Capstone Project - The Battle of the Neighborhoods

Guillermo Velasco

Table of contents

Introduction: Business Problem

Madrid is the capital of Spain and has a population of 3.300.000 citizens. With more than 9.400 restaurants, Madrid is considered a great place to enjoy almost any type of cuisine. The negative part of having such a restaurant abundance is that finding the location for a new restaurant is not an easy task.

In this project I will look at the best possible place to open a pizza place in Madrid. The goal would be to find an area without any or few pizza restaurants in the area combined with a high population density.

To obtain this information, I will be using the knowledge acquired during the data science professional certificate course from IBM. The analysis will provide an understanding of the data and insisghts on where would it be better to locate a new pizza place.

Data

To answer the problem stated above, the following information, including the source, needs to be obtained:

- List of neighbourhoods in Madrid including population and area. Source: http://www-2.munimadrid.es/CSE6/control/seleccionDatos?numSerie=14010100012

- Coordinates for each neighbourhood. Source: Geocoder ibrary for Python.

- Pizza restaurants in each neighbourhood. Source: Foursquare API.

Neighborhouds data

The list of neighbourhoods in Madrid is provided by the city of Madrid in an Excel format. The file includes all neighbourhoods sorted by district and the respective area and population. The Excel file is cleaned and imported to Python.

In [1]:

```
import pandas as pd
df = pd.read_excel('/Users/guillermo/Python/NeighbourhoodsMadrid.xls', sheet_name='Sheet1')
df
```

Out[1]:

	District	Neighbourhood	Area (Ha)	Population
0	Centro	Palacio	146.99	23593
1	Centro	Embajadores	103.37	47048
2	Centro	Cortes	59.19	10771
3	Centro	Justicia	73.94	18021
4	Centro	Universidad	94.80	33418
...	...	...	...	...
126	Barajas	Alameda de Osuna	197.03	19820
127	Barajas	Aeropuerto	2962.61	1900
128	Barajas	Casco Histórico de Barajas	54.94	7683
129	Barajas	Timón	509.45	12853
130	Barajas	Corralejos	468.25	7754

131 rows × 4 columns

A column with the population density is calculated and added to the data set, with density equals to population divided by area.

In [2]:

```
df["Density"]=df["Population"]/df["Area (Ha)"]
df
```

Out[2]:

	District	Neighbourhood	Area (Ha)	Population	Density
0	Centro	Palacio	146.99	23593	160.507518
1	Centro	Embajadores	103.37	47048	455.141724
2	Centro	Cortes	59.19	10771	181.973306
3	Centro	Justicia	73.94	18021	243.724642
4	Centro	Universidad	94.80	33418	352.510549
...	...	...	...	...	...
126	Barajas	Alameda de Osuna	197.03	19820	100.593818
127	Barajas	Aeropuerto	2962.61	1900	0.641326
128	Barajas	Casco Histórico de Barajas	54.94	7683	139.843466
129	Barajas	Timón	509.45	12853	25.229169
130	Barajas	Corralejos	468.25	7754	16.559530

131 rows × 5 columns

To obtain the latitude and longitude for each neighbourhood the Geocoder librabry is used

In [4]:

```
import geocoder

latitude=[]
longitude=[]
for code in df['Neighbourhood']:
    g = geocoder.arcgis('{}, Madrid, Madrid'.format(code))
    while (g.latlng is None):
        g = geocoder.arcgis('{}, Madrid, Madrid'.format(code))
    latlng = g.latlng
    latitude.append(latlng[0])
    longitude.append(latlng[1])
```

In [5]:

```
df["Latitude"]=latitude
df["Longitude"]=longitude
```

In [6]:

```
df.head()
```

Out[6]:

	District	Neighbourhood	Area (Ha)	Population	Density	Latitude	Longitude
0	Centro	Palacio	146.99	23593	160.507518	40.41517	-3.71273
1	Centro	Embajadores	103.37	47048	455.141724	40.40803	-3.70067
2	Centro	Cortes	59.19	10771	181.973306	40.41589	-3.69636
3	Centro	Justicia	73.94	18021	243.724642	40.42479	-3.69308
4	Centro	Universidad	94.80	33418	352.510549	40.42565	-3.70726

In [7]:

```
#Using geocoder library to get the latitude and longitude values of Madrid.

g = geocoder.arcgis('Madrid, Madrid')
latlng = g.latlng
latitudeMadrid=latlng[0]
longitudeMadrid=latlng[1]
print('The geograpical coordinates of Madrid are {}, {}.'.format(latitudeMadrid, longitudeMadrid))
```

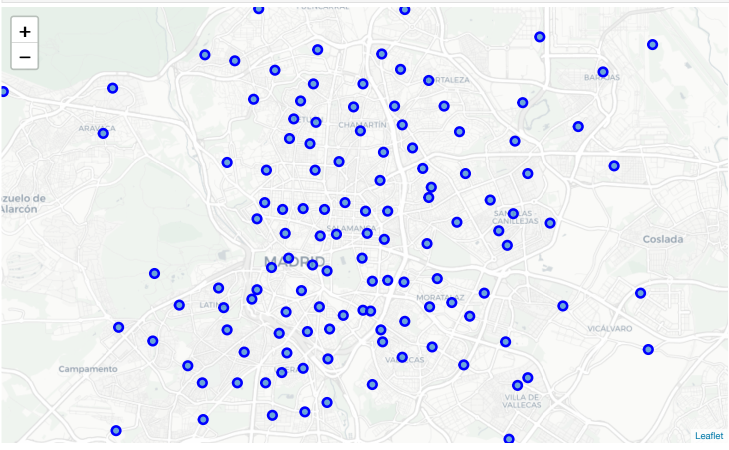
```
The geograpical coordinates of Madrid are 40.41955000000007, -3.6919599999999377.
```

Now I will plot all neighbourhoods in a map

In [8]:

```
# create map of Madrid using latitude and longitude values

import folium # map rendering library

map_madrid = folium.Map(location=[latitudeMadrid, longitudeMadrid], zoom_start=12)
folium.TileLayer('cartodbpositron').add_to(map_madrid) #cartodbpositron cartodbdark_matter

# add markers to map
for lat, lng, district, neighbourhood in zip(df['Latitude'], df['Longitude'], df['District'], df['Neighbourhoo
    label = '{}, {}'.format(neighbourhood, district)
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=5,
        popup=label,
        color='blue',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.7,
        parse_html=False).add_to(map_madrid)

map_madrid
```

Out[8]:



Pizza places obtained from Foursquare

By calling the Foursquare API we will obtain all pizza restaurants in the city of Madrid

In [9]:

```
#Define Foursquare Credentials and Version
CLIENT_ID = 'SE3FSDLHCBUETUXV0P5ANSUJ0HV0NCDCYIEVUJXOY1MVTSVC' # your Foursquare ID
CLIENT_SECRET = 'SGV2YDC3E3G2PN1A1UO32TDRYT5IR2OKKR4OYHGVIJBJUWG2' # your Foursquare Secret
VERSION = '20180605' # Foursquare API version
LIMIT = 100 # A default Foursquare API limit value
```

In [10]:

```
import json # library to handle JSON files
import requests # library to handle requests
from pandas import json_normalize # tranform JSON file into a pandas dataframe
```

In [11]:

```
#function to get nearby pizzerias for all the neighborhoods in a radius of 1km
def getNearbyVenues(names, latitudes, longitudes, radius=1000):

    venues_list=[]
    for name, lat, lng in zip(names, latitudes, longitudes):

        # create the API request URL
        url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={}&v={}&ll={},{}&radiu
            CLIENT_ID,
            CLIENT_SECRET,
            VERSION,
            lat,
            lng,
            radius,
            LIMIT,
            "4bf58dd8d48988d1ca941735") # PIZZA PLACE CATEGORY ID


        # make the GET request
        results = requests.get(url).json()["response"]['groups'][0]['items']
```

```
        # return only relevant information for each nearby venue
        venues_list.append([(
            name,
            lat,
            lng,
            v['venue']['name'],
            v['venue']['location']['lat'],
            v['venue']['location']['lng'],
            v['venue']['categories'][0]['name']) for v in results])

    nearby_venues = pd.DataFrame([item for venue_list in venues_list for item in venue_list])
    nearby_venues.columns = ['Neighbourhood',
                  'Neighbourhood Latitude',
                  'Neighbourhood Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']

    return(nearby_venues)
```

In [12]:

```
#get nearby pizzeria for all the neighborhoods in Madrid
madrid_pizzerias = getNearbyVenues(names=df['Neighbourhood'],
                                   latitudes=df['Latitude'],
                                   longitudes=df['Longitude']
                                  )
```

In [13]:

```
madrid_pizzerias
```

Out[13]:

	Neighbourhood	Neighbourhood Latitude	Neighbourhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Palacio	40.41517	-3.71273	Trattoria Malatesta	40.416788	-3.707182	Italian Restaurant
1	Palacio	40.41517	-3.71273	Al Settimo Cielo	40.410509	-3.710321	Pizza Place
2	Palacio	40.41517	-3.71273	Ópera : Pizza	40.417915	-3.708965	Pizza Place
3	Palacio	40.41517	-3.71273	El Horno Azul	40.421598	-3.710031	Pizza Place
4	Palacio	40.41517	-3.71273	López & López	40.409499	-3.704046	Pizza Place
...	...	...	...	...	...	...	...
1449	Alameda de Osuna	40.45818	-3.58953	L'Incontro Trattoria	40.457505	-3.585463	Pizza Place
1450	Alameda de Osuna	40.45818	-3.58953	Pizzamascalzone	40.465177	-3.592820	Pizza Place
1451	Casco Histórico de Barajas	40.47482	-3.57951	Telepizza	40.472902	-3.579102	Pizza Place
1452	Casco Histórico de Barajas	40.47482	-3.57951	Pizzeria La Piazzeta	40.471107	-3.571191	Pizza Place
1453	Corralejos	40.46540	-3.61164	Telepizza	40.469282	-3.616347	Pizza Place

1454 rows × 7 columns

Now that we have all the pizzerias per neighbourhood in a 1km radius lets plot them in a map

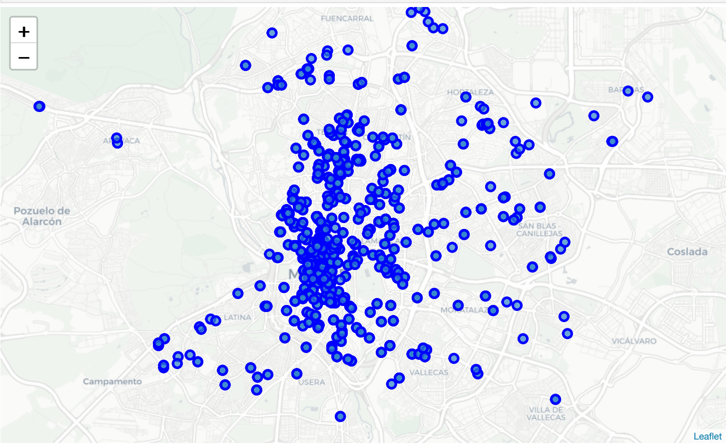
In [14]:

```
from folium import plugins
from folium.plugins import HeatMap
import numpy as np
```

In [15]:

```
map_madrid = folium.Map(location=[latitudeMadrid, longitudeMadrid], zoom_start=12)
folium.TileLayer('cartodbpositron').add_to(map_madrid) #cartodbpositron cartodbdark_matter
# add markers to map
for lat, lng, venue in zip(madrid_pizzerias['Venue Latitude'], madrid_pizzerias['Venue Longitude'], madrid_piz
    label = '{}'.format(venue)
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=5,
        popup=label,
        color='blue',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.7,
        parse_html=False).add_to(map_madrid)

map_madrid
```

Out[15]:



Now we have all the needed data for our analysis.

Methodology

In this project we will focus on first identifying neighbourhoods of Madrid with a low number of pizza restaurants and second on figuring out which of these neighbourhoods has a high population density.

In the first step we have collected the required data for our analysis. It consists of all neighbourhoods of Madrid including population, area and location plus all pizza places within a 1km radius of the center of each neighbourhood.

In the second step of the analysis I will be exploring the density of pizza restaurants in Madrid. I will be using heat maps to identify promising locations without pizza restaurants.

In third and final step I will be calculating the number of pizza places per neighbourhood and contrast it against the population denisty. This will let us identify which neighbourhoods are more promising in terms of potential. Plus, this information added with the heatmap will allow us to identify new places to locate a pizza restaurant.

Analysis

I will be exploring the density of pizza restaurants in Madrid. I will be using a heat map to identify promising locations without pizza restaurants.

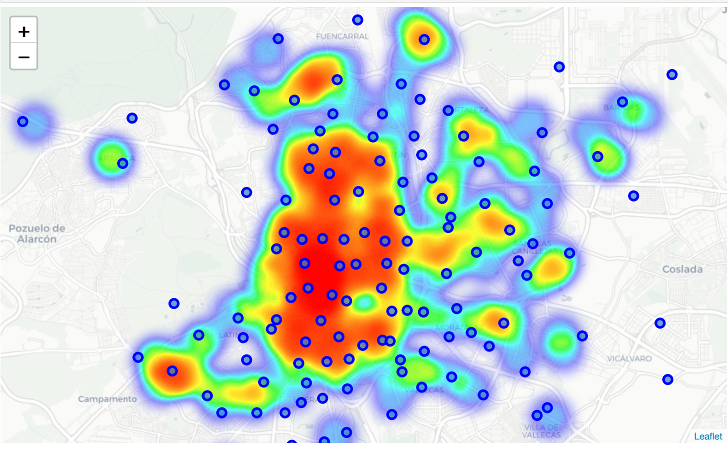
In [16]:

```
# Ensure you're handing floats and not strings
madrid_pizzerias['Venue Latitude'] = madrid_pizzerias['Venue Latitude'].astype(float)
madrid_pizzerias['Venue Longitude'] = madrid_pizzerias['Venue Longitude'].astype(float)

# List comprehension to make out list of lists
pizzerias_latlons = [[row['Venue Latitude'],row['Venue Longitude']] for index, row in madrid_pizzerias.iterrow
```

In [17]:

```
map_madrid = folium.Map(location=[latitudeMadrid, longitudeMadrid], zoom_start=12)
folium.TileLayer('cartodbpositron').add_to(map_madrid) #cartodbpositron cartodbdark_matter
HeatMap(pizzerias_latlons).add_to(map_madrid)

# add neighbourhoods to map
for lat, lng, district, neighbourhood in zip(df['Latitude'], df['Longitude'], df['District'], df['Neighbourhoo
    label = '{}, {}'.format(neighbourhood, district)
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=5,
        popup=label,
        color='blue',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.7,
        parse_html=False).add_to(map_madrid)

map_madrid
```

Out[17]:



Now we have a heat map showing all areas with a high density of pizzerias but most important, it shows areas where there are not many or not at all, which are the places that we would like to put a new pizza place.

Of course there are lot of areas, so in order to find the best options, we can proceed to analyse the numbers for the neighbourhoods.

We can start by sorting the neighbourhoods by number of pizzerias.

In [28]:

```
neighbourhood_pizzerias=madrid_pizzerias.groupby('Neighbourhood').count()[['Venue']]
neighbourhood_pizzerias
```

Out[28]:

	Venue
Neighbourhood	
Abrantes	3
Acacias	21
Adelfas	6
Alameda de Osuna	3
Almagro	28
...	...
Vinateros	3
Vista Alegre	3
Zofío	1
Águilas	8
Ángeles	7

115 rows × 1 columns

In [29]:

```
neighbourhood_pizzerias.sort_values(by=['Venue'])[['Venue']]
```

Out[29]:

	Venue
Neighbourhood	
Mirasierra	1

Corralejos	1
San Fermín	1
San Isidro	1
Ensanche de Vallecas	1
...	...
Embajadores	55
Justicia	63
Cortes	80
Universidad	84
Sol	92

115 rows × 1 columns

Now that we have all neighbourhoods sorted by number of pizzerias, we can focus on those that have a low number of them. However it could be that they have few pizzerias because there are few people living in the area. To address this concern, I will look into the population data and sort the neighbourhoods by population density.

In [36]:

```
neighbourhood_density=df.sort_values(by=['Density'])[['District','Neighbourhood','Density']]
neighbourhood_density
```

Out[36]:

	District	Neighbourhood	Density
43	Fuencarral-El Pardo	El Pardo	0.185091
127	Barajas	Aeropuerto	0.641326
117	Vicálvaro	El Cañaveral	2.267762
50	Fuencarral-El Pardo	El Goloso	7.187532
51	Moncloa-Aravaca	Casa de Campo	7.560450
...	...	...	...
38	Chamberí	Arapiles	427.257004
13	Retiro	Pacífico	445.298694
16	Retiro	Ibiza	447.012195
1	Centro	Embajadores	455.141724
37	Chamberí	Gaztambide	460.592300

131 rows × 3 columns

Combining the data of Density and number of pizzerias for each neighbourhood we can create a new dataframe that would allow to rank the neighbourhoods

In [41]:

```
expected_result = pd.merge(neighbourhood_density, neighbourhood_pizzerias, on = 'Neighbourhood', how = 'left')
expected_result
```

Out[41]:

	District	Neighbourhood	Density	Venue
0	Fuencarral-El Pardo	El Pardo	0.185091	NaN
1	Barajas	Aeropuerto	0.641326	NaN
2	Vicálvaro	El Cañaveral	2.267762	NaN
3	Fuencarral-El Pardo	El Goloso	7.187532	NaN
4	Moncloa-Aravaca	Casa de Campo	7.560450	NaN
...	...	...	...	...
126	Chamberí	Arapiles	427.257004	53.0
127	Retiro	Pacífico	445.298694	8.0
128	Retiro	Ibiza	447.012195	20.0
129	Centro	Embajadores	455.141724	55.0
130	Chamberí	Gaztambide	460.592300	28.0

131 rows × 4 columns

In [49]:

```
#I will make the NaN equal to 0 to calculate aftewards with those values
expected_result['Venue'] = expected_result['Venue'].fillna(0)
expected_result
```

Out[49]:

	District	Neighbourhood	Density	Venue	Population Density/Pizzerias
0	Fuencarral-El Pardo	El Pardo	0.185091	0.0	NaN
1	Barajas	Aeropuerto	0.641326	0.0	NaN
2	Vicálvaro	El Cañaveral	2.267762	0.0	NaN
3	Fuencarral-El Pardo	El Goloso	7.187532	0.0	NaN
4	Moncloa-Aravaca	Casa de Campo	7.560450	0.0	NaN
...	...	...	...	...	...
126	Chamberí	Arapiles	427.257004	53.0	8.061453
127	Retiro	Pacífico	445.298694	8.0	55.662337
128	Retiro	Ibiza	447.012195	20.0	22.350610
129	Centro	Embajadores	455.141724	55.0	8.275304
130	Chamberí	Gaztambide	460.592300	28.0	16.449725

131 rows × 5 columns

With both number of pizzerias (Venue) and Density, we can calculate the number of pizzerias per population density, which will tell us the places where it would be more interesting to open a pizza place

In [50]:

```
expected_result["Population Density/Pizzerias"]=expected_result["Density"]/expected_result["Venue"]
expected_result.sort_values(by=['Population Density/Pizzerias'])
```

Out[50]:

	District	Neighbourhood	Density	Venue	Population Density/Pizzerias
20	Retiro	Los Jerónimos	36.959715	40.0	0.923993
9	Arganzuela	Atocha	16.460514	15.0	1.097368
63	Centro	Sol	171.165506	92.0	1.860495
69	Centro	Cortes	181.973306	80.0	2.274666
56	Centro	Palacio	160.507518	45.0	3.566834
...	...	...	...	...	...
3	Fuencarral-El Pardo	El Goloso	7.187532	0.0	inf
2	Vicálvaro	El Cañaveral	2.267762	0.0	inf
1	Barajas	Aeropuerto	0.641326	0.0	inf
39	Ciudad Lineal	Colina	116.473111	0.0	inf
0	Fuencarral-El Pardo	El Pardo	0.185091	0.0	inf

131 rows × 5 columns

Now that we have the Population Density/Pizzerias per each neighbourhood, we can focus on those neighbourhoods with higher values. Notice that when the number of Pizzerias is 0 the result is infinite so in those cases is intersting to look at the population density of the neighbourhood. But once we have this information, we can focus on the most attractice neighbourhoods and then go back to the heat map to select an area within the neighbourhood that has no pizza places nearby to avoid competition.

Results and Discussion

Our analysis shows that although there is a great number of pizzerias in Madrid, there are pockets of low number of pizzerias. However, it is interesting to check that could potentially look like a good place to open a pizzeria might not be that much. For that reason it is important to check at the population density of the different neighbourhoods. Taking this information and combining it with the number of pizzerias per neighbourhood we can obtain a good ratio Population Density/Pizzerias that will tell us, the higher the ratio, the more interesting is to open a pizzeria. Of course once we have that information and we have selected the most intersting neighbourhoods, it is important to look into the heat map to precisely select a location without competition nearby.

Conclusion

This analysis has turned out to be a good and quick method to locate possible place to open a pizzeria within the city of Madrid. The analysis takes into consideration the two most important criteria when chosing to open a restaurant, which is population density and competition. However, such an important decision cannot rely on only this two criteria. For a future analysis it could be interesting to add more socioeconomical factors such as income, age, type of buildings and further more that could make the decision a lot more accurate.

In [ ]: