REPORT on Exploring & Profiling Adjacent Neighborhoods

Coursera Capstone Project for

IBM Data Science Specialization

Christos Glymidakis – enaeuro@yahoo.com

Introduction – Business Problem

Problem Overview

It is often the case that someone would like to know “what is around” a certain location of explicit interest. For example, when we visit a new place we only know the address of the hotel we have booked, or of the house of the friends who are going to accommodate us. However, most commonly we will not know anything about the neighborhoods around this single point. To overcome this challenge, in this capstone project I have proposed the following advisory solution:

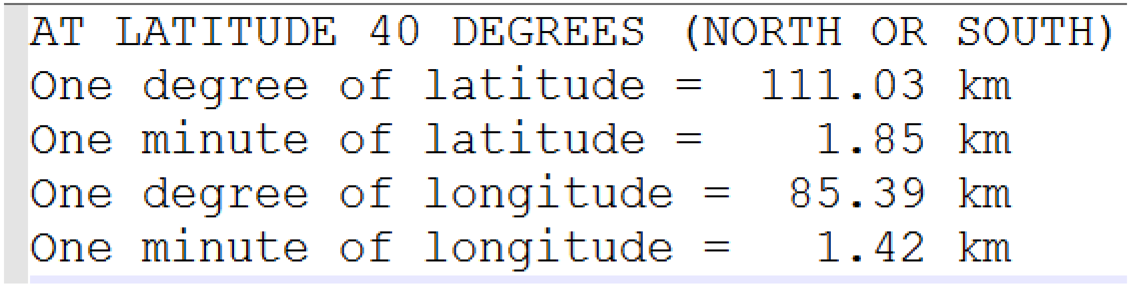
- Select the single pin point on a map, around which we would like to do some profile exploration. For this single point we will need either a valid and complete address or the exact coordination (latitude and longitude).
- Automatically generate a grid of 5x5 points (this can be shrank or expanded, depending on how far away we would like to explore), which have a distance of approximately 1-2 km from each other. This grid essentially covers an area of roughly 16-64 square kilometers (rectangular with almost 4-8 km each side).
- Using the Foursquare API fetch the most common venues for each area and consider their venue type.
- Apply a clustering algorithm with an average of 5 distinct profiles (i.e. k=5) and therefore, group the areas into similar profiles.
- In this way we can tell if the neighborhoods around the focal point are similar to the one where we plan to stay, and towards which direction we might have alternative patterns, landscapes and type of venues.

Data Overview

Data Overview

To solve this challenge we will need the following data:

- An initial,=]
- valid and complete address with which we can call the Google geocode API and retrieve the exact coordinate of the point of interest.
- The shape and size of the grid we would like to explore around the central point. As a rule of thumb, we can use the following steps:

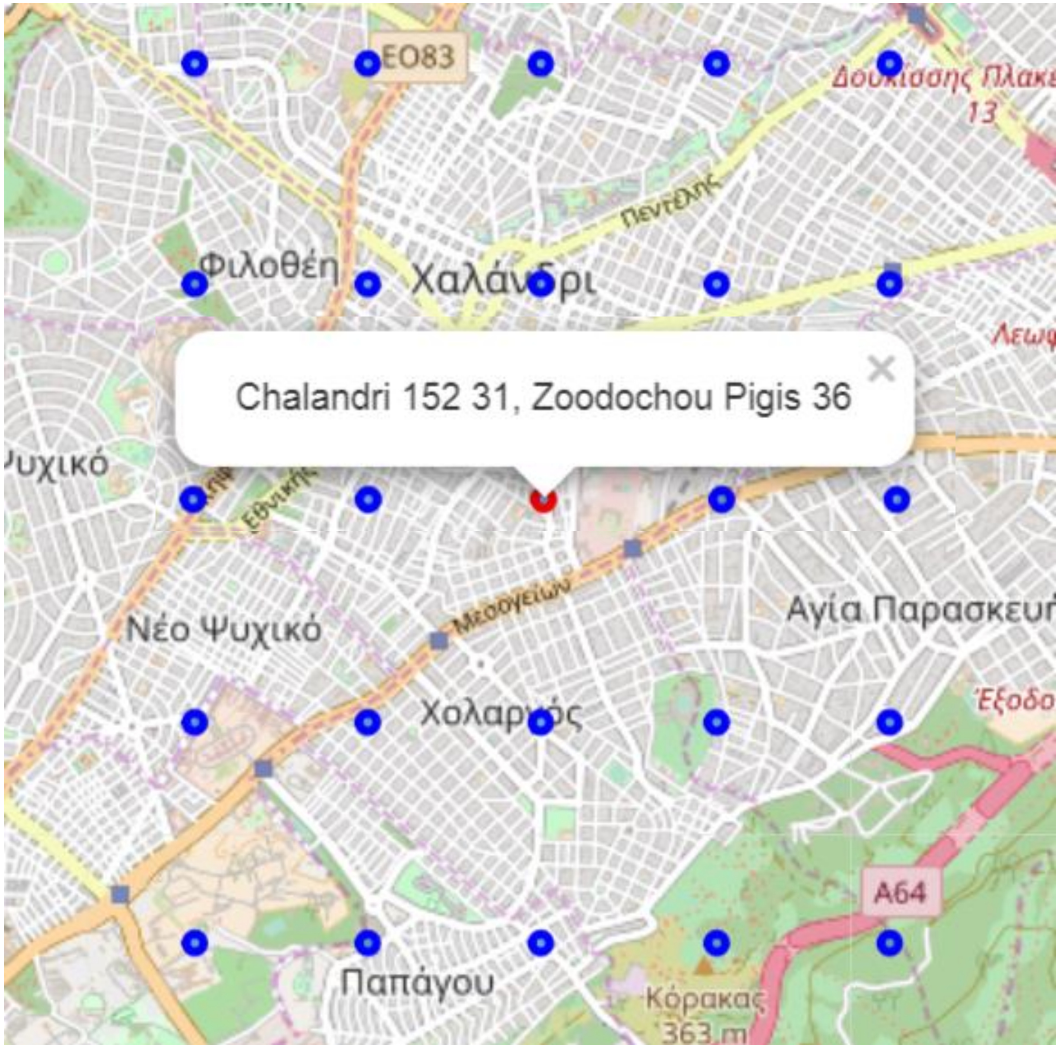


```
AT LATITUDE 40 DEGREES (NORTH OR SOUTH)
One degree of latitude =  111.03 km
One minute of latitude =    1.85 km
One degree of longitude =  85.39 km
One minute of longitude =   1.42 km
```

- Knowing the points of the grid allows us to use the Google API and retrieve the names of the places on the grid, like the street, the neighborhood, the city and the country. Among the various addresses that might fall into the same neighborhood, we simply keep only the first. Anyhow, this is mostly for informative purposes.
- For each one of the points on the virtual grid, we call the Foursquare API and retrieve the containing venues, with a limit of 100 per request.
- Using the venues type, we create a profile for each neighborhood and finally we run a clustering algorithm to derive the basic, distinct profiles of all the neighborhoods around our main point of interest.

Example of a virtual grid on a map

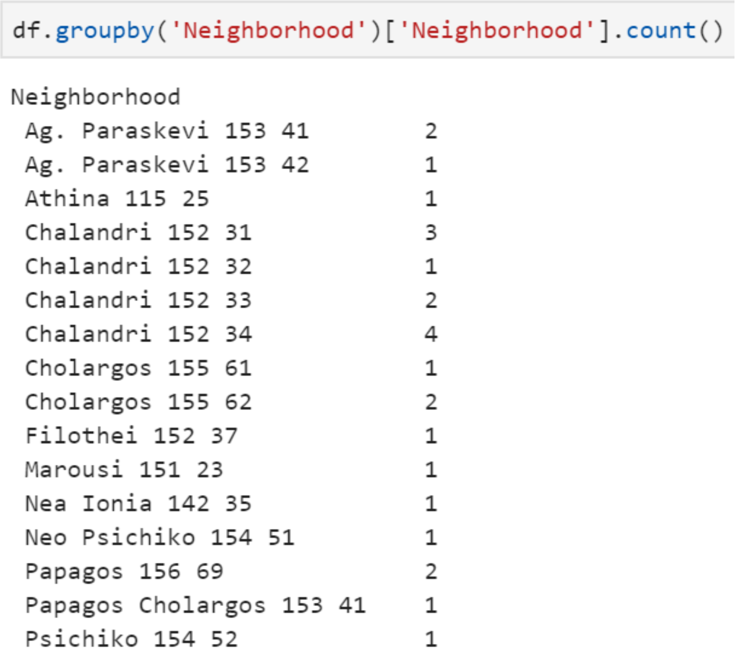
Example of a virtual grid on a real map



Methodology (1 of 3)

Methodology Overview

The solution starts with retrieving the exact latitude and longitude of the given real world address. For this step, it uses the Google geocode API.
It then goes on to calculate a list of the 25 point on the virtual grid around the central point. Some basic exploratory analysis shows the street, neighborhoods, city and country of the respective points on the map. The grid in the submitted notebook is 5x5 in size with a distance of 1/100 of a degree between them. These parameters can be adjusted, depending on the area and the landscape around the central point, especially when there is not much diversification in the neighborhood names. The following table shows the neighborhoods around the selected central point of the submitted document.



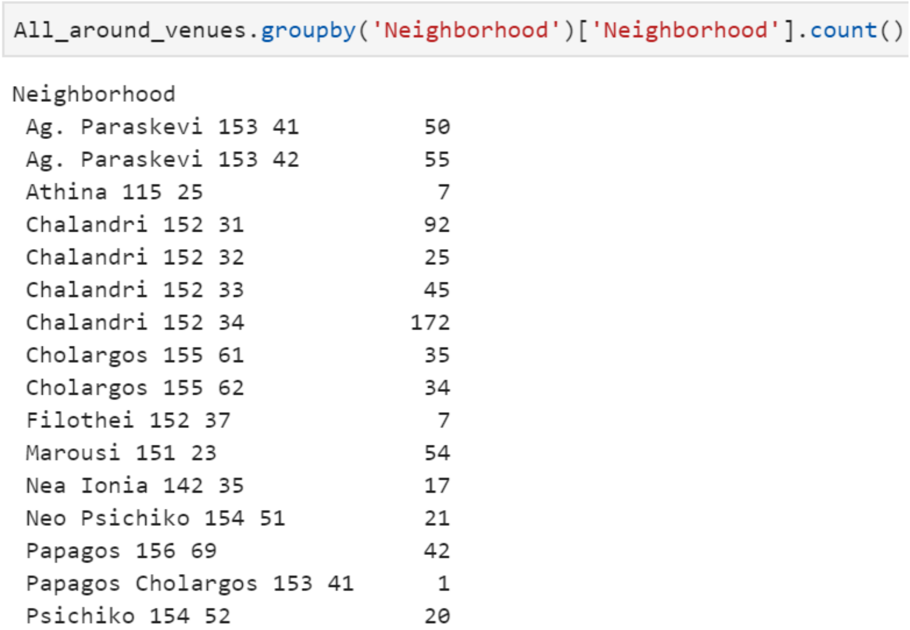
```
df.groupby('Neighborhood')['Neighborhood'].count()
```

```
Neighborhood
 Ag. Paraskevi 153 41        2
 Ag. Paraskevi 153 42        1
 Athina 115 25               1
 Chalandri 152 31            3
 Chalandri 152 32            1
 Chalandri 152 33            2
 Chalandri 152 34            4
 Cholargos 155 61            1
 Cholargos 155 62            2
 Filothei 152 37             1
 Marousi 151 23              1
 Nea Ionia 142 35            1
 Neo Psichiko 154 51         1
 Papagos 156 69              2
 Papagos Cholargos 153 41    1
 Psichiko 154 52             1
```

Methodology (2 of 3)

Methodology Overview

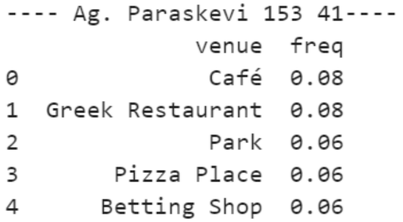
The next step is to retrieve the list of venues for each one of the 25 points in the list. The Foursquare API is used for this purpose. We can then explore the concentration of available venues in each neighborhood:



```
All_around_venues.groupby('Neighborhood')['Neighborhood'].count()
```

```
Neighborhood
 Ag. Paraskevi 153 41          50
 Ag. Paraskevi 153 42          55
 Athina 115 25                  7
 Chalandri 152 31              92
 Chalandri 152 32              25
 Chalandri 152 33              45
 Chalandri 152 34             172
 Cholargos 155 61              35
 Cholargos 155 62              34
 Filothei 152 37                7
 Marousi 151 23                54
 Nea Ionia 142 35              17
 Neo Psichiko 154 51           21
 Papagos 156 69                42
 Papagos Cholargos 153 41       1
 Psichiko 154 52               20
```

For each neighborhood, we also generate the list of counters for the number of venues by venue type and convert to percentages. We can then show the top 5 venue types for each neighborhood:



```
---- Ag. Paraskevi 153 41----
              venue  freq
0              Café  0.08
1  Greek Restaurant  0.08
2              Park  0.06
3       Pizza Place  0.06
4      Betting Shop  0.06
```

Methodology (3 of 3)

Methodology Overview

The last step is to run a clustering algorithm which is expected to group together the neighborhoods in the virtual grid that are more similar to each other and therefore, provide a good overview of the overall patterns and profile towards all directions around the focal point. The results can be visualized on the map

Alternatively, the top 10 venue types for the neighborhoods in each cluster might be displayed, thus giving a good understanding of the venues that are mostly located in the areas all around.

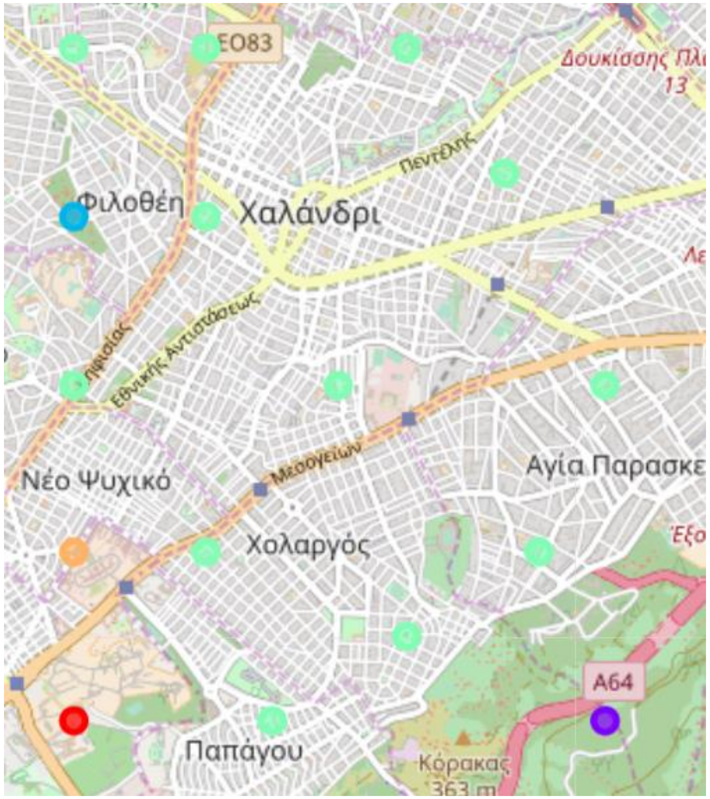
Note: Like all clustering algorithms, the labels of the clusters are not provided and they must be derived from the profile of the cluster members.

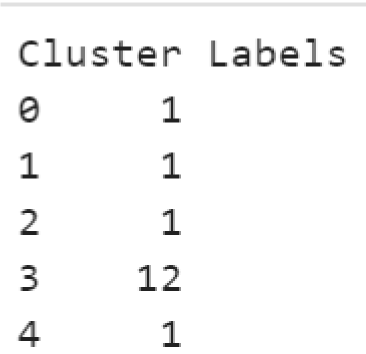
Results

Results

The clustering algorithm was instructed to generate k=5 groups, because this number is aligned to both the number of available neighborhoods in the grid and the interpretation effort required to comprehend the profile of the derived clusters.

In the case of the submitted notebook, the majority of the neighborhoods were grouped into a single cluster (light green), with the exception inevitably of 4 neighborhoods which constituted the remaining 4 clusters (blue, orange, red and purple).





```
Cluster Labels
0       1
1       1
2       1
3      12
4       1
```

Discussion

Discussion

There are a few observations that become obvious while the data get enriched and evolve with the addition of the neighborhoods and the venues inside these neighborhoods. Specifically:

- The exercise makes sense when there is a variety of neighborhoods around the central point. If the central point is pinned in the middle of nowhere, most probably the results will be meaningless.
- The same is true with regards to the number and diversification of the venues and their type. If the majority of the neighborhoods in the virtual grid do not have many venues in Foursquare, then the results might be trivial or incomplete.
- Finally, it should be noted that adjacent points on the grid which fall into the same neighborhood are implicitly grouped together into a single point, representing all the points in this common neighborhood. If this process ends up in too few neighborhoods, then the results do not bear much value.

The above 3 challenges might require remedial actions, in order for the entire process and methodology to make sense. In particular, one should either change the location of the central point, or expand the grid by increasing the distance between the points, or extend the grid by adding more points (e.g. make it 6 X 6 or 7 X 7). Alternatively, it might be necessary to change the number of required clusters in the respective algorithms to a smaller number if there is less diversity or to a higher number if there are more neighborhoods and many venues.

As a last note, it would be interesting to run the same exercise for a central point in New York, or increase the distance between the points so that the grid covers a complete Metropolitan area.

Conclusion

Conclusion

The overall exercise seems to work well and provide some useful insights for the area of interest.

In the particular example included in the submitted notebook, the key message is that around the central point (which happens to be my own neighborhood in Athens), the majority of the adjacent places have a very similar profile, silhouette and outline.

On the other hand, as we move farther from the central point, the areas change drastically and include commuting centers, parks and scenic lookout areas, movie theaters and plazas, or mostly restaurants.