Covid-19 cases locations in Hong-Kong

Daniel PONT

April 3, 2020

1. Introduction

The 2019–20 coronavirus pandemic is an ongoing pandemic of coronavirus disease 2019 (COVID-19), caused by severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2). The outbreak started in Wuhan, Hubei Province, China, in December 2019. The World Health Organization (WHO) declared the outbreak to be a Public Health Emergency of International Concern on 30 January 2020 and recognised it as a pandemic on 11 March 2020

(source : https://en.wikipedia.org/wiki/2019%E2%80%9320_coronavirus_pandemic).

In order to better understand and fight this pandemic, geospatial data are crucial. Countries that have been able to track accurately locations of outbreaks (South Corea, Taiwan…), often using smartphones geolocation features, are also those who have been able to take the most efficient measures.

In this report we will focus on covid-19 cases in Hong-Kong :

- In the first part we will visualize on a map buildings where cases have been reported. We will also show how the buildings locations can be segmented using k-means algorithm.

- In the second part, we will search (using foursquares API) venues located within a 150m radius of each building linked to cases. We will sort these venues according to the number of buildings with covid-19 cases located in their neighborhood and provide informations (name, category, position) retrieved via foursquares.

2. Data

The original dataset is a csv file : **building_list_eng.csv** updated daily on this website : https://data.gov.hk/en-data/dataset/hk-dh-chpsebcddr-novel-infectious-agent

We downloaded it on march 25. It contains the following data (see table next page) :

	District	Building name	Last date of residence of the case(s)	Related probable/confirmed cases
0	Sai Kung	Yee Ching House Yee Ming Estate	NaN	58,128
1	Wan Chai	Envoy Garden	NaN	114, 213
2	Southern	Block 28, Baguio Villa	NaN	Case notified by the health authority of Canad...
3	Tai Po	Heng Tai House, Fu Heng Estate	NaN	119, 124, 140
4	Tuen Mun	On Hei House, Siu Hei Court	NaN	120, 121
...	...	...	...	...
432	Central & Western	33 Des Voeux Road Central (non-residential)	23/03/2020	402
433	Central & Western	Wah Po Building (non-residential)	24/03/2020	319
434	Yau Tsim Mong	Silvercord (non-residential)	24/03/2020	388
435	Sham Shui Po	Lander Hotel Prince Edward (non-residential)	24/03/2020	388
436	Yau Tsim Mong	Mary Building (non-residential)	24/03/2020	388

437 rows × 4 columns

Thanks to google geocoding API, we can find latitude and longitude of each building from their name and district :

	District	Building name	Last date of residence of the case(s)	Related probable/confirmed cases	Latitude	Longitude
0	Sai Kung	Yee Ching House Yee Ming Estate	NaN	58,128	22.307034	114.263220
1	Wan Chai	Envoy Garden	NaN	114, 213	22.264584	114.188604
2	Southern	Block 28, Baguio Villa	NaN	Case notified by the health authority of Canad...	22.262689	114.131696
3	Tai Po	Heng Tai House, Fu Heng Estate	NaN	119, 124, 140	22.457911	114.170456
4	Tuen Mun	On Hei House, Siu Hei Court	NaN	120, 121	22.373186	113.967726
...	...	...	...	...	...	...
432	Central & Western	33 Des Voeux Road Central (non-residential)	23/03/2020	402	22.283049	114.157233
433	Central & Western	Wah Po Building (non-residential)	24/03/2020	319	22.283902	114.129211
434	Yau Tsim Mong	Silvercord (non-residential)	24/03/2020	388	22.297496	114.169367
435	Sham Shui Po	Lander Hotel Prince Edward (non-residential)	24/03/2020	388	22.326621	114.164267
436	Yau Tsim Mong	Mary Building (non-residential)	24/03/2020	388	22.296243	114.171714

437 rows × 6 columns

Then, using foursquares API, we can find venues located within a 150 m radius of each building :

	District	Building	Building Latitude	building Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Wan Chai	Envoy Garden	22.264584	114.188604	Happy Valley (Upper) Bus Stop 跑馬地（上）巴士站	22.264392	114.188966	Bus Stop
1	Wan Chai	Envoy Garden	22.264584	114.188604	Yogo Happy Valley	22.264002	114.188833	Cupcake Shop
2	Wan Chai	Envoy Garden	22.264584	114.188604	Envoy Garden Bus Stop (安蔭苑巴士站)	22.264389	114.188804	Bus Stop
3	Southern	Block 28, Baguio Villa	22.262689	114.131696	Baguio Villa (Lower) Minibus Stop 碧瑤灣（下）小巴站	22.261945	114.132117	Bus Stop
4	Southern	Block 28, Baguio Villa	22.262689	114.131696	The Arcade, Cyberport (數碼港商場)	22.263127	114.132820	Shopping Mall

3. Methodology

The metodology is the following :

1. Load data in a jupyter notebook
2. Add geocoordinates by querying google geocding API
3. Study clustering of builldings reporting cases
4. Explore venues nearby buildings with cases

3.1. Load data in a jupyter notebook

```
df = pd.read_csv('./data/building_list_eng.csv')
```

3.2. Add geocordinates by querying google geocoding API

```
df['Latitude']=np.nan
df['Longitude']=np.nan
gmaps = googlemaps.Client(key='XXXXXXXXXXXXXXXXXXXXXXXXXX')
for row in df.itertuples():
     address = df.at[row.Index, 'Building name']+', '+df.at[row.Index, 'District']+'
District, '+'Hong Kong'
     geocode_result = gmaps.geocode(address)
     if ((len(geocode_result) > 0 )) and ('geometry' in geocode_result[0]) :
        df.at[row.Index, 'Latitude'] = geocode_result[0]['geometry']['location']['lat']
        df.at[row.Index, 'Longitude'] =
geocode_result[0]['geometry']['location']['lng']./data/building_list_eng.csv')
```

3.3 Clustering of cases locations

First we have to find the number k of clusters : we calculate the the silhouette score for k-means clustering :

```
# Instantiate a scikit-learn K-Means model
model = KMeans(random_state=0)

# Instantiate the KElbowVisualizer with the number of clusters and the metric
visualizer = KElbowVisualizer(model, k=(2,10), metric='silhouette', timings=False)

# Fit the data and visualize
visualizer.fit(df[['Latitude','Longitude']])
visualizer.poof()
```

Then we appy k-means clustering to the geodata (latitude,longitude) :

```
# set number of clusters
kclusters = 2
# run k-means clustering
kmeans = KMeans(n_clusters=kclusters, random_state=0).fit(df[['Latitude','Longitude']])

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

df.insert(6, 'Cluster Labels', kmeans.labels_)
```

Finally we can visualize clusters on a folium map :

```
hongkong_map = folium.Map(location=[22.3526632,113.9876166], zoom_start=11,tiles='Stamen
Toner')
cluster_colors=['red','blue']
for lat, lon, building, cluster in zip(df['Latitude'], df['Longitude'],df['Building
name'],df['Cluster Labels']):
    folium.CircleMarker([lat, lon],
                        radius=4,
                        color=cluster_colors[cluster],
                       popup =('Building: ' + str(building) + '<br>'),
                        fill_color=cluster_colors[cluster],
                        fill_opacity=0.7 ).add_to(hongkong_map)
```

3.4. Exploration of venues nearby buildings with cases

Venues within a 150 m radius of buildings where cases have been reported can be found by applying the following function on our dataset :

```
def getNearbyVenues(districts, buildings, latitudes, longitudes, radius=150):

    venues_list=[]
    for district, building, lat, lng in zip(districts, buildings, latitudes,
longitudes):

        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
        response = requests.get(url).json()["response"]

        if(('groups' in response) and (len(response['groups']) > 0) and
(len(response['groups'][0]['items']) > 0)) :
            results = response['groups'][0]['items']

            # return only relevant information for each nearby venue
            venues_list.append([(
                district, building, 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 = [
                  'District', 'Building', 'Building Latitude', 'building Longitude',
                  'Venue', 'Venue Latitude', 'Venue Longitude', 'Venue Category']

    return(nearby_venues)
```

We can then sort these venues according to the number of buildings with covid-19 cases located in their neighborhood :

```
df_venues_count_building=df_venues.groupby([
        'Venue','Venue Latitude','Venue Longitude','Venue Category']).
        count()[['Building']].
        sort_values(by='Building', ascending=False)
```

Next we can find wich categories are the most frequent amongst the venues of interest :

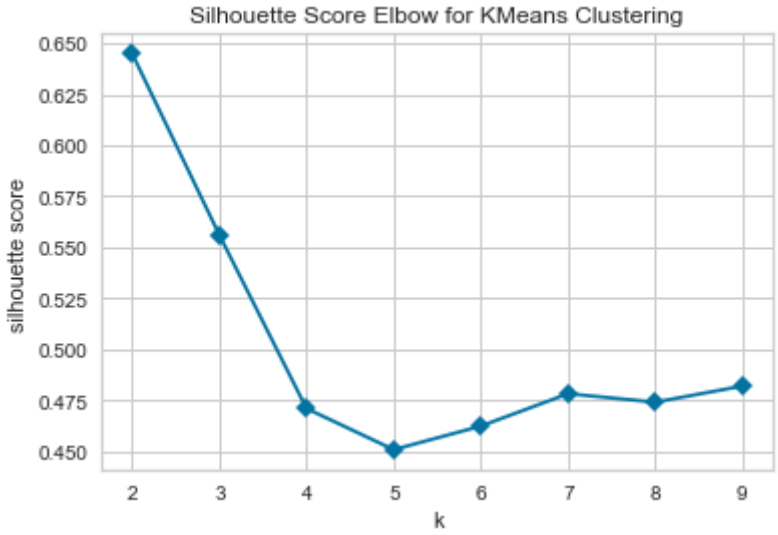
```
df_venues_count_building.groupby('Venue Category').
        sum()[['Building']].
        sort_values(by='Building', ascending=False)[0:50]
```

Finally we can visualize venues which have more than 3 buildings with cases in their neighborhood on a folium map :

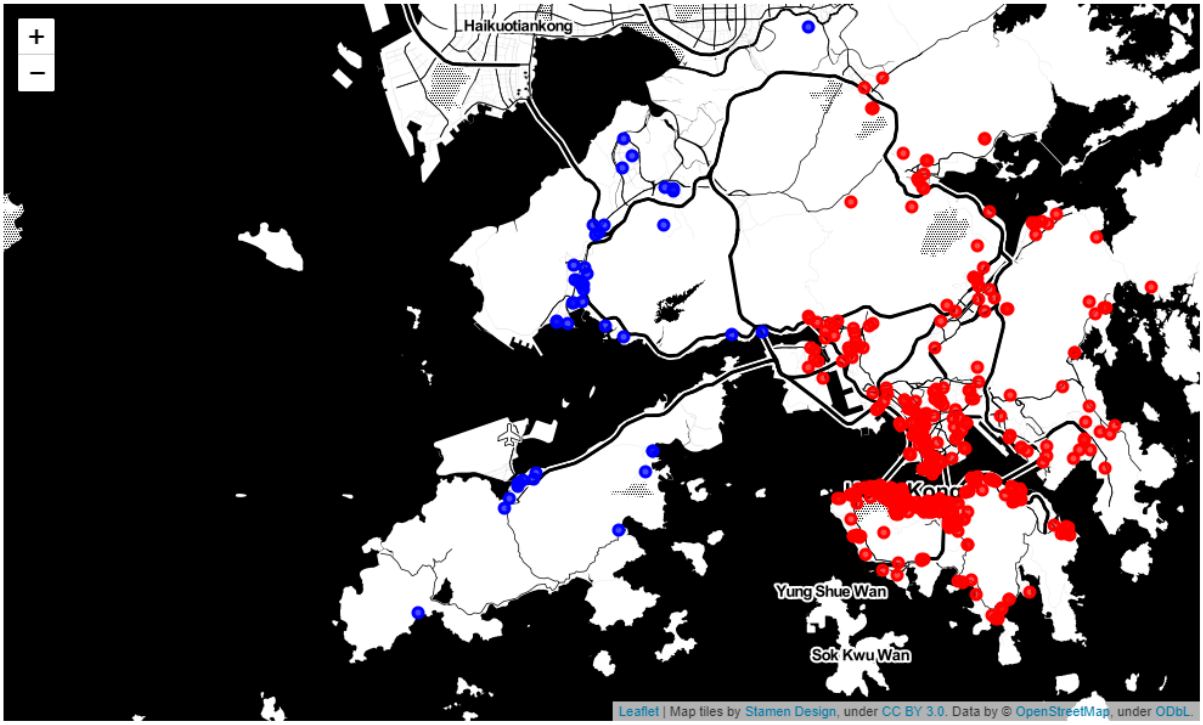
```
df_main_venues=df_venues_count_building.query('Building>3').reset_index()
venues_map = folium.Map(location=[22.3526632,113.9876166], zoom_start=11,tiles='Stamen
Toner')
colors_array = cm.YlOrRd(np.linspace(0, 1, 15))
colgrad = [colors.rgb2hex(i) for i in colors_array]
for lat, lon, venue, category, building in zip(df_main_venues['Venue Latitude'],
df_main_venues['Venue Longitude'],
                                    df_main_venues['Venue'],df_main_venues['Venue
Category'], df_main_venues['Building']):
    folium.CircleMarker([lat, lon],
                        radius=5,
                        color=colgrad[building-1],
                        popup =('Venue: ' + venue +'<br> Category : '+category+'<br>
Nb.:'+str(building) + ' <br>'),
                        fill_color=colgrad[building-1],
                        fill_opacity=0.7 ).add_to(venues_map)
```

4. Results

The silhouette score for k-means clustering, displayed below, indicate that k=2 is the optimal number of clusters :



These clusters are displayed on the map next page :



We note that one cluster is on the west side of Hong Kong, the other (the main one in term of casualties) is on the east side.

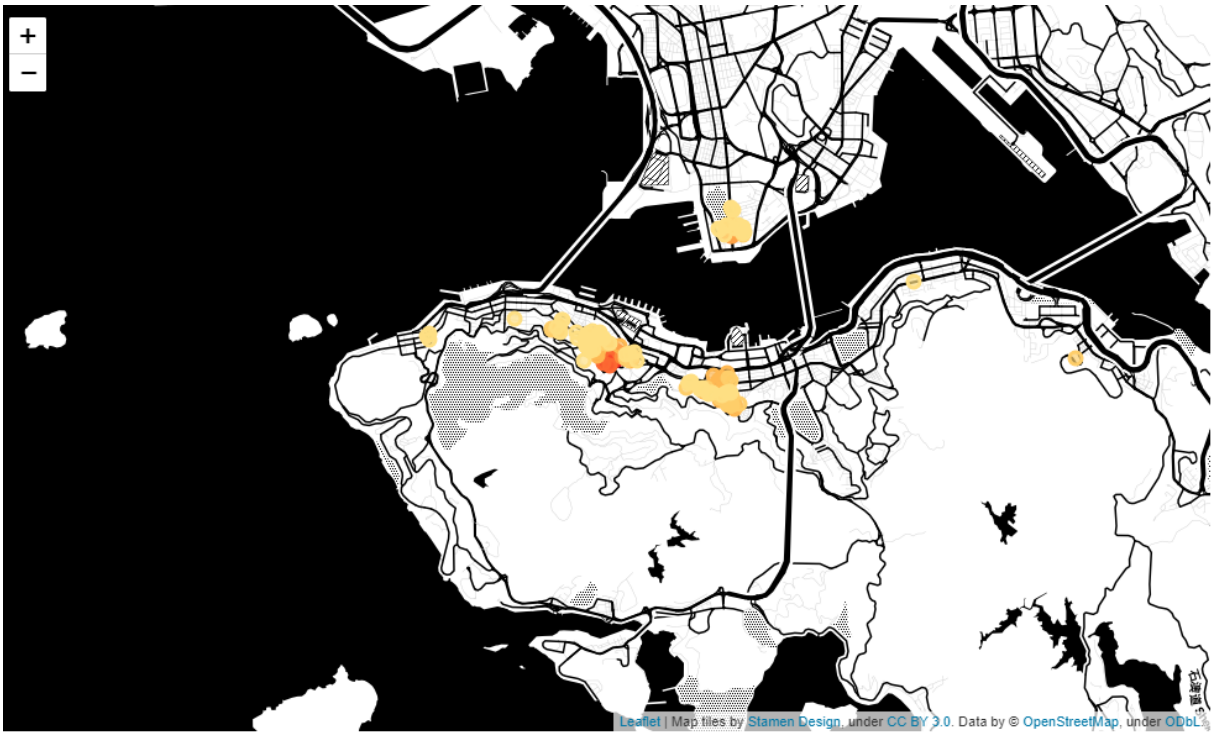
Venues which counts the largest numbers of buildings with reported cases in their neighborhoods are the followings :

Venue	Venue Latitude	Venue Longitude	Venue Category	Building
Grill ขุนทอง	22.281279	114.155596	BBQ Joint	15
Le Boudoir	22.281470	114.154787	Cocktail Bar	15
Fringe Club Roof Garden	22.280302	114.155958	Bar	14
Porterhouse	22.281173	114.155580	Steakhouse	14
pherform	22.280430	114.156019	Gym	14
Hard Rock Café Hong Kong	22.280517	114.155297	American Restaurant	14
Tango Argentinian Steakhouse	22.281835	114.154728	Argentinian Restaurant	14
Flex Studio Central	22.280431	114.156020	Yoga Studio	14
Chrono	22.280819	114.155595	Karaoke Bar	14
Pure Fitness	22.280941	114.155457	Gym / Fitness Center	14
Chaiwala	22.280896	114.154886	Indian Restaurant	14
Pure Yoga	22.280698	114.154960	Yoga Studio	14
Bar de Luxe	22.280977	114.155881	Cocktail Bar	14
Sake Bar Ginn	22.280859	114.155308	Japanese Restaurant	14
Gao's Foot Massage (古法足道)	22.280955	114.155872	Massage Studio	14
Carbone	22.280698	114.155441	Italian Restaurant	14
Casa Lisboa	22.281394	114.154851	Portuguese Restaurant	14
New Punjab Club	22.280250	114.155475	Pakistani Restaurant	13
Iso Fit - Pilates and Gyrotonic Studio	22.281147	114.154723	Gym	13
Bombay Dreams	22.281560	114.154409	Indian Restaurant	13

These venues seem to be mostly restaurants and bars, a fact we can confirm by counting the buildings for each venue category :

Venue Category	Building
Coffee Shop	191
Chinese Restaurant	184
Café	167
Japanese Restaurant	155
Hotel	131
Italian Restaurant	108
Cantonese Restaurant	105
French Restaurant	100
Fast Food Restaurant	100
Bar	97
Noodle House	93
Cocktail Bar	85
Sushi Restaurant	75
Shopping Mall	74
Bakery	70
Hong Kong Restaurant	69
Cha Chaan Teng	66
Dessert Shop	64
Vietnamese Restaurant	60
Indian Restaurant	54

These venues, unsurprisngly, are located mostly in the central neighborhood of Hong Kong Island.



5. Conclusion

Google geocoding API let us visualize on a folium map buildings where Covid-19 cases have been reported in Hong Kong.

By applying k-means, we showed that these cases can be segmented in two clusters, one on the west side of Hong Kong, the other on the east side.

The free version of Foursquare API enables data scientist to find venues nearby buildings where cases have been reported. Being able to spot those venues (mostly bars, coffees, restaurants) can be used in several ways :

- Close venues wich feature a high risk,
- Warn customers of these venues they should get diagnosed,
- Study how the virus is transmitted.