

Battle of World Capitals!!!

Capstone Project, Coursera-IBM Specialization

Aníbal J Guerra S, ajguerras@gmail.com

Medellín, Colombia

In this project we are meant to put a lot of the knowledge we have gotten through the whole IBM data science specialization. My work is intended to exhibit the application of a methodology, the usage of the right tools, data processing, tools for visualization and to provide a final analysis to try to answer the initial questions.

1. Introduction

I remember the days when playing the simulation game SIM City ® I could be the hero of my very own city as I designed and created a beautiful and bustling metropolis. Through a long chain of apparently simple decisions, like creating schools, libraries, hospitals, entertainment places, etc.; you could make your city a good and attractive place to live, so it would get larger and more intricate. That could be defined as success. On the opposite case, people in your town would start leaving and the whole system finally would fail. A perfect balance which sounds very hard to find.

I grew up in Latin America, in a country that we were told that we were “on the road to become a first world country”. But it always seemed like a never-ending process. After traveling a lot, and visiting very organized cities with a high standards for living, I started to ask myself: what factors really needed for a city to achieve the perfect balance that makes it attractive, providing enough quality of life to consider such city as a well-developed city.

2. Problem Description

Far beyond my youthful meditations, finding the answers to such questions is a very relevant problem. For public servers in the government (i.e. major, governors) having clarity about such factors would give them clarity about the decisions that should be made to keep the city on the road to development. Two questions seem obvious: where on that road is the city right now (what does the city “have”) and what should we “have” as a city in order to be considered as developed city in the future.

As I am not an expert in such extent, I find those answers hard to find. But for sure data science has the answers. I would use the Foursquare data related to all of the venues that the capitals of the world have right now, and see if through data science I get to stratify (or cluster) those cities to see if there are specific venues that are characteristic of the cities development in each class.

Hypothesis: It is possible to discriminate groups of cities according to the amount of certain types of venues in them.

If so, I will have to figure out if at least one of such groups correspond to an already known classification (i.e. in the class of well-developed cities).

3. Data Description

a. Capitals of the world and along with their geo-localization data.

I used python's BeautifulSoup library to get the initial data, the geo-localization of each capital from this website:

`"https://lab.lmnixon.org/4th/worldcapitals.html")` .

The following figure is an example of the initial input. For every capital of the world, the respective geo-localization coordinates and the country they belong to.

Country	Capital	Latitude	Longitude
Afghanistan	Kabul	34.28N	69.11E
Albania	Tirane	41.18N	19.49E
Algeria	Algiers	36.42N	03.08E
American Samoa	Pago Pago	14.16S	170.43W
Andorra	Andorra la Vella	42.31N	01.32E
Angola	Luanda	08.50S	13.15E

Figure 1. Example of the original data

I had to make a couple of transformations to get such data in a suitable format for the Machine Learning process I was going to do after. Final data looked like this:

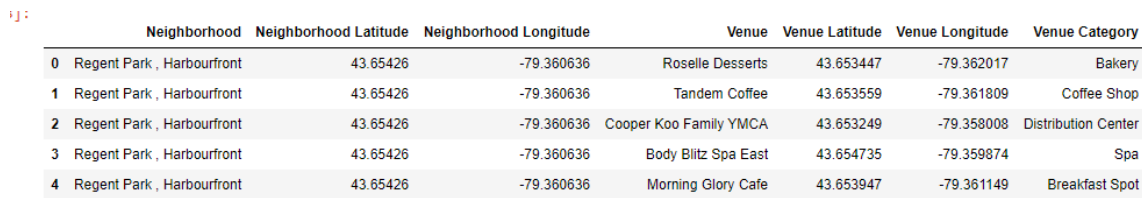


	Country	Capital	Lat	Long
0	Afghanistan	Kabul	34.28	69.11
1	Albania	Tirane	41.18	19.49
2	Algeria	Algiers	36.42	3.08
3	American Samoa	Pago Pago	-14.16	-170.43
4	Andorra	Andorra la Vella	42.31	1.32

Figure 2. Example of the original data after being modified to suite the project's objectives.

b. Venue's data

From Foursquare database I retrieved relevant venue's information, which was finally structured in the following format:



The image shows a Jupyter Notebook interface with a code cell containing a pandas DataFrame. The DataFrame has 8 columns: an index column, Neighborhood, Neighborhood Latitude, Neighborhood Longitude, Venue, Venue Latitude, Venue Longitude, and Venue Category. It contains 5 rows of data for venues in Regent Park, Harbourfront.

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Regent Park , Harbourfront	43.65426	-79.360636	Roselle Desserts	43.653447	-79.362017	Bakery
1	Regent Park , Harbourfront	43.65426	-79.360636	Tandem Coffee	43.653559	-79.361809	Coffee Shop
2	Regent Park , Harbourfront	43.65426	-79.360636	Cooper Koo Family YMCA	43.653249	-79.358008	Distribution Center
3	Regent Park , Harbourfront	43.65426	-79.360636	Body Blitz Spa East	43.654735	-79.359874	Spa
4	Regent Park , Harbourfront	43.65426	-79.360636	Morning Glory Cafe	43.653947	-79.361149	Breakfast Spot

Figure 3. Example of the venue's data after being structured in a pandas dataframe.

A maximum of 100 venues is returned by Foursquare for each request. Also, the maximum number of requests per day was 950, which was very little considering that I worked with more than 200 capitals cities of world. That limited significantly the amount of runs I could do per day.

4. Methodology

a. Folium Maps

In the notebook, all the maps and their respective clusters will be displayed using the Folium library. Folium builds on the data wrangling strengths of the Python ecosystem and the mapping strengths of the leaflet.js library. So, I manipulate my data in Python using pandas and NumPy, and then I visualize it in on a Leaflet map via folium.

Folium makes it easy to visualize data manipulated in Python on an interactive leaflet map. It enables both the binding of data to a map for choropleth visualizations as well as passing rich vector/raster/HTML visualizations as markers on the map. This latter fits my goals in this project. For more information, visit: <https://python-visualization.github.io/folium/>

This is the initial view of the world capitals using folium:



Figure 4. Worldwide map of the country capitals.

b. Data preprocessing

I needed to extract and structure to the information retrieved from the Foursquare database. I crated functions for such goals based my on the useful code provided in the 3rd week of the IBM Capstone project. Here is an example of the function to convert the json file received from Foursquare request into a pandas dataframe.

```
[ ] def get_nearby_venues(results):
    venues = results['response']['groups'][0]['items']

    nearby_venues = json_normalize(venues) # flatten JSON

    #print ("1*****\n",nearby_venues)#quitar

    # filter columns
    filtered_columns = ['venue.name', 'venue.categories', 'venue.location.lat', 'venue.location.lng']
    nearby_venues =nearby_venues.loc[:, filtered_columns]

    #print ("2*****\n",nearby_venues)#quitar

    # filter the category for each rowyy
    nearby_venues['venue.categories'] = nearby_venues.apply(get_category_type, axis=1)

    #print ("3*****\n",nearby_venues)#quitar

    # clean columns
    nearby_venues.columns = [col.split(".")[1] for col in nearby_venues.columns]

    #print ("4*****\n",nearby_venues)#quitar

    nearby_venues["name"] = nearby_venues["name"].str.lower()
    nearby_venues["categories"] = nearby_venues["categories"].str.lower()

    return (nearby_venues)
```

An important issue was that I had to make a request for every single city in the dataset, so it was necessary to iterate over the original dataframe (the one with the capitals coordinates), build a request URL for every specific capital and then execute it.

```
#building calls
def Get_Capitals_Info(Capitals):
    CLIENT_ID = '5KCQKMMUJFG2QNC2HJQE4QL3H2K3CRWFVORFSV0X02FEVUIC' # your Foursquare ID
    CLIENT_SECRET = 'E0W2XTHVMGOT0Y2PL2MVFB2HPYK3ASESU4IHM40014FEVXEM' # your Foursquare Secret
    VERSION = '20180605' # Foursquare API version
    List=['Airport','Culture','Food','Finance','Spectacle','Stores','Education','Tourism','Leisure','Lodging','Industry','Transport','Health','Empty','TotalResp']
    Capitals=pd.concat([Capitals,pd.DataFrame(columns=List)],sort=False)

    for index, row in Capitals.iterrows():
        #print(index, row["Lat"],row["Long"])
        url= buildURL (CLIENT_ID, CLIENT_SECRET, VERSION, latitude=row["Lat"], longitude=row["Long"], radius=40000, LIMIT=500) #1000
        results = requests.get(url).json()

        Capitals['TotalResp'][index]=results['response']['totalResults']
        if (results['response']['totalResults']==0) :
            Capitals['Empty'][index]=1
        else:
            nearby_venues=get_nearby_venues(results)
            #This next line allowed me to know the 15 most common words per capital city
            print(index, nearby_venues['categories'].value_counts()[:15].index.tolist()) #row["Lat"],row["Long"],
            print(Capitals['TotalResp'][index], " "+'{} venues returned by Fours filtered.'.format(nearby_venues.shape[0]))
```

As shown before, the foursquare API returns lots of fields and many categories of venues. In consequence, the decision model would be based on the amount of venues from certain categories per city.

To reduce data dimension and ensure an appropriate clustering, I decided to encode data. However, instead of using the hot encoding learned before I wanted to see the effect of using a different encoding approach. After analyzing the most common words in the dataset, I designed a filter that create classes inside the venue's dataset. Once I segmented venues in different categories, I counted the amount of venues corresponding to each category in the dataframe. These are the categories I defined, along with some of the keywords considered for each class:

- 1.- Airport: has the word "airport", plane, etc.
- 2.- Food services: Restaurant, coffee, bakery, bistro, diner, cafe, etc.
- 3.- Culture: Art, museum, etc.
- 4.- Finance: Bank, etc.
- 5.- Spectacles, and big events: Stadium, theater, hall, etc.
- 6.- Shops: Store, shop, boutique, market, etc.
- 7.- Education services: College, school, etc.
- 8.- Tourism: travel, tourism, etc.
- 9.- Entertainment, Activities, leisure: Trail, gym, club, music, pool, park, playground, plaza, aquarium, bar, church, park, mall, etc.
- 10.- Lodging services: hostel, hotel, etc.
- 11.- Technological services: IT services, etc.
- 12.- Transport facilities: train, transportation, gas station, bus, rail, etc.

13.- Health services: hospital, pharmacy, dentist, etc.

And this is an example of the instructions used to perform the classification:

```
#Count categories appearance
Capitals[List[0]][index]=nearby_venues.categories.str.count("airport|plane").sum() #1.Airport
Capitals[List[1]][index]=nearby_venues.categories.str.count("art|museum|culture|opera").sum() #Culture:
Capitals[List[2]][index]=nearby_venues.categories.str.count("pizza|burger|sandwich|cream|chicken|restaurant|coffee|bakery|bist
Capitals[List[3]][index]=nearby_venues.categories.str.count("bank|finance|credit").sum() #Finance
Capitals[List[4]][index]=nearby_venues.categories.str.count("stadium|theater|hall|event|scenic").sum()#5.Spectacle
Capitals[List[5]][index]=nearby_venues.categories.str.count("store|shop|boutique|market").sum()#6.Stores
Capitals[List[6]][index]=nearby_venues.categories.str.count("college|school|university").sum()# 7. Education
Capitals[List[7]][index]=nearby_venues.categories.str.count("travel|tourist|tourism").sum()#8. Tourism
Capitals[List[8]][index]=nearby_venues.categories.str.count("ski|spa|zoo|taverna|forest|nature|farm|waterfall|pub|plaza|histor
```

So, the final encoding results in the next dataframe :

	Country	Capital	Lat	Long	Airport	Culture	Food	Finance	Spectacle	Stores	Education	Tourism	Leisure	Lodging	Industry	Transport	Health	F
0	Afghanistan	Kabul	34.28	69.11	1	1	10	0	0	3	0	0	5	4	0	1	0	
1	Albania	Tirane	41.18	19.49	0	1	58	0	0	9	0	0	21	13	0	0	0	
2	Algeria	Algiers	36.42	3.08	1	3	27	0	0	5	0	0	9	9	0	1	0	
3	American Samoa	Pago Pago	-14.16	-170.43	0	0	2	0	0	1	0	0	3	0	0	0	0	
4	Andorra	Andorra la Vella	42.31	1.32	0	1	50	0	3	7	0	0	36	10	0	2	0	

Figure 5. Example of the input data encoding

The next thing I did was eliminating non-numeric columns in order to perform data normalization. You'll see in the following figures that I eliminated the 'Finance' category, because in data exploration I observed this field did not allow separating the cities in classes.

And then I defined functions to perform the data normalization.

```
def std_norm(df, column):
    c = df[column]
    df[column] = (c - c.mean())/c.std()

std_norm(Input, 'Health')
std_norm(Input, 'Transport')
std_norm(Input, 'Industry')
std_norm(Input, 'Lodging')
std_norm(Input, 'Leisure')
std_norm(Input, 'Tourism')
std_norm(Input, 'Education')
std_norm(Input, 'Stores')
std_norm(Input, 'Spectacle')
```

In summary, after all the preprocessing and data normalization, I got the next result; which is the final input for the clustering algorithm.

	Airport	Culture	Food	Spectacle	Stores	Education	Tourism	Leisure	Lodging	Industry	Transport	Health
0	0.384252	-0.351402	-0.87272	-0.859855	-0.790625	-0.210187	-0.234738	-0.952632	-0.344169	-0.289318	-0.417542	-0.308614
1	-0.491485	-0.351402	2.02342	-0.859855	-0.222303	-0.210187	-0.234738	0.162922	1.31009	-0.289318	-0.922717	-0.308614
2	0.384252	0.525985	0.152995	-0.859855	-0.601184	-0.210187	-0.234738	-0.673743	0.574865	-0.289318	-0.417542	-0.308614
3	-0.491485	-0.790096	-1.35541	-0.859855	-0.980066	-0.210187	-0.234738	-1.09208	-1.0794	-0.289318	-0.922717	-0.308614
4	-0.491485	-0.351402	1.54073	0.59719	-0.411744	-0.210187	-0.234738	1.20875	0.758671	-0.289318	0.0876324	-0.308614

Figure 6. Example of the final structure for the input data

c. Machine Learning

Since I had no labeled data as input, neither a known way to label it; it was clear that I had to choose a non-supervised approach for Machine Learning. I selected a clustering approach based on the K-means algorithm. The intention was to cluster similar cities in order to figure out if a can find a common criterion (venue category) that allows to discriminate groups.

In order to select the optimal number of clusters, I used the silhouette score metric. It measures how similar is an element to its own cluster (cohesion) in comparison to how different it is to other clusters (separation). It was calculated from 2 to 20 as shown in the code below).

```
for kclusters in range(2, 20) :
    kgc = Input
    kmeans = KMeans(n_clusters = kclusters, init = 'k-means++', random_state = 0).fit_predict(kgc)
    score = silhouette_score(kgc, kmeans)
    indices.append(kclusters)
    scores.append(score)

print (scores)
plot_(indices, scores)
```

Then I plot the results, which are presented in the following graphic. The silhouette score gets better as it gets closer to 1.

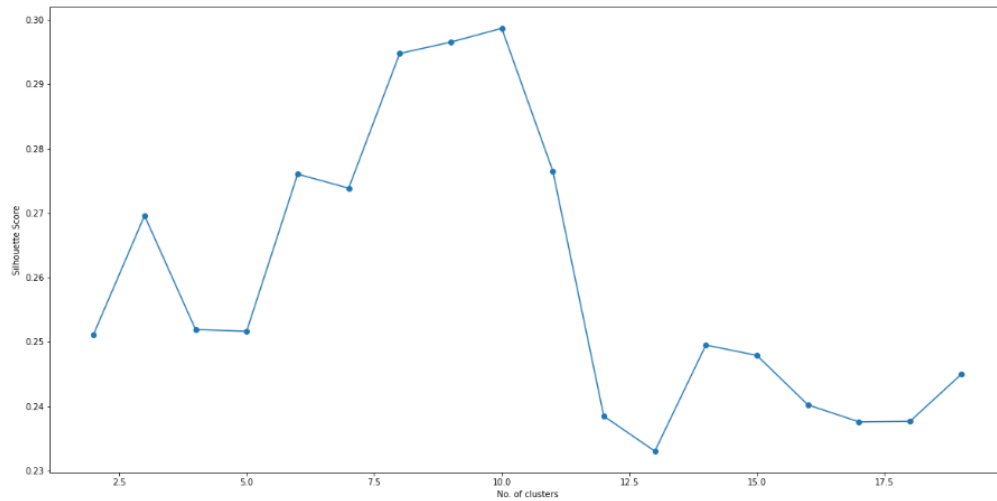


Fig. 7 Silhouette score.

As you can see in $k=10$ I find the optimal value. However, since there is very little difference between the score for $k=8$ and $k=10$, I decided to use the smaller useful value for k .

5. Results

The world capitals were divided into 8 clusters, according to the optimal value found. Using Folium, I generated the worldwide map showing the cities belonging to each cluster, each in a different color.

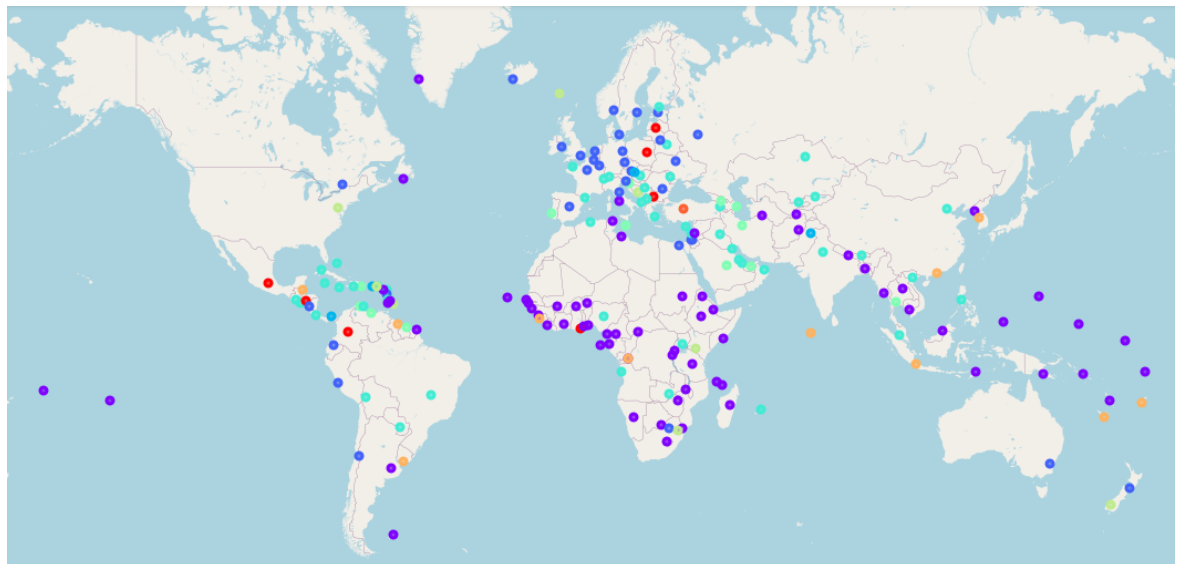


Figure 8. Worldwide map of the capital cities clustered, using k means algorithm with $k=8$.

6. Discussion

When observing the results, the dark blue cluster immediately called my attention. It includes most of the cities in north Europe, also Ottawa in Canada, Moscow, Pretoria and Santiago de Chile. After reviewing it, most of those cities are what I consider very well-developed cities in the world. So, I decided to dive in to see what this cluster was about.

This was cluster number 2

	Top 1	Top 2	Top 3	City
10	Food	Leisure	Stores	Canberra
11	Leisure	Food	Stores	Vienna
18	Food	Leisure	Stores	Brussels
33	Food	Stores	Leisure	Ottawa
38	Food	Stores	Leisure	Santiago
48	Food	Leisure	Lodging	Prague
50	Food	Airport	Culture	Copenhagen
55	Food	Stores	Leisure	Quito
56	Food	Leisure	Stores	Cairo
60	Food	Lodging	Leisure	Tallinn
66	Leisure	Stores	Food	Paris
72	Food	Leisure	Stores	Berlin
76	Food	Stores	Leisure	Basse-Ter
85	Food	Leisure	Lodging	Reykjavik
90	Leisure	Food	Lodging	Dublin
91	Food	Leisure	Stores	Jerusalem
94	Food	Leisure	Lodging	Amman
107	Food	Leisure	Stores	Vilnius
108	Food	Leisure	Lodging	Luxembourg
127	Food	Leisure	Stores	Amsterdam
130	Food	Stores	Leisure	Wellington
131	Food	Stores	Lodging	Managua
137	Food	Stores	Leisure	Oslo
144	Stores	Food	Leisure	Lima
151	Leisure	Food	Stores	Bucuresti
152	Food	Lodging	Stores	Moskva
159	Stores	Leisure	Food	San Marin
165	Leisure	Lodging	Transport	Ljubljana
168	Food	Stores	Leisure	Pretoria
169	Food	Leisure	Stores	Madrid

Figure 9. List of the cities in cluster #2, along with the top 3 venues per city.

After exploring such table, it was clear that this cluster was dominated in more than 90% by Food, Leisure, and stores venues; and secondary by lodging services. All that categories are indicators of commercial activities which could also be traduced as a sign of a healthy economy. If I count those 4 categories, they describe 98% of the venues in cluster #2. So, it seems that, according to the Foursquare limited dataset (100 venues per city), the number of venues in such four categories allows clustering those cities, and finally that might be interpreted as an indicator for city development. I also observed the effects of the culture, since oriental cites were categorized in a different cluster.

An issue that must be discussed, is the effect of the population size in each city, which clearly would affect the number of venues in every city. For a second iteration of this data analysis I would consider normalizing the amount of venues respect to the population size of each city. Another important issue is the limited number of venues returned by Foursquare, since 100 is a very small number and that surely impacted results.

7. Conclusions

In this report, I have presented a model to cluster world cities according to the most common venues in them.

One of the major challenges of public servers at local governments is to equilibrate the growth in every sector in order to have well-balanced cities. Nowadays, the commerce and service venues have become an important part of a city development, and according to the results presented here they could be interpreted as a sign of a healthy economy.

If small cities really want to grow and develop, they must watch the receipt that has been followed by major cities around the globe. The economy and governance are delicate fields, multiple factors must be coordinated in order to maintain a balance of growth in the cities. Now that in this modern era data has become available and computing capabilities have grown significantly, data science could be the key tool to answer the most complex questions.

The results presented here should be subjected to an in-depth comparative analysis with the data available in the archive of Key Short-Term Economic Indicators available at the website of the organization for economic co-operation and development (<https://stats.oecd.org>); in order to find greater coincidences and a deeper interpretation of them.