Casual Travel Planner:
Exploring Venues around Post Offices with Web APIs:
Hokkaido, Japan

Coursera:
IBM Data Science Professional Certificate:
Capstone Project

July 31, 2019

Hisato Kato

Introduction

As a domestic and worldwide traveler, I used to use guidebooks to explore cities and popular places and seek advices from hotel concierges for local attractions. It has been fun so far, but having visited many places over the years, I started asking myself "Isn't it just re-experiencing what other people had experienced?" With this in mind, I exploit the new technology I learned in this course, geo-locations and Web APIs, to offer to myself quite new experiences with not-so-popular places.

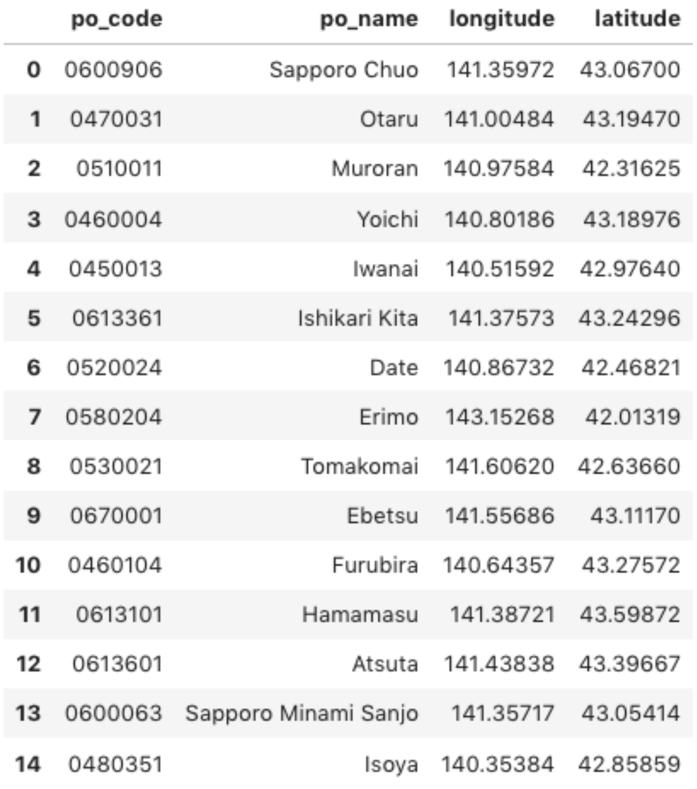
In using geo-search Web APIs, choosing good pivotal locations, or seed locations, is important. Using the name of the city for Web search produces results not much different from guidebooks. Looking for good seed locations, I encountered post offices: they are finer-grained than cities and their latitudes and longitudes are available; there are 24,376 post offices whereas there are 744 cities in Japan. Using post office locations also matches my travel preference. I like somewhat civilized places, meaning not too natural or wild. I use rent-a-car and stay at hotels/inns. In sparse-populated prefectures such as Hokkaido, a post office means there are sizable inhabitants around and there are ATMs and tellers for *Yuucho* Bank and *Kampo* Insurance who speak standard-accent Japanese: No worries while traveling unfamiliar places.

In this report, I describe how I applied data science approach and techniques to plan my next travel. In the next section, data used this time is described, which is followed by methodology, results, discussion, and conclusion sections.

Data

Various geo-location data are available from the Japan Geographic Data Center[i], from which I downloaded 99 post office information available free of charge[ii]. The data corresponds to a part of Hokkaido, which is the largest and the northern-most prefecture in Japan. I extracted necessary fields for my geo-search (post office code, post office name, latitude, and longitude), transliterated the name into English, transformed into CSV format, and placed it on GitHub[iii]. Table 1 shows the first 15 rows of the data.

Table 1. Post office location data (sampled)



	po_code	po_name	longitude	latitude
0	0600906	Sapporo Chuo	141.35972	43.06700
1	0470031	Otaru	141.00484	43.19470
2	0510011	Muroran	140.97584	42.31625
3	0460004	Yoichi	140.80186	43.18976
4	0450013	Iwanai	140.51592	42.97640
5	0613361	Ishikari Kita	141.37573	43.24296
6	0520024	Date	140.86732	42.46821
7	0580204	Erimo	143.15268	42.01319
8	0530021	Tomakomai	141.60620	42.63660
9	0670001	Ebetsu	141.55686	43.11170
10	0460104	Furubira	140.64357	43.27572
11	0613101	Hamamasu	141.38721	43.59872
12	0613601	Atsuta	141.43838	43.39667
13	0600063	Sapporo Minami Sanjo	141.35717	43.05414
14	0480351	Isoya	140.35384	42.85859

In addition, I use geographical map and venues data available through Web APIs. I use the Folium package[iv] to render maps and the Foursquare Places API[v] to access trending venues, venue recommendations, and venue categories.

Methodology

First, I draw a map of Hokkaido and place the 99 post office data points on the map with the Folium API, centering on the average of all latitudes and longitudes. Then I go on to narrow down the candidate locations, excluding the places I already visited and excluding big cities I know of. Then I try the Foursquare "Get Trending Venues" endpoint [vi]to check if there are places marked trending by Foursquare around the remaining post office locations.

Out of left-over post offices, I select some locations with Pandas operations based on our knowledge that larger latitude means that the place is further north, smaller longitude means that the place is further west, and so on. Then, I take another way of selecting places, clicking on a data point on the interactive Folium map and read the corresponding post office name.

Next, I calculate the distance between some of the target local locations and primal cities where hotels are available. I use a Stack Overflow sample code[vii] with necessary wrapper function.

I then use the Foursquare “Get Venue Recommendations” endpoint[viii] to look for interesting places around the destination,

Next, I call the Foursquare “Get Venue Categories” endpoint[ix], format the output, and find out the number of categories and the structure thereof.

Grouping nearby locations within specified distance is a good practice for checking the feasibility of visiting multiple locations in one day. I apply the density-based clustering technique DBSCAN (Density-Based Spatial Clustering of Applications with Noise)[x] to display a clustering result on the map. Changing the parameter ε (epsilon: distance between two points) gives us various clustering results (number of clusters and number of outliers), which I illustrate with Matplotlib line plots.

Results

A Jupyter Notebook contains all the results obtained from executing the above procedure[xi]. Figure 1 shows an initial map of Hokkaido centering around the centroid of 99 post office locations and Figure 2 shows 99 post office locations on the map, each with a blue dot surrounded by a yellow circle: when displayed on Jupyter Notebook and clicked, the corresponding post office name pops up. We see many overlapping data points around Sapporo, the capital city of Hokkaido. Excluding post offices with major city names (Sapporo, Otaru, Yoichi, Muroran, Tomakomai) narrowed down the places from 99 to 78.

Searching trending venues around 78 locations gives us just an airport.

```
Chitose Kuukou (Airport) : New Chitose Airport (CTS) (新千歳空港) : Airport
```

It looks it is not very useful for finding out trending venues in Hokkaido, but it is OK for my purpose because I am interested in not-so-popular places.

```
[5]: #Display an initial map
     map3 = folium.Map(location=[latitude, longitude], zoom_start=7)
     map3
```

[5]:

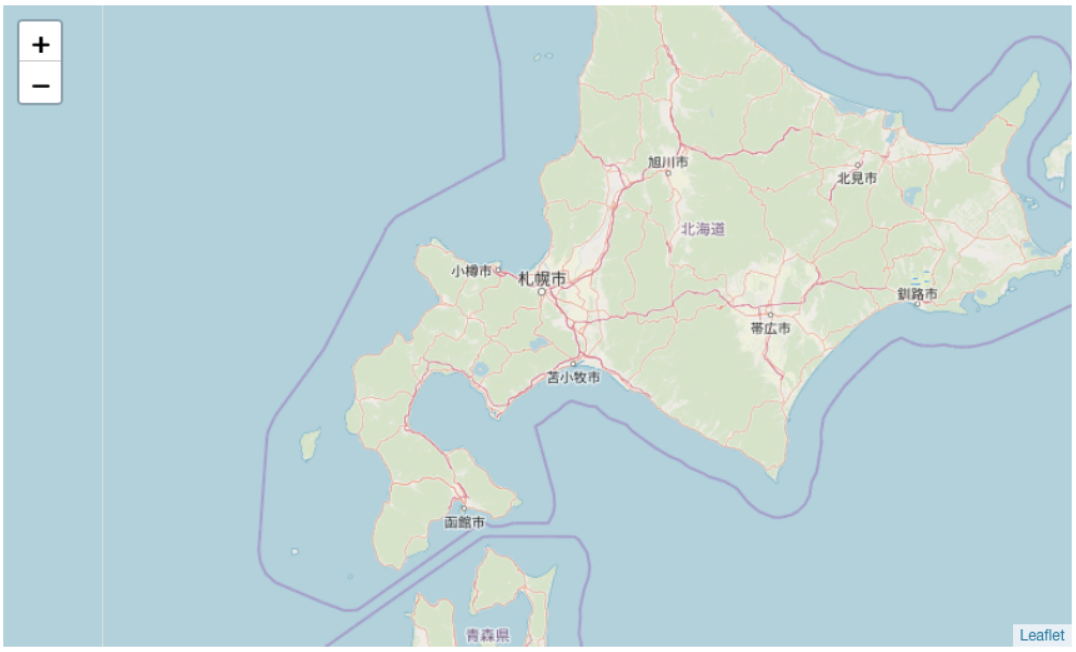


Figure 1. Initial map of Hokkaido centered around the centroid of 99 data points

[6]:

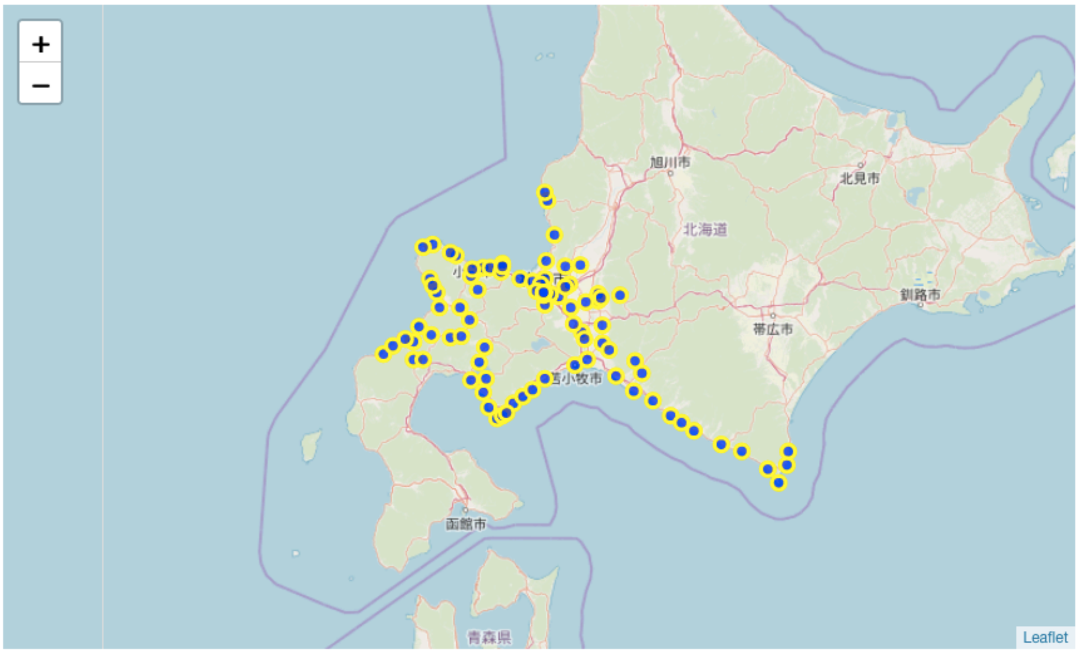


Figure 3. Ninety-nine post office locations. (An interactive map on Jupyter Notebook.)

I then used Pandas min/max operations to pick up several north-end, east-end, and west-end locations as follows:

- Top 2 latitude (North-end): Horo, Hamamasu
- Top 4 longitude (East-end): Hidaka Meguro, Shoya, Erimo Misaki, Erimo
- Bottom 2 longitude (West-end): Shimamaki, Honme

These places are surrounded by blue rectangles in Figure 4.
In addition, I search for a couple of inland places, clicking on the blue markers to display the name, and selected those with interesting (exotic) names as follows:

- Mid-south-east: Niwan, Biratori
- Western part: Nakoma, Konbu

These places are surrounded by orange rectangles in Figure 4.

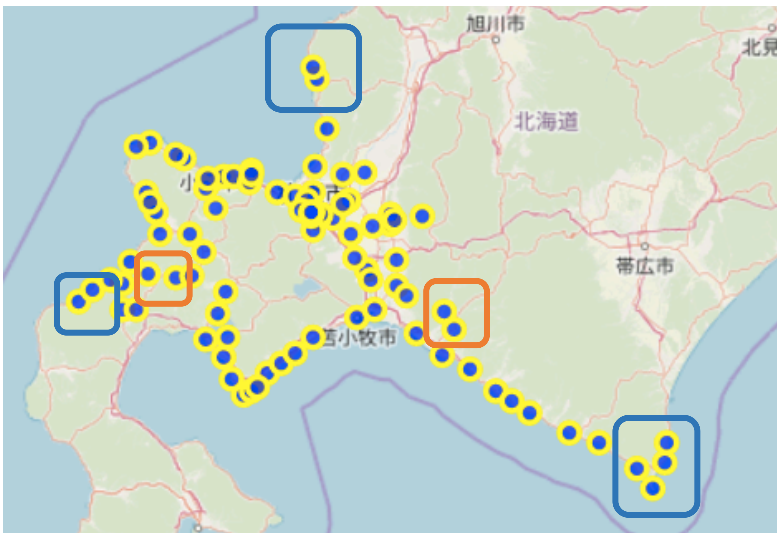
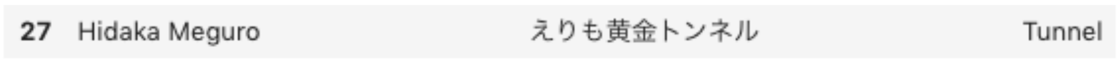


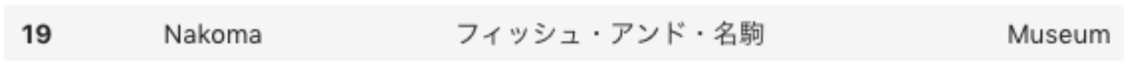
Figure 4. Candidate travel locations. (An interactive map on Jupyter Notebook.)

Then, I listed up neighborhood venues around the places I picked up. For example, the Erimo Golden Tunnel was identified as a neighborhood venue of "Hidaka Meguro".



27	Hidaka Meguro	えりも黄金トンネル	Tunnel

Another neighborhood search result is the "Fish and Nakoma museum" which is close to "Nakoma" post office.



19	Nakoma	フィッシュ・アンド・名駒	Museum

An example of single distance calculation result follows:

- Hidaka Meguro - Erimo Misaki: 22.2 km

Using Pandas Data Frame and apply() function, multiple distance calculation can be done at once, for example,

- from Sapporo Chuo to Nakoma: 79.3 km
- from Otaru to Nakoma: 61.9 km
- from Muroran to Nakoma: 70.1 km
- from Tomakomai to Nakoma: 96.5 km

Foursquare has a couple of levels of categories: for example, “food” category has subcategories such as “Asian Restaurant” and “Latin American Restaurant”, and “Asian Restaurant” in turn has sub-subcategories such as “Japanese Restaurant” and “Chinese Restaurant”, and so on. Analyzing the “Get Venue Categories” call result, I found out categories are nested in varying depths, five levels maximum. The maximum depth case is the following:

Food > Latin American Restaurant > South American Restaurant
> Brazilian Restaurant > {Acai House, Baiano Restaurant, ⋯, Tapiocaria}

Number of categories at each level is summarized in Table 2.

Table 2. Foursquare categories

Level	# of entries	Example
1	10	Food
2	456	Latin American Restaurant
3	367	South American Restaurant
4	91	Brazilian Restaurant
5	13	Tapiocaria

An example of applying the DBSCAN density-based clustering to the original 99 post office locations is shown in Figure 5. This shows which locations are close together: locations with the same color means they are in a cluster where each location is within a specified distance (ε=15km) from a neighboring location.

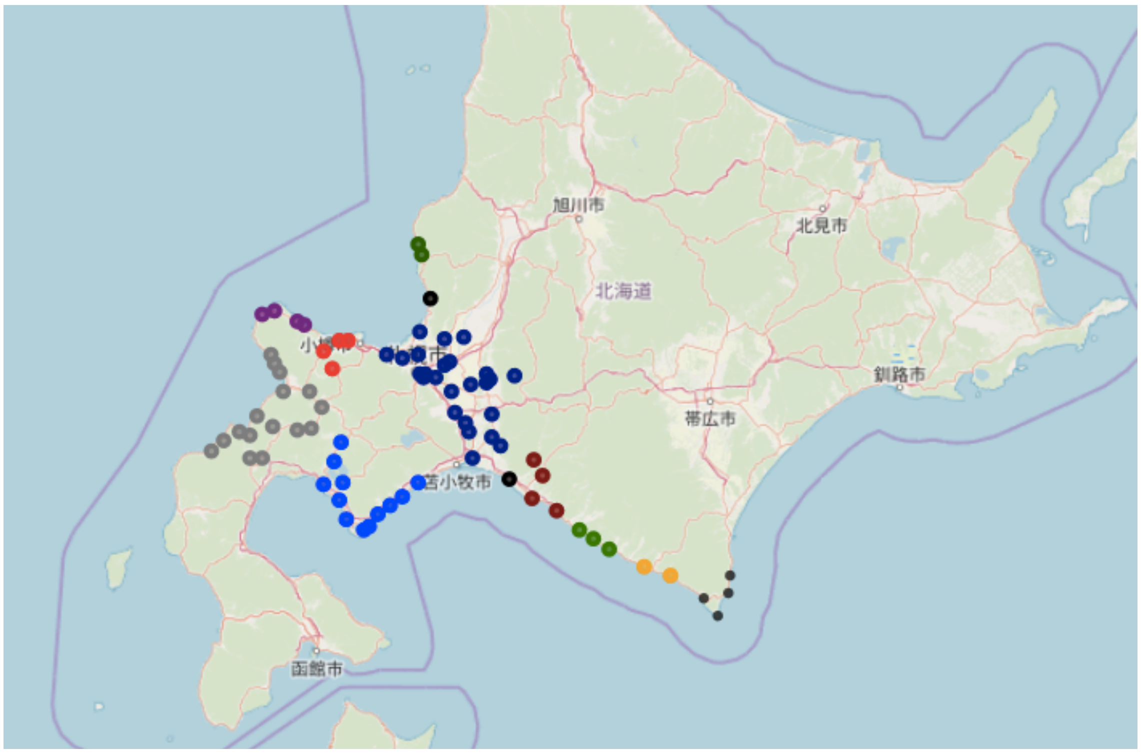


Figure 5. Density-based clustering (ε=15km)

Changing the parameter ε gives us a different clustering results (the number of clusters and the number of outliers), which is plotted in Figure 6. We see the number of outliers (singular points without a neighbor) monotonically decreases to zero, whereas the number of clusters increases, then decreases to one, which means all the locations are in one cluster. Choosing ε=10km~15km looks good for practical travel planning.

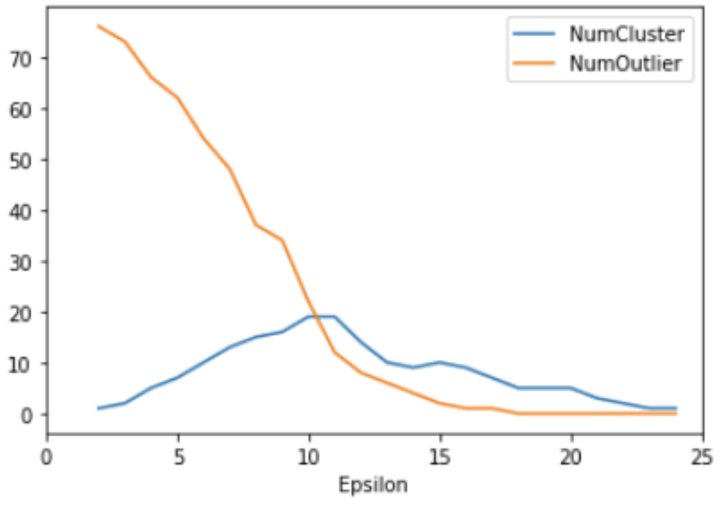


Figure 6. Density-based clustering with varying parameter (ε)

Discussion

As described above, post office location data and Web APIs (Folium, Foursquare) can be used for a new way of travel planning. As for travel experience, I have the following scenario in mind:

Post offices are, historically, a pivot of civilization in each local vicinity.

Post office tellers speak standard-accent Japanese.

Use their business service first, for example, ask for a "YuuPack" gift catalog, choose a gift, pay for it, and arrange delivery via post office network.

Ask a post office teller (or the head of post office) to introduce interesting places around. (Only if they are not busy working. Usually they aren't.)

By the way, doing such things are quite safe and effective in Japan

Conclusions

I have come up with a new travel planning approach, which I will put into practice soon. Future plans are to obtain all post office geo-location data and explore other prefectures. In reality, we need other web services to complete a travel planning such as car navigation[xii], train navigation[xiii] and the Gourmet navigation[xiv].

---

[i] http://www.kokudo.or.jp/

[ii] http://www.kokudo.or.jp/database/005.html

[iii] https://github.com/hisakato/Capstone/blob/master/2019-07_post-office-Hokkaido1e.csv

[iv] https://python-visualization.github.io/folium/

[v] https://developer.foursquare.com/places-api

[vi] https://developer.foursquare.com/docs/api/venues/trending

[vii] https://stackoverflow.com/questions/19412462/getting-distance-between-two-points-based-on-latitude-longitude

[viii] https://developer.foursquare.com/docs/api/venues/explore

[ix] https://developer.foursquare.com/docs/api/venues/categories

[x] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html

[xi] https://github.com/hisakato/Capstone/blob/master/Kato_Hokkaido4e.ipynb

[xii] www.google.com/maps

[xiii] ekitan.com

[xiv] www.gnavi.co.jp