IBM Capstone Project — The Battle of Neighborhoods in Berlin: Restaurants

Location options for opening a restaurant in Berlin

-----based on Venues Data Analysis in all neighborhoods of Berlin

As a part of the IBM Data Science Professional Certificate, you will find in this post an overview of my final capstone project.

As prepared for the assignment, I go through the problem description, data preparation and final analysis section step by step. Detailed codes are given in Github and link can be found at the end of the post.

Lingyan Fu

14.08.2020

Berlin

1. Introduction/Business Understanding

1.1 Descrption of the problem

The business problem we are currently posing is: At the end of the summer when the epidemic is facing a second outbreak in 2020, Opening which type of restaurants in which neighborhood of Berlin can be the most profitable choice for investors?

1.2 Discussion of the background

As the capital and biggest city of Germany, the second most populous city in the European Union, Berlin has nearly 3.8 million residents from more than 190 countries with a population density of 4,200 people per km², the ciry is divided into 12 boroughs, 95 neighborhoods.

From the data, we can see that Istanbul is a global ciry with high population and population density, but the special layout of the city of Berlin is, one-third of the city's land consists of forests, parks, gardens, rivers and lakes, commercial availability Lower than the EU average. When we consider the ideas of investors in the catering industry, we hope to give them suggestions that the cost of opening a restaurant is low and the types of restaurants they want to open are not so intense (the same type of restaurant density is low). . At the same time, they may wish to choose a region based on the density of social venues. But at the moment, it is difficult for us to obtain information that can guide investors in the catering industry in this direction.

Even though the world is facing the Coronavirus crisis, Berlin welcomed 700,000 tourists in July.2020 (only 60% of the same period in 2019), and it is expected that there will still be in August and September. More than 600,000 tourists visit Berlin, which means that the number of tourists per month accounts for 15%-20% of the population. There are **** restaurants in Berlin, as showed in restaurant category there are more than 64 kinds. How to capture the needs of local residents and tourists for restaurants based on the region?
When we comprehensively consider all this problem, we can solve this problem by creating a map and information chart that shows the real distribution of local restaurants in Berlin and clustering each area according to the density of the place.

So, how do we use Foursquare location data and machine learning to help us make decisions for diners and investors? I want to solve this problem in this pinnacle project taking Berlin as an example. In this project, I will use Foursquare location data and clustering methods to divide regions into different groups based on their restaurant location information.

2. Data requirements

For this project, we need the following data:

Berlin data, which contains the list of Boroughs and neighborhoods and their latitudes and longitudes.

Data source: https://en.wikipedia.org/wiki/Boroughs_and_neighborhoods_of_Berlin

Description: We will discard the Berlin area (district) table through Wikipedia and use the geocoder class of the Geopy client to obtain the coordinates of these 12 main areas.

Restaurants in each neighborhood in Berlin:

Data source: Foursquare API

Description: By using this API, we will obtain all venues in each community. We can filter these places to get only restaurants.

3. Methodology

3.1 Data Preparation

3.1.1 Scraping Berlin Boroughs and neighborhoods table from Wikipedia

I first make use of Boroughs and neighborhoods of Berlin from Wiki to scrap the table to create a data-frame. For this, I’ve used pandas to transform the data in the table on the Wikipedia page into a dataframe containing name of the 96 neighborhoods of Berlin, Area, population. After little manipulation, the data-fram is obtained as below:

```
In [25]: #set new index for df
df=df.reset_index(drop=True)
df
```

Out[25]:

	Neighborhood
0	Mitte (locality)
1	Moabit
2	Hansaviertel
3	Tiergarten (Berlin)
4	Wedding (Berlin)
5	Gesundbrunnen (Berlin)
6	Friedrichshain
7	Kreuzberg
8	Prenzlauer Berg
9	Weißensee (Berlin)
10	Blankenburg (Berlin)

3.1.2 Getting Coordinates of 96 neighborhoods : Geopy Client

Next objective is to get the coordinates of these 96 neighborhoods using geocoder class of Geopy client as follow:

```
35]: # define a function to get coordinates
def get_latlng(neighborhood):
    # initialize your variable to None
    lat_lng_coords = None
    # loop until you get the coordinates
    while(lat_lng_coords is None):
        g = geocoder.arcgis('{}, Berlin, Germany'.format(neighborhood))
        lat_lng_coords = g.latlng
    return lat_lng_coords
# call the function to get the coordinates, store in a new list using list comprehension
coords = [ get_latlng(neighborhoodlist) for neighborhoodlist in df["Neighborhood"].tolist() ]
coords
```

```
print(len(coords))
df_coords = pd.DataFrame(coords, columns=['Latitude', 'Longitude'])
df['Latitude'] = df_coords['Latitude']
df['Longitude'] = df_coords['Longitude']
print(df.shape)
df
```

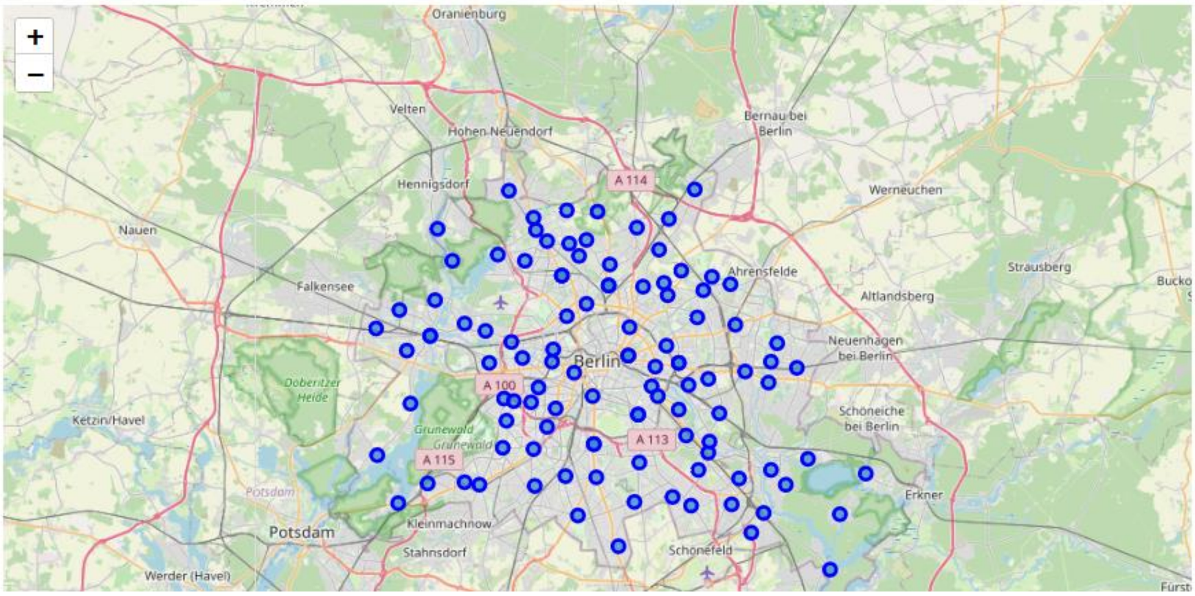
```
96
(96, 3)
```

]:

	Neighborhood	Latitude	Longitude
0	Mitte (locality)	52.52119	13.42414
1	Moabit	52.52570	13.34005
2	Hansaviertel	52.51679	13.33835
3	Tiergarten (Berlin)	52.50993	13.36393
4	Wedding (Berlin)	52.54781	13.35473
5	Gesundbrunnen (Berlin)	52.55619	13.37710
6	Friedrichshain	52.51402	13.45403
7	Kreuzberg	52.49382	13.38358
8	Prenzlauer Berg	52.54083	13.42575

I used python **folium** library to visualize geographic details of Berlin and its 96 neighborhoods and I created a map of Berlin with boroughs superimposed on top. I used latitude and longitude values to get the visual as below:

```
map_berlin = folium.Map(location=[latitude, longitude], zoom_start=10)
# add markers to map
for lat, lng, neighborhood in zip(df['Latitude'], df['Longitude'], df['Neighborhood']):
    label = '{}'.format(neighborhood)
    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_berlin)

map_berlin
```



3.2. Exploratory Data Analysis: Using Foursquare Location Data

Firstly, I will use exploratory data analysis (EDA) to uncover hidden properties of data and provide useful insights to the reader, both future diners and investors.

Finally, let's make use of Foursquare API and get the top 200 venues that are in Berlin within a radius of 2000 meters, And extract the rows containing "Restaurant" in the VenueCatergory column. How we got the radius of 2000 meters: Total area of Berlin/96 neighborhoods, Then enter the area formula to find the radius:1700 meter. I conclude that the total number of restaurants in Berlin is 1,246, which is significantly lower than the real level, but the data has sufficient sample size for data analysis for reference.

(1) We notice that 45 unique venue categories were returned by Foursquare, Italian and German Restaurant in the top of the list as we can see above.

```
#extract the rows containing "Restaurant" in the VenueCatergory column.
print('There are {} uniques categories.'.format(len(venues_df['VenueCategory'].unique())))
```

```
There are 63 uniques categories.
```

```
#各个餐厅种类各有多少餐厅
#How many restaurants are there for each type of restaurant-catogety
venues_df.loc[:,'VenueCategory'].value_counts()
```

```
7]: Italian Restaurant                 231
German Restaurant                  153
Restaurant                          69
Vietnamese Restaurant               61
Greek Restaurant                    60
Fast Food Restaurant                56
Asian Restaurant                    49
Chinese Restaurant                  48
Indian Restaurant                   47
Falafel Restaurant                  44
Thai Restaurant                     35
Sushi Restaurant                    32
Doner Restaurant                    31
Vegetarian / Vegan Restaurant       30
Mexican Restaurant                  21
Turkish Restaurant                  21
Middle Eastern Restaurant           20
Japanese Restaurant                 19
```

So, the first finding is:

Six of most popular type of restaurants in Berlin are: Italian, German, Vietnamese, Greek and fast food.

(2) Let's get the top ten neighborhoods in total restaurants

```
#pick top 10 neighborhoods
top_10_venues_df_restaurant_neighborhood=venues_df_restaurant_neighborhood[0:10]
top_10_venues_df_restaurant_neighborhood
```

]:

Neighborhood	Latitude	Longitude	VenueName	VenueLatitude	VenueLongitude	VenueCategory
Charlottenburg	41	41	41	41	41	41
Grunewald	38	38	38	38	38	38
Friedenau	36	36	36	36	36	36
Halensee	35	35	35	35	35	35
Steglitz	32	32	32	32	32	32
Lichtenberg (locality)	32	32	32	32	32	32
Wilmersdorf	32	32	32	32	32	32
Kreuzberg	32	32	32	32	32	32
Prenzlauer Berg	32	32	32	32	32	32
Wedding (Berlin)	28	28	28	28	28	28

So, the second finding is:

Top ten neighborhoods in total restaurant are: Charlottenburg, Grunewald, Firedenau, Halensee, Steglitz, Lichtenberg, Wilmersdor, Kreuzberg, Prenzlauer Berg and Wedding.

So, we proceed as follows:

First, create a data-frame with pandas one hot encoding for the venue categories.

```
#Let’s analyze each neighborhood to know about the top 5 venues of each one.
#one hot encoding
bl_onehot = pd.get_dummies(venues_df_restaurant[['VenueCategory']], prefix="", prefix_sep="")

# add neighborhood column back to dataframe
bl_onehot['Neighborhoods'] = venues_df_restaurant['Neighborhood']

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

print(bl_onehot.shape)
bl_onehot.head()
```

```
(1246, 64)
```

	Neighborhoods	African Restaurant	American Restaurant	Argentinian Restaurant	Asian Restaurant	Austrian Restaurant	Brazilian Restaurant	Cantonese Restaurant	Caribbean Restaurant	Caucasian Restaurant	Chinese Restaurant	Cuban Restaurant
1	Mitte (locality)	0	0	0	0	0	0	0	0	0	0	0
31	Mitte (locality)	0	0	0	0	0	0	0	0	0	0	0
33	Mitte (locality)	0	0	0	0	0	0	0	0	0	0	0
40	Mitte (locality)	0	0	0	0	1	0	0	0	0	0	0
41	Mitte (locality)	0	0	0	0	0	0	0	0	0	0	0

Second, use pandas groupby on neighborhood column and calculate the mean of the frequency of occurrence of each venue category.

```
#Use pandas groupby on neighborhood column and calculate the mean of the frequency of occurrence of each venue categ
#在邻域列上使用pandas groupby并计算每个场所类别出现频率的平均值。
bl_grouped=bl_onehot.groupby("Neighborhoods").mean().reset_index()
bl_grouped
```

]:

	Neighborhoods	African Restaurant	American Restaurant	Argentinian Restaurant	Asian Restaurant	Austrian Restaurant	Brazilian Restaurant	Cantonese Restaurant	Caribbean Restaurant	Caucasian Restaurant	Chinese Restaurant	Cu Restau
0	Adlershof	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
1	Alt-Hohenschönhausen	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
2	Alt-Treptow	0.055556	0.000000	0.000000	0.055556	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
3	Altglienicke	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
4	Baumschulenweg	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.125000	0.000
5	Biesdorf (Berlin)	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
6	Blankenburg (Berlin)	0.000000	0.000000	0.000000	0.333333	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
7	Bohnsdorf	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000

Third,

```
#Output each neighborhood along with the top 5 most common venues:
num_top_venues=5

for hood in bl_grouped['Neighborhoods']:
    print('---'+hood+'---')
    temp=bl_grouped[bl_grouped['Neighborhoods']==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(num_top_venues))
    print('\n')
```

```
---Adlershof---
                venue  freq
0  German Restaurant   0.3
1   Sushi Restaurant   0.2
2   Greek Restaurant   0.2
3         Restaurant   0.1
4  Korean Restaurant   0.1


---Alt-Hohenschönhausen---
                   venue  freq
0       Greek Restaurant  0.29
1      German Restaurant  0.29
2  Fast Food Restaurant  0.29
3      Indian Restaurant  0.14
4     African Restaurant  0.00


---Alt-Treptow---
```

I will use *prescriptive analytics* to help an investor decide a location and VenueCategory to open a restaurant. I will use *clustering* (KMeans).

Finally, we try to cluster these 96 neighborhoods based on the venue categories and use K-Means clustering. So, our expectation would be based on the similarities of venue categories, these districts will be clustered. I have used the code below:

```
#Run k-means to cluster the neighborhoods in Kuala Lumpur into 3 clusters.

# set number of clusters
kclusters = 5

bl_clustering = bl_grouped.drop(["Neighborhoods"], 1)

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

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

```
]: array([0, 0, 0, 0, 0, 1, 2, 1, 0, 0])
```

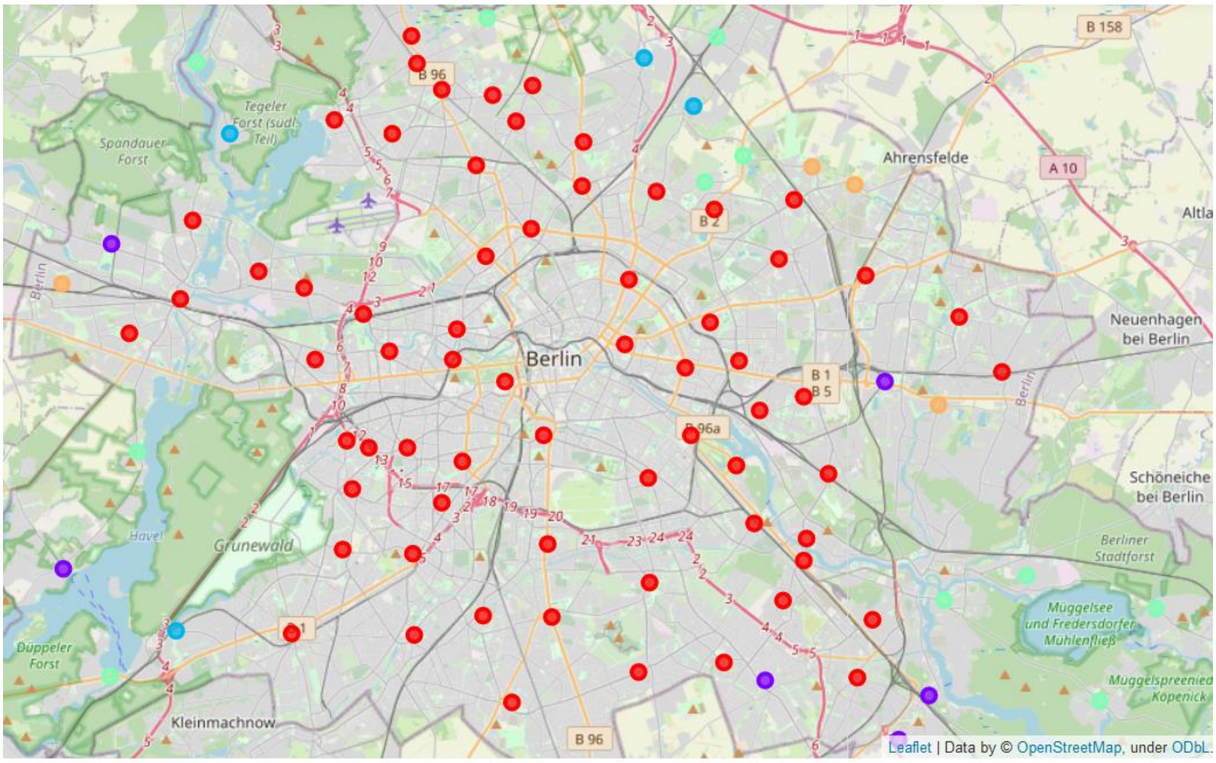
```python
# merge toronto_grouped with toronto_data to add latitude/longitude for each neighborhood
bl_merged = bl_merged.join(df.set_index("Neighborhood"), on="Neighborhood")

print(bl_merged.shape)
bl_merged.head() # check the last columns!
```

```
(95, 67)
```

]:

ian ant	Szechuan Restaurant	Tapas Restaurant	Thai Restaurant	Theme Restaurant	Turkish Home Cooking Restaurant	Turkish Restaurant	Vegetarian / Vegan Restaurant	Venezuelan Restaurant	Vietnamese Restaurant	Yemeni Restaurant	Cluster Labels	Latitude	Longitude
0.0	0.0	0.000000	0.000000	0.0	0.0	0.0	0.000000	0.0	0.000000	0.0	0	52.43779	13.54778
0.0	0.0	0.000000	0.000000	0.0	0.0	0.0	0.000000	0.0	0.000000	0.0	0	52.54706	13.50055
0.0	0.0	0.055556	0.055556	0.0	0.0	0.0	0.111111	0.0	0.111111	0.0	0	52.49350	13.45711
0.0	0.0	0.000000	0.000000	0.0	0.0	0.0	0.000000	0.0	0.000000	0.0	0	52.42006	13.53969
0.0	0.0	0.000000	0.000000	0.0	0.0	0.0	0.000000	0.0	0.125000	0.0	0	52.46669	13.48840

We can represent these 5 clusters in a leaflet map using Folium library as below:



4. Results & discussion

We got a glimpse of the Restaurants in Berlin and were able to find out some interesting insights which might be useful to investors who plan to open a reataurant in Berlin. Let’s summarize our findings:

· Italian restaurants top the charts of most common venues in 96 neighborhoods.

: Charlottenburg, Grunewald and Firedenau have maximum number of restaurants.

· Since the clustering was based only on the category of restaurants on each neighborhood, top-10-neighborhoods in total restaurant all fall in the same cluster 0, which indicate that each of those neighborhoods presents a similar experience to investors in terms of category of food.

· It’s also important to note that top-10-neighborhoods all fall within the Yamanote Line which make them accessible and easy to move between them.

· Französisch Buchholz, Wartenberg, Kladow, Konradshöhe and Falkenhagener Feld have 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 stations, range of prices of restaurants, Michelin Restaurants and so on, since we don’t have such data and it would be difficult to farm it for a small exploratory study like ours. Hence, our analysis only helps travelers to get an overview of Restaurants distribution by categories in the 96 neighborhoods of Tokyo.

5. Conclusion

In the current digital age, there are many real-life problems/cases. We can find corresponding solutions by searching for data-analyzing data. As seen in the example above, the data content is based on the distribution of the most common dining places (restaurants) in the Berlin 96 neighborhoods. The results of the analysis can help investors determine the most suitable areas for investment.

I used some commonly used python libraries to extract web data, used the Foursquare API to explore the main areas of Berlin, and used the Folium leaflet map to see the results of the region segmentation.

Similarly, data can also be used to solve other problems that most people face in large cities. The potential for such analysis in real life is discussed in detail. In addition, some shortcomings and improvement opportunities are mentioned to represent more realistic pictures.