Latitude	Ward Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Ward 8	10.797914	106.674852	EASTIN Grand Hotel Saigon	10.796807	106.673363	Hotel
1	Ward 8	10.797914	106.674852	Tung Garden	10.796832	106.673386	Cantonese Restaurant
2	Ward 8	10.797914	106.674852	The Open Space	10.796959	106.674827	Coffee Shop
3	Ward 8	10.797914	106.674852	The Fig	10.797200	106.676869	Café
4	Ward 8	10.797914	106.674852	Yeebo	10.795833	106.675274	Chinese Restaurant

In [24]:
```
#check how many venues were returned for each area
print('There are {} uniques categories in District 1.'.format(len(Dist1_venues
['Venue Category'].unique())))
Dist1_venues.groupby('Ward_name').count()
```

```
There are 94 uniques categories in District 1.
```

Out[24]:

Ward_name	Ward Latitude	Ward Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
Ben Nghe Ward	84	84	84	84	84	84
Ben Thanh Ward	38	38	38	38	38	38
Cau Kho Ward	21	21	21	21	21	21
Cau Ong Lanh Ward	37	37	37	37	37	37
Co Giang Ward	15	15	15	15	15	15
Da Kao Ward	100	100	100	100	100	100
Nguyen Cu Trinh Ward	31	31	31	31	31	31
Nguyen Thai Binh Ward	72	72	72	72	72	72
Pham Ngu Lao Ward	100	100	100	100	100	100
Tan Dinh Ward	34	34	34	34	34	34

In [25]:
```
#check how many venues were returned for each area
print('There are {} uniques categories in District 3.'.format(len(Dist3_venues
['Venue Category'].unique())))
Dist3_venues.groupby('Ward_name').count()
```

```
There are 77 uniques categories in District 3.
```

Out[25]:

Ward_name	Ward Latitude	Ward Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
Ward 1	15	15	15	15	15	15
Ward 10	11	11	11	11	11	11
Ward 11	37	37	37	37	37	37
Ward 12	11	11	11	11	11	11
Ward 13	23	23	23	23	23	23
Ward 14	28	28	28	28	28	28
Ward 2	53	53	53	53	53	53
Ward 3	37	37	37	37	37	37
Ward 4	32	32	32	32	32	32
Ward 5	5	5	5	5	5	5
Ward 6	92	92	92	92	92	92
Ward 7	63	63	63	63	63	63
Ward 8	32	32	32	32	32	32
Ward 9	29	29	29	29	29	29

Analyze District 1

In [26]:
```
# one hot encoding
Dist1_onehot = pd.get_dummies(Dist1_venues[['Venue Category']], prefix="", pre
fix_sep="")

# add neighborhood column back to dataframe
Dist1_onehot['Ward_name'] = Dist1_venues['Ward_name']

# move neighborhood column to the first column
fixed_columns = [Dist1_onehot.columns[-1]] + list(Dist1_onehot.columns[:-1])
Dist1_onehot = Dist1_onehot[fixed_columns]

#examine the new dataframe size after one hot encoding
print('{} rows were returned after one hot encoding.'.format(Dist1_onehot.shap
e[0]))

#group rows by neighborhood and by taking the mean of the frequency of occurre
nce of each category
Dist1_grouped = Dist1_onehot.groupby('Ward_name').mean().reset_index()

#examine the new dataframe size after one hot encoding
print('{} rows were returned after grouping.'.format(Dist1_grouped.shape[0]))
```

```
532 rows were returned after one hot encoding.
10 rows were returned after grouping.
```

In [27]:
```
#print each neighborhood along with the top 5 most common venues
num_top_venues = 5

for hood in Dist1_grouped['Ward_name']:
    print("----"+hood+"----")
    temp = Dist1_grouped[Dist1_grouped['Ward_name'] == 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).hea
d(num_top_venues))
    print('\n')
```

```
----Ben Nghe Ward----
            venue  freq
0     Coffee Shop  0.12
1            Café  0.08
2           Hotel  0.07
3  Massage Studio  0.05
4    Cocktail Bar  0.05


----Ben Thanh Ward----
                   venue  freq
0                  Hotel  0.16
1  Vietnamese Restaurant  0.13
2         Sandwich Place  0.08
3                   Café  0.05
4                    Spa  0.05


----Cau Kho Ward----
                   venue  freq
0       Asian Restaurant  0.14
1  Vietnamese Restaurant  0.14
2     Chinese Restaurant  0.10
3                   Café  0.10
4             Restaurant  0.05


----Cau Ong Lanh Ward----
                   venue  freq
0  Vietnamese Restaurant  0.16
1                  Hotel  0.05
2           Burger Joint  0.05
3      Indian Restaurant  0.05
4                 Hostel  0.05


----Co Giang Ward----
                           venue  freq
0                          Hotel  0.20
1          Vietnamese Restaurant  0.13
2  Vegetarian / Vegan Restaurant  0.13
3             Seafood Restaurant  0.07
4                         Hostel  0.07


----Da Kao Ward----
                   venue  freq
0                   Café  0.21
1  Vietnamese Restaurant  0.18
2    Japanese Restaurant  0.06
3      French Restaurant  0.05
4            Coffee Shop  0.05


----Nguyen Cu Trinh Ward----
                   venue  freq
0  Vietnamese Restaurant  0.16
```

```
1                   Café  0.10
2     Seafood Restaurant  0.06
3       Asian Restaurant  0.06
4           Noodle House  0.03


----Nguyen Thai Binh Ward----
                   venue  freq
0  Vietnamese Restaurant  0.19
1                  Hotel  0.08
2                   Café  0.08
3           Burger Joint  0.06
4            Coffee Shop  0.06


----Pham Ngu Lao Ward----
                           venue  freq
0                          Hotel  0.12
1          Vietnamese Restaurant  0.11
2                         Hostel  0.07
3  Vegetarian / Vegan Restaurant  0.05
4                    Coffee Shop  0.04


----Tan Dinh Ward----
                   venue  freq
0  Vietnamese Restaurant  0.24
1                   Café  0.15
2            Coffee Shop  0.09
3         Breakfast Spot  0.06
4       Asian Restaurant  0.06
```

In [28]:
```
#put into a pandas dataframe

#write a function to sort the venues in descending order
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]

#create the new dataframe and display the top 10 venues for each neighborhood
num_top_venues = 8

indicators = ['st', 'nd', 'rd']

# create columns according to number of top venues
columns = ['Ward_name']
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
areas_venues_sorted = pd.DataFrame(columns=columns)
areas_venues_sorted['Ward_name'] = Dist1_grouped['Ward_name']

for ind in np.arange(Dist1_grouped.shape[0]):
    areas_venues_sorted.iloc[ind, 1:] = return_most_common_venues(Dist1_groupe
d.iloc[ind, :], num_top_venues)

areas_venues_sorted.head()
```

Out[28]:

	Ward_name	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	
0	Ben Nghe Ward	Coffee Shop	Café	Hotel	Cocktail Bar	Massage Studio	Spa	Asian Restaurant	R
1	Ben Thanh Ward	Hotel	Vietnamese Restaurant	Sandwich Place	Café	Food Court	Spa	Noodle House	
2	Cau Kho Ward	Vietnamese Restaurant	Asian Restaurant	Chinese Restaurant	Café	Coffee Shop	Fast Food Restaurant	Bookstore	R
3	Cau Ong Lanh Ward	Vietnamese Restaurant	Hostel	Hotel	Coffee Shop	Juice Bar	Indian Restaurant	Burger Joint	V R
4	Co Giang Ward	Hotel	Vietnamese Restaurant	Vegetarian / Vegan Restaurant	Hostel	Food	Middle Eastern Restaurant	Seafood Restaurant	R

K-mean Cluster District 1

In [29]:
```
from sklearn.cluster import KMeans

# set number of clusters
kclusters = 3

Dist1_grouped_clustering = Dist1_grouped.drop('Ward_name', 1)

# run k-means clustering
kmeans = KMeans(n_clusters=kclusters, random_state=0).fit(Dist1_grouped_cluste
ring)

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

#create a new dataframe that includes the cluster as well as the top 10 venues
for each neighborhood.
Dist1_merged = Dist1_latlong

# add clustering labels
Dist1_merged['Cluster Labels'] = kmeans.labels_

# merge grouped with data to add latitude/longitude for each neighborhood
Dist1_merged = Dist1_merged.join(areas_venues_sorted.set_index('Ward_name'), o
n='Ward_name')

Dist1_merged.head()
```

Out[29]:

	No.	District code	District name	Ward_name	Ward code	Latitude	Longitude	Cluster Labels	1st Most Common Venue	2nd I Com Ve
0	1	760	District 1	Tan Dinh Ward	26734	10.793203	106.690203	1	Vietnamese Restaurant	
1	2	760	District 1	Da Kao Ward	26737	10.788476	106.698318	2	Café	Vietnar Restau
2	3	760	District 1	Ben Nghe Ward	26740	10.781234	106.702650	1	Coffee Shop	
3	4	760	District 1	Ben Thanh Ward	26743	10.772866	106.694300	2	Hotel	Vietnar Restau
4	5	760	District 1	Nguyen Thai Binh Ward	26746	10.768828	106.698797	0	Vietnamese Restaurant	I

In [30]:
```
# Matplotlib and associated plotting modules
import matplotlib.cm as cm
import matplotlib.colors as colors

#Finally, let's visualize the resulting clusters
# create map 10.793203, 106.690203
D1_clusters = folium.Map(location=[10.793203,106.690203], zoom_start=13)

# 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(Dist1_merged['Latitude'], Dist1_merged['Longitude'], Dist1_merged['Ward_name'], Dist1_merged['Cluster Labels']):
    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(D1_clusters)

D1_clusters
```

Out[30]:

Analyze District 3

In [31]:
```
# one hot encoding
Dist3_onehot = pd.get_dummies(Dist3_venues[['Venue Category']], prefix="", pre
fix_sep="")

# add neighborhood column back to dataframe
Dist3_onehot['Ward_name'] = Dist3_venues['Ward_name']

# move neighborhood column to the first column
fixed_columns = [Dist3_onehot.columns[-1]] + list(Dist3_onehot.columns[:-1])
Dist3_onehot = Dist3_onehot[fixed_columns]

#examine the new dataframe size after one hot encoding
print('{} rows were returned after one hot encoding.'.format(Dist3_onehot.shap
e[0]))

#group rows by neighborhood and by taking the mean of the frequency of occurre
nce of each category
Dist3_grouped = Dist3_onehot.groupby('Ward_name').mean().reset_index()

#examine the new dataframe size after one hot encoding
print('{} rows were returned after grouping.'.format(Dist3_grouped.shape[0]))
```

```
468 rows were returned after one hot encoding.
14 rows were returned after grouping.
```

In [32]:
```
#print each neighborhood along with the top 5 most common venues
num_top_venues = 5

for hood in Dist3_grouped['Ward_name']:
    print("----"+hood+"----")
    temp = Dist3_grouped[Dist3_grouped['Ward_name'] == 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).hea
d(num_top_venues))
    print('\n')
```

```
----Ward 1----
                   venue  freq
0  Vietnamese Restaurant  0.20
1                   Café  0.20
2            Pizza Place  0.07
3            Coffee Shop  0.07
4              BBQ Joint  0.07


----Ward 10----
                venue  freq
0                Café  0.18
1  Seafood Restaurant  0.18
2          Restaurant  0.09
3           BBQ Joint  0.09
4        Tennis Court  0.09


----Ward 11----
                   venue  freq
0                   Café  0.32
1  Vietnamese Restaurant  0.11
2            Coffee Shop  0.11
3                  Diner  0.05
4     Chinese Restaurant  0.05


----Ward 12----
                   venue  freq
0  Vietnamese Restaurant  0.27
1                   Café  0.27
2                  Diner  0.09
3            Pizza Place  0.09
4           Tennis Court  0.09


----Ward 13----
                   venue  freq
0  Vietnamese Restaurant  0.22
1                   Café  0.17
2     Seafood Restaurant  0.13
3                  Diner  0.09
4            Coffee Shop  0.09


----Ward 14----
                   venue  freq
0                   Café  0.21
1  Vietnamese Restaurant  0.18
2                  Diner  0.11
3            Coffee Shop  0.07
4     Seafood Restaurant  0.07


----Ward 2----
                   venue  freq
0                   Café  0.13
```

```
1            Coffee Shop  0.13
2  Vietnamese Restaurant  0.08
3    Japanese Restaurant  0.08
4       Asian Restaurant  0.06


----Ward 3----
                   venue  freq
0  Vietnamese Restaurant  0.22
1       Asian Restaurant  0.11
2                   Café  0.08
3             Food Truck  0.05
4           Dessert Shop  0.05


----Ward 4----
                   venue  freq
0                   Café  0.34
1  Vietnamese Restaurant  0.12
2            Coffee Shop  0.06
3         Ice Cream Shop  0.03
4                 Market  0.03


----Ward 5----
                  venue  freq
0           Flea Market   0.2
1                  Food   0.2
2                 Hotel   0.2
3  Gym / Fitness Center   0.2
4        Breakfast Spot   0.2


----Ward 6----
                           venue  freq
0          Vietnamese Restaurant  0.21
1               Asian Restaurant  0.10
2                           Café  0.09
3                    Coffee Shop  0.08
4  Vegetarian / Vegan Restaurant  0.04


----Ward 7----
                   venue  freq
0                   Café  0.25
1  Vietnamese Restaurant  0.22
2            Coffee Shop  0.10
3       Asian Restaurant  0.06
4             Restaurant  0.03


----Ward 8----
                   venue  freq
0            Coffee Shop  0.25
1                   Café  0.22
2  Vietnamese Restaurant  0.06
3     Chinese Restaurant  0.06
```

```
4    American Restaurant  0.03


----Ward 9----
                   venue  freq
0  Vietnamese Restaurant  0.21
1                   Café  0.14
2     Seafood Restaurant  0.14
3            Coffee Shop  0.07
4            Music Venue  0.07
```

In [33]:
```
#create the new dataframe and display the top 10 venues for each neighborhood
num_top_venues = 8

indicators = ['st', 'nd', 'rd']

# create columns according to number of top venues
columns = ['Ward_name']
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
areas_venues_sorted = pd.DataFrame(columns=columns)
areas_venues_sorted['Ward_name'] = Dist3_grouped['Ward_name']

for ind in np.arange(Dist3_grouped.shape[0]):
    areas_venues_sorted.iloc[ind, 1:] = return_most_common_venues(Dist3_groupe
d.iloc[ind, :], num_top_venues)

areas_venues_sorted.head()
```

Out[33]:

	Ward_name	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
0	Ward 1	Vietnamese Restaurant	Café	Bookstore	Snack Place	Diner	Pizza Place	Coffee Shop
1	Ward 10	Seafood Restaurant	Café	Coffee Shop	Music Venue	Convenience Store	Restaurant	Gym / Fitness Center
2	Ward 11	Café	Coffee Shop	Vietnamese Restaurant	Diner	Chinese Restaurant	Smoothie Shop	Japanese Restaurant
3	Ward 12	Vietnamese Restaurant	Café	Hotel	Tennis Court	Pizza Place	Diner	Steakhouse
4	Ward 13	Vietnamese Restaurant	Café	Seafood Restaurant	Diner	Coffee Shop	Smoothie Shop	Asian Restaurant

K-mean Cluster District 3

In [34]:

```
# set number of clusters
kclusters = 3

Dist3_grouped_clustering = Dist3_grouped.drop('Ward_name', 1)

# run k-means clustering
kmeans = KMeans(n_clusters=kclusters, random_state=0).fit(Dist3_grouped_cluste
ring)

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

#create a new dataframe that includes the cluster as well as the top 10 venues
for each neighborhood.
Dist3_merged = Dist3_latlong

# add clustering labels
Dist3_merged['Cluster Labels'] = kmeans.labels_

# merge grouped with data to add latitude/longitude for each neighborhood
Dist3_merged = Dist3_merged.join(areas_venues_sorted.set_index('Ward_name'), o
n='Ward_name')

Dist3_merged.head()
```

Out[34]:

	No.	District code	District name	Ward_name	Ward code	Latitude	Longitude	Cluster Labels	1st Most Common Venue	2nd I Com V
0	135	770	District 3	Ward 8	27121	10.797914	106.674852	1	Coffee Shop	
1	136	770	District 3	Ward 7	27124	10.780620	106.686400	0	Café	Vietnar Resta
2	137	770	District 3	Ward 14	27127	10.789566	106.678634	0	Café	Vietnar Resta
3	138	770	District 3	Ward 12	27130	10.788233	106.673674	1	Vietnamese Restaurant	
4	139	770	District 3	Ward 11	27133	10.792952	106.674998	1	Café	C ;

In [35]:

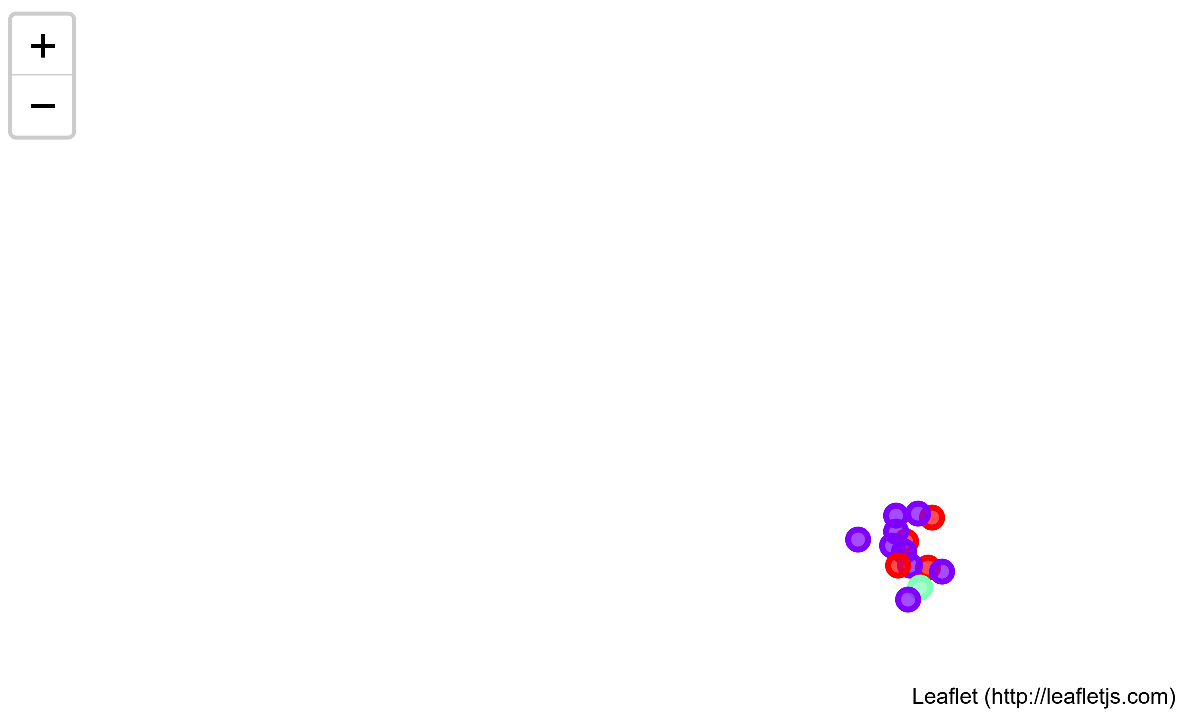
```
#Visualize the resulting clusters
# create map
Dist3_clusters = folium.Map(location=[latitude, longitude], zoom_start=11)

# 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(Dist3_merged['Latitude'], Dist3_merged['Long
itude'], Dist3_merged['Ward_name'], Dist3_merged['Cluster Labels']):
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=Tru
e)
    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(Dist3_clusters)

Dist3_clusters
```

Out[35]:



5.Results

In [36]:
```
#Cluster 1 for District 1
Dist1_merged.loc[Dist1_merged['Cluster Labels'] == 0, Dist1_merged.columns[[2]
+ list(range(5, Dist1_merged.shape[1]))]]
```

Out[36]:

	District name	Latitude	Longitude	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue
4	District 1	10.768828	106.698797	0	Vietnamese Restaurant	Hotel	Café	Burger Joint	Coffee Shop

In [37]:
```
#Cluster 2 for District 1
Dist1_merged.loc[Dist1_merged['Cluster Labels'] == 1, Dist1_merged.columns[[2]
+ list(range(5, Dist1_merged.shape[1]))]]
```

Out[37]:

	District name	Latitude	Longitude	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th M Comn Ve
0	District 1	10.793203	106.690203	1	Vietnamese Restaurant	Café	Coffee Shop	Vegetarian / Vegan Restaurant	Break S
2	District 1	10.781234	106.702650	1	Coffee Shop	Café	Hotel	Cocktail Bar	Mass Stu
5	District 1	10.766707	106.692001	1	Hotel	Vietnamese Restaurant	Hostel	Vegetarian / Vegan Restaurant	Co S
6	District 1	10.765459	106.696588	1	Vietnamese Restaurant	Hostel	Hotel	Coffee Shop	Juice
9	District 1	10.757525	106.688900	1	Vietnamese Restaurant	Asian Restaurant	Chinese Restaurant	Café	Co S

In [38]:
```
#Cluster 3 for District 1
Dist1_merged.loc[Dist1_merged['Cluster Labels'] == 2, Dist1_merged.columns[[2] + list(range(5, Dist1_merged.shape[1]))]]
```

Out[38]:

	District name	Latitude	Longitude	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5t Cc
1	District 1	10.788476	106.698318	2	Café	Vietnamese Restaurant	Japanese Restaurant	French Restaurant	Coffe
3	District 1	10.772866	106.694300	2	Hotel	Vietnamese Restaurant	Sandwich Place	Café	Foo
7	District 1	10.762010	106.693365	2	Hotel	Vietnamese Restaurant	Vegetarian / Vegan Restaurant	Hostel	
8	District 1	10.762397	106.686651	2	Vietnamese Restaurant	Café	Seafood Restaurant	Asian Restaurant	Conve

In [39]:
```
#Cluster 1 for Disrtrict 3
Dist3_merged.loc[Dist3_merged['Cluster Labels'] == 0, Dist3_merged.columns[[2] + list(range(5, Dist3_merged.shape[1]))]]
```

Out[39]:

	District name	Latitude	Longitude	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5 C
1	District 3	10.780620	106.686400	0	Café	Vietnamese Restaurant	Coffee Shop	Asian Restaurant	Re
2	District 3	10.789566	106.678634	0	Café	Vietnamese Restaurant	Diner	Smoothie Shop	Coff
8	District 3	10.781472	106.676094	0	Seafood Restaurant	Café	Coffee Shop	Music Venue	Conv
12	District 3	10.797398	106.687658	0	Café	Coffee Shop	Japanese Restaurant	Vietnamese Restaurant	Re

In [40]:
```
#Cluster 2 for District 3
Dist3_merged.loc[Dist3_merged['Cluster Labels'] == 1, Dist3_merged.columns[[2]
+ list(range(5, Dist3_merged.shape[1]))]]
```

Out[40]:

	District name	Latitude	Longitude	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Co
0	District 3	10.797914	106.674852	1	Coffee Shop	Café	Vietnamese Restaurant	Chinese Restaurant	Am Rest
3	District 3	10.788233	106.673674	1	Vietnamese Restaurant	Café	Hotel	Tennis Court	
4	District 3	10.792952	106.674998	1	Café	Coffee Shop	Vietnamese Restaurant	Diner	Cl Rest
5	District 3	10.786314	106.677848	1	Vietnamese Restaurant	Café	Seafood Restaurant	Diner	(
6	District 3	10.779185	106.691148	1	Vietnamese Restaurant	Asian Restaurant	Café	Coffee Shop	Vege / Rest
7	District 3	10.781610	106.680207	1	Vietnamese Restaurant	Seafood Restaurant	Café	Music Venue	F Rest
10	District 3	10.790434	106.662371	1	Hotel	Gym / Fitness Center	Flea Market	Food	Bre
11	District 3	10.770193	106.679057	1	Vietnamese Restaurant	Asian Restaurant	Café	Food Truck	E Tea
13	District 3	10.798837	106.682654	1	Vietnamese Restaurant	Café	Bookstore	Snack Place	

In [41]:
```
#Cluster 3 for District 3
Dist3_merged.loc[Dist3_merged['Cluster Labels'] == 2, Dist3_merged.columns[[2]
+ list(range(5, Dist3_merged.shape[1]))]]
```

Out[41]:

	District name	Latitude	Longitude	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue
9	District 3	10.773762	106.683225	2	Café	Vietnamese Restaurant	Coffee Shop	Ice Cream Shop	Restaurant

6.Discussion

Based on cluster for each district above, I believe that classification for each cluster can be done better with calculation of venues categories (most common) in each district in Hochiminh city. Refering to each clsuter, I can't deterimine clearly what represent in each cluster by using Foursquare - Most Common Venue data.

Assumed each cluster as follow:

- Cluster 1: District 1: Tourism/Business place
- Cluster 2: District 1: Residental
- Cluster 3: District 1: Mix
- Cluster 1: District 3: Tourism/Business placeResidental
- Cluster 2: District 3: Residental
- Cluster 3: District 3: Mix What is lacking at this point is a systematic, quantitative way to identify and distinguish different district and to describe the correlation most common venues as recorded in Foursquare. The reality is however more complex: similar cities might have or might not have similar common venues. A further step in this classification would be to find a method to extract these common venues and integrate the spatial correlations between different of areas or district.

Conclusion

The data was captured by using Foursquare API is common places all around the world. Using Foursquare which is to determine the similarity or dissimilarirty of both districts and classification of area located inside District whether it is residential/tourism-bussiness places/others In conclusion, both district 1 and district 3 are the center of attraction among Hochiminh city. However, to declare both districts are similar or dissimilar base on common venues visited is quite difficult because both districts are similar in some venues also dissimilar in certain venues.

Thank you.

Vu Hoang Ha Linh