IBM Data Science Professional Certificate

Author: Manuel Cirulli

Date: 31/01/2021

DATA SCIENCE FOR RETAIL LOCATION SELECTION IN MILAN

Introduction

In the retail industry, the selection of a new location is an important strategic decision that can largely determine the success of the new store and affect the company's bottom line for years to come. Because of its importance, this problem has received a lot of attention from academia and industry practitioners.

With a population of around 1.4 million people, over 3.26 million in the metropolitan area, and over 5.2 million people in its urban area, Milan is the second-most populous city in Italy after Rome and one of the largest urban areas in the EU.

The goal of this project is to leverage open source data science tools and the Foursquare location data to identify the best areas to open a new supermarket in the city of Milan.

Data

The data I used for this project is all available online. For the general data about Milan city population, boroughs, and districts I scraped the Wikipedia website at the following link:

https://en.wikipedia.org/wiki/Municipalities_of_Milan

For the coordinates of each borough, I turned to the Milan Open Data website (available at https://dati.comune.milano.it/), a useful open data web resource curated by the city of Milan where a lot of useful information about the city can be found. Below the link to the JSON file on the portal from which I uploaded the coordinates to the data frame:

https://dati.comune.milano.it/dataset/a7f54a9a-8331-4825-bc52-6a69a10b0bd3/resource/162ffc23-419f-420d-b14f-bccfe28d920a/download/municipi_sedi__final.json

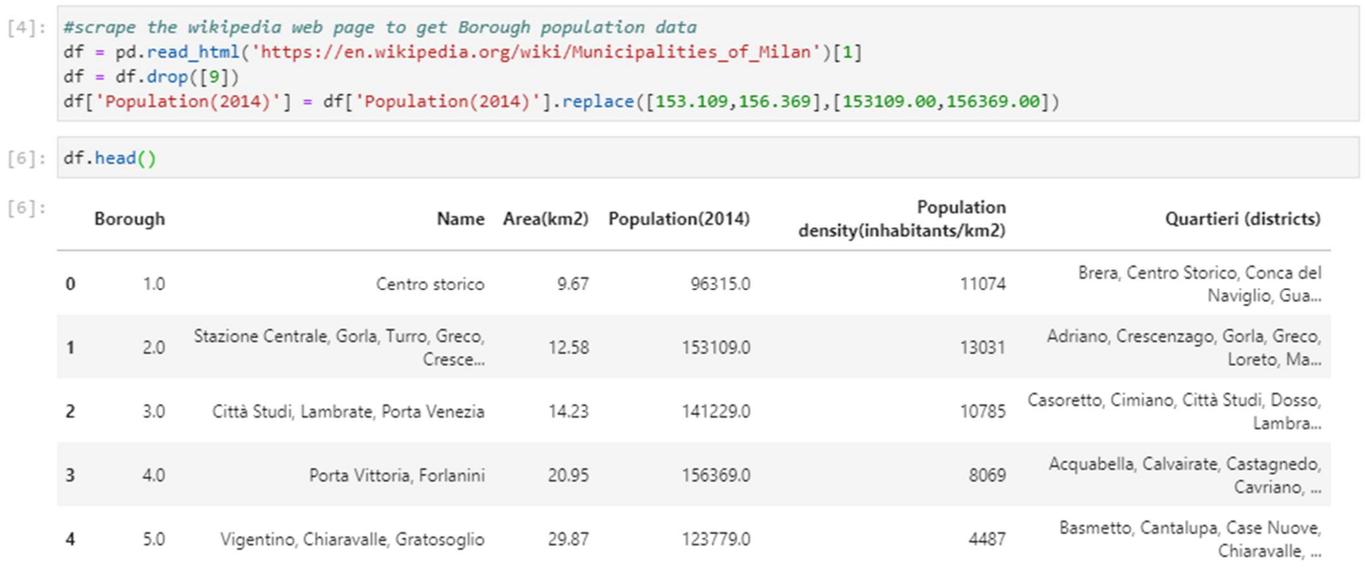
Finally, I used the Foursquare API (available at: https://developer.foursquare.com/) to get the data about supermarkets and other food stores in Milan.

Methodology

The goal of this project was to suggest the most attractive area (or areas) to open a new store in the city of Milan to a potential food retailer based on publicly available data of population and the competition.

To do this, I needed to find reliable data about the population in the city of Milan for each Borough/district. I found a table summarizing the data I needed on Wikipedia (https://en.wikipedia.org/wiki/Municipalities_of_Milan), so I imported the table to a data frame called *df* using *pandas.* For some boroughs, the values were mistakenly expressed in decimal form in the original table, so I had to use a replace method to change the values.

Below a screenshot of the code I used, and the first 5 rows of data frame *df*:



```
[4]: #scrape the wikipedia web page to get Borough population data
df = pd.read_html('https://en.wikipedia.org/wiki/Municipalities_of_Milan')[1]
df = df.drop([9])
df['Population(2014)'] = df['Population(2014)'].replace([153.109,156.369],[153109.00,156369.00])
```

```
[6]: df.head()
```

[6]:

	Borough	Name	Area(km2)	Population(2014)	Population density(inhabitants/km2)	Quartieri (districts)
0	1.0	Centro storico	9.67	96315.0	11074	Brera, Centro Storico, Conca del Naviglio, Gua...
1	2.0	Stazione Centrale, Gorla, Turro, Greco, Cresce...	12.58	153109.0	13031	Adriano, Crescenzago, Gorla, Greco, Loreto, Ma...
2	3.0	Città Studi, Lambrate, Porta Venezia	14.23	141229.0	10785	Casoretto, Cimiano, Città Studi, Dosso, Lambra...
3	4.0	Porta Vittoria, Forlanini	20.95	156369.0	8069	Acquabella, Calvairate, Castagnedo, Cavriano, ...
4	5.0	Vigentino, Chiaravalle, Gratosoglio	29.87	123779.0	4487	Basmetto, Cantalupa, Case Nuove, Chiaravalle, ...

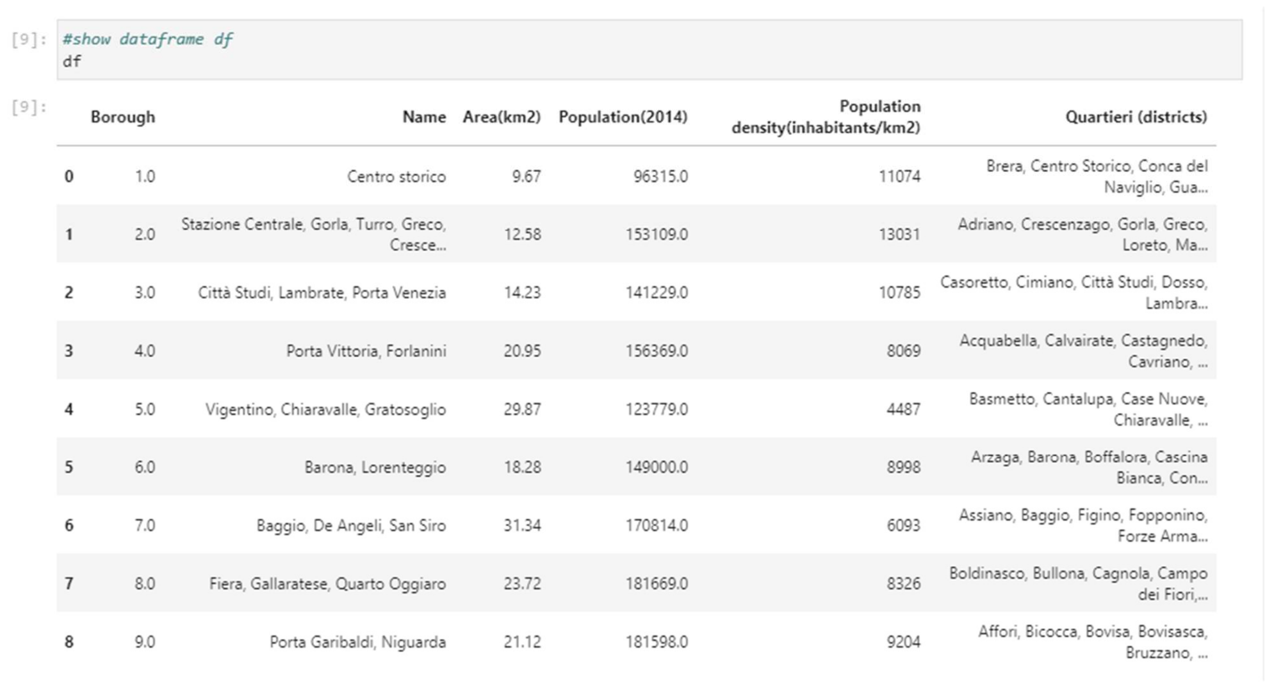
The underlying assumption of the analysis is that, given two areas with the same population, the area with fewer options to buy would be more attractive to a new potential food retailer.

However, we should remember that people can move among areas to find alternatives that better suit their preferences, so it would be a mistake to consider areas as perfectly independent from one another. This is even truer for adjacent areas, areas with limited extension, or where people routinely commute for work or other reasons.

Taking fully into account all these aspects would have greatly complicated the analysis, and it is beyond the scope of this project.

However, considering what I mentioned above, basing the location analysis on single districts did not seem like the best choice, because of the limited area each district covers. Boroughs, on the other hand, span over wider areas and include multiple districts. Hence, boroughs were selected as the cornerstone for the location analysis.

A simple look at dataframe *df* reveals that the city of Milan is divided into 9 Boroughs, each of them containing several city districts.



```
[9]: #show dataframe df
     df
```

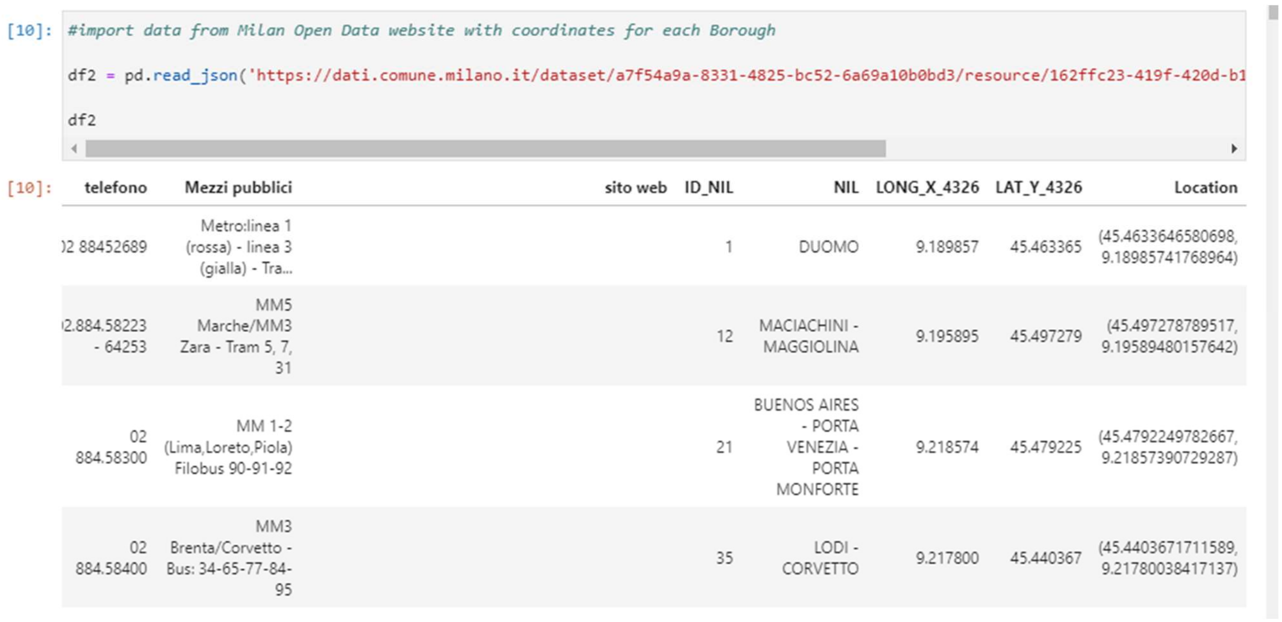
[9]:

	Borough	Name	Area(km2)	Population(2014)	Population density(inhabitants/km2)	Quartieri (districts)
0	1.0	Centro storico	9.67	96315.0	11074	Brera, Centro Storico, Conca del Naviglio, Gua...
1	2.0	Stazione Centrale, Gorla, Turro, Greco, Cresce...	12.58	153109.0	13031	Adriano, Crescenzago, Gorla, Greco, Loreto, Ma...
2	3.0	Città Studi, Lambrate, Porta Venezia	14.23	141229.0	10785	Casoretto, Cimiano, Città Studi, Dosso, Lambra...
3	4.0	Porta Vittoria, Forlanini	20.95	156369.0	8069	Acquabella, Calvairate, Castagnedo, Cavriano, ...
4	5.0	Vigentino, Chiaravalle, Gratosoglio	29.87	123779.0	4487	Basmetto, Cantalupa, Case Nuove, Chiaravalle, ...
5	6.0	Barona, Lorenteggio	18.28	149000.0	8998	Arzaga, Barona, Boffalora, Cascina Bianca, Con...
6	7.0	Baggio, De Angeli, San Siro	31.34	170814.0	6093	Assiano, Baggio, Figino, Fopponino, Forze Arma...
7	8.0	Fiera, Gallaratese, Quarto Oggiaro	23.72	181669.0	8326	Boldinasco, Bullona, Cagnola, Campo dei Fiori,...
8	9.0	Porta Garibaldi, Niguarda	21.12	181598.0	9204	Affori, Bicocca, Bovisa, Bovisasca, Bruzzano, ...

While very useful and informative, dataframe df still misses a key element: Boroughs are not georeferenced.

The geographic coordinates of each Borough were not available on the Wikipedia web page, but the city of Milan maintains an Open Data website with many interesting freely available datasets and statistics about the city: one of these datasets happened to include coordinates I was looking for.

I imported the data to a new dataframe called *df2* directly from the website using *pandas*. Below a screenshot of the code and the first few rows of dataframe *df2*:



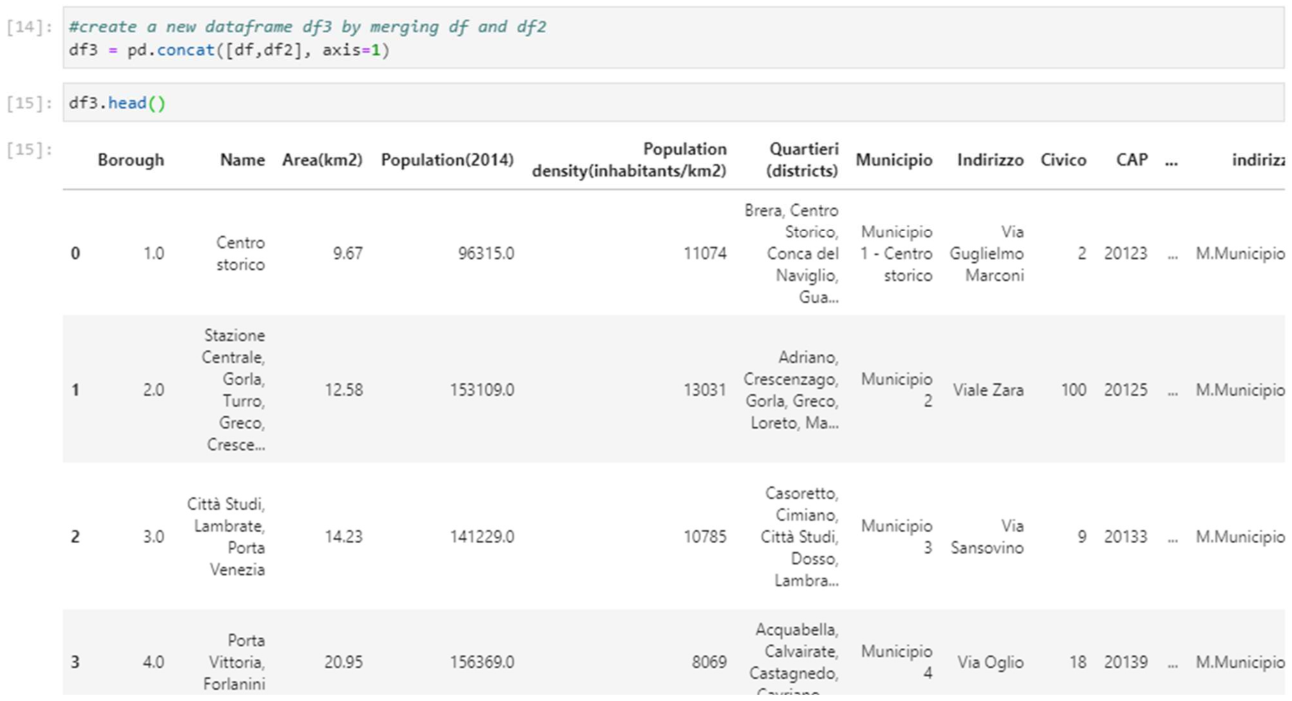
```
[10]: #import data from Milan Open Data website with coordinates for each Borough

      df2 = pd.read_json('https://dati.comune.milano.it/dataset/a7f54a9a-8331-4825-bc52-6a69a10b0bd3/resource/162ffc23-419f-420d-b1

      df2
```

[10]:

telefono	Mezzi pubblici	sito web	ID_NIL	NIL	LONG_X_4326	LAT_Y_4326	Location
)2 88452689	Metro:linea 1 (rossa) - linea 3 (gialla) - Tra...		1	DUOMO	9.189857	45.463365	(45.4633646580698, 9.18985741768964)
)2.884.58223 - 64253	MM5 Marche/MM3 Zara - Tram 5, 7, 31		12	MACIACHINI - MAGGIOLINA	9.195895	45.497279	(45.497278789517, 9.19589480157642)
02 884.58300	MM 1-2 (Lima,Loreto,Piola) Filobus 90-91-92		21	BUENOS AIRES - PORTA VENEZIA - PORTA MONFORTE	9.218574	45.479225	(45.4792249782667, 9.21857390729287)
02 884.58400	MM3 Brenta/Corvetto - Bus: 34-65-77-84-95		35	LODI - CORVETTO	9.217800	45.440367	(45.4403671711589, 9.21780038417137)

Then, using the *concat* method, I created a new dataframe *df3* by merging the two dataframes *df* and *df2* into one, to have all the data I needed in one dataframe.

Below a screenshot of the code, and the first few rows dataframe *df3*:

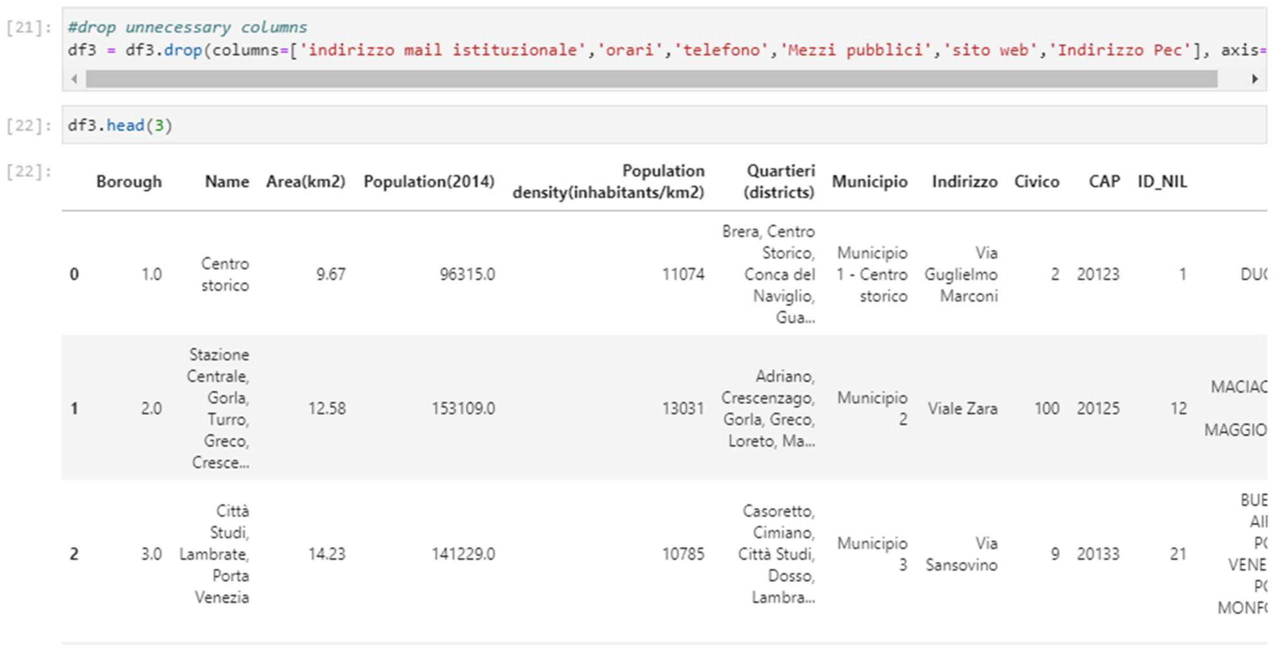


```
[14]: #create a new dataframe df3 by merging df and df2
      df3 = pd.concat([df,df2], axis=1)

[15]: df3.head()
```

[15]:

	Borough	Name	Area(km2)	Population(2014)	Population density(inhabitants/km2)	Quartieri (districts)	Municipio	Indirizzo	Civico	CAP	...	indiriz
0	1.0	Centro storico	9.67	96315.0	11074	Brera, Centro Storico, Conca del Naviglio, Gua...	Municipio 1 - Centro storico	Via Guglielmo Marconi	2	20123	...	M.Municipio
1	2.0	Stazione Centrale, Gorla, Turro, Greco, Cresce...	12.58	153109.0	13031	Adriano, Crescenzago, Gorla, Greco, Loreto, Ma...	Municipio 2	Viale Zara	100	20125	...	M.Municipio
2	3.0	Città Studi, Lambrate, Porta Venezia	14.23	141229.0	10785	Casoretto, Cimiano, Città Studi, Dosso, Lambra...	Municipio 3	Via Sansovino	9	20133	...	M.Municipio
3	4.0	Porta Vittoria, Forlanini	20.95	156369.0	8069	Acquabella, Calvairate, Castagnedo,	Municipio 4	Via Oglio	18	20139	...	M.Municipio

Then, I dropped unnecessary and redundant columns from the combined dataframe:



```
[21]: #drop unnecessary columns
      df3 = df3.drop(columns=['indirizzo mail istituzionale','orari','telefono','Mezzi pubblici','sito web','Indirizzo Pec'], axis=

[22]: df3.head(3)
```

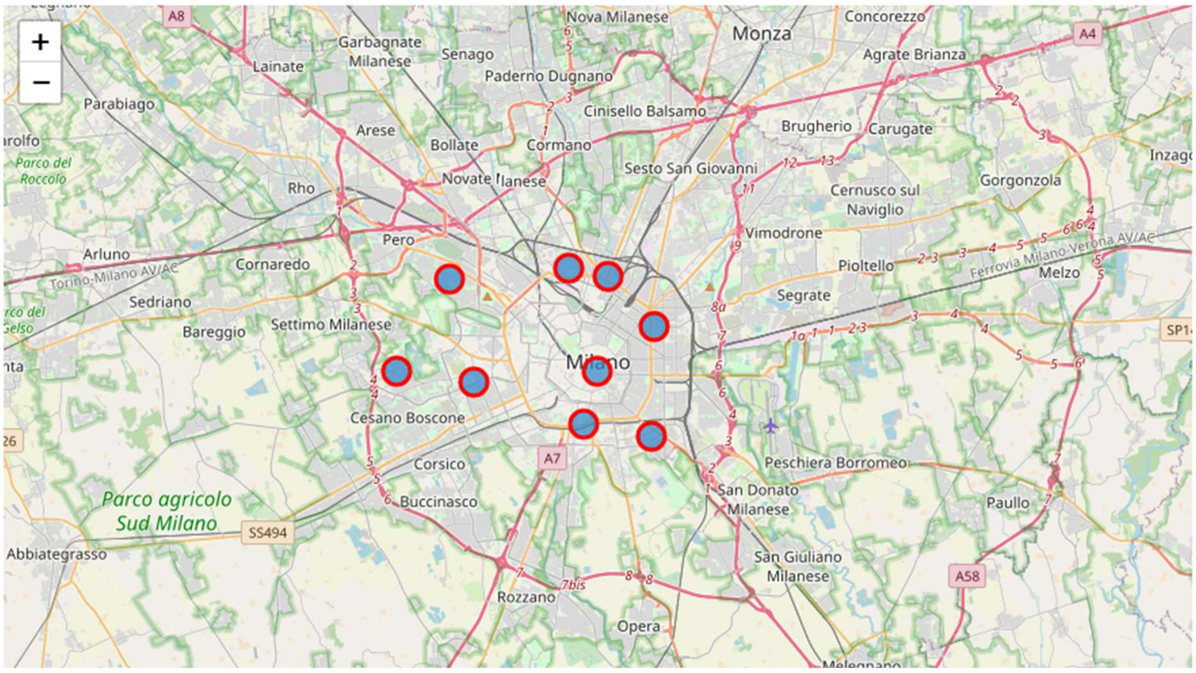
[22]:

	Borough	Name	Area(km2)	Population(2014)	Population density(inhabitants/km2)	Quartieri (districts)	Municipio	Indirizzo	Civico	CAP	ID_NIL	
0	1.0	Centro storico	9.67	96315.0	11074	Brera, Centro Storico, Conca del Naviglio, Gua...	Municipio 1 - Centro storico	Via Guglielmo Marconi	2	20123	1	DU
1	2.0	Stazione Centrale, Gorla, Turro, Greco, Cresce...	12.58	153109.0	13031	Adriano, Crescenzago, Gorla, Greco, Loreto, Ma...	Municipio 2	Viale Zara	100	20125	12	MACIAC MAGGIO
2	3.0	Città Studi, Lambrate, Porta Venezia	14.23	141229.0	10785	Casoretto, Cimiano, Città Studi, Dosso, Lambra...	Municipio 3	Via Sansovino	9	20133	21	BUE AII P VENE P MONF

At this point, using the georeferenced dataframe *df3* and the *folium* library, I was able to create a map of Milan with superimposed markers of each borough:

```
map_milan
```

[26]:



To assess the attractiveness of each borough for a food retailer searching for a location to open a supermarket, the following data was needed:

- Potential customers: population of each borough;
- Competition: number of supermarkets and other food stores in each borough;

While I could get the population data from my newly built dataframe df3, data about supermarkets and other food stores for each borough was still not available: that's when the Foursquare API comes into play.

After defining the function *getNearbyVenues* I leveraged the Fourquare API to retrieve all the venues (shops, stores, restaurants, etc.) within a 3000 meters radius from each borough in Milan. In the function, I set the maximum number of locations for each borough equal to 100.

The application of the *getNearbyVenues* function to the dataframe *df3* returned a total of 881 venues throughout Milan:

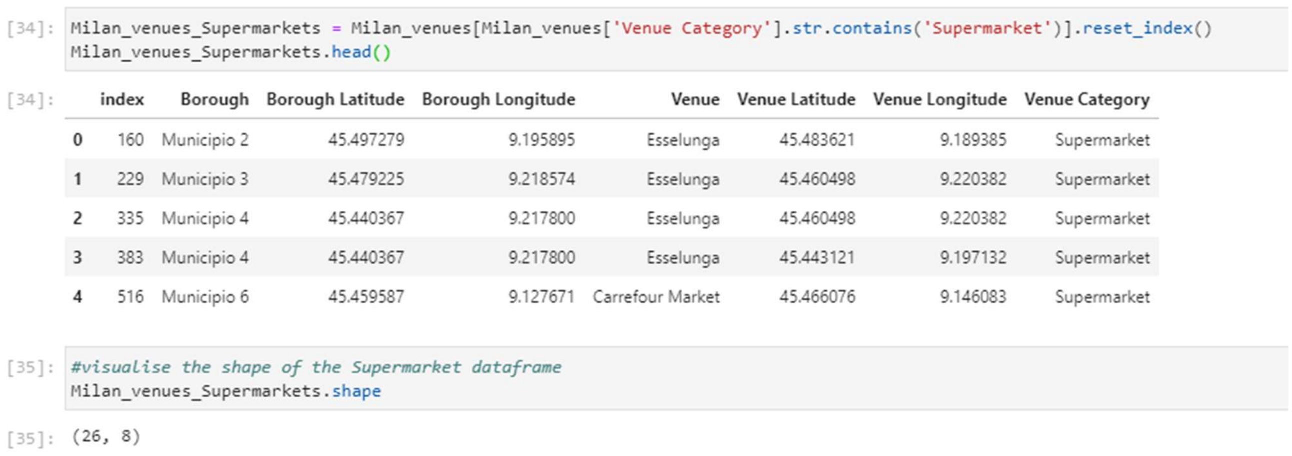
[30]:
```
#get the venues in Milan
Milan_venues = getNearbyVenues(names=df3['Municipio'],
                               latitudes=df3['LAT_Y_4326'],
                               longitudes=df3['LONG_X_4326']
                              )
```

```
Municipio 1 - Centro storico
Municipio 2
Municipio 3
Municipio 4
Municipio 5
Municipio 6
Municipio 7
Municipio 8
Municipio 9
```

[31]:
```
#visualise the shape of the Milan_venues dataframe

Milan_venues.shape
```

[31]:
```
(881, 7)
```

The *Milan_venues* dataframe includes all kinds of venues (shops, supermarkets, restaurants, etc.). However, I was only interested in supermarkets and other food stores. Hence, I created a new dataframe by filtering the *Milan_venues* dataframe for venue category “Supermarket”:



```
[34]: Milan_venues_Supermarkets = Milan_venues[Milan_venues['Venue Category'].str.contains('Supermarket')].reset_index()
      Milan_venues_Supermarkets.head()
```

[34]:

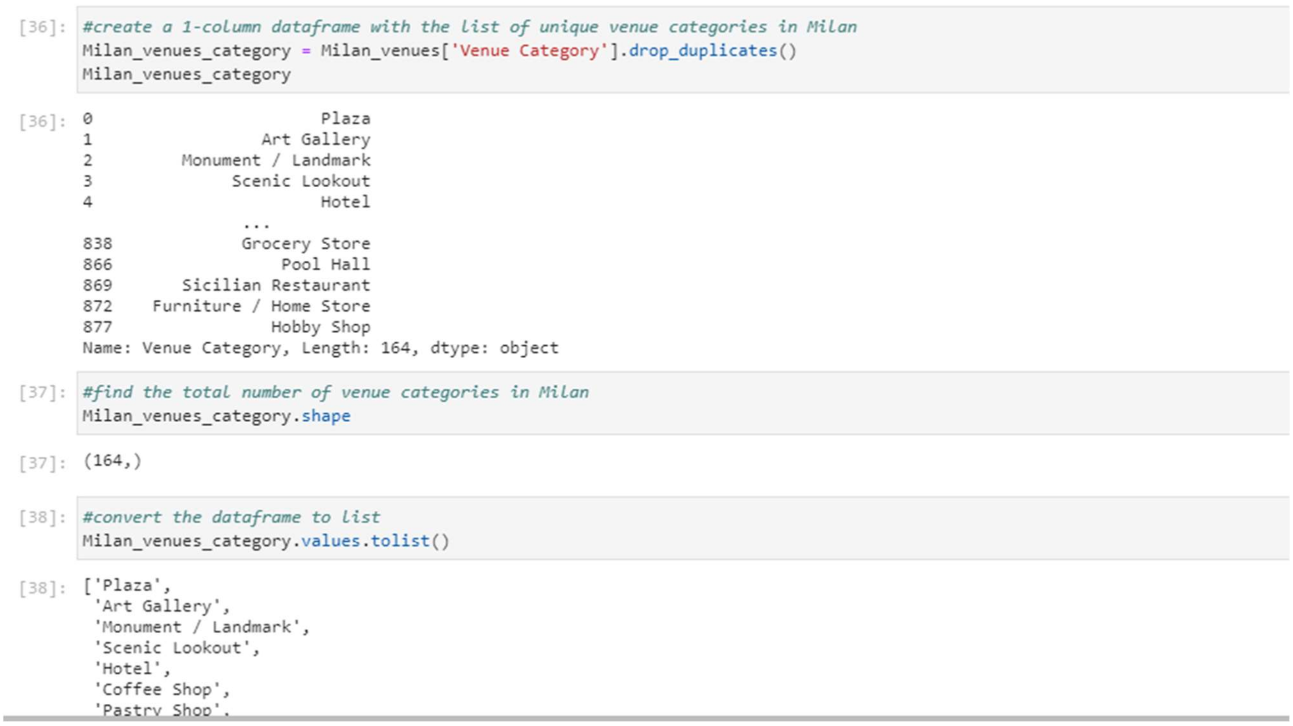
	index	Borough	Borough Latitude	Borough Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	160	Municipio 2	45.497279	9.195895	Esselunga	45.483621	9.189385	Supermarket
1	229	Municipio 3	45.479225	9.218574	Esselunga	45.460498	9.220382	Supermarket
2	335	Municipio 4	45.440367	9.217800	Esselunga	45.460498	9.220382	Supermarket
3	383	Municipio 4	45.440367	9.217800	Esselunga	45.443121	9.197132	Supermarket
4	516	Municipio 6	45.459587	9.127671	Carrefour Market	45.466076	9.146083	Supermarket

```
[35]: #visualise the shape of the Supermarket dataframe
      Milan_venues_Supermarkets.shape

[35]: (26, 8)
```

A total of 26 supermarkets were returned by the API call.

While supermarkets are certainly the most relevant kind of competition, other venue categories may be relevant as well. So, I create a list of all venue categories and went through the list in search of other useful categories:

Below a screenshot with all the passages to get to the list:



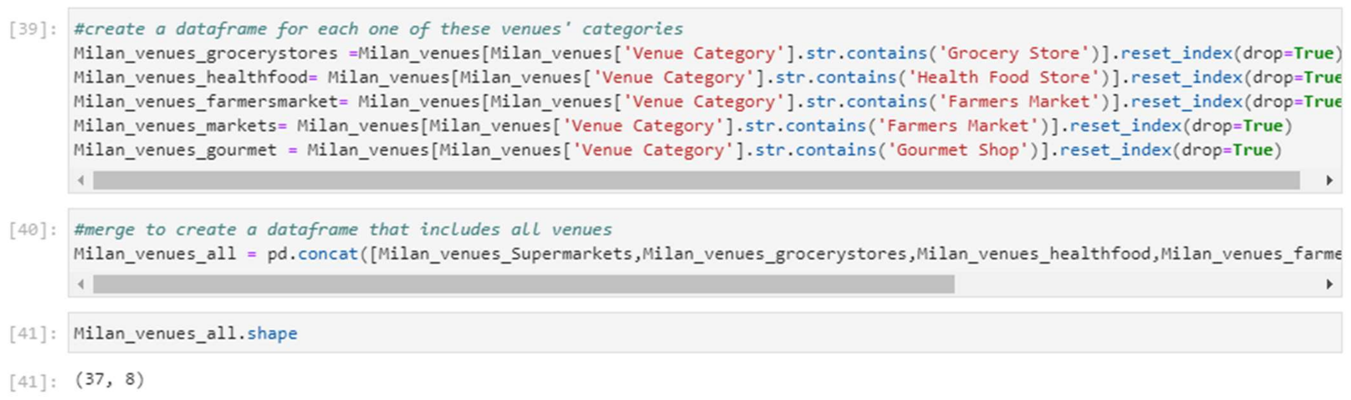
```
[36]: #create a 1-column dataframe with the list of unique venue categories in Milan
      Milan_venues_category = Milan_venues['Venue Category'].drop_duplicates()
      Milan_venues_category

[36]: 0                       Plaza
      1                 Art Gallery
      2          Monument / Landmark
      3              Scenic Lookout
      4                       Hotel
                             ...
      838            Grocery Store
      866                Pool Hall
      869       Sicilian Restaurant
      872      Furniture / Home Store
      877               Hobby Shop
      Name: Venue Category, Length: 164, dtype: object

[37]: #find the total number of venue categories in Milan
      Milan_venues_category.shape

[37]: (164,)

[38]: #convert the dataframe to list
      Milan_venues_category.values.tolist()

[38]: ['Plaza',
       'Art Gallery',
       'Monument / Landmark',
       'Scenic Lookout',
       'Hotel',
       'Coffee Shop',
       'Pastry Shop',
```

As expected, there were other relevant categories in the list, namely:

- Grocery Store
- Health Food Store
- Farmers Market

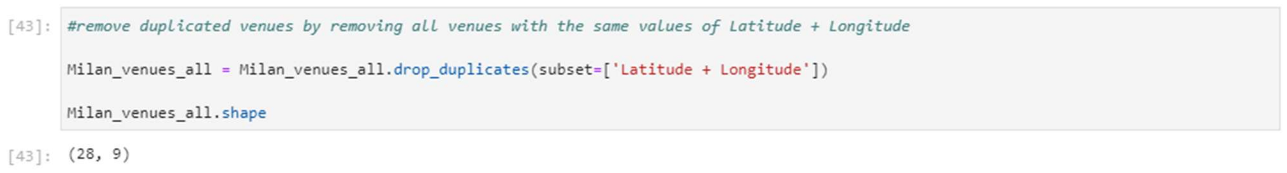
- Market
- Gourmet Shop

Based on these findings, I created a new dataframe for each one of the newly identified categories, and then I merged all venues into a single dataframe:



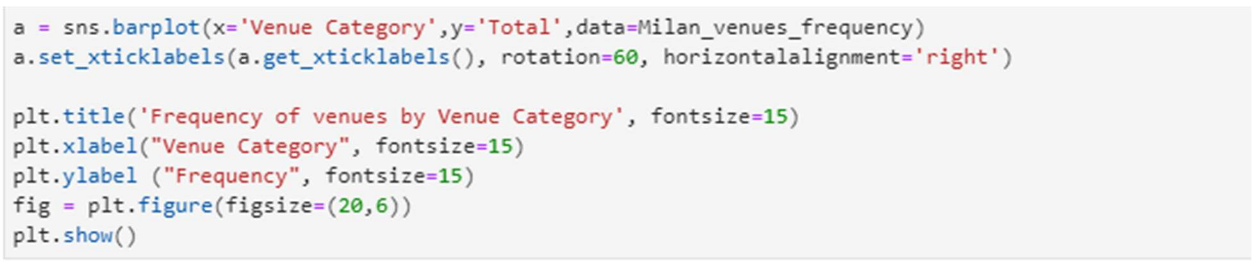
```
[39]: #create a dataframe for each one of these venues' categories
      Milan_venues_grocerystores =Milan_venues[Milan_venues['Venue Category'].str.contains('Grocery Store')].reset_index(drop=True)
      Milan_venues_healthfood= Milan_venues[Milan_venues['Venue Category'].str.contains('Health Food Store')].reset_index(drop=True
      Milan_venues_farmersmarket= Milan_venues[Milan_venues['Venue Category'].str.contains('Farmers Market')].reset_index(drop=True
      Milan_venues_markets= Milan_venues[Milan_venues['Venue Category'].str.contains('Farmers Market')].reset_index(drop=True)
      Milan_venues_gourmet = Milan_venues[Milan_venues['Venue Category'].str.contains('Gourmet Shop')].reset_index(drop=True)

[40]: #merge to create a dataframe that includes all venues
      Milan_venues_all = pd.concat([Milan_venues_Supermarkets,Milan_venues_grocerystores,Milan_venues_healthfood,Milan_venues_farme

[41]: Milan_venues_all.shape

[41]: (37, 8)
```

After removing duplicates from the dataframe, only 28 venues remained.

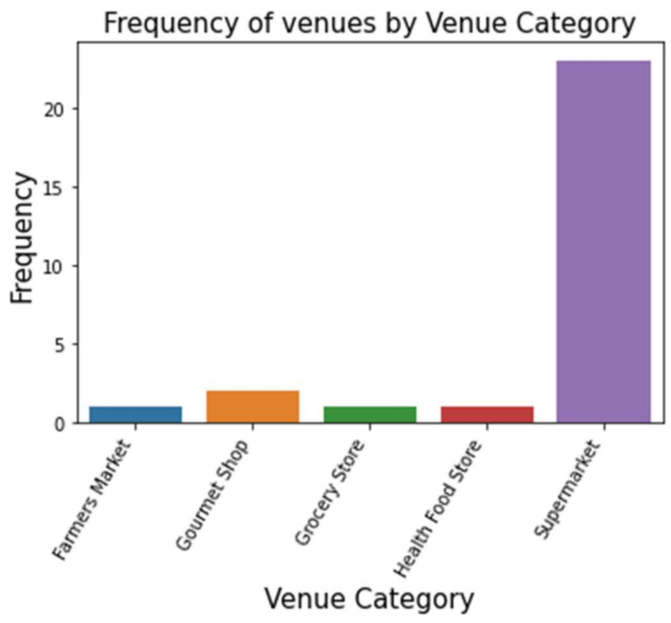


```
[43]: #remove duplicated venues by removing all venues with the same values of Latitude + Longitude

      Milan_venues_all = Milan_venues_all.drop_duplicates(subset=['Latitude + Longitude'])

      Milan_venues_all.shape

[43]: (28, 9)
```

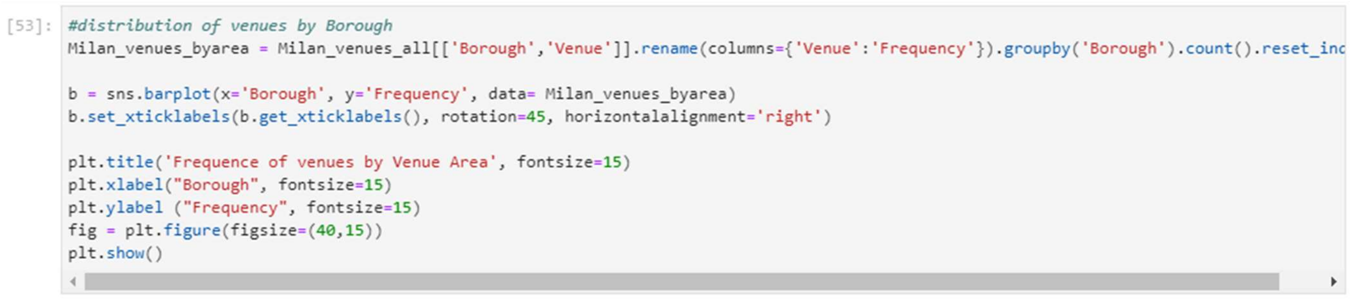
Then, I calculated the frequency of each venue, and plotted a bar chart using the seaborn library:



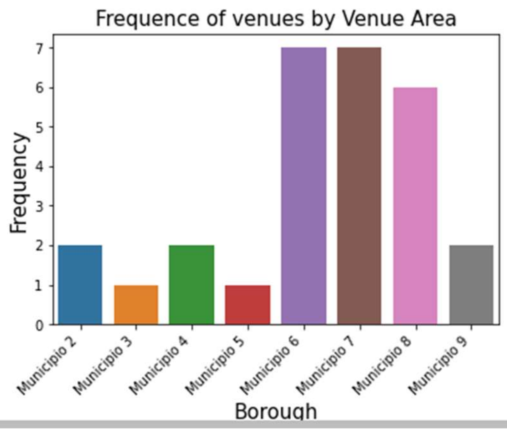
```
a = sns.barplot(x='Venue Category',y='Total',data=Milan_venues_frequency)
a.set_xticklabels(a.get_xticklabels(), rotation=60, horizontalalignment='right')

plt.title('Frequency of venues by Venue Category', fontsize=15)
plt.xlabel("Venue Category", fontsize=15)
plt.ylabel ("Frequency", fontsize=15)
fig = plt.figure(figsize=(20,6))
plt.show()
```



As the plot clearly shows, supermarkets are the most frequent venue category by far, with the others only occupying a small fraction of the total. Below a plot of the number of venues in each area:



```
[53]: #distribution of venues by Borough
      Milan_venues_byarea = Milan_venues_all[['Borough','Venue']].rename(columns={'Venue':'Frequency'}).groupby('Borough').count().reset_inc

      b = sns.barplot(x='Borough', y='Frequency', data= Milan_venues_byarea)
      b.set_xticklabels(b.get_xticklabels(), rotation=45, horizontalalignment='right')

      plt.title('Frequence of venues by Venue Area', fontsize=15)
      plt.xlabel("Borough", fontsize=15)
      plt.ylabel ("Frequency", fontsize=15)
      fig = plt.figure(figsize=(40,15))
      plt.show()
```



The plot shows something really interesting: there are no venues in the first borough (*Municipio 1 – Centro Storico*). This is surprising because this borough lies at the heart of the city, spanning over its entire historical center, including many of Milan's most popular venues.

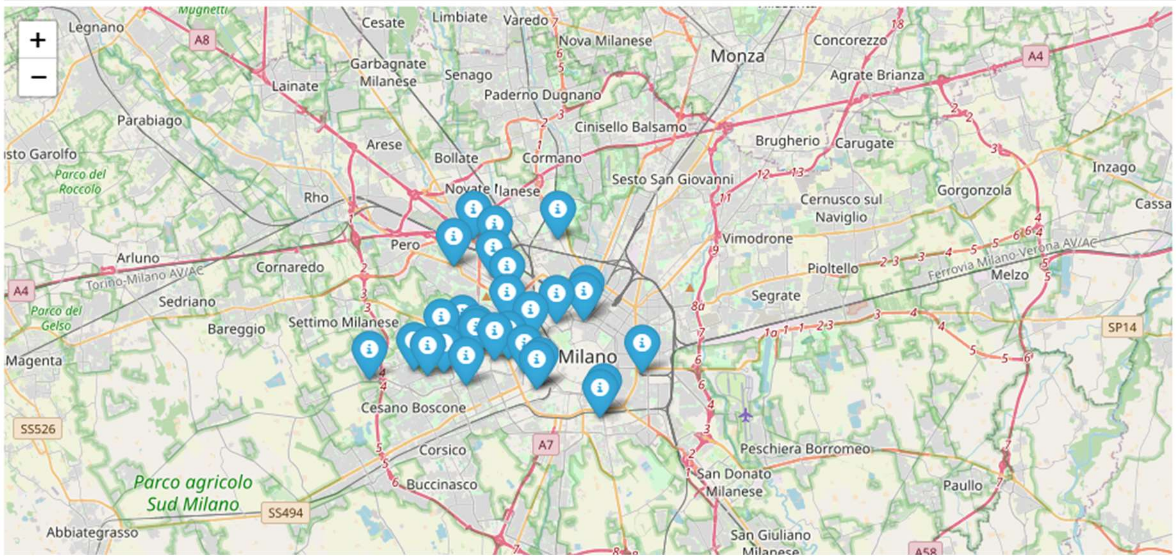
One explanation for this could be that the Foursquare API call missed certain venues, or included them into a different borough because they didn't fall within the radius defined into the *GetNearbyVenues* function. It could also mean that certain venues simply haven't been mapped by Foursquare users yet.

Another interpretation could be that supermarkets tend to be positioned mostly outside of the city center, and even the people who live in *Municipio 1* are more likely to shop for food in an adjacent borough rather than their own.

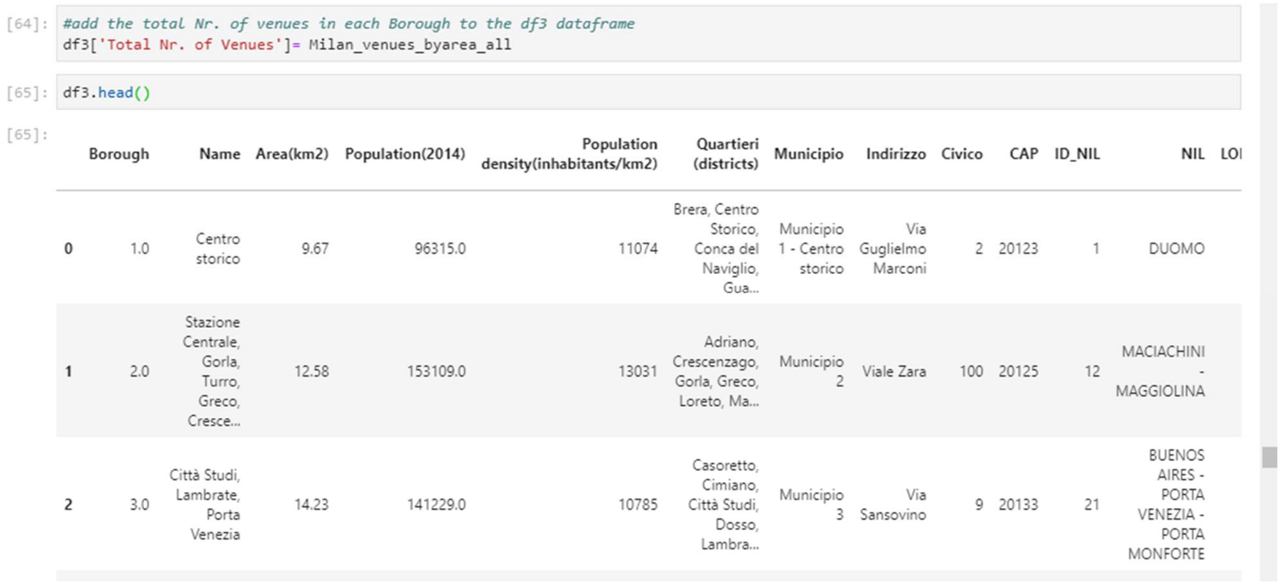
Continuing with the analysis, I created a Folium map displaying all the food store venues in Milan:

```
map_venues_all
```

[56]:



Then, I added a column containing the total number of venues in each borough to the *df3* dataframe:



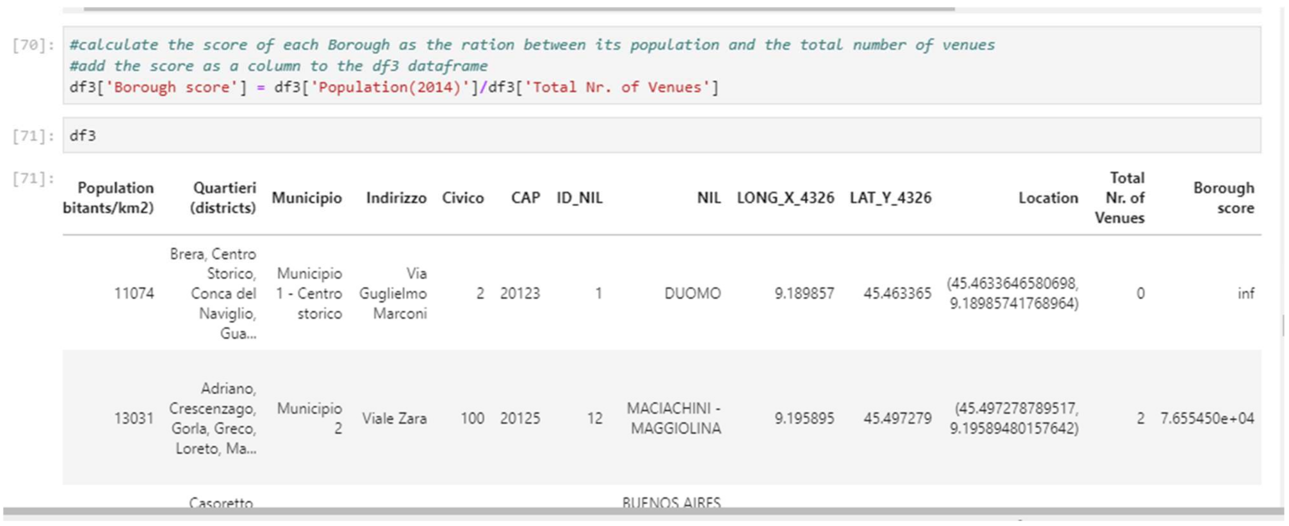
[64]:
```
#add the total Nr. of venues in each Borough to the df3 dataframe
df3['Total Nr. of Venues']= Milan_venues_byarea_all
```

[65]:
```
df3.head()
```

[65]:

	Borough	Name	Area(km2)	Population(2014)	Population density(inhabitants/km2)	Quartieri (districts)	Municipio	Indirizzo	Civico	CAP	ID_NIL	NIL	LOI
0	1.0	Centro storico	9.67	96315.0	11074	Brera, Centro Storico, Conca del Naviglio, Gua...	Municipio 1 - Centro storico	Via Guglielmo Marconi	2	20123	1	DUOMO	
1	2.0	Stazione Centrale, Gorla, Turro, Greco, Cresce...	12.58	153109.0	13031	Adriano, Crescenzago, Gorla, Greco, Loreto, Ma...	Municipio 2	Viale Zara	100	20125	12	MACIACHINI - MAGGIOLINA	
2	3.0	Città Studi, Lambrate, Porta Venezia	14.23	141229.0	10785	Casoretto, Cimiano, Città Studi, Dosso, Lambra...	Municipio 3	Via Sansovino	9	20133	21	BUENOS AIRES - PORTA VENEZIA - PORTA MONFORTE	

And I could finally calculate the score of each borough as the ratio between the total number of venues within it and its population:

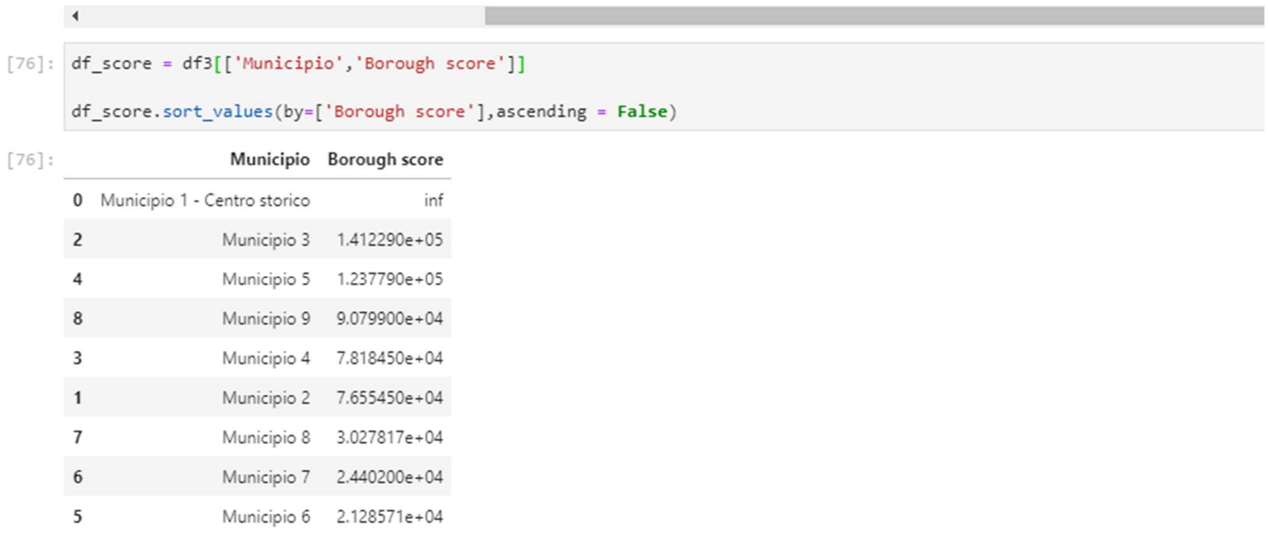


```
[70]: #calculate the score of each Borough as the ration between its population and the total number of venues
      #add the score as a column to the df3 dataframe
      df3['Borough score'] = df3['Population(2014)']/df3['Total Nr. of Venues']

[71]: df3
```

[71]:

Population bitants/km2)	Quartieri (districts)	Municipio	Indirizzo	Civico	CAP	ID_NIL	NIL	LONG_X_4326	LAT_Y_4326	Location	Total Nr. of Venues	Borough score
11074	Brera, Centro Storico, Conca del Naviglio, Gua...	Municipio 1 - Centro storico	Via Guglielmo Marconi	2	20123	1	DUOMO	9.189857	45.463365	(45.4633646580698, 9.18985741768964)	0	inf
13031	Adriano, Crescenzago, Gorla, Greco, Loreto, Ma...	Municipio 2	Viale Zara	100	20125	12	MACIACHINI - MAGGIOLINA	9.195895	45.497279	(45.497278789517, 9.19589480157642)	2	7.655450e+04
	Casoretto						BUENOS AIRES					

Then, I ordered the boroughs in descending order based on their score:



```
[76]: df_score = df3[['Municipio','Borough score']]

      df_score.sort_values(by=['Borough score'],ascending = False)
```

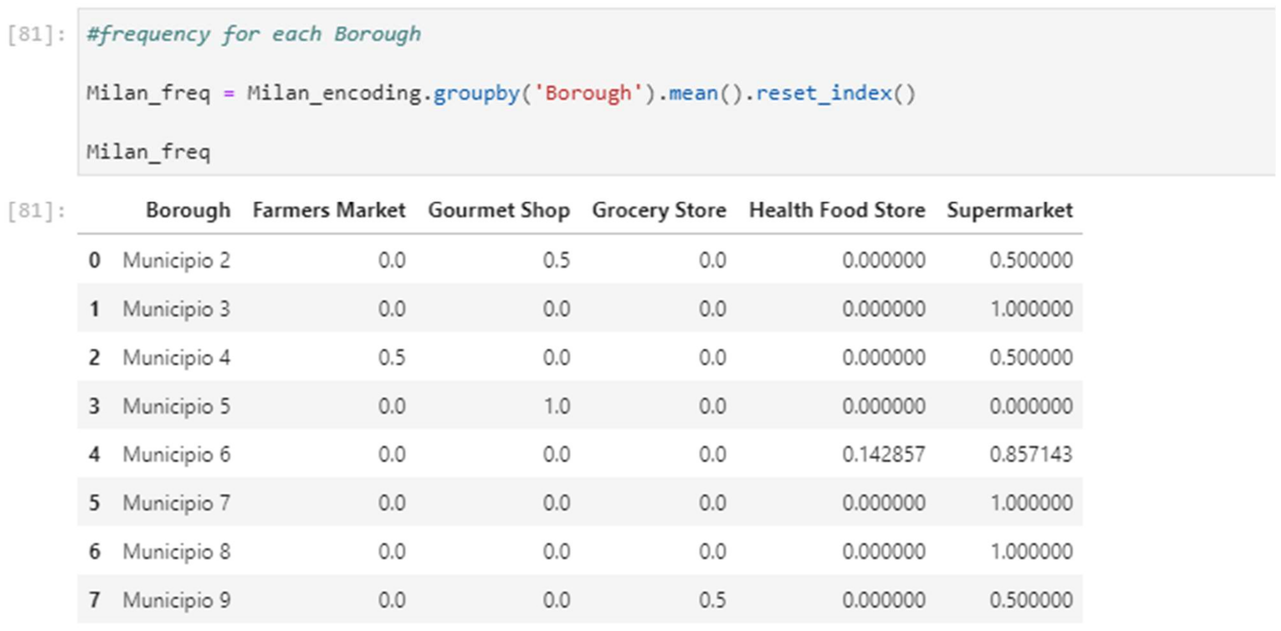
[76]:

	Municipio	Borough score
0	Municipio 1 - Centro storico	inf
2	Municipio 3	1.412290e+05
4	Municipio 5	1.237790e+05
8	Municipio 9	9.079900e+04
3	Municipio 4	7.818450e+04
1	Municipio 2	7.655450e+04
7	Municipio 8	3.027817e+04
6	Municipio 7	2.440200e+04
5	Municipio 6	2.128571e+04

The table shows that the first borough (*Municipio 1*) is the most attractive to open a new supermarket based on its population/venues ratio, followed by *Municipio 3*, with the least attractive being *Municipio 6*.

The analysis was carried out by including multiple venue categories, and the score makes no distinction between them. A food retailer, however, may be more interested in a specific venue category, for example, supermarkets, and might want to understand in which areas supermarkets are the most prevalent food store venue. This additional insight could help the retailer differentiate the boroughs of the city in terms of buying patterns and further inform the final decision.

To do this, I first applied one-hot encoding technique, and calculated the frequency of each location in every borough:

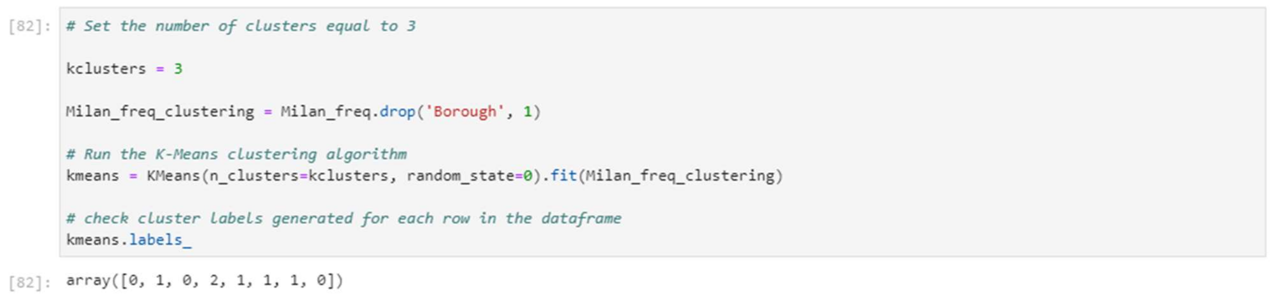


```
[81]: #frequency for each Borough

      Milan_freq = Milan_encoding.groupby('Borough').mean().reset_index()

      Milan_freq
```

[81]:

	Borough	Farmers Market	Gourmet Shop	Grocery Store	Health Food Store	Supermarket
0	Municipio 2	0.0	0.5	0.0	0.000000	0.500000
1	Municipio 3	0.0	0.0	0.0	0.000000	1.000000
2	Municipio 4	0.5	0.0	0.0	0.000000	0.500000
3	Municipio 5	0.0	1.0	0.0	0.000000	0.000000
4	Municipio 6	0.0	0.0	0.0	0.142857	0.857143
5	Municipio 7	0.0	0.0	0.0	0.000000	1.000000
6	Municipio 8	0.0	0.0	0.0	0.000000	1.000000
7	Municipio 9	0.0	0.0	0.5	0.000000	0.500000

Then, I applied the *k-means (k=3) clustering algorithm* to group together boroughs with similar characteristics. The resulting clusters are in the array below:

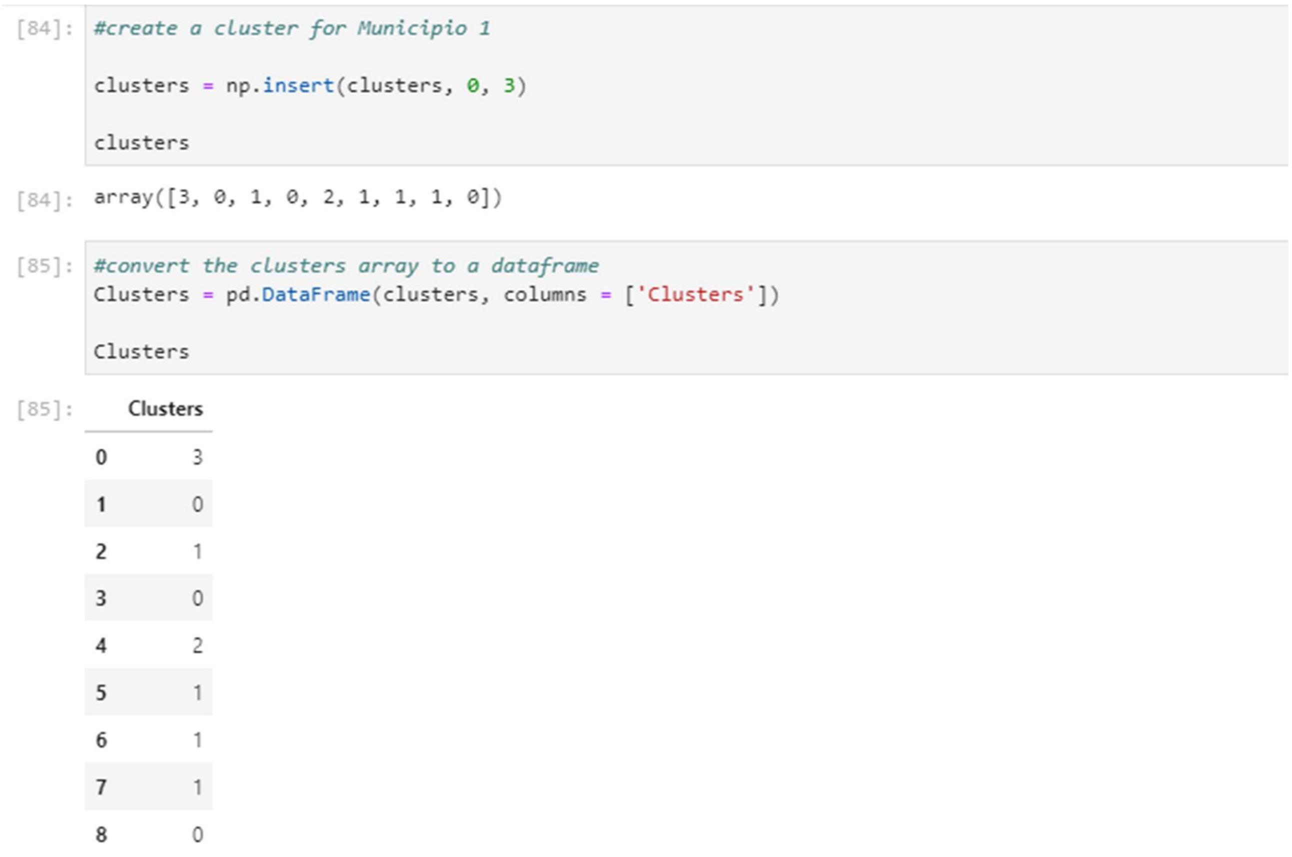


```
[82]: # Set the number of clusters equal to 3

      kclusters = 3

      Milan_freq_clustering = Milan_freq.drop('Borough', 1)

      # Run the K-Means clustering algorithm
      kmeans = KMeans(n_clusters=kclusters, random_state=0).fit(Milan_freq_clustering)

      # check cluster labels generated for each row in the dataframe
      kmeans.labels_

[82]: array([0, 1, 0, 2, 1, 1, 1, 0])
```

We notice that only 8 boroughs have been clustered: this is consistent with the fact that borough 1 (*Municipio 1*) does not contain any venues, so I assigned it to a cluster of its own. This results in the following list of clusters:

- CLUSTER 0: 50% of venues in this cluster are supermarkets. The cluster includes 3 boroughs: *Municipio 2, Municipio 4, Municipio 9*.
- CLUSTER 1: In this borough, more than 50% of venues are supermarkets. The cluster includes four boroughs: *Municipio 3, Municipio 6, Municipio 7, Municipio 8*.
- CLUSTER 2: there are no supermarkets among the venues of this cluster. The cluster only includes one borough, *Municipio 5*.
- CLUSTER 3: This cluster only includes one borough (*Municipio 1*). There are no food stores among the venues of this cluster.

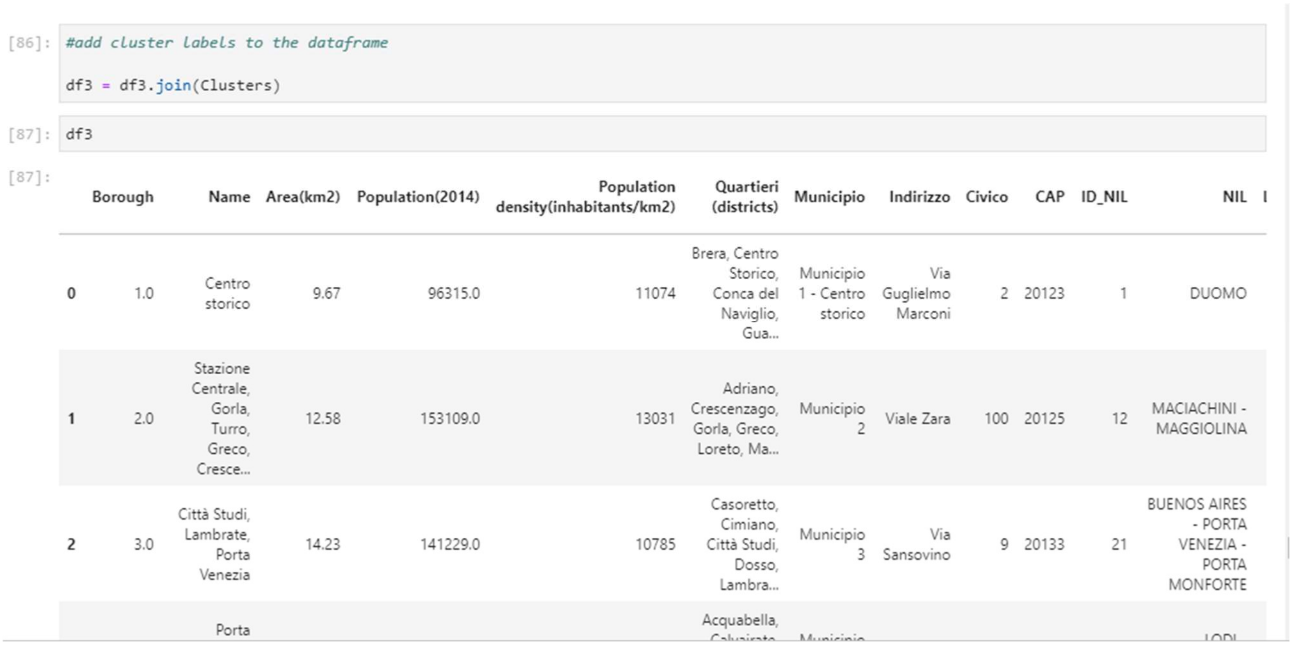
To add a cluster for *Municipio 1*, I added the number 3 to the first position of the clusters' array, then converted it to a dataframe:



```
[84]: #create a cluster for Municipio 1

      clusters = np.insert(clusters, 0, 3)

      clusters

[84]: array([3, 0, 1, 0, 2, 1, 1, 1, 0])

[85]: #convert the clusters array to a dataframe
      Clusters = pd.DataFrame(clusters, columns = ['Clusters'])

      Clusters

[85]:
```

	Clusters
0	3
1	0
2	1
3	0
4	2
5	1
6	1
7	1
8	0

Then, I added the clusters to the dataframe *df3* using the *join* method:

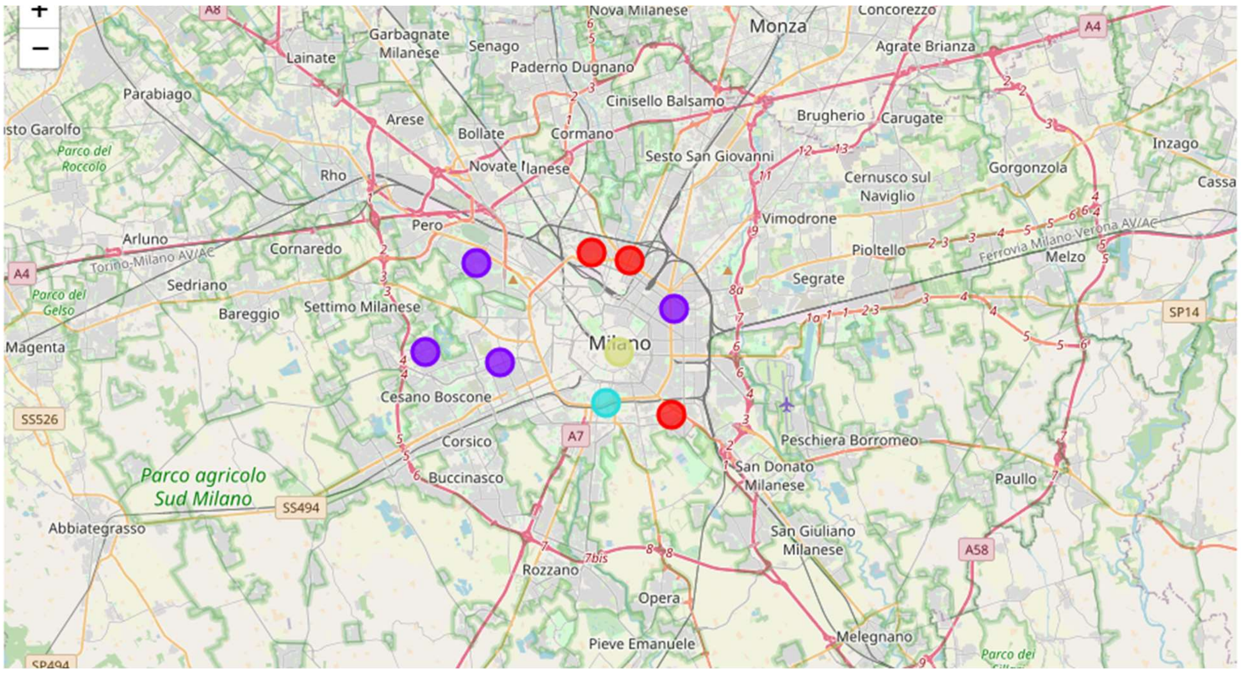


```
[86]: #add cluster labels to the dataframe

      df3 = df3.join(Clusters)

[87]: df3

[87]:
```

	Borough	Name	Area(km2)	Population(2014)	Population density(inhabitants/km2)	Quartieri (districts)	Municipio	Indirizzo	Civico	CAP	ID_NIL	NIL
0	1.0	Centro storico	9.67	96315.0	11074	Brera, Centro Storico, Conca del Naviglio, Gua...	Municipio 1 - Centro storico	Via Guglielmo Marconi	2	20123	1	DUOMO
1	2.0	Stazione Centrale, Gorla, Turro, Greco, Cresce...	12.58	153109.0	13031	Adriano, Crescenzago, Gorla, Greco, Loreto, Ma...	Municipio 2	Viale Zara	100	20125	12	MACIACHINI - MAGGIOLINA
2	3.0	Città Studi, Lambrate, Porta Venezia	14.23	141229.0	10785	Casoretto, Cimiano, Città Studi, Dosso, Lambra...	Municipio 3	Via Sansovino	9	20133	21	BUENOS AIRES - PORTA VENEZIA - PORTA MONFORTE
		Porta				Acquabella,						

Having added the cluster labels to the *df3* dataframe, I was able to create a *folium* map of Milan displaying the four clusters with different colors.

Below a screenshot of the final map with clusters:

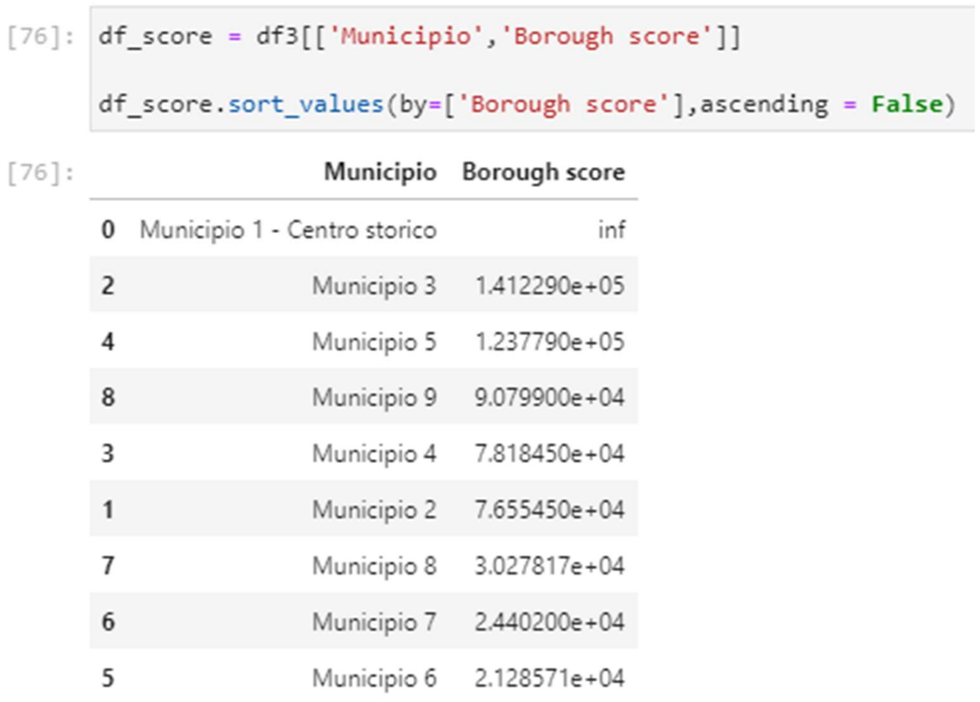


- CLUSTER 0 (RED): 50% of venues in this cluster are supermarkets. The cluster includes 3 boroughs: *Municipio 2, Municipio 4, Municipio 9*.
- CLUSTER 1 (VIOLET): In this borough, more than 50% of venues are supermarkets. The cluster includes four boroughs: *Municipio 3, Municipio 6, Municipio 7, Municipio 8*.
- CLUSTER 2 (BLUE): there are no supermarkets among the venues of this cluster. The cluster only includes one borough, *Municipio 5*.
- CLUSTER 3 (YELLOW): This cluster only includes one borough (*Municipio 1*). There are no food stores among the venues of this cluster.

Results

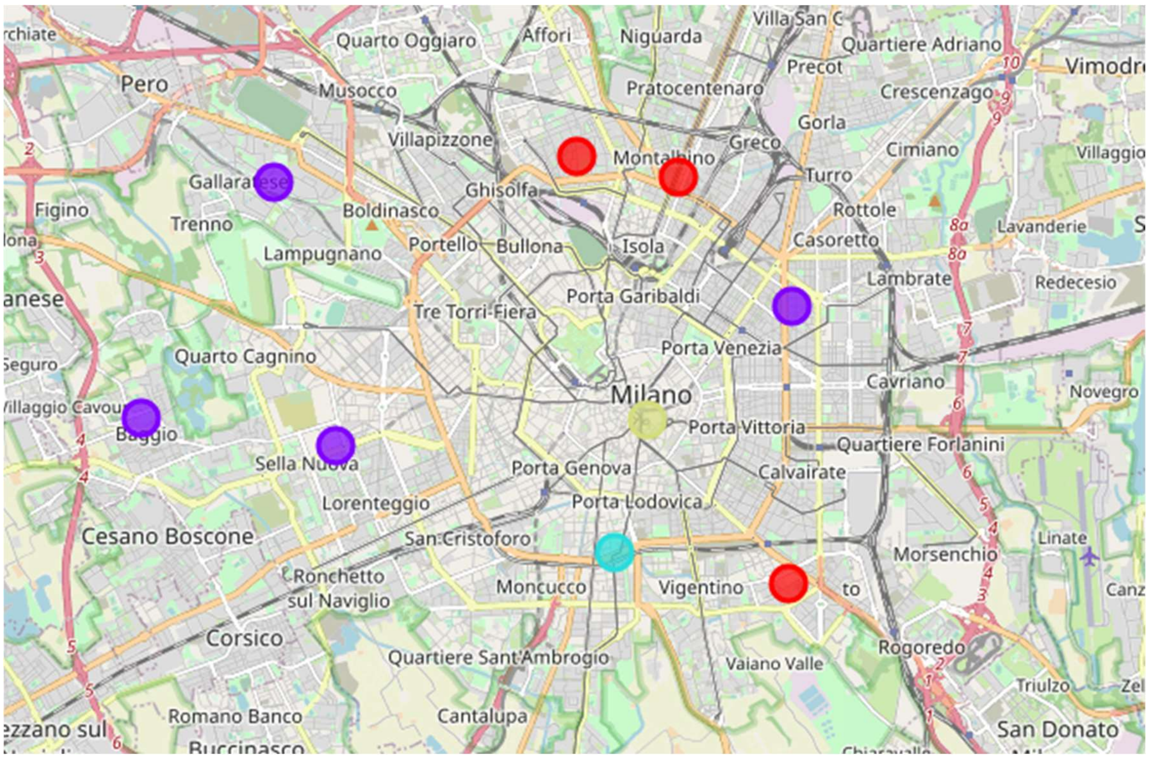
Based on the insights gained from the analysis, the following observations can be made:

- A score was assigned to each borough in Milan. The score measures the "attractiveness" of each borough as a location for a new supermarket, and it is defined as the ratio between the population of the borough and the total number of other food retail stores in the same area. Based on this score, Milan's boroughs were ranked as follows:



```
[76]: df_score = df3[['Municipio','Borough score']]

      df_score.sort_values(by=['Borough score'],ascending = False)

[76]:                 Municipio  Borough score
      0  Municipio 1 - Centro storico           inf
      2               Municipio 3  1.412290e+05
      4               Municipio 5  1.237790e+05
      8               Municipio 9  9.079900e+04
      3               Municipio 4  7.818450e+04
      1               Municipio 2  7.655450e+04
      7               Municipio 8  3.027817e+04
      6               Municipio 7  2.440200e+04
      5               Municipio 6  2.128571e+04
```

- The solution included among the potential competition not only supermarkets but also other food stores. To provide additional insight and make the analysis more relevant to a potential supermarket chain, the boroughs were clustered based on the "density" of supermarkets within their boundaries. Below a map showing the four resulting clusters:



- CLUSTER 0 (RED): 50% of venues in this cluster are supermarkets. The cluster includes 3 boroughs: Municipio 2, Municipio 4, Municipio 9.

- CLUSTER 1 (VIOLET): In this borough, more than 50% of venues are supermarkets. The cluster includes four boroughs: Municipio 3, Municipio 6, Municipio 7, Municipio 8.

- CLUSTER 2 (BLUE): there are no supermarkets among the venues of this cluster. The cluster only includes one borough, Municipio 5.

- CLUSTER 3 (YELLOW): This cluster only includes one borough (Municipio 1). There are no food stores among the venues of this cluster.

Discussion

Based on these results, a potential retailer searching for a location to open a supermarket in Mian would be able to:

- Understand the level of competition in each borough.
- Differentiate the boroughs based on population, competition, and purchasing habits.

Conclusions

The problem of selecting the best location for a new retail store is a complex one, and this project does not aspire to provide a roadmap for solving it or to address it fully.

While the analysis provided in this report can be useful, there are many other aspects, both qualitative and quantitative, that would need to be considered to address the problem in a real-world scenario. Addressing all these aspects would have complicated the analysis, and it is beyond the scope of this work.

The goal of this project was to demonstrate through a simplified example of a real-world application, how leveraging open source tools, APIs, and applied data science techniques can help tackle complex problems and extract meaningful insights from data.

References

- Links to data sources:

https://en.wikipedia.org/wiki/Municipalities_of_Milan

https://dati.comune.milano.it

https://dati.comune.milano.it/dataset/a7f54a9a-8331-4825-bc52-6a69a10b0bd3/resource/162ffc23-419f-420d-b14f-bccfe28d920a/download/municipi_sedi__final.json

- Links to reports, articles, presentations and Jupiter Notebooks:

https://link.springer.com/article/10.1007/s40196-013-0015-6

https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass/DP0701EN/sample_submission/Predicting_the_Improvement_of_NBA_players_Report.pdf

https://cocl.us/coursera_capstone_notebook

https://towardsdatascience.com/clustering-food-venues-in-sydney-bf48877650d5

https://www.linkedin.com/pulse/housing-sales-prices-venues-data-analysis-ofistanbul-sercan-y%C4%B1ld%C4%B1z/

https://www.linkedin.com/pulse/battle-neighborhoods-marie-kathrin-tritschel/?articleId=6674287855189155841

https://www.linkedin.com/pulse/applied-data-science-capstone-project-restaurant-wagner-mba/?articleId=6670274875946622976

https://github.com/JohWagner/Coursera_Capstone/blob/master/Week%205.ipynb

https://github.com/JohWagner/Coursera_Capstone/blob/master/200605_Report.pdf

https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass/DP0701EN/sample_submission/Predicting_the_Improvement_of_NBA_players_Presentation.pdf