Clustering cities for similar liveability

Hung Dinh
February, 2020

1. Business problem

It is just about time in your life that someday you may want to move to another city. Could be a new job, could be relocation closer to your family, or could be that you want to try life in a different country. If that day comes, the most nature question you will have is “how is the new city different/similar from my current place”. In this project, I collect 6 democratic information from popular cities in the world, categories their differences, and put similar cities in groups.

The information I collect contributes to the so-called liveability index from this Business Insider article, including:

- crime rate
- education
- culture
- nature
- health
- infrastructure.

They help you depict most aspect of a city to be consider for living potential.

What you can get from the result? If you have to relocate, you can tell if the new city is similar to your place. If you aim for new experience, you can use the cluster to find cities that fit your purpose. Given this, you can have reasonable expectation and plans prior to moving.

2. Data

The clustering will be carried out on basic components of selected cities from all over the world. These cities can be chosen based on their size (eg. area, population or economy). Spatial area does not necessary reflect city popularity. Similar reason is applied for population. The economic strength is the most suitable indicator for our purpose. A well-developed city can be large (eg, Vancouver) or small (London), with high population (eg, Beijing) or low (Brussels). I get a list of such cities in this wiki page. With these, we need to acquire information mentioned above.

2.1. Crime rate

For crime rate, I use the data provided at Numbeo website. The database is city-indexed and can be requested directly from city name.

2.2. Education, culture, nature, health, and infrastructure

For the other 5 criteria, I use the Foursquare database to find venues in each category. To request this, I first need each city's coordinate. The latitude and longitude can be requested similar to the crime rate (directly from city name) from Geodatos website.

For each city represented by a pair of latitude/longitude, I make an explore query by foursquare api, selecting the most popular 100 venues in an area with radius of 5 km. The returned venues' categories will be put into 5 groups. Each group contains specific non-overlap foursquare categories summarized on [Foursquare development page](#).

Ideally, after acquiring and cleaning data, what I have is a table with the following 9 columns: city name, latitude, longitude, crime index, counts of venues in 5 criteria groups. I will then run the unsupervised clustering on this data. In the result, if an user is interested in looking for places that are similar to his/her interested city, the cities that have the same cluster will be suggested.

3. Methodology

As described above, my project starts with getting the list of cities from a Wikipedia page (3.1). Exploratory analysis consists of removing nan and summarizing cities in each country.

I then get the crime rate for each of these cities from a third-party website (3.2). Because of the differece in city naming convention, some entries need to be edited before acquiring the data. Moreover, some cities have no data. Exploratory analysis consists of removing nan and summarizing cities with highest and lowest crime rate.

The next step is to obtain geocoordinates of these cities from another third-party website (3.3). Again, I need to edit some city names to match its database before crawling.

Before acquiring Foursquare venues, I need to build a classification of their categories based on the 5 criteria (3.4). This step involves getting all possible categories from Foursquare first, and then grouping them. How to build the 5 groups is described in more detail later.

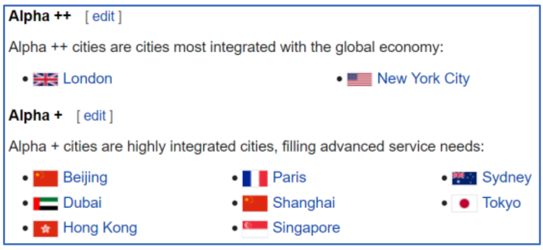
The last data acquisition stage is getting popular venues in each city (3.5). This is straighforwardly done by calling the personal api.

After obtaining necessary data, for each city, I group all venues into groups and count how many places each group has (3.6). This step helps to find the most typical living criteria of the cities. Then this data is fed into an unsupervised clustering (3.7). This process finds cities that have similar living indexes. This result (4) will be analyzed later (5) to see the differences between group.

Finally, I provide an example of application to find and visualize similar cities from a specific city defined by the user (6).

3.1. Get list of cities

The data is available on the wiki page provided above. To get them, I first acquire the page use the combination of `request` and `beautifulsoup` packages. The returned webpage is sliced to get the title between `<li>` tags as shown below. This holds the city name information that I want.



Alpha ++ [edit]

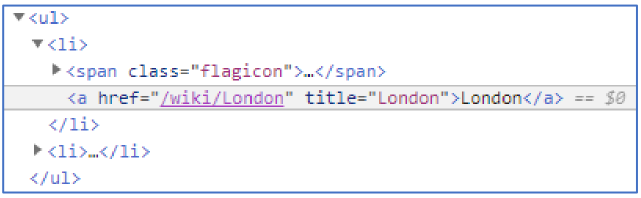
Alpha ++ cities are cities most integrated with the global economy:

- London
- New York City

Alpha + [edit]

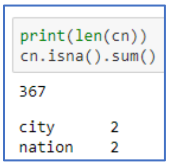
Alpha + cities are highly integrated cities, filling advanced service needs:

- Beijing
- Dubai
- Hong Kong
- Paris
- Shanghai
- Singapore
- Sydney
- Tokyo



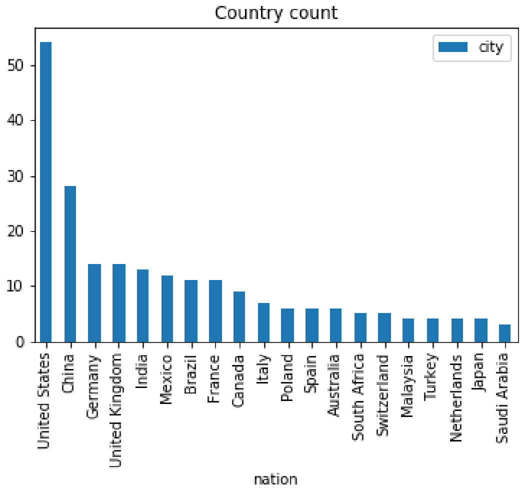
```
▼<ul>
  ▼<li>
    ▶<span class="flagicon">…</span>
     <a href="/wiki/London" title="London">London</a> == $0
   </li>
  ▶<li>…</li>
 </ul>
```

After check the *Nan* values, it is found that the data has 367 entries, in which 2 are *Nan*. Removing them gives me 365 entries. With some specific cities (like *Mexico city*), I also have to remove the *"city"* part from their names for data consistency.



```
print(len(cn))
cn.isna().sum()

367

city      2
nation    2
```

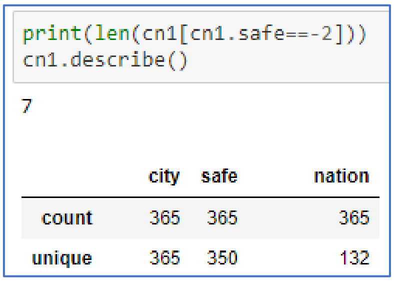
Next, I make a plot to see how many cities selected in each country (Please note that this figure will be referenced later in the Discussion):



It's clear that the US has most many cities in the data, mostly because the criteria for choosing cities is ones that have impactful economic strength. The next is, not surprisingly, China. Then, most of G8 countries are in the list.

3.2. Get Crime data

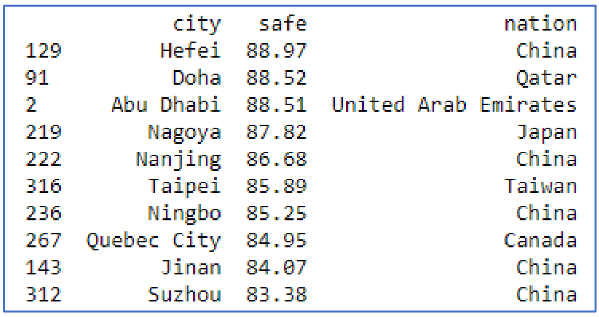
The data is available on Numbeo website. However, because its city name is stored slightly different from my city list, I need to manually edit those to match. The modification is mostly changing Unicode characters into ASCII like this: `cn.loc[cn['city'] == "Zürich", 'city'] = "Zurich"`.



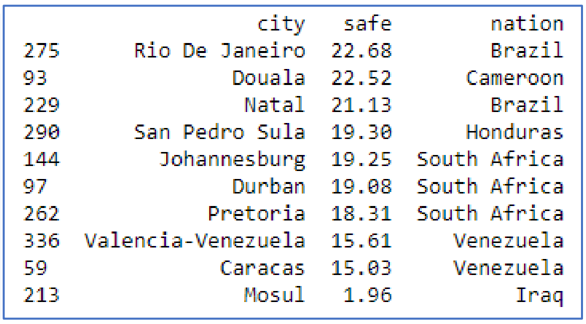
```
print(len(cn1[cn1.safe==-2]))
cn1.describe()

7

        city  safe  nation
count    365   365     365
unique   365   350     132
```

Once the data is acquired, I check if there is any city that is not found. It turns out that there are 7 entries not found on the website. I simply remove them from the data. The remaining data now has 358 cities without any *Nan* value.

Below are 10 cities with highest (left) and lowest (right) crime rate. We can see the most dangerous cities are found in China, and the safest cities are in South America/South Africa.



```
            city   safe                nation
129        Hefei  88.97                 China
91          Doha  88.52                 Qatar
2      Abu Dhabi  88.51  United Arab Emirates
219       Nagoya  87.82                 Japan
222      Nanjing  86.68                 China
316       Taipei  85.89                Taiwan
236       Ningbo  85.25                 China
267  Quebec City  84.95                Canada
143        Jinan  84.07                 China
312       Suzhou  83.38                 China
```

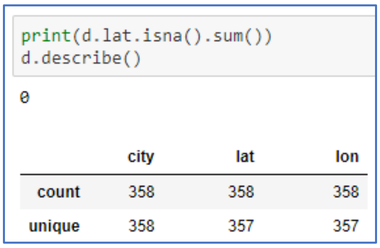


```
                  city   safe        nation
275     Rio De Janeiro  22.68        Brazil
93              Douala  22.52      Cameroon
229              Natal  21.13        Brazil
290     San Pedro Sula  19.30      Honduras
144       Johannesburg  19.25  South Africa
97              Durban  19.08  South Africa
262           Pretoria  18.31  South Africa
336 Valencia-Venezuela  15.61     Venezuela
59             Caracas  15.03     Venezuela
213              Mosul   1.96          Iraq
```

3.3. Get coordinates

The data is available at Geodatos website. Similar to Numbeo website, I need to slightly edit the city names to match the online database. The acquisition simply involves a `request` command with city name as the query.

The returned data appears to have no *Nan* value. After this stage, my data consists of 358 cities.



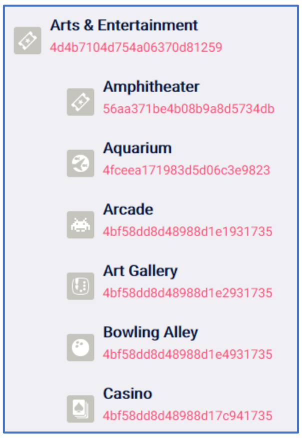
```
print(d.lat.isna().sum())
d.describe()
```

```
0
```

	city	lat	lon
count	358	358	358
unique	358	357	357

3.4. Get foursquare categories

This stage starts with getting all available categories that is provided by Foursquare. I convert them (left) into a hierarchical table starts with `cat.`, followed by parents and children numbers (right) as shown below. In total, 941categories are found.



Arts & Entertainment
4d4b7104d754a06370d81259

Amphitheater
56aa371be4b08b9a8d5734db

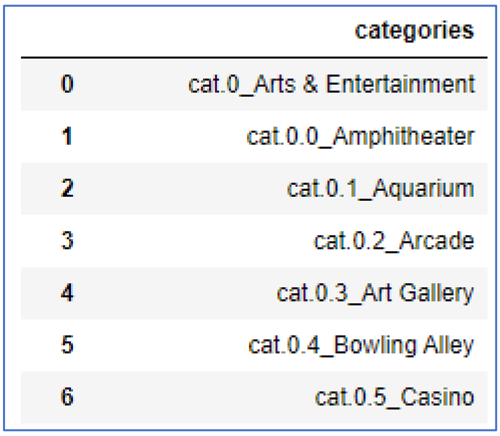
Aquarium
4fceea171983d5d06c3e9823

Arcade
4bf58dd8d48988d1e1931735

Art Gallery
4bf58dd8d48988d1e2931735

Bowling Alley
4bf58dd8d48988d1e4931735

Casino
4bf58dd8d48988d17c941735



	categories
0	cat.0_Arts & Entertainment
1	cat.0.0_Amphitheater
2	cat.0.1_Aquarium
3	cat.0.2_Arcade
4	cat.0.3_Art Gallery
5	cat.0.4_Bowling Alley
6	cat.0.5_Casino

To group them into 5 criteria as described in 2.2, I classify them as below:

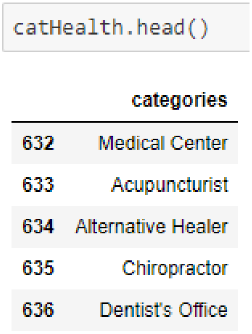
- Health:
    - Medical Center (6.22)
- Culture:
    - Art and Entertainment (0)
    - Nightlife Spot (4)
- Environment:
    - Outdoors & Recreation (5) except Athletics & Sports (5.0) and States & Municipalities (5.52)
- Education: school and the like in Foursquare categories
    - College & University (1)
    - School (6.34)
    - Daycare (8.25)

- Infrastructure: focus on public transportation (bus stop, train station), commercial offices and the like in Foursquare categories
  - Travel & Transport (9) except Hotel (9.13)

An example of classifying for Infrastructure index is shown below:

```
catInfras = categ[categ["categories"].str.startswith('cat.9')]
catInfras = catInfras[~catInfras["categories"].str.startswith('cat.9.13')]
```

The final example of Health index criteria has following categories:



```
catHealth.head()
```

	categories
632	Medical Center
633	Acupuncturist
634	Alternative Healer
635	Chiropractor
636	Dentist's Office

To prepare for the next step (summarizing Foursquare venues for each city), I add extra 5 empty columns (associating to 5 criteria) into the data. At this stage, my data has 358 cities.

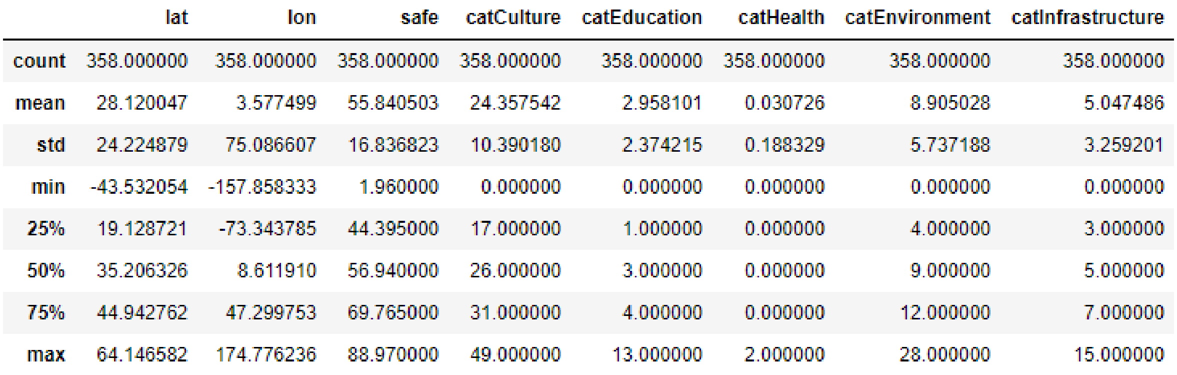
3.5. Get Foursquare venues

I acquire the most 100 places in the radius of 5 km in these cities. This covers an area of about 80 km^2, or a medium size city. I start with defining a function to crape and write data to the output, then calling this function for each city in the data, given its latitude and longitude found before in section 3.3.

The function is done by using the free Foursquare api, requesting `venues`, `explore` query. This process is completed without much hassle. Each city now has a 100 row sub-database listing its popular places and their categories.

3.6. Group returned venues into the 5 category criteria

I then group the venues' categories and count how many places fall into each of the 5 living criteria. Here is the final summary of the grouped data:



	lat	lon	safe	catCulture	catEducation	catHealth	catEnvironment	catInfrastructure
count	358.000000	358.000000	358.000000	358.000000	358.000000	358.000000	358.000000	358.000000
mean	28.120047	3.577499	55.840503	24.357542	2.958101	0.030726	8.905028	5.047486
std	24.224879	75.086607	16.836823	10.390180	2.374215	0.188329	5.737188	3.259201
min	-43.532054	-157.858333	1.960000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	19.128721	-73.343785	44.395000	17.000000	1.000000	0.000000	4.000000	3.000000
50%	35.206326	8.611910	56.940000	26.000000	3.000000	0.000000	9.000000	5.000000
75%	44.942762	47.299753	69.765000	31.000000	4.000000	0.000000	12.000000	7.000000
max	64.146582	174.776236	88.970000	49.000000	13.000000	2.000000	28.000000	15.000000

The data summary shows very low `catHealth` index. This implies that Health index might not contribute to the classification. For demonstration purpose, it is kept in this analysis but one can remove `catHealth` to see if there is any difference. My initial examination suspects there is not though.

3.7. k-means clustering

For this project, I choose unsupervised learning because it's one of the most effective way to get insights about the data without human bias (normally introduced by selecting training data).

I pick k-means clustering because it works for numerical data and the result is easy to interpret.

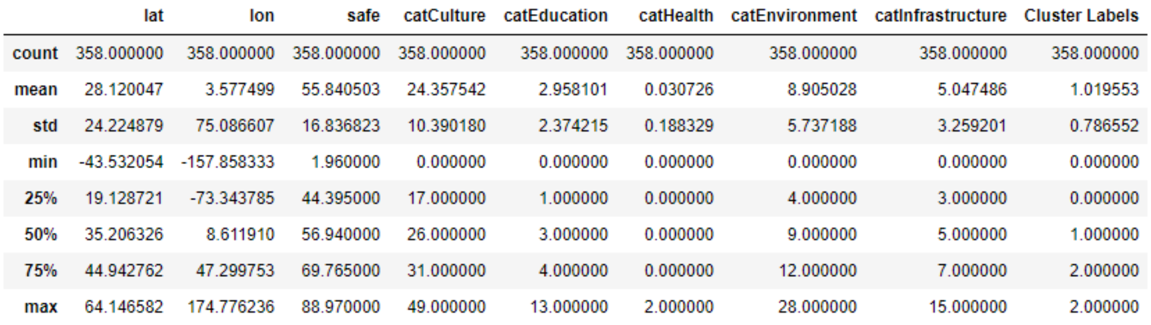
In this project, I select *k=3* simply because of a natural thinking toward numeric number range: "low" - "mid" - "high". Of course, one can have a better idea, but for a blind unbiased approach, this is a good starting point.

In terms of preconditioning, it should be noted that the output venue count is in percentage unit that matches the safety crime index. Therefore, normalization is not required because relative venues ratio in each city is reached with 100 samples.

4. Results

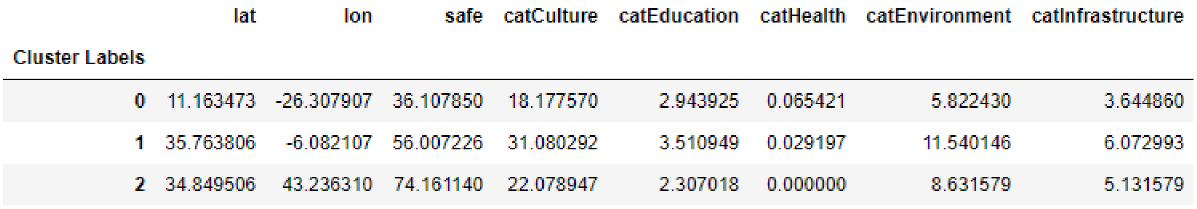
In order to analyze this result, we need to compare the statistics of each group with the whole data.

Let's look at the statistics of the grouped data:



	lat	lon	safe	catCulture	catEducation	catHealth	catEnvironment	catInfrastructure	Cluster Labels
count	358.000000	358.000000	358.000000	358.000000	358.000000	358.000000	358.000000	358.000000	358.000000
mean	28.120047	3.577499	55.840503	24.357542	2.958101	0.030726	8.905028	5.047486	1.019553
std	24.224879	75.086607	16.836823	10.390180	2.374215	0.188329	5.737188	3.259201	0.786552
min	-43.532054	-157.858333	1.960000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	19.128721	-73.343785	44.395000	17.000000	1.000000	0.000000	4.000000	3.000000	0.000000
50%	35.206326	8.611910	56.940000	26.000000	3.000000	0.000000	9.000000	5.000000	1.000000
75%	44.942762	47.299753	69.765000	31.000000	4.000000	0.000000	12.000000	7.000000	2.000000
max	64.146582	174.776236	88.970000	49.000000	13.000000	2.000000	28.000000	15.000000	2.000000

and now the statistics of each cluster:



	lat	lon	safe	catCulture	catEducation	catHealth	catEnvironment	catInfrastructure
Cluster Labels								
0	11.163473	-26.307907	36.107850	18.177570	2.943925	0.065421	5.822430	3.644860
1	35.763806	-6.082107	56.007226	31.080292	3.510949	0.029197	11.540146	6.072993
2	34.849506	43.236310	74.161140	22.078947	2.307018	0.000000	8.631579	5.131579

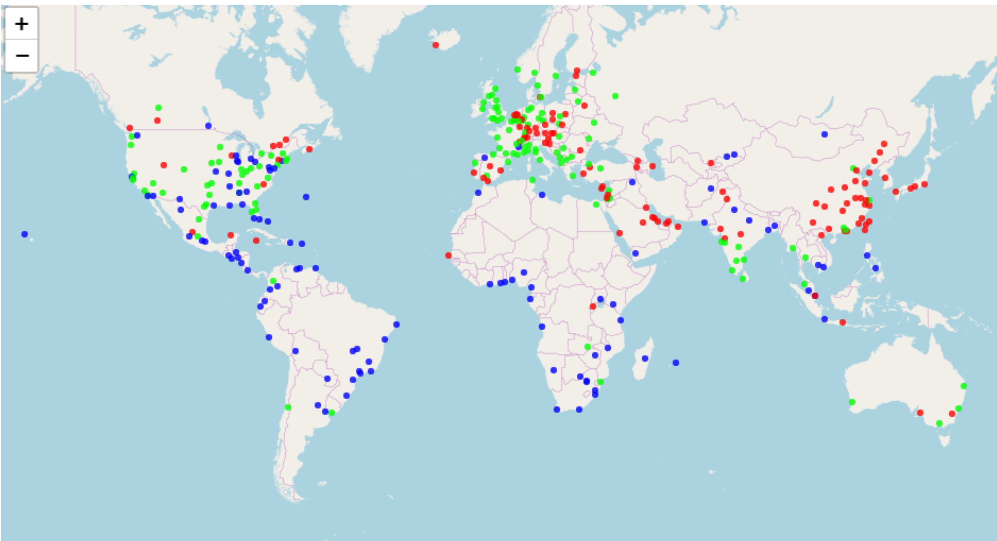
Compare this table with the previous, we can see the difference between the three groups in terms of each criteria:

- Safety, three intervals are clearly distinct with mean values round the mean of the whole safety (55): group 1 is around this value, groups 0 and 2 have values of 36 and 74, smaller than the 25% quantile (44) and larger than 75% quantile (69).
- Culture, class 0 has high value of 31, equal the 75% quantile of this criteria. Classes 1 and 2 are similar and below average.
- Education, three classes are not siginificantly different.
- Health, in the description table, the variance is even larger than the mean. This indicate this criteria doesn't contribute into the clustering. Please note that my standard for health is counting the number of hospital and health department, not necessary the quality.
- Environment, class 1 has highest score, while classes 0 and 2 are lower distinctly.
- Infrastructure, classes 0 and 2 have similar values to the 25% and 50% quantiles, while class 1 is a bit lower than the 75% quantile.

With these observations, I have my interpretation as follow:

- Class 0 - developing cities: lowest crime rate and other living indexes except education (for development).
- Class 1 - working cities: best for democratic indexes but medium crime rate - best for career and financial dynamic life (high crime rate is quite common for such cities).
- Class 2 - trade-off cities: average in living indicators but high crime index.

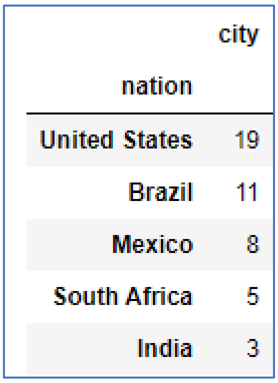
Then I make a map to display the distribution of these classes. This is done by using folium packages, plotting each city according to its coordinates, and color coded based on the cluster (Blue: class 0, Green: class 1, Red: class 2).



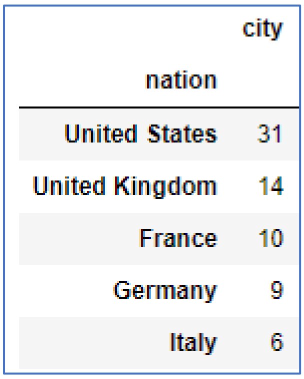
On the interactive version, I can access the information of each city by clicking on its point. In this project, the city name and its cluster name will be displayed.

5. Discussion

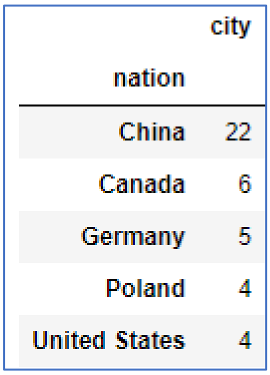
Now let's see if there is any country that has most cities fall into these classes. Their counting is shown below, from left to right as ordered from 0 to 2, respectively:



	city
nation	
United States	19
Brazil	11
Mexico	8
South Africa	5
India	3



	city
nation	
United States	31
United Kingdom	14
France	10
Germany	9
Italy	6



	city
nation	
China	22
Canada	6
Germany	5
Poland	4
United States	4

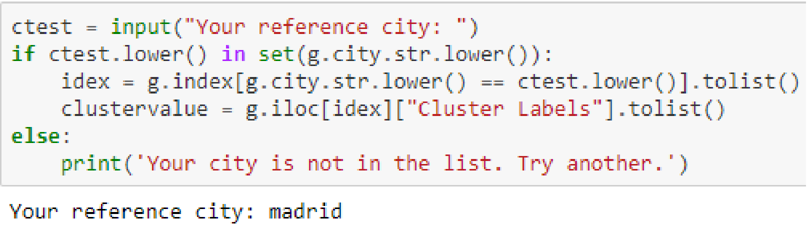
No surprisingly, as the most developed country, the US mainly has developing or full work cities (classes 0 and 1, respectively). China with the reputation of pollution and closed politics has highest crime rate (found in 3.2) and ranks 1st in the trade-off cities (class 2). These findings can be easily spotted on the class map above: the bad cities (class 2 - red) in China, developing (class 0 - blue) and full working (class 1 - green) cities in the US.

Of course, a factor that these 2 countries make up most of the data (found in Figure 1) infers they can rank the first in any statistics. If we instead look at the distribution in each country for now, it is found that most UK cities are good working cities, followed by France and Germany. We can also find that most of the EU are good (except for Central European cities); South America and South Africa are best for development; Middle East and Japan are also trade-off cities between development and stability.

All this information can be used as a recommendation guideline for ones who are using the tool for making decision.

6. Application

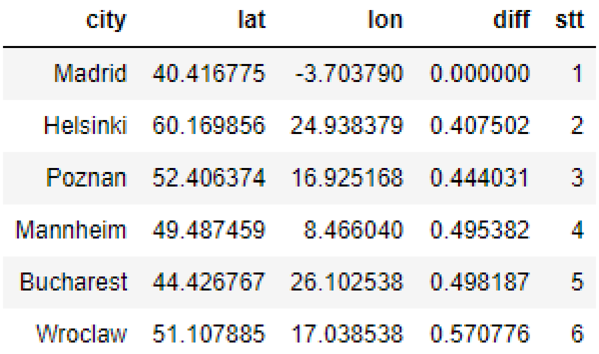
Now one can try finding places similar to his. I build a simple form to an user to input. In the example below, Madrid (Spain) is chosen.



```
ctest = input("Your reference city: ")
if ctest.lower() in set(g.city.str.lower()):
    idex = g.index[g.city.str.lower() == ctest.lower()].tolist()
    clustervalue = g.iloc[idex]["Cluster Labels"].tolist()
else:
    print('Your city is not in the list. Try another.')
```

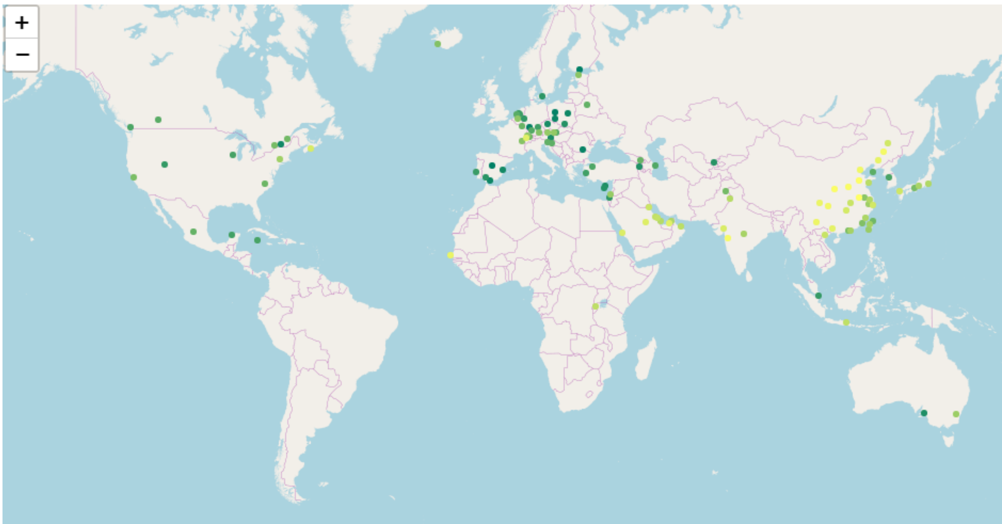
```
Your reference city: madrid
```

The idea is to tell the top 5 similar cities that are similar to the user’s choice. In order to obtain this, I first extract all cities that have the same cluster as the chosen one's. I then normalize the 6 living indexes of these cities and calculate the L2 difference to the user's city. This result is sorted and the cities with lowest difference are shown. In this example, the most similar city to Madrid is Helsinki (Poland):



city	lat	lon	diff	stt
Madrid	40.416775	-3.703790	0.000000	1
Helsinki	60.169856	24.938379	0.407502	2
Poznan	52.406374	16.925168	0.444031	3
Mannheim	49.487459	8.466040	0.495382	4
Bucharest	44.426767	26.102538	0.498187	5
Wroclaw	51.107885	17.038538	0.570776	6

To make the result even more intuitive, I create another map now displaying all similar cities to the use's input, color coded by the similarity, the darker the closer. Here the visualization indicates better information that is more useful for the user for his decision.



7. Conclusions

The project has grouped most popular cities in the world. The similarity between these groups are measured based on the 6 main living indexes, including crime, health, culture, nature, education, and infrastructure. Three main city groups are determined associating to work, to develop and to beware of a possible trade-off with crime.

Closer examination into each group and their international relationship also reveals interesting findings between countries. Possible explanations are provided to support them. An applied example is shown to demonstrate how one can use the result to get meaningful information. I hope the users find this work and its results useful.

8. Acknowledgements

I acknowledge Anton Biryukov for inspiring me to pursue the Machine learning and Data science path since October 2017. Special thanks to Si Le who have been always supported me and, in this project, graciously assisted me playing around with Foursquare category data. Final thanks to IBM Data Science Professional Certificate lecturers and classmates who have helped me follow this intensive specialization.