A battle of cities

Analyzing similarities between neighborhoods in two cities

MARCH 29

IBM Data Science Professional – Final Project
Authored by: Raúl Cano Argamasilla

Introduction

Business Problem

In the coming weeks, I am changing my job from Germany to Chile. Since I will need to relocate there, chosing a new apartment can be overwhelming given all the different districts in the new city, let alone the lack of knowledge of what it is like to live there.

Obviously, I would like to choose an area to live where I feel comfortable. An approximate solution is to find a neighborhood similiar to the one I live in or any other I know in my current city, based on the services they offer.

By clustering neighborhoods in both cities, using their services as the features of the algorithm, we could find neighborhoods in the new city that are similar to the ones where I live currently. In this way, I will mitigate the uncertainty what I will find in my new location.

> ***"Find even more easy-to-use tools on the Insert tab, such as to add a hyperlink or insert a comment"***

Audience

This situation is a very frequent one that employees all over the world have to face when they are relocated. If generalized to multiple cities, any human resources department could leverage the applicability of this algorithm to ease the on-boarding of employees moving to different cities.
In addition, I intend to use this project for my personal use to look for an apartment in Santiago.

Data

Data processing

After an initial attempt to find location data on this two cities, there does not seem to be an official source with structured data. Therefore, the approach will be:

1. Extract the postal codes of each city.
2. Find the coordinates of each postal code using Nominatim or any other service offering such information.
3. Structure the extracted data coordinates properly

Data sources

The following sources of information for this phase have been identified:

Postal codes and city neighborhoods data

- https://www.geonames.org/postal-codes/
- For Santiago
  https://www.geonames.org/postalcode-search.html?q=santiago&country=CL
- For Darmstadt
  https://www.geonames.org/postalcode-search.html?q=Darmstadt&country=DE

Venues information

- https://developer.foursquare.com/

With this approach, I foresee that the solution is transformed in a general one with any tuple of cities. In order to execute successful query, we need:

- The name of the cities to search.
  For this data, one needs to type the names manually in the corresponding variable.
- The correct country codes that are used in the Geonames website, such as CL for Chile, or DE for Germany.
  As we see in the website, we can extract all available country codes by inspecting the dropdown menu.

```
<select name="country">
<option value="" selected=""> all countries</option>
<option value="DZ"> Algeria</option>
<option value="AS"> American Samoa</option>
<option value="AD"> Andorra</option>
…
</select>
```

To obtain the postal codes, a query to Geonames including city and country, will output a HTML table with each postal code and coordinates in their rows, among other data. Our work will consist then in requesting such HTML and extract the relevant items from the table. For example, the URL https://www.geonames.org/postalcode-search.html?q=santiago&country=CL produces the following HTML table (I cut some rows to make visualization easier):

	Place	Code	Country	Admin1	Admin2	Admin3
1	Santiago	8320000	Chile	Región Metropolitana	Provincia de Santiago	Santiago
	-33.454/-70.656					
2	Providencia	7500000	Chile	Región Metropolitana	Provincia de Santiago	Providencia
	-33.436/-70.609					
3	Las Condes	7550000	Chile	Región Metropolitana	Provincia de Santiago	Las Condes
	-33.421/-70.502					
4	Vitacura	7630000	Chile	Región Metropolitana	Provincia de Santiago	Vitacura

Methodology

Input

- *cities*: Dictionary with cities and country codes where the clustering will be done.
  For example:

```
cities = [
{'city' : 'Cologne', 'country' : 'DE'},
{'city' : 'Santiago', 'country' : 'CL'},
]
```

This algorithm will allow to enter as many cities as desired, but in this exercise we are focusing in only two.

Output

- A list of clusters grouping areas in both cities, so one can inspect visually the similarity between them

Results

asdfasdf

Discussion

asdfasdf

Conclusion

asdfasdf