Capstone Project

11.2018

Allison

Introduction

When we decide to visit a country, we may need lots of information pertaining to some target places, i.e. a city, scenery, or national park, etc. In this project, I'd like to develop a system to help tourists visit Taipei City which is the capital and a special municipality of Taiwan. By analyzing the 12 administrative districts in the Taipei City, including Songshan, Xinyi, Daan, Zhongshan, Zhongzheng, Datong, Wanhua, Wenshan, Nangang, Neihu, Shilin, and Beitou, tourists may have initial knowledge of Taipei City which may help them to organize their itinerary. Moreover, when we plan a journey, we may search where to stay with online marketplaces and filter out lots of searches. To help people make a decision more easily, the system will show the regional characteristics with the Foursquare data pertaining to a short list of accommodation choices. More specifically, the project will provide a function that analyzes the data obtained from Foursquare and assists people to search as well as decide the accommodations when they are traveling, especially, in Taipei.

Objective & Data

Objective

1. Analyze the districts in Taipei
2. Recommend the ordered accommodations from the aspect of the specified attributes.

Data

The district data of Taipei City will be scraped from "https://en.wikipedia.org/wiki/Taipei". For the second objective, a short list of accommodation addresses is prepared in text file format and uploaded to my github "https://github.com/allisonyuplayground/Coursera_Capstone/blob/master/accommodation_list.txt". Another aspect of this project is the Foursquare data, such as venue name, venue category, venue latitude, venue longitude, etc., which are mainly used for segmentation and clustering.

Methodology 1/17

✱ Analyze the districts in Taipei - Data wrangling

```
df = pd.read_html( "https://en.wikipedia.org/wiki/Taipei" )[8]

# Drop the first and second rows that we don't need
df.drop( df.columns[[0, 1]], axis = 0, inplace = True )

# Shift left by one column of the row index 2
df.loc[[2]] = df.loc[[2]].shift( -1, axis = 1 )

# Drop the columns we don't need
df.drop( df.columns[[2, 3, 4, 8]], axis = 1, inplace = True )

# Fill the missing value
df.loc[[2], [5]] = 257922

# Reset the row indexes
df.reset_index( drop = True, inplace = True )

# Define the dataframe columns
df.columns = ['eng_name', 'cn_name', 'population', 'area_km2', 'postal_code']

# Convert data into expected format
print( "The data type of column 'postal_code' and 'population' are '{}' and '{}' originally.".format( d
df['postal_code'] = df['postal_code'].astype( np.int64 )
df['population'] = df['population'].astype( np.int64 )
print( "The data type of column 'postal_code' and 'population' are '{}' and '{}' now.".format( df['post
```

```
The data type of column 'postal_code' and 'population' are 'float64' and 'object' originally.
The data type of column 'postal_code' and 'population' are 'int64' and 'int64' now.
```

	eng_name	cn_name	population	area_km2	postal_code
0	Beitou	北投區	257922	56.8216	112
1	Da'an	大安區	312909	11.3614	106
2	Datong	大同區	131029	5.6815	103
3	Nangang	南港區	122296	21.8424	115
4	Neihu	內湖區	287726	31.5787	114
5	Shilin	士林區	290682	62.3682	111
6	Songshan	松山區	209689	9.2878	105
7	Wanhua	萬華區	194314	8.8522	108
8	Wenshan	文山區	275433	31.5090	116
9	Xinyi	信義區	229139	11.2077	110
10	Zhongshan	中山區	231286	13.6821	104
11	Zhongzheng	中正區	162549	7.6071	100

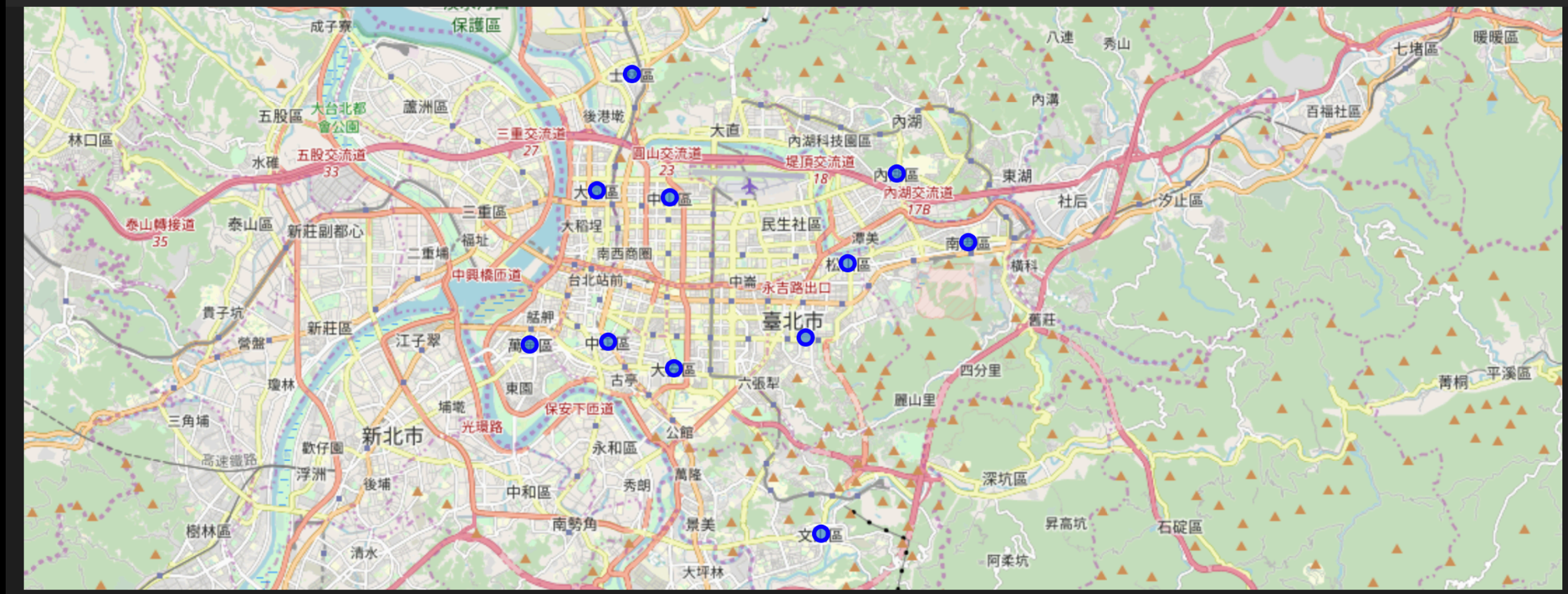
Methodology 2/17

Use geopy library to get the latitude and longitude values of each district for further use. Through geopy package, the latitude and longitude values of each district can be obtained.

	eng_name	cn_name	population	area_km2	postal_code	lat	lng
0	Beitou	北投區	257922	56.8216	112	25.131931	121.498593
1	Da'an	大安區	312909	11.3614	106	25.026515	121.534395
2	Datong	大同區	131029	5.6815	103	25.065986	121.515514
3	Nangang	南港區	122296	21.8424	115	25.054578	121.606600
4	Neihu	內湖區	287726	31.5787	114	25.069664	121.588998
5	Shilin	士林區	290682	62.3682	111	25.091840	121.524207
6	Songshan	松山區	209689	9.2878	105	25.049885	121.577272
7	Wanhua	萬華區	194314	8.8522	108	25.031933	121.499332
8	Wenshan	文山區	275433	31.5090	116	24.989786	121.570458
9	Xinyi	信義區	229139	11.2077	110	25.033345	121.566896
10	Zhongshan	中山區	231286	13.6821	104	25.064361	121.533468
11	Zhongzheng	中正區	162549	7.6071	100	25.032361	121.518267

Methodology 3/17

For visualization, create a map of Taipei City with districts superimposed on top.



Methodology 4/17

- Analyze the districts in Taipei - Data exploration

  Utilize the Foursquare API to explore the characteristics of each district and segment them for tourists.

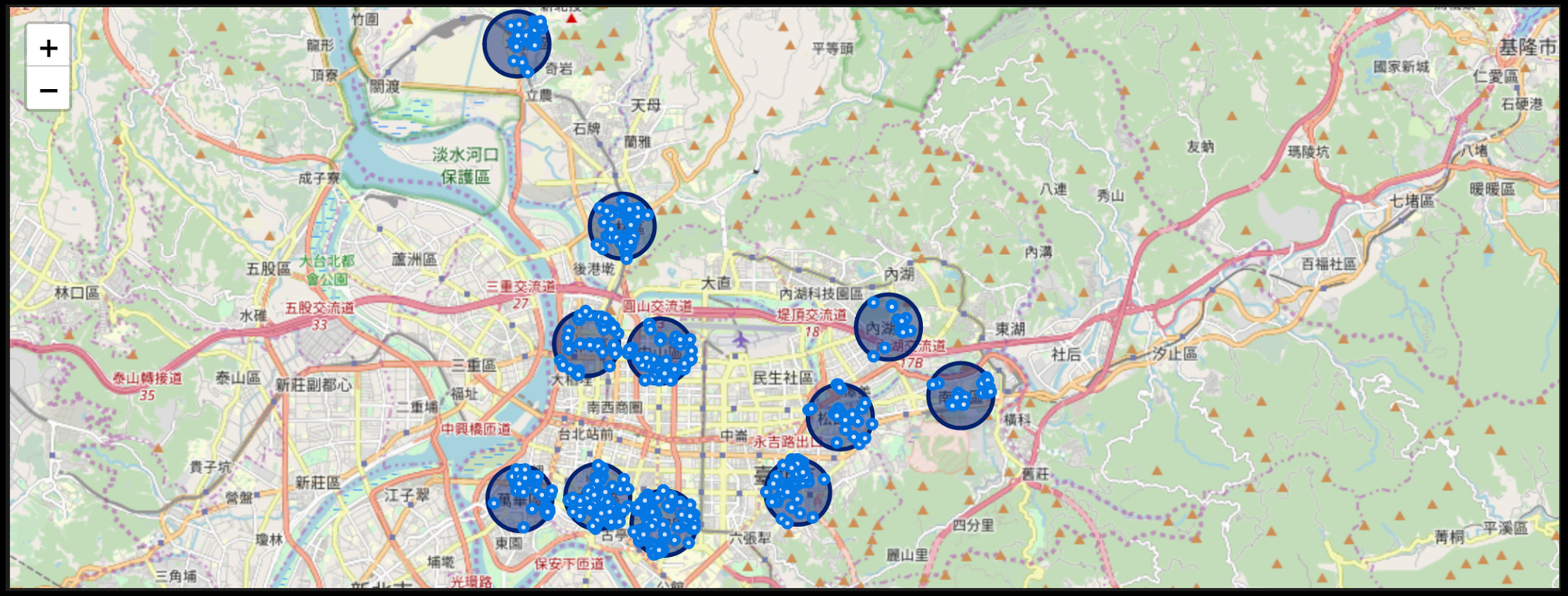
```
1. tp_venues.shape is (619, 6), that is, there are 619 venues returned.

2. There are 133 uniques categories.

3. Here is the amount of returned venues for each district:
```

district	v_name	v_lat	v_lng	v_dis	category
Beitou	34	34	34	34	34
Da'an	100	100	100	100	100
Datong	50	50	50	50	50
Nangang	29	29	29	29	29
Neihu	13	13	13	13	13
Shilin	68	68	68	68	68
Songshan	31	31	31	31	31
Wanhua	47	47	47	47	47
Wenshan	30	30	30	30	30
Xinyi	100	100	100	100	100
Zhongshan	56	56	56	56	56
Zhongzheng	61	61	61	61	61

```
There are  34 top venues searched within 800 meters in     Beitou district
There are 100 top venues searched within 800 meters in      Da'an district
There are  50 top venues searched within 800 meters in     Datong district
There are  29 top venues searched within 800 meters in    Nangang district
There are  13 top venues searched within 800 meters in     Neihu district
There are  68 top venues searched within 800 meters in     Shilin district
There are  31 top venues searched within 800 meters in   Songshan district
There are  47 top venues searched within 800 meters in     Wanhua district
There are  30 top venues searched within 800 meters in    Wenshan district
There are 100 top venues searched within 800 meters in      Xinyi district
There are  56 top venues searched within 800 meters in  Zhongshan district
There are  61 top venues searched within 800 meters in Zhongzheng district
```

Methodology 5/17

Create a map of Taipei City with the returned venues superimposed on top.



Check the dataframe returned.

	district	v_name	v_lat	v_lng	v_dis	category	category_2
0	Beitou	蔡元益紅茶（總店）	25.131896	121.502012	344	Tea Room	dessert shop
1	Beitou	阿馬非 Coffee. Pizza.pasta	25.132340	121.497882	85	Italian Restaurant	food shop
2	Beitou	傳統之最豆花堂	25.133637	121.498719	190	Dessert Shop	dessert shop
3	Beitou	Beitou Park (北投公園)	25.136605	121.504432	785	Park	tourist spot
4	Beitou	張吳記什錦麵	25.127777	121.500023	484	Noodle House	food shop

Methodology 6/17

- Analyze the districts in Taipei - Characteristic analysis of Taipei districts

  Next, create a function to convert some categories into expected types.

```
new_category = ['coffee shop', 'dessert shop', 'food shop', 'leisure place', 'hotel', 'shop', 'tourist
def cvtCategory( ori_category ):
    if any( ele in ori_category.lower() for ele in ['café', 'coffee shop'] ):
        ori_category = new_category[0]
    if any( ele in ori_category.lower() for ele in ['ice cream', 'snack', 'dessert shop', 'tea', 'bake'
        ori_category = new_category[1]
    if any( ele in ori_category.lower() for ele in ['restaurant', 'soup', 'noodle', 'breakfast', 'deli'
        ori_category = new_category[2]
    if any( ele in ori_category.lower() for ele in ['theater', 'playground', 'gym', 'laser tag', 'bowli
        ori_category = new_category[3]
    if any( ele in ori_category.lower() for ele in ['hotel', 'hostel', 'boarding house'] ):
        ori_category = new_category[4]
    if any( ele in ori_category.lower() for ele in ['duty-free', 'market', 'mall', 'store', 'plaza', 'l
        ori_category = new_category[5]
    if any( ele in ori_category.lower() for ele in ['museum', 'monument', 'planetarium', 'hall', 'histo
        ori_category = new_category[6]
    if any( ele in ori_category.lower() for ele in ['bar', 'club', 'lounge', 'speakeasy'] ):
        ori_category = new_category[7]
    return ori_category

# Convert category types and add into a new column
tp_venues['category_2'] = tp_venues[['category']].apply( lambda x: cvtCategory( *x ), axis = 1 )
```

Methodology 7/17

Do one-hot encoding for further analysis.

```
# One hot encoding
tp_onehot = pd.get_dummies( tp_df[['category_2']], prefix = "", prefix_sep = "" )

# Add district column back to dataframe
tp_onehot['district'] = tp_df['district']
tp_onehot['ori_category'] = tp_df['category']

# Move district and category column to the first and second columns
fixed_columns = [tp_onehot.columns[-2], tp_onehot.columns[-1]] + list( tp_onehot.columns[:-2] )
tp_onehot = tp_onehot[fixed_columns]

# Check the new dataframe tp_onehot
print( "tp_onehot.shape = {}".format( tp_onehot.shape ) )
tp_onehot.head()
```

```
tp_onehot.shape = (612, 10)
```

	district	ori_category	bar	coffee shop	dessert shop	food shop	hotel	leisure place	shop	tourist spot
0	Beitou	Tea Room	0	0	1	0	0	0	0	0
1	Beitou	Italian Restaurant	0	0	0	1	0	0	0	0
2	Beitou	Dessert Shop	0	0	1	0	0	0	0	0
3	Beitou	Park	0	0	0	0	0	0	0	1
4	Beitou	Noodle House	0	0	0	1	0	0	0	0

Methodology 8/17

Group rows by district and by taking the mean of the frequency of occurrence of each category which specified by us previously.

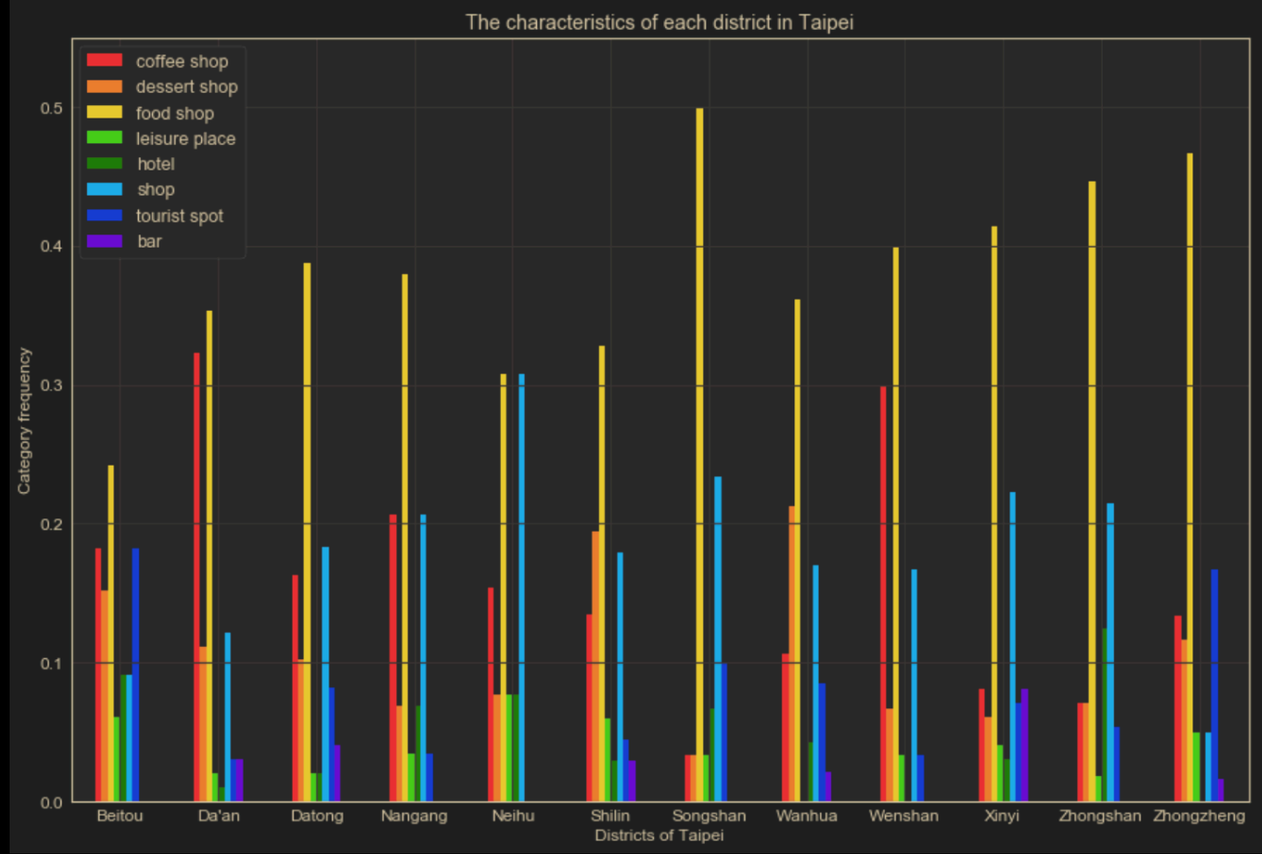
```
tp_grouped = tp_onehot.groupby( 'district' ).mean().reset_index()

# Add venue density for further use
tp_grouped['venue_density'] = ( tp_venues.groupby( 'district' ).count()['v_name'] / ( ( RADIUS * 0.001
tp_grouped
#tp_venues.groupby( 'district' ).count()['v_name'] / ( ( RADIUS * 0.001 ) * ( RADIUS * 0.001 ) )
```

	district	bar	coffee shop	dessert shop	food shop	hotel	leisure place	shop	tourist spot	venue_density
0	Beitou	0.000000	0.181818	0.151515	0.242424	0.090909	0.060606	0.090909	0.181818	69.387755
1	Da'an	0.030303	0.323232	0.111111	0.353535	0.010101	0.020202	0.121212	0.030303	204.081633
2	Datong	0.040816	0.163265	0.102041	0.387755	0.020408	0.020408	0.183673	0.081633	102.040816
3	Nangang	0.000000	0.206897	0.068966	0.379310	0.068966	0.034483	0.206897	0.034483	59.183673
4	Neihu	0.000000	0.153846	0.076923	0.307692	0.076923	0.076923	0.307692	0.000000	26.530612
5	Shilin	0.029851	0.134328	0.194030	0.328358	0.029851	0.059701	0.179104	0.044776	138.775510
6	Songshan	0.000000	0.033333	0.033333	0.500000	0.066667	0.033333	0.233333	0.100000	63.265306
7	Wanhua	0.021277	0.106383	0.212766	0.361702	0.042553	0.000000	0.170213	0.085106	95.918367
8	Wenshan	0.000000	0.300000	0.066667	0.400000	0.000000	0.033333	0.166667	0.033333	61.224490
9	Xinyi	0.080808	0.080808	0.060606	0.414141	0.030303	0.040404	0.222222	0.070707	204.081633
10	Zhongshan	0.000000	0.071429	0.071429	0.446429	0.125000	0.017857	0.214286	0.053571	114.285714
11	Zhongzheng	0.016667	0.133333	0.116667	0.466667	0.000000	0.050000	0.050000	0.166667	124.489796

Methodology 9/17

After grouping the data, we can use the bar plot to visualize the frequency distribution of venue categories for each district.

```
colors = ['#ea3333','#ea7f33','#eacb33','#49cc21', '#287a0f','#1cacea','#173cd3','#6b0cd1']
ax = tp_grouped[new_category].plot( kind = 'bar',
                                    title = "The characteristics of each district in Taipei",
                                    figsize = ( 15, 10 ),
                                    legend = True,
                                    fontsize = 12,
                                    color = colors )
ax.set_xlabel( "Districts of Taipei", fontsize = 12 )
ax.set_ylabel( "Category frequency", fontsize = 12 )
ax.set_xticklabels( tp_grouped['district'], rotation = 0, fontsize = 12 )
ax.margins( 0.1 )
plt.show()
```



Methodology 10/17

* Analyze the districts in Taipei - K-means clustering

  Run k-means to cluster the neighborhood into 3 clusters

```
from sklearn.cluster import KMeans
K_CLUSTER = 3
kmeans = KMeans( n_clusters = K_CLUSTER, random_state = 0 ).fit( tp_grouped[new_category + ['venue_dens
# check cluster labels generated for each row in the dataframe
kmeans.labels_
```

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

Methodology 11/17

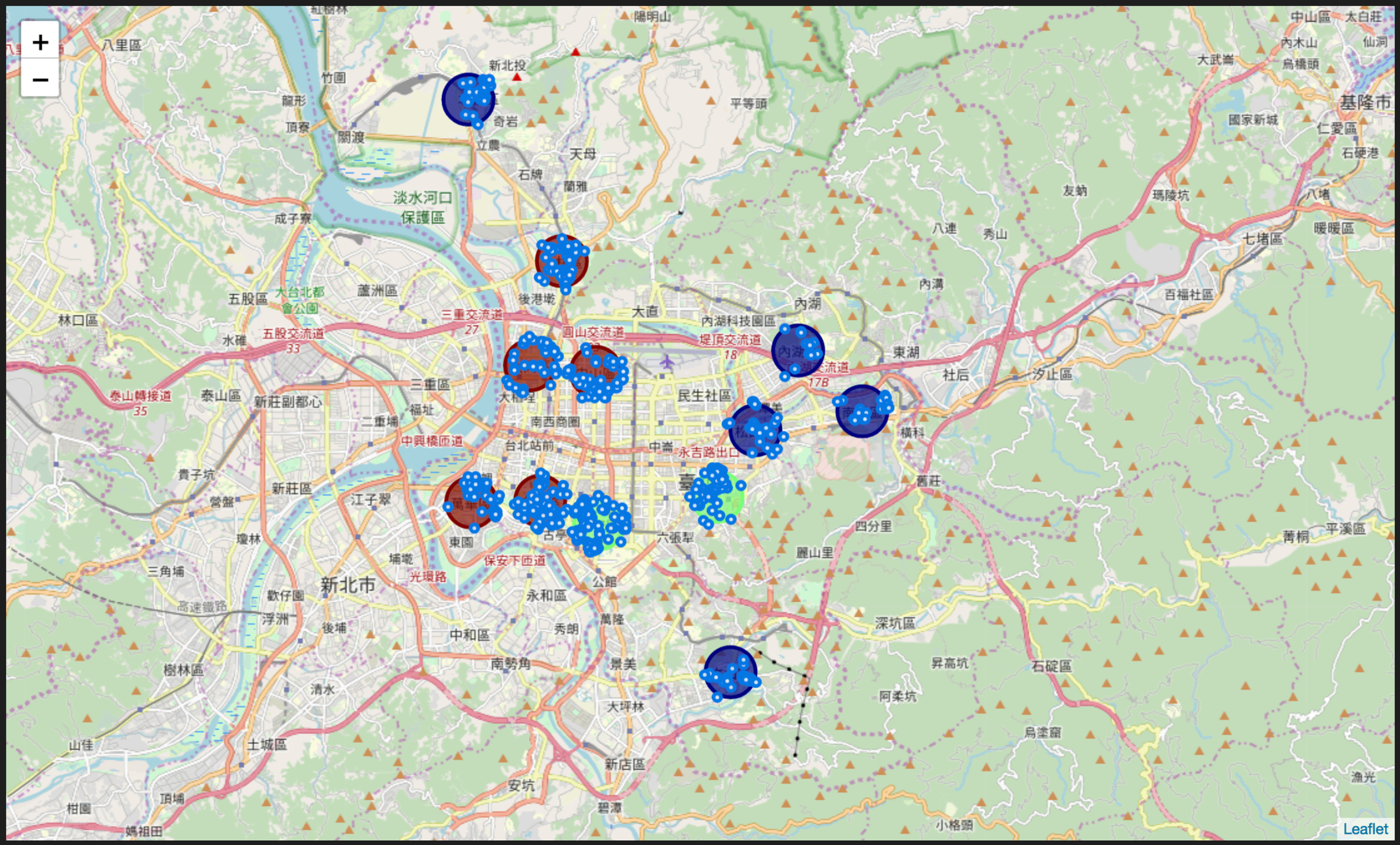
Analyze the districts in Taipei - Data aggregation

Sort venues for districts respectively and join the analyzed data into the first dataframe df as df_joined. Also add clustering results by k-means into df_joined.

```
# Prepare the new column names
temp_columns = ['district']
for idx in np.arange( len( new_category ) ):
    temp_columns.append( '{}_common_venue'.format( idx + 1 ) )

# Create a function to sort venues for Taipei districts respectively, and return lists of lists structure
def sort_rows_by_freq( data, columns ):
    res_list = []
    for idx in range( 0, data.shape[0], 1 ):
        v_sorted = data.loc[[idx], columns].sort_values( by = idx, ascending = False, axis = 1 )
        # Append the distrct name and its sorted venues
        res_list.append( list( data.loc[[idx], 'district'].values ) + v_sorted.columns.tolist() )
    return res_list

# Execute venue sorting and store into a dataframe
temp_data = sort_rows_by_freq( tp_grouped, new_category )
temp_df = pd.DataFrame( temp_data, columns = temp_columns )
df_joined = df.join( temp_df.set_index( 'district' ), on = 'eng_name' )

# Add clustering labels
df_joined['cluster_label'] = kmeans.labels_

# Add venue density
df_joined['venue_density'] = tp_grouped['venue_density'].values
```

Methodology 12/17

Visualize the map with the clustering results.



Methodology 13/17

- Recommend the ordered accommodations from the aspect of the specified attributes

Check the data

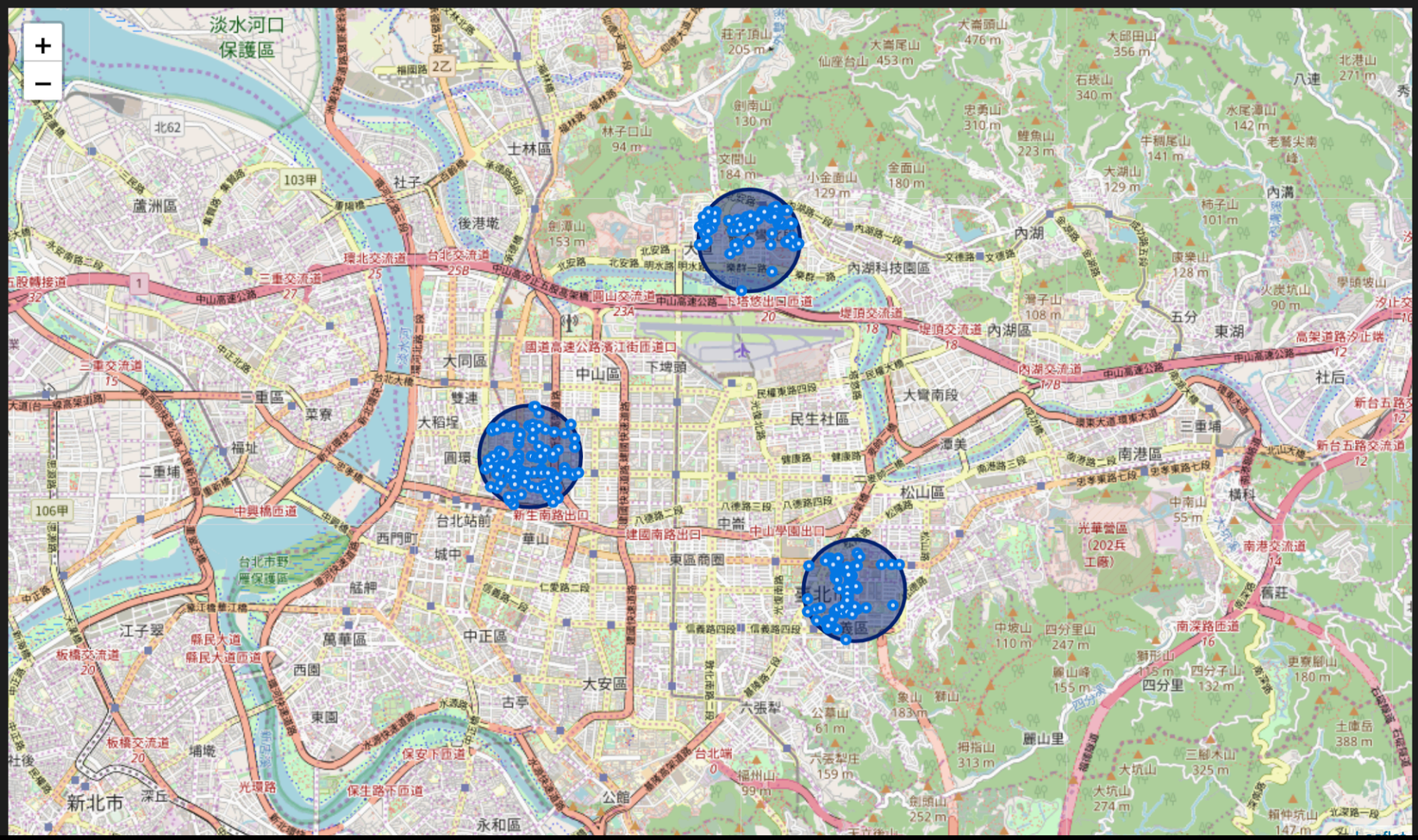
```
print( "Attribute: {}".format( new_category ) )

# Read address from the given text file
with open( 'accommodation_list.txt', 'r') as f:
    content = f.readlines()
addresses = [x.strip().replace( "No.", "" ).\
                       replace( "Rd.", "Road" ).\
                       replace( "Dist.", "District" ).\
                       replace( ", R.O.C.", "" ) for x in content]
print( "Accommodation: {}".format( addresses ) )
```

```
Attribute: ['coffee shop', 'dessert shop', 'food shop', 'leisure place', 'hotel', 'shop', 'tourist spot',
'bar']
Accommodation: ['38, Songren Road, Xinyi District, Taipei City Taiwan', '55, Lequn 2nd Road, Zhongshan Dist
rict, Taipei 10462, Taiwan', 'Regent Taipei']
```

Methodology 14/17

Find and show venues pertaining to the accommodation list.

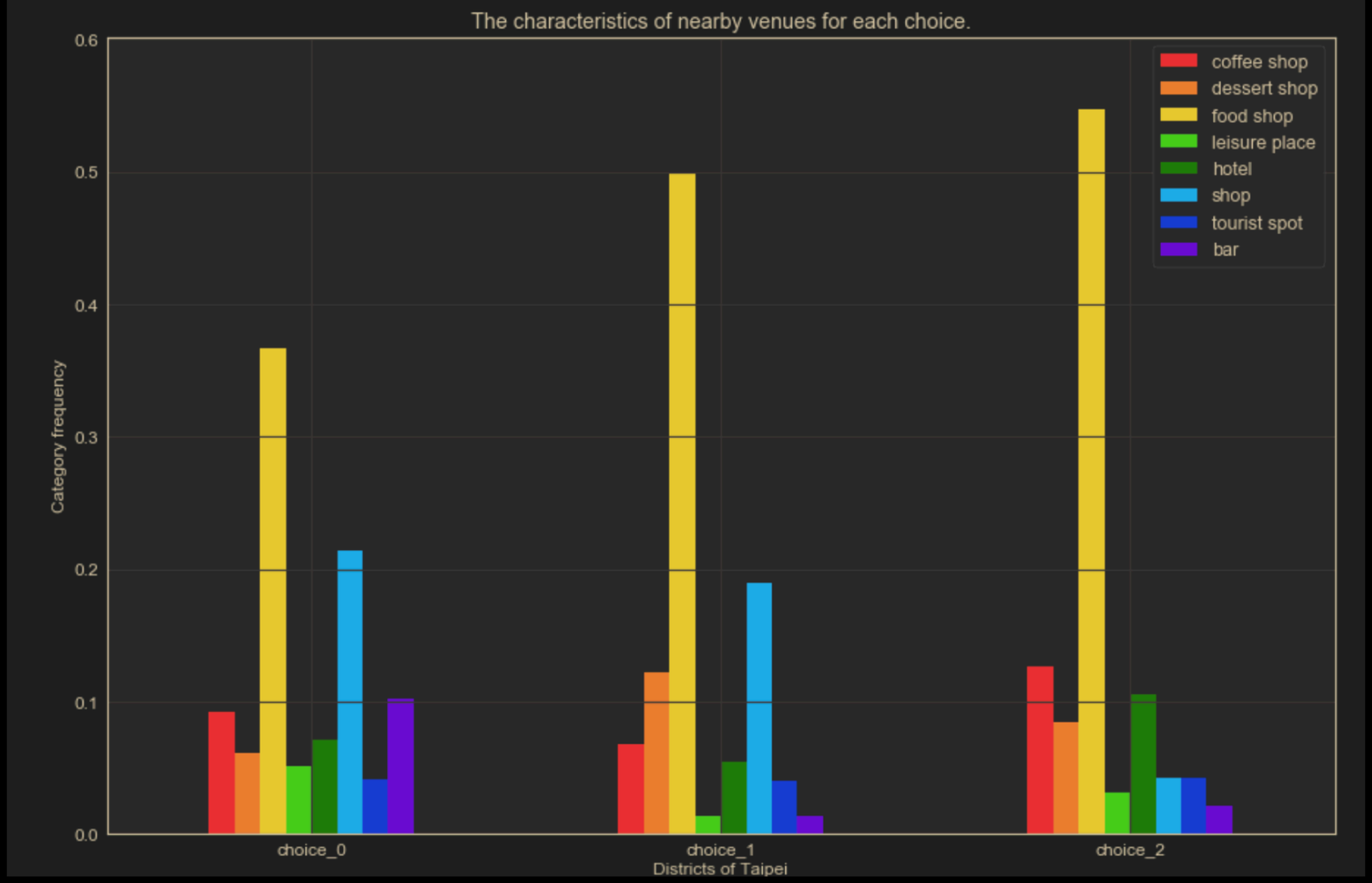
```
The geograpical coordinate of '38, Songren Road, Xinyi District, Taipei City Taiwan' are 25.0378219, 121.5
67956116648
The geograpical coordinate of '55, Lequn 2nd Road, Zhongshan District, Taipei 10462, Taiwan' are 25.080545
4, 121.553787376954
The geograpical coordinate of 'Regent Taipei' are 25.05417225, 121.524227099279
There are 100 top venues searched within 700 meters in choice_0
There are  82 top venues searched within 700 meters in choice_1
There are 100 top venues searched within 700 meters in choice_2
```



Methodology 15/17

Analyze nearby venues pertaining to the accommodation list.

```
There are 90 unique categories in column 'category'
There are 18 unique categories in column 'category_2'.

The shape of original venue dataframe in expected category: (282, 7)
The shape of new venue dataframe in expected category: (267, 7)
v_onehot.shape = (267, 10)
```



Methodology 16/17

Aggregate venue data into a dataframe data pertaining to the accommodation list.

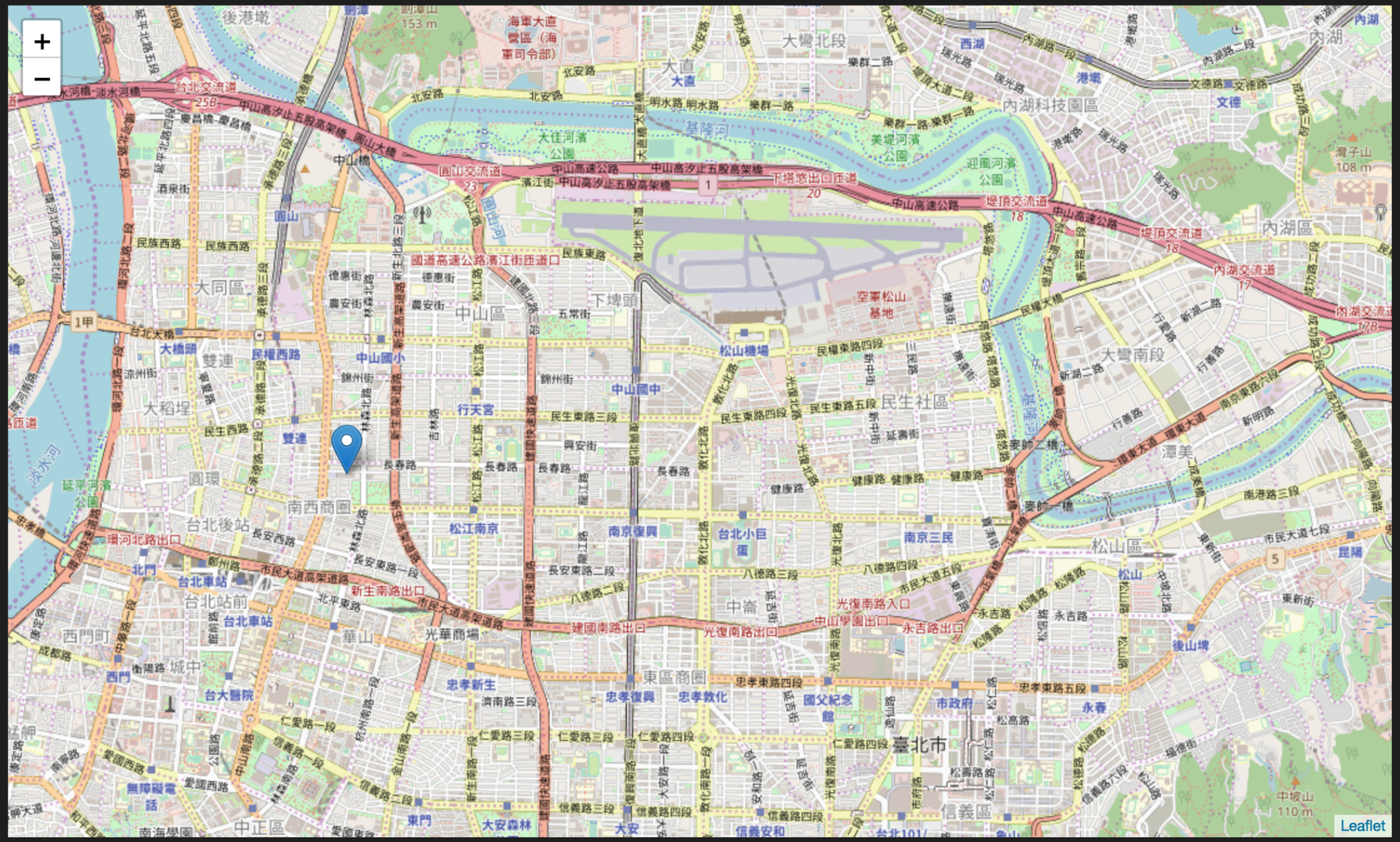
```
# Prepare the new column names
choice_columns = ['choice']
for idx in np.arange( len( new_category ) ):
    choice_columns.append( '{}_common_venue'.format( idx + 1 ) )

# Create a function to sort venues for Taipei districts respectively, and return lists of lists structur
def sort_rows_by_freq( data, columns ):
    res_list = []
    for idx in range( 0, data.shape[0], 1 ):
        data_sorted = data.loc[[idx], columns].sort_values( by = idx, ascending = False, axis = 1 )
        # Append the distrct name and its sorted venues
        res_list.append( list( data.loc[[idx], 'choice'].values ) + data_sorted.columns.tolist() )
    return res_list

# Execute venue sorting and store into a dataframe
v_temp_df = pd.DataFrame( sort_rows_by_freq( v_grouped, new_category ), columns = choice_columns )

# Join the sorted data into the original accommodation data_dict
v_df_joined = pd.DataFrame.from_dict( data_dict ).join( v_temp_df.set_index( 'choice' ), on = 'accommc
```

	accommodation_choice	address	latitude	longitude	1_common_venue	2_common_venue	3_common_venue	4_common_venue	5_common_venue	6_common_venue	7_common_venue	8_common_venue
0	choice_0	38, Songren Road, Xinyi District, Taipei City ...	25.037822	121.567956	food shop	shop	bar	coffee shop	hotel	dessert shop	leisure place	tourist spot
1	choice_1	55, Lequn 2nd Road, Zhongshan District, Taipei...	25.080545	121.553787	food shop	shop	dessert shop	coffee shop	hotel	tourist spot	leisure place	bar
2	choice_2	Regent Taipei	25.054172	121.524227	food shop	coffee shop	hotel	dessert shop	shop	tourist spot	leisure place	bar

Methodology 17/17

Recommend an accommodation according to a user's preference.

```
Your favorite category: 'coffee shop',
we recommend: ['Regent Taipei']
```

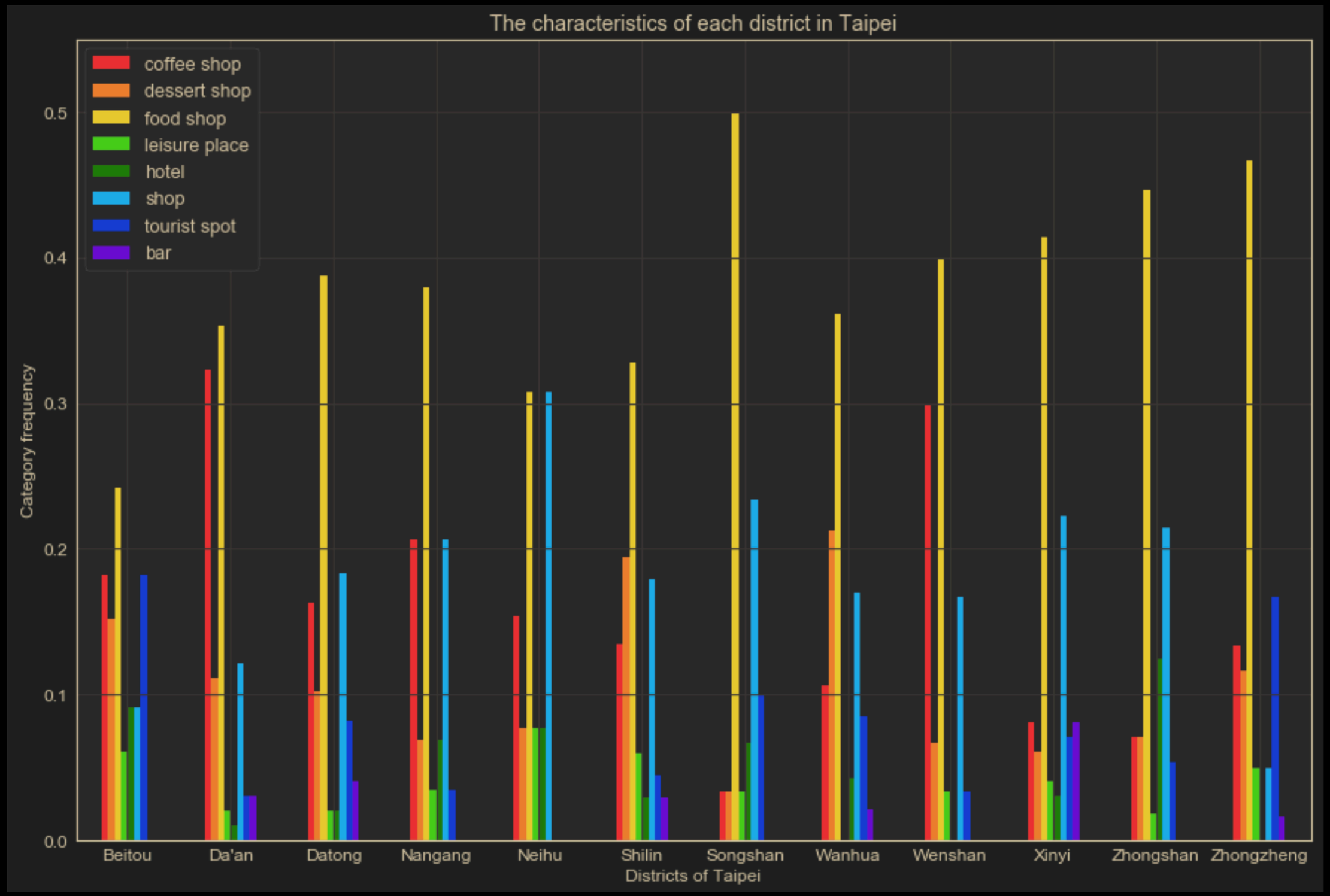


Result 1/5

1. Analyze the districts in Taipei - Distinctive venue distribution of each district

	eng_name	venue_density	1_common_venue	2_common_venue	3_common_venue	4_common_venue	5_common_venue	6_common_venue	7_common_venue	8_common_venue
0	Beitou	69.387755	food shop	coffee shop	tourist spot	dessert shop	hotel	shop	leisure place	bar
1	Da'an	204.081633	food shop	coffee shop	shop	dessert shop	tourist spot	bar	leisure place	hotel
2	Datong	102.040816	food shop	shop	coffee shop	dessert shop	tourist spot	bar	leisure place	hotel
3	Nangang	59.183673	food shop	coffee shop	shop	dessert shop	hotel	leisure place	tourist spot	bar
4	Neihu	26.530612	food shop	shop	coffee shop	dessert shop	leisure place	hotel	tourist spot	bar
5	Shilin	138.775510	food shop	dessert shop	shop	coffee shop	leisure place	tourist spot	hotel	bar
6	Songshan	63.265306	food shop	shop	tourist spot	hotel	coffee shop	dessert shop	leisure place	bar
7	Wanhua	95.918367	food shop	dessert shop	shop	coffee shop	tourist spot	hotel	bar	leisure place
8	Wenshan	61.224490	food shop	coffee shop	shop	dessert shop	leisure place	tourist spot	hotel	bar
9	Xinyi	204.081633	food shop	shop	coffee shop	bar	tourist spot	dessert shop	leisure place	hotel
10	Zhongshan	114.285714	food shop	shop	hotel	coffee shop	dessert shop	tourist spot	leisure place	bar
11	Zhongzheng	124.489796	food shop	tourist spot	coffee shop	dessert shop	leisure place	shop	bar	hotel

Result 2/5


The characteristics of each district in Taipei
coffee shop
dessert shop
food shop
leisure place
hotel
shop
tourist spot
bar
Category frequency
0.0
0.1
0.2
0.3
0.4
0.5
Beitou
Da'an
Datong
Nangang
Neihu
Shilin
Songshan
Wanhua
Wenshan
Xinyi
Zhongshan
Zhongzheng
Districts of Taipei

Result 3/5

2. Analyze the districts in Taipei - Cluster 1 for Taipei

	eng_name	cluster_label	venue_density	1_common_venue	2_common_venue	3_common_venue	4_common_venue	5_common_venue	6_common_venue	7_common_venue	8_common_venue
2	Datong	0	102.040816	food shop	shop	coffee shop	dessert shop	tourist spot	bar	leisure place	hotel
5	Shilin	0	138.775510	food shop	dessert shop	shop	coffee shop	leisure place	tourist spot	hotel	bar
7	Wanhua	0	95.918367	food shop	dessert shop	shop	coffee shop	tourist spot	hotel	bar	leisure place
10	Zhongshan	0	114.285714	food shop	shop	hotel	coffee shop	dessert shop	tourist spot	leisure place	bar
11	Zhongzheng	0	124.489796	food shop	tourist spot	coffee shop	dessert shop	leisure place	shop	bar	hotel

3. Analyze the districts in Taipei - Cluster 2 for Taipei

	eng_name	cluster_label	venue_density	1_common_venue	2_common_venue	3_common_venue	4_common_venue	5_common_venue	6_common_venue	7_common_venue	8_common_venue
0	Beitou	1	69.387755	food shop	coffee shop	tourist spot	dessert shop	hotel	shop	leisure place	bar
3	Nangang	1	59.183673	food shop	coffee shop	shop	dessert shop	hotel	leisure place	tourist spot	bar
4	Neihu	1	26.530612	food shop	shop	coffee shop	dessert shop	leisure place	hotel	tourist spot	bar
6	Songshan	1	63.265306	food shop	shop	tourist spot	hotel	coffee shop	dessert shop	leisure place	bar
8	Wenshan	1	61.224490	food shop	coffee shop	shop	dessert shop	leisure place	tourist spot	hotel	bar

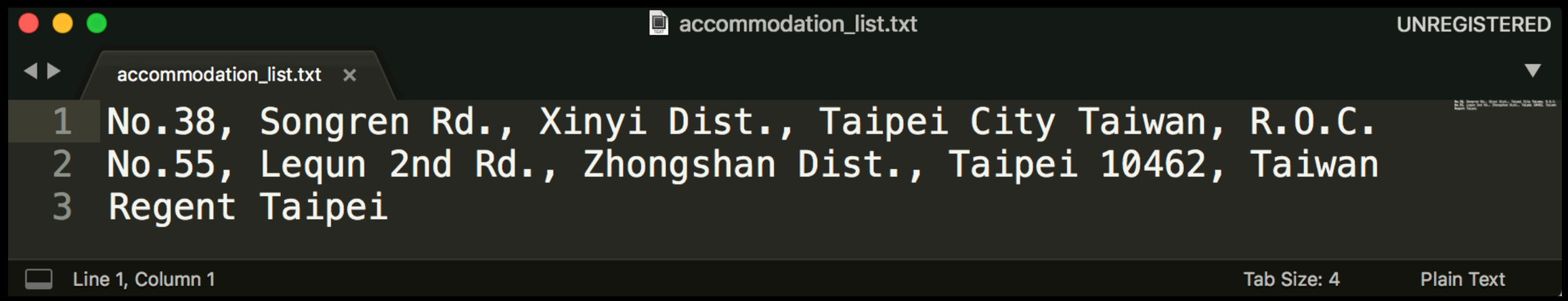
4. Analyze the districts in Taipei - Cluster 3 for Taipei

	eng_name	cluster_label	venue_density	1_common_venue	2_common_venue	3_common_venue	4_common_venue	5_common_venue	6_common_venue	7_common_venue	8_common_venue
1	Da'an	2	204.081633	food shop	coffee shop	shop	dessert shop	tourist spot	bar	leisure place	hotel
9	Xinyi	2	204.081633	food shop	shop	coffee shop	bar	tourist spot	dessert shop	leisure place	hotel

Result 4/5

5. Recommend the ordered accommodations from the aspect of the specified attributes

with an accommodation list:



with an attribute list to choose:

```
['coffee shop', 'dessert shop', 'food shop', 'leisure place', 'hotel', 'shop', 'tourist spot', 'bar']
```

Result 5/5

```
# With the accommodation list: accommodation_list.txt
# new_category:
#  ['coffee shop', 'dessert shop', 'food shop', 'leisure place', 'hotel', 'shop', 'tourist spot', 'bar']
for item in new_category:
    res_df = getRecommendation( item )
```

```
Your favorite category: 'coffee shop',
we recommend: ['Regent Taipei']

Your favorite category: 'dessert shop',
we recommend: ['55, Lequn 2nd Road, Zhongshan District, Taipei 10462, Taiwan']

Your favorite category: 'food shop',
we recommend: ['38, Songren Road, Xinyi District, Taipei City Taiwan'
 '55, Lequn 2nd Road, Zhongshan District, Taipei 10462, Taiwan'
 'Regent Taipei']

Your favorite category: 'leisure place',
we recommend: ['Regent Taipei']

Your favorite category: 'hotel',
we recommend: ['Regent Taipei']

Your favorite category: 'shop',
we recommend: ['38, Songren Road, Xinyi District, Taipei City Taiwan'
 '55, Lequn 2nd Road, Zhongshan District, Taipei 10462, Taiwan']

Your favorite category: 'tourist spot',
we recommend: ['55, Lequn 2nd Road, Zhongshan District, Taipei 10462, Taiwan']

Your favorite category: 'bar',
we recommend: ['38, Songren Road, Xinyi District, Taipei City Taiwan']
```

Discussion 1/2

- We can found Datong, Shilin, Wanhua, Zhongshan, and Zhongzheng district are similar, which are classified as cluster 1.

- We can found Beitou, Nangong, Neihu, Songshan, and Wenshan district are similar, which are classified as cluster 2.

- We can found Da'an, and Xinyi district are similar, which are classified as cluster 3.

- From the clustering results, the parameter 'venue_density' have great impact on the clustering results can be observed. The values of cluster 1 is around 115, around 55 for cluster 2, and around 204 for cluster 3.

Discussion 2/2

- For the first objective, we can discover that all districts in Taipei are full of food to eat, which is in line with reality on purchasing food commodities in Taipei City, since the first common venue is "food shop" for all of them. Moreover, we also can found that the category "bar" is less than other category relatively in Taipei. If travelers who prefer walking along colorful markets, from the analysis results, Da'an or Xinyi district is recommended. While if travelers who focus on tourist spot, Zhongzheng, Beitou, or Songshan district may be more suitable for them comparing to other districts.

- For the second objective, we can input a list of accommodations in Taipei City, and then a suggestion based on our preference and the accommodation list will be shown.

Conclusion

With Pandas, Geopy, Foursquare APIs, we can capture quiet a few useful information to explore the characteristics of a region conveniently and achieve the goals of this project as follows:

1. Analyze the districts in Taipei
2. Recommend the ordered accommodations from the aspect of the specified attributes.

Moreover, Folium and Matplotlib are practical tools to visualize the structured data, which can be observed in the previous demonstrations.

As for the district analysis of Taipei City, the characteristics of each district and the results of K-means clustering are in line with the residents' perceptions even the data collected from a Wiki page and Foursquare database only. Hope in the future, the content of this project can be reorganized and modulized in a more systematic way for further use.

Thanks