2. Data acquisition and cleaning

2.1 Data sources

Data including Japanese postal code, city name, latitude and longitude can be obtained from the following.
https://www.aggdata.com/free/japan-postal-codes
However, this data includes data from all over Japan. Therefore, it is necessary to extract only the necessary part. In addition, the granularity of the data is so fine that multiple city names with the same latitude and longitude may be included.
The other data is regional information obtained from Foursquare.

2.2 Data Cleaning

There are several problems for the datasets.

	postalcode	Neighborhood	state	Borough	Latitude	Longitude
0	490-1401	Rokujocho	Aichi Ken	Yatomi Shi	34.9	137.15
1	490-1402	Gotoyama	Aichi Ken	Yatomi Shi	34.9	137.15
2	490-1403	Toriganjicho	Aichi Ken	Yatomi Shi	34.9	137.15
3	490-1403	Toriganji	Aichi Ken	Yatomi Shi	34.9	137.15
4	490-1404	Ikadaba	Aichi Ken	Yatomi Shi	34.9	137.15

```
df.shape
```

```
(123695, 6)
```

First, the dataset is huge because it contains all addresses in Japan, and there are 123695 rows. So, after making only the data whose "state" is "Tokyo to", I made only the data that "ku"(it means “ward”) was included in "borough".

Second, the data was very detailed and included multiple addresses with the same "longitude" and "latitude". Therefore, the duplication of “longitude" and "latitude" was dropped.

The following data set was obtained by the above processing.

	postalcode	Neighborhood	state	Borough	Latitude	Longitude
108164	130-0000	Ikanikeisaiganaibaai	Tokyo To	Sumida Ku	35.7068	139.8072
108165	130-0001	Azumabashi	Tokyo To	Sumida Ku	35.7096	139.8031
108166	130-0002	Narihira	Tokyo To	Sumida Ku	35.7079	139.8134
108167	130-0003	Yokokawa	Tokyo To	Sumida Ku	35.7048	139.8156
108168	130-0004	Honjo	Tokyo To	Sumida Ku	35.7047	139.8021

```
df_tokyo4.shape
```

```
(651, 6)
```

I will work with this dataset and Foursquare geodata to solve the problem.