The battle of neighborhoods Barcelona city

Case study: What is the ideal location for a Greek Restaurant?

Business problem

Background

Barcelona, the capital of Catalonia, has a population of 1.6M people and is at the heart of a metropolitan region of 5M inhabitants

Context

Despite the large and ethnically diverse population there is only a handful of Greek restaurants which offer a high quality menu for a middle level target audience.

Problem statement

The stake holder wants to fill this gap by opening a greek restaurant. What is the ideal location to open a Greek restaurant in Barcelona?

Criteria

- Density of other restaurants
- Other greek, or similar cuisine (e.g. spanish, mediterranean) restaurants in the neighborhood
- Population density
- Distance from city centre

Methodology

- Beautiful Soup for web scraping tables with neighborhood/district data.
- Nominatim geolocator module for longitude/latitudes of these data
- Folium package for visualization
- k-means clustering from sklearn package, as the machine learning technique used
- The rest of analysis relies on pandas, numpy, matplotlib packages.

Web scraping

```
# extract data using Beautiful Soup
url='https://en.wikipedia.org/wiki/Districts_of_Barcelona'
res = requests.get(url)

soup = BeautifulSoup(res.content,'lxml')
table = soup.find_all('table')
data = pd.read_html(str(table))
df = pd.DataFrame(data[7])
```

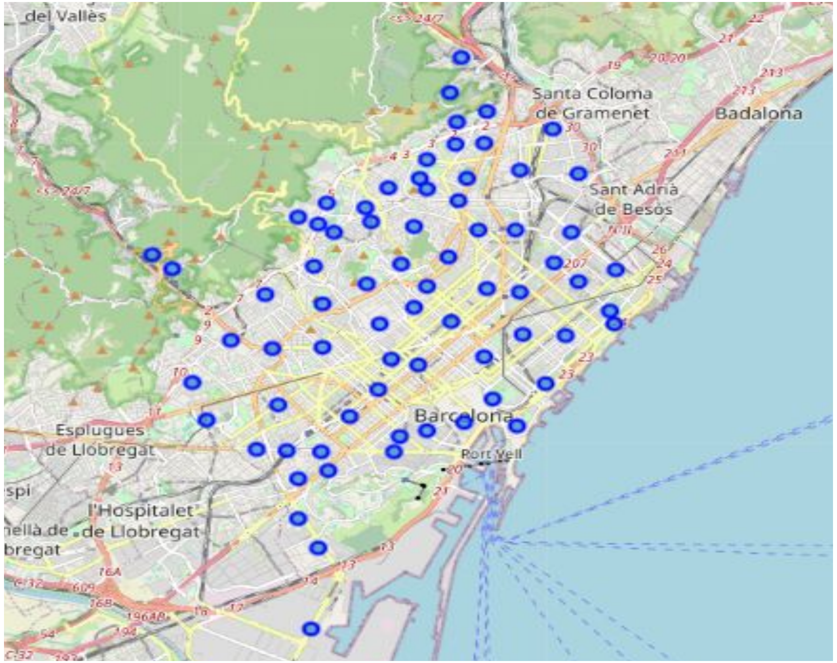
Extract neighborhood.districts data for Barcelona using Beautiful Soup

Geolocation

```
# address = 'Sant Andreu de Palomar,'
def find_lon_lat(address):
    geolocator = Nominatim(user_agent="ny_explorer ")
    location = geolocator.geocode(address,timeout=10000)
    latitude = location.latitude
    longitude = location.longitude
    return [latitude, longitude]
# test it
find_lon_lat('El Coll Barcelona, Spain')
```

Use Nominatim module to assess longitudes/latitudes for the neighborhoods

Visualization



Visualizing data with the Folium package

The first dataframe

```
# Append distance to bcn_restaursnts
bcn_restaurants.insert(3, 'Distance from centre', distance)
bcn_restaurants.head()
```

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Distance from centre	Venue	Venue Latitude	Venue Longitude	Venue Category
2	La Barceloneta	41.380653	2.189927	1.8	Somorrostro	41.379156	2.189100	Spanish Restaurant
3	La Barceloneta	41.380653	2.189927	1.8	La Cova Fumada	41.379254	2.189254	Tapas Restaurant
5	La Barceloneta	41.380653	2.189927	1.8	Rumbanroll	41.380597	2.187807	Mediterranean Restaurant
7	La Barceloneta	41.380653	2.189927	1.8	La Bombeta	41.380521	2.187573	Tapas Restaurant
8	La Barceloneta	41.380653	2.189927	1.8	La Barra Carles Abellan	41.379838	2.187712	Restaurant

We obtain restaurants from Foursquare api, and construct a dataframe with this information, along with the distance from the city centre

Using machine learning to analyse neighborhoods

```
# one hot encoding
bcn_onehot = pd.get_dummies(bcn_restaurants[['Venue Category']], prefix="", prefix_sep="")

# add neighborhood column back to dataframe
bcn_onehot['Neighborhood'] = bcn_venues['Neighborhood']

# move neighborhood column to the first column
fixed_columns = [bcn_onehot.columns[-1]] + list(bcn_onehot.columns[:-1])
bcn_onehot = bcn_onehot[fixed_columns]


print (bcn_onehot.shape)
```

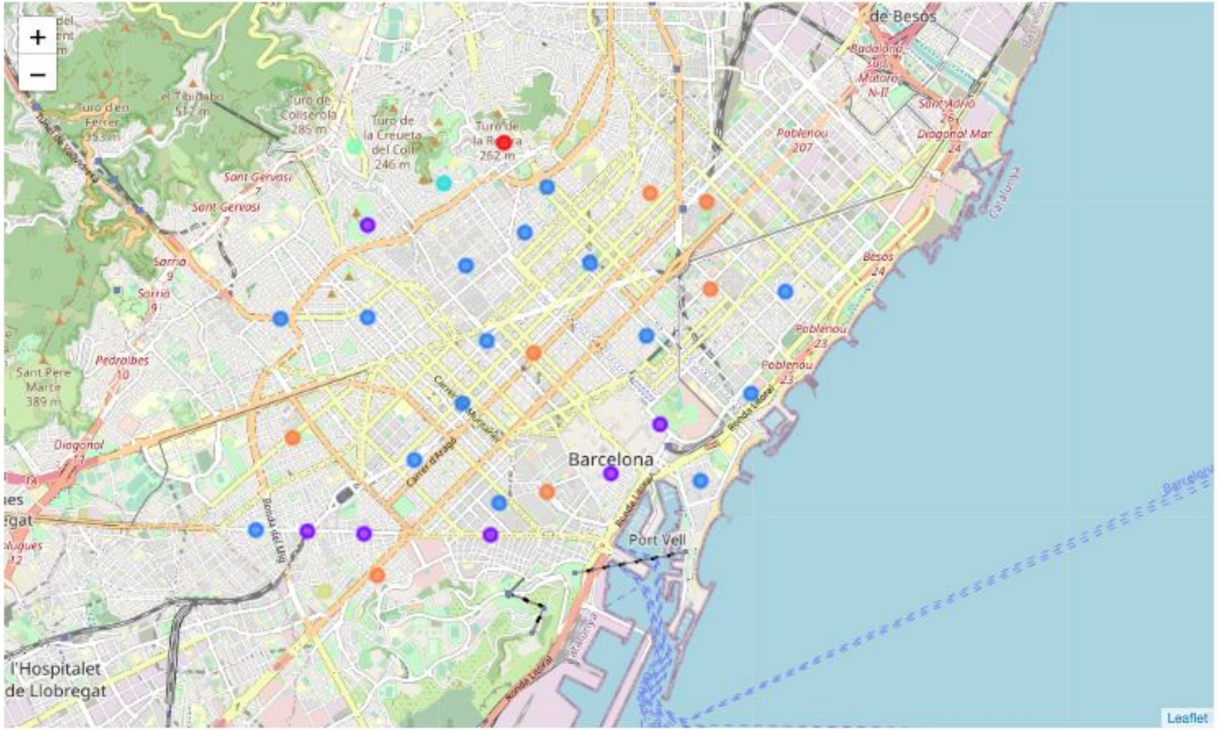
```
(744, 60)
```

```
# import k-means from clustering stage
from sklearn.cluster import KMeans
```

```
# set number of clusters
kclusters = 7

bcn_grouped_clustering = bcn_grouped.drop('Neighborhood', 1)

# run k-means clustering
kmeans = KMeans(n_clusters=kclusters, random_state=0).fit(bcn_grouped_clustering)

# check cluster labels generated for each row in the dataframe
kmeans.labels_[0:10]
```

```
array([2, 0, 2, 6, 6, 6, 5, 6, 1, 2], dtype=int32)
```

Visualizing the resulting clusters



Analyzing the resulting clusters

```
cluster1= bcn_merged.loc[bcn_merged['Cluster Labels'] == 1]

t1 = cluster1[ cluster1['Venue Category'].str.contains("greek|italian|tapas|mediterranean|spanis
h", case=False)].\
groupby('Neighborhood').count ()

t2 = cluster1.groupby('Neighborhood').count()

# t1/t2 is the ratio of similar-to-greek cuisine restaurants to total restaurants in each neighbo
rhood
t1/t2
# cluster1[ cluster1['Venue Category'].str.contains("greek|italian|tapas|mediterranean|spanish",
case=False)].\
# groupby('Neighborhood').count ()/test
```

	Neighborhood Latitude_x	Neighborhood Longitude_x	Distance from centre_x	Venue	Venue Latitude_x	Venue Longitude_x	Venue Category	Cluster Labels	Neighborl Latitude_
Neighborhood									
El Poble-sec	0.695652	0.695652	0.695652	0.695652	0.695652	0.695652	0.695652	0.695652	0.695652
Gothic	0.818182	0.818182	0.818182	0.818182	0.818182	0.818182	0.818182	0.818182	0.818182
Hostafrancs	0.565217	0.565217	0.565217	0.565217	0.565217	0.565217	0.565217	0.565217	0.565217
Santa Caterina i la Ribera	0.625000	0.625000	0.625000	0.625000	0.625000	0.625000	0.625000	0.625000	0.625000
Sants	0.550000	0.550000	0.550000	0.550000	0.550000	0.550000	0.550000	0.550000	0.550000
El Putget i Farró	0.615385	0.615385	0.615385	0.615385	0.615385	0.615385	0.615385	0.615385	0.615385

We use as criteria the density of similar-to-greek restaurants per neighborhood. We pick the cluster which has the overall lowest density

Further refining the clusters

We further refine the search by using the additional criteria:

1. The population in each neighborhood/cluster
2. The total number of restaurants in each neighborhood/cluster

Ideally we should opt for the cluster where restaurants/population is small, thus more potential customers.

We use population/neighborhood data

```
pop = pd.read_csv("Population_barrios_Bcn.txt", sep=',\t+',delimiter=',')
# pop[['Neighborhood']]

pop['Neighborhood'] = pop["Neighborhood"].str.strip()
pop['Population'] = pop["Population"].str.strip()
pop
```

	Neighborhood	Population
0	el Raval	48.297
1	el Barri Gòtic	19.180
2	la Barceloneta	15.173
3	Sant Pere Santa Caterina i la Ribera	23.170
4	el Fort Pienc	32.649
...	...	...
68	Diagonal Mar i el Front Marítim del Poblenou	13.625
69	el Besòs i el Maresme	24.660
70	Provençals del Poblenou	21.303
71	Sant Martí de Provençals	26.168
72	la Verneda i la Pau	28.883

73 rows × 2 columns

Examining density (venues/population)

```
testdf = cluster2_pop.groupby('Population').count().reset_index()
testdf1= testdf['Population'].astype(float)
testdf2= testdf['Neighborhood']
testdf2/testdf1
# cluster2_pop.head()
```

```
0    3.361234
1    0.808003
2    0.312509
3    0.700099
4    0.610200
dtype: float64
```

We examine the density (venues/population) per neighborhood for the chosen cluster and find that the Neighborhoods with less density (venues/population) is numbers 2, 4 & 3

Which are the neighborhoods with less density of venues/population?

```
pop_reduced[pop_reduced['Population'].astype(float)==testdf1[2]]
```

	Neighborhood	Population
31	Camp d'en Grassot i Gràcia Nova	35.199

```
pop_reduced[pop_reduced['Population'].astype(float)==testdf1[4]]
```

	Neighborhood	Population
30	Vila de Gràcia	50.803

```
pop_reduced[pop_reduced['Population'].astype(float)==testdf1[3]]
```

	Neighborhood	Population
9	Sant Antoni	38.566

Conclusions

After the final refinement, based on the number of restaurants per population, our final conclusion consists of 3 neighborhoods:

- **Camp d'en Grassot i Gràcia Nova**
- **Vila de Gràcia**
- **Sant Antoni**

Discussion

In this analysis we did not take other factors into account, like:

- economic affluence in residents in each neighborhood,
- tourist movement,
- number of hotels in the area,
- rental/buying prices for property,
- public transport and accessibility,
- crime rate.

Based on the machine learning techniques, our conclusion can be a first step to a more thorough analysis.