Opening a Pizza Place in the Cities of Essen or Duesseldorf

INTRODUCTION

A fictitious agent lives between Essen (North Rhine Westphalia, Germany) and Duesseldorf (North Rhine Westphalia, Germany). He has a plan to open a pizza place in Essen or in Duesseldorf. He already has precise ideas about the target group he would like to address in particular: People who do not want to spend too much money and at the same time are interested in getting something tasty to eat quickly.

One starting point is to first look at where pizza places are located in Essen and Duesseldorf. Are there places where there are particularly many pizza places? Are there places where there are no pizza places? It stands to reason that locations where there are a lot of pizza places are in high demand. This could argue for opening a pizza place nearby there as well. At the same time, the competition there is high and the risk of not reaching a sufficient number of customers to enable the long-term maintenance of the restaurant is not negligible.

DATA

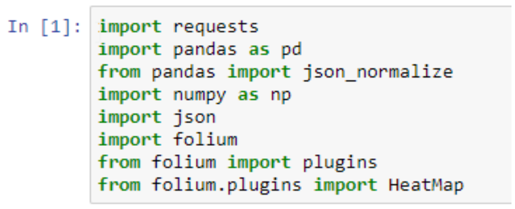
For the reasons mentioned above, other data points are added: Locations of colleges and universities. Why? Classes at schools and universities often last into the afternoon.

We assume that both groups of people do not feel like eating in the cafeteria every day and will therefore also eat at a pizza place nearby. Because of that we combine data points from pizza places with schools and universities. We use Foursquare, an independent platform that provides location data. This allows us to get the exact geodata of Pizza Places, Colleges and Universities. First, we visually represent the locations of the pizza places on a map to provide an initial overview of their locations. Subsequently, we use a heat map to show the concentration of pizza restaurants more clearly. This way we can directly see where most of the pizza places are located.

Now we add locations of schools and universities to the map. Thereby, we can underline where there are educational institutions, but no or few pizza places. If we find such places, they could be a potential location for opening the pizza place.

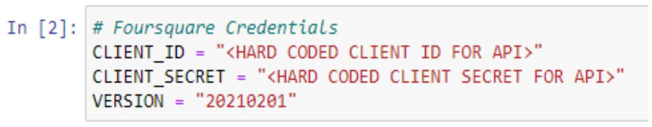
METHODOLOGY

First, all necessary libraries are imported, which are used in this project for data acquisition, data processing and analysis.



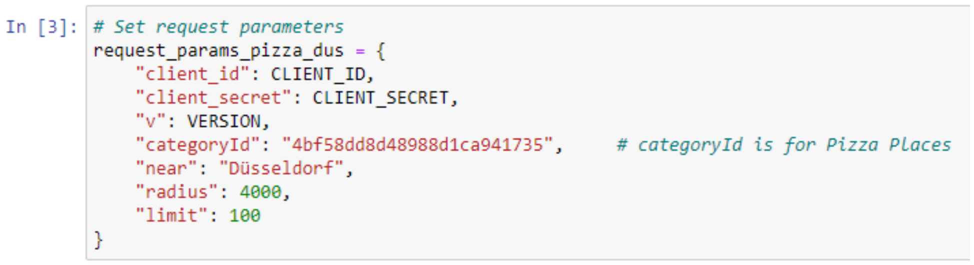
```
In [1]: import requests
        import pandas as pd
        from pandas import json_normalize
        import numpy as np
        import json
        import folium
        from folium import plugins
        from folium.plugins import HeatMap
```

Foursquare is used to visualize the geodata accordingly. We use the API of this service to retrieve relevant data. In order to use the API, we need a client ID as well as the client secret.



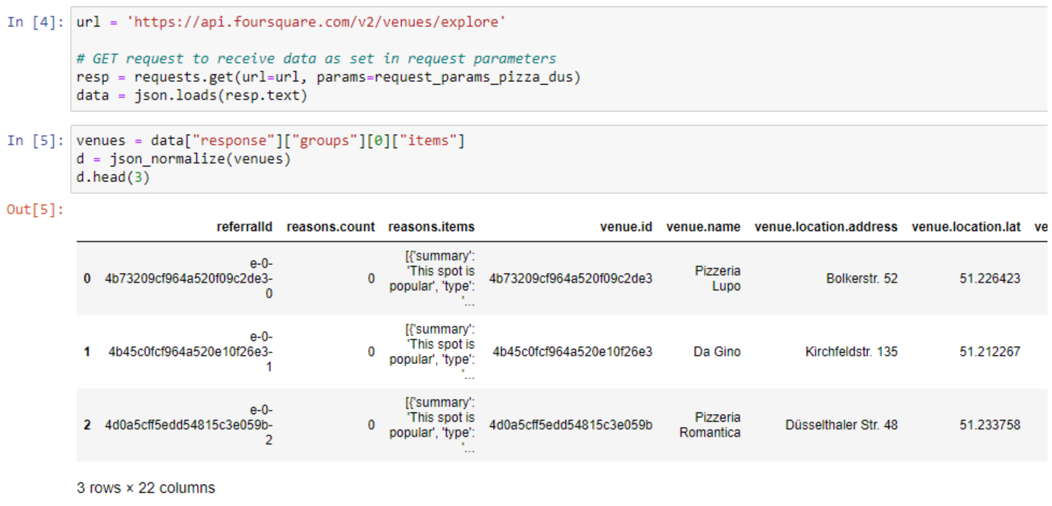
```
In [2]: # Foursquare Credentials
        CLIENT_ID = "<HARD CODED CLIENT ID FOR API>"
        CLIENT_SECRET = "<HARD CODED CLIENT SECRET FOR API>"
        VERSION = "20210201"
```

Next, we use a dictionary to set the request parameters. The category ID you see is a unique identifier for Pizza Places on Foursquare. We first want to get information about Duesseldorf and search for Pizza Places near Duesseldorf in a radius of 4000 meters. The search results are limited to 100 Pizza Places. This should list enough Pizza Places to get an adequate impression of the location and number of Pizza Places in Duesseldorf.



```
In [3]: # Set request parameters
        request_params_pizza_dus = {
            "client_id": CLIENT_ID,
            "client_secret": CLIENT_SECRET,
            "v": VERSION,
            "categoryId": "4bf58dd8d48988d1ca941735",      # categoryId is for Pizza Places
            "near": "Düsseldorf",
            "radius": 4000,
            "limit": 100
        }
```

We use the URL "https://api.foursquare.com/v2/venues/explore" to send a GET request with the request parameters. The response is Pizza Places with various associated attributes such as name, address, lattitude, longitutude and others.



```
In [4]: url = 'https://api.foursquare.com/v2/venues/explore'

        # GET request to receive data as set in request parameters
        resp = requests.get(url=url, params=request_params_pizza_dus)
        data = json.loads(resp.text)
```

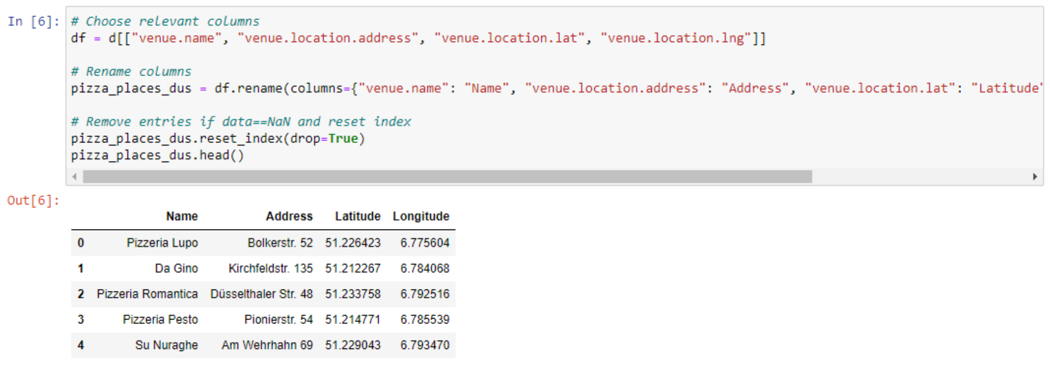
```
In [5]: venues = data["response"]["groups"][0]["items"]
        d = json_normalize(venues)
        d.head(3)
```

Out[5]:

	referralId	reasons.count	reasons.items	venue.id	venue.name	venue.location.address	venue.location.lat	ve
0	e-0-4b73209cf964a520f09c2de3-0	0	[{'summary': 'This spot is popular', 'type': '...	4b73209cf964a520f09c2de3	Pizzeria Lupo	Bolkerstr. 52	51.226423	
1	e-0-4b45c0fcf964a520e10f26e3-1	0	[{'summary': 'This spot is popular', 'type': '...	4b45c0fcf964a520e10f26e3	Da Gino	Kirchfeldstr. 135	51.212267	
2	e-0-4d0a5cff5edd54815c3e059b-2	0	[{'summary': 'This spot is popular', 'type': '...	4d0a5cff5edd54815c3e059b	Pizzeria Romantica	Düsselthaler Str. 48	51.233758	

3 rows × 22 columns

For our analysis we are only interested in the name, address and geodata (longitude and latitude). We store these attributes in a new DataFrame "df", which we will work with from now on. We rename the columns of our DataFrame to make the attributes understandable and readable.



```
In [6]: # Choose relevant columns
        df = d[["venue.name", "venue.location.address", "venue.location.lat", "venue.location.lng"]]

        # Rename columns
        pizza_places_dus = df.rename(columns={"venue.name": "Name", "venue.location.address": "Address", "venue.location.lat": "Latitude'

        # Remove entries if data==NaN and reset index
        pizza_places_dus.reset_index(drop=True)
        pizza_places_dus.head()
```

Out[6]:

	Name	Address	Latitude	Longitude
0	Pizzeria Lupo	Bolkerstr. 52	51.226423	6.775604
1	Da Gino	Kirchfeldstr. 135	51.212267	6.784068
2	Pizzeria Romantica	Düsselthaler Str. 48	51.233758	6.792516
3	Pizzeria Pesto	Pionierstr. 54	51.214771	6.785539
4	Su Nuraghe	Am Wehrhahn 69	51.229043	6.793470

The next step is to visualize the obtained data. We can use Longitude and Latitude to display the location of the Pizza Places exactly on a map. We use the Folium library to create a map instance for Duesseldorf ("dus_map").

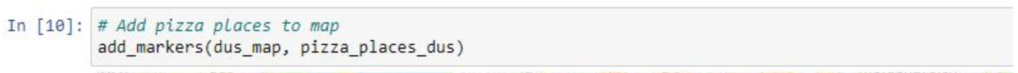
In [7]:
```python
# Düsseldorf geolocation
lat = "51.221332448"
long = "6.785496858"

# Create Map instance für Düsseldorf
dus_map = folium.Map(location=[lat, long], zoom_start=13)
dus_map
```

Out[7]:

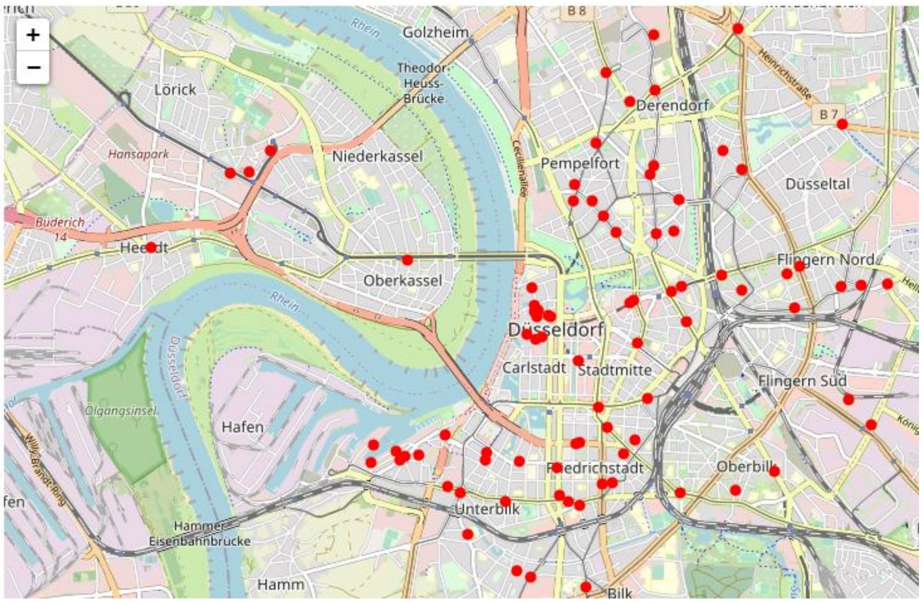
+
−
Lörick
Heuss-Brücke
Derendorf
B 7
Hansapark
Niederkassel
Pempelfort
Düsseltal
Büderich
Heerdt
Oberkassel
Flingern Nord
Rhein
Düsseldorf
Carlstadt
Stadtmitte
Flingern Süd
Olgangsinsel
Hafen
Friedrichstadt
Oberbilk
Willy-Brandt-Ring
Unterbilk
Hammer Eisenbahnbrücke
Hamm
Bilk
B 326
Südpark
Neuss-Hafen

To display the Pizza Places on the map, we create a function "add_markers", which is passed the map instance and the DataFrame containing necessary attributes.

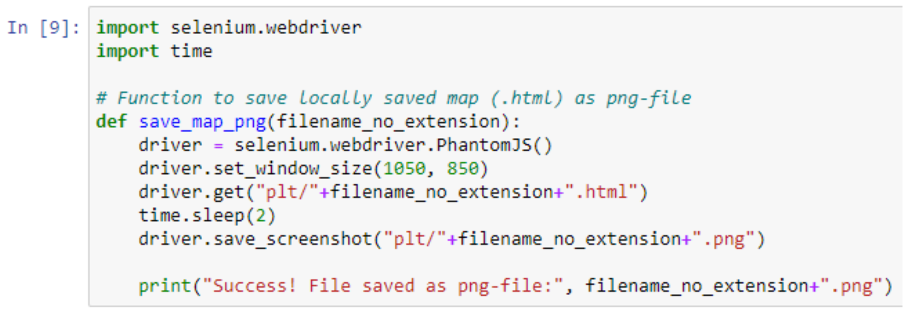
In [8]:
```python
# Function to add markers from df to df_map
def add_markers(df_map, df, color="red"):
    for lat, long, name in zip(df["Latitude"],
                               df["Longitude"],
                               df["Name"]):
        folium.CircleMarker(
            [lat, long],
            radius=2.5,
            fill=True,
            color=color,
            fill_color=color,
            fill_opacity=1
            ).add_to(df_map)

    display(df_map)
```



```
In [10]: # Add pizza places to map
         add_markers(dus_map, pizza_places_dus)
```



Further we create the function "save_map_png", which has no necessity for the correct execution of this script, but saves the passed map instance as a png file, that we can use separately - for example for presentations.

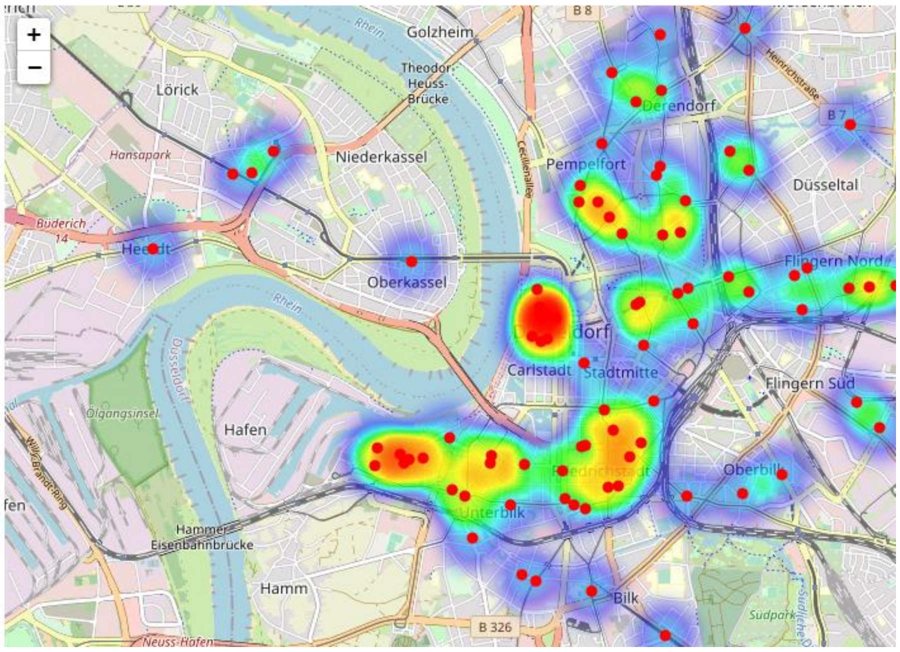


```
In [9]: import selenium.webdriver
        import time

        # Function to save locally saved map (.html) as png-file
        def save_map_png(filename_no_extension):
            driver = selenium.webdriver.PhantomJS()
            driver.set_window_size(1050, 850)
            driver.get("plt/"+filename_no_extension+".html")
            time.sleep(2)
            driver.save_screenshot("plt/"+filename_no_extension+".png")

            print("Success! File saved as png-file:", filename_no_extension+".png")
```

Based on the Pizza Places locations, we now create a Heatmap to show where there are many Pizza Places and where there are few. To do this, we create a Heatmap layer using the Heatmap function provided by the Folium library. This Heatmap layer is added to the map instance "dus_map" using the "add_child" function, which is also in the Folium library.

```
In [11]: # Add HeatMap
         heatmap_data = pizza_places_dus[["Latitude", "Longitude"]].to_numpy().tolist()
         dus_map.add_child(plugins.HeatMap(heatmap_data))

         dus_map
```

Out[11]:



Now that we have represented the Pizza Places by markers and a heatmap, the question is where colleges and universities are located in Duesseldorf. Again we use the API of Foursquare with our request parameters. We only adjust the Category ID to get the data about Colleges and Universities as response.

```
In [13]: # Set request parameters
         request_params_college_dus = {
             "client_id": CLIENT_ID,
             "client_secret": CLIENT_SECRET,
             "v": VERSION,
             "categoryId": "4d4b7105d754a06372d81259",     # categoryId is for Colleges and University
             "near": "Düsseldorf",
             "radius": 4000,
             "limit": 100
         }
```

```
In [14]: url = 'https://api.foursquare.com/v2/venues/explore'

         # GET request to receive data as set in request parameters
         resp = requests.get(url=url, params=request_params_college_dus)
         data = json.loads(resp.text)
```

```
In [15]: venues = data["response"]["groups"][0]["items"]
         d = json_normalize(venues)
```

We store the necessary data in a DataFrame "colleges_dus" and again adjust the names of the attributes.

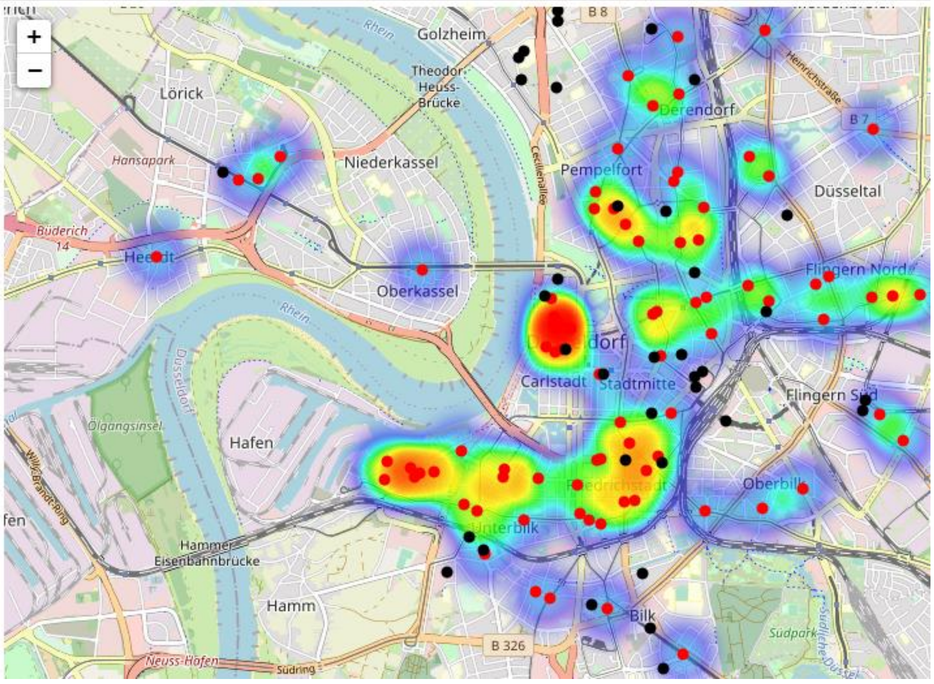
In [16]:
```
# Choose relevant columns
df = d[["venue.name", "venue.location.address", "venue.location.lat", "venue.location.lng"]]

# Rename columns
colleges_dus = df.rename(columns={"venue.name": "Name", "venue.location.address": "Address", "venue.lo

# Remove entries if data==NaN and reset index
colleges_dus.reset_index(drop=True)
colleges_dus.head()
```

Out[16]:

	Name	Address	Latitude	Longitude
0	Walter-Eucken-Berufskolleg Dependance	NaN	51.231055	6.791989
1	Konfuzius Institut	Graf-Adolf Straße 63	51.219497	6.786359
2	Die Brücke am Heinrich-Heine-Platz (VHS)	Kasernenstraße 6	51.224711	6.774901
3	Zentralbibliothek Düsseldorf	Bertha-von-Suttner-Platz 1	51.218803	6.796055
4	DAPR - Deutsche Akademie für Public Relations	NaN	51.229192	6.772290

We can then pass this DataFrame to the function "add_markers" and plot the locations of the colleges and universities as black dots on the map.

In [17]:
```
add_markers(dus_map, colleges_dus, "black")
```



We use the same approach to analyze the city of Essen in terms of the location of Pizza Places, Colleges and Universities. Once again, we use the API of Foursquare to request necessary data.

```
In [19]: # GET request to receive data as set in request parameters
         request_params = {
             "client_id": CLIENT_ID,
             "client_secret": CLIENT_SECRET,
             "v": VERSION,
             "categoryId": "4bf58dd8d48988d1ca941735",        # categoryId is for Pizza Places
             "near": "Essen",
             "radius": 4000,
             "limit": 100
         }

         url = 'https://api.foursquare.com/v2/venues/explore'

         # GET request to receive data as set in request parameters
         resp = requests.get(url=url, params=request_params)
         data = json.loads(resp.text)
```

```
In [20]: venues = data["response"]["groups"][0]["items"]
         d = json_normalize(venues)
```

We use a heatmap to show the number of Pizza Places in a given region.

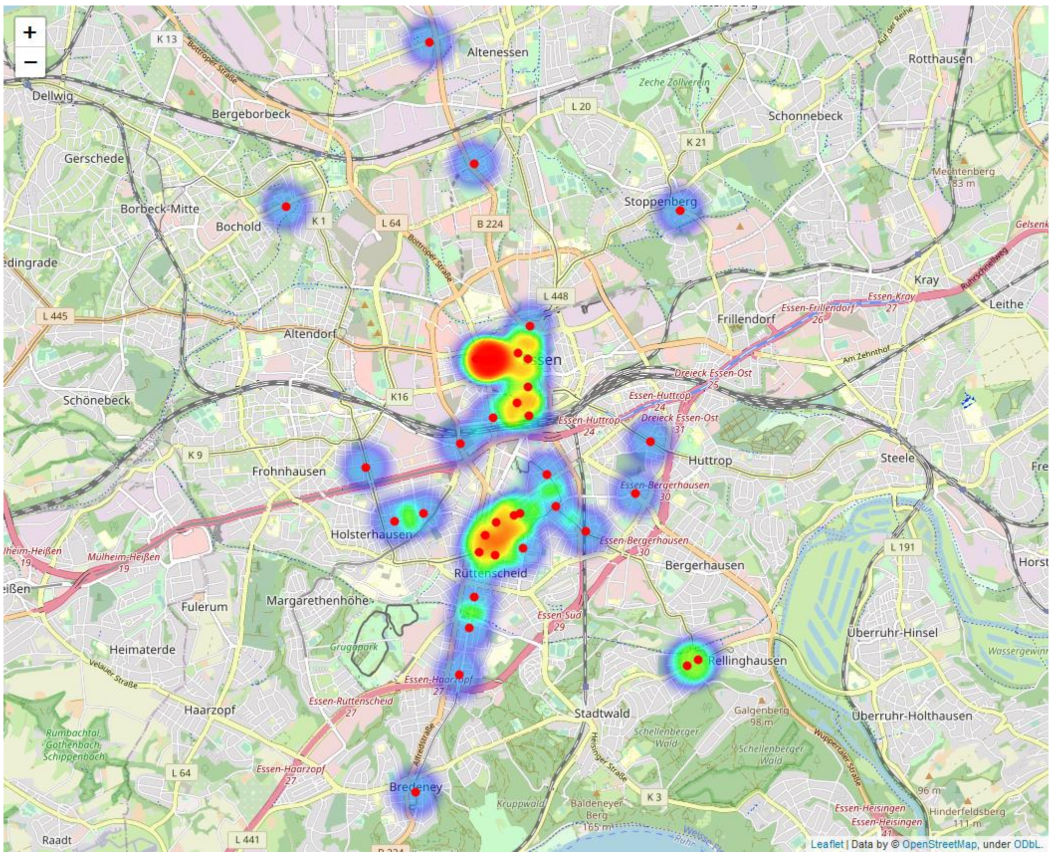


Figure 1: Heatmap Pizza Places in Essen

Finally, using the Foursquare API, we query the locations of colleges and universities to display them on the map for the city of Essen.

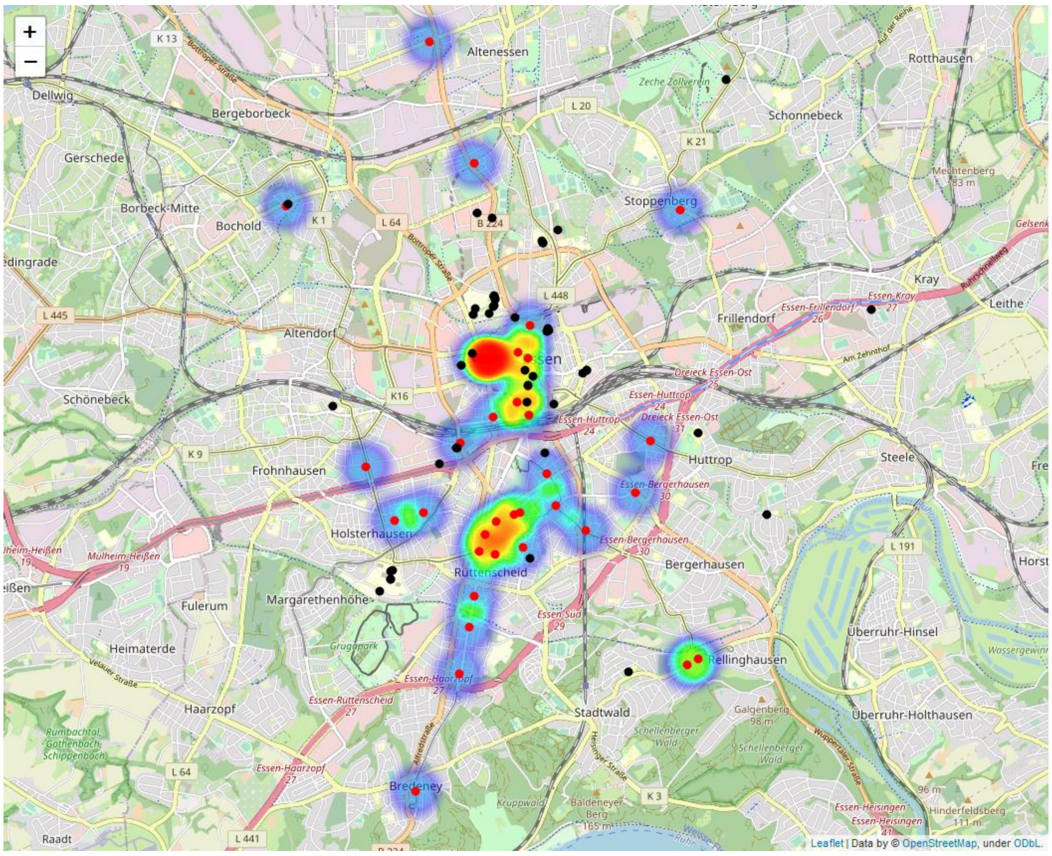


Figure 2: Heatmap Pizza Places and Colleges/ Universities in Essen

RESULTS

There are many Pizza Places in Duesseldorf as well as in Essen. Clusters of Pizza Places can be found in both cities, suggesting that there is sufficiently high demand in these regions.

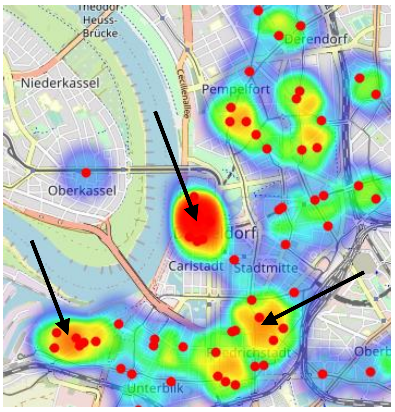


Figure 3: Cluster Pizza Places DUS

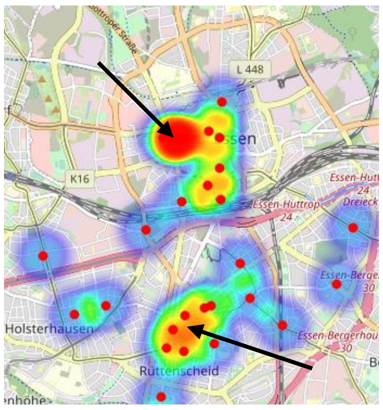


Figure 4: Cluster Pizza Places ESSEN

Likewise, there are also areas where Pizza Places exist, but only sporadically. On this basis, it can be seen that there are regions in both cities where there are no or hardly any Pizza Places, but there are Colleges and/ or Universities.

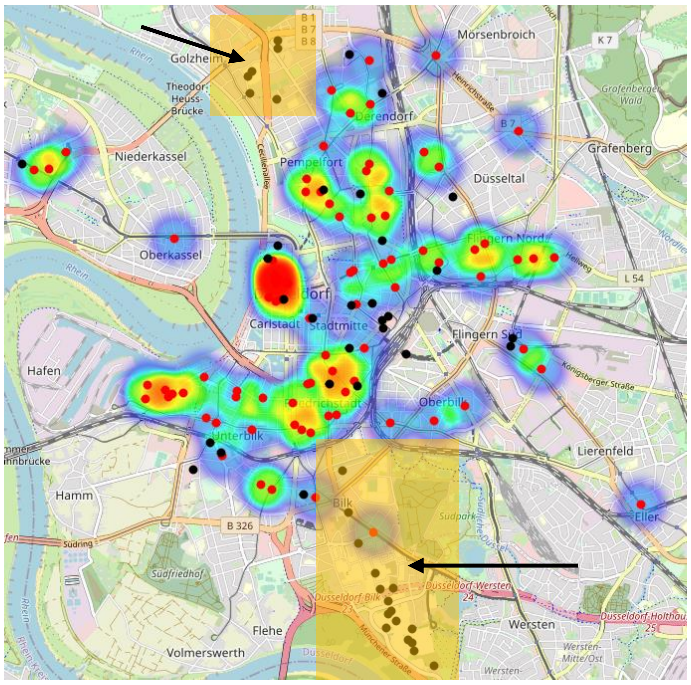


Figure 5: Colleges/ Universities, no Pizza Places DUS

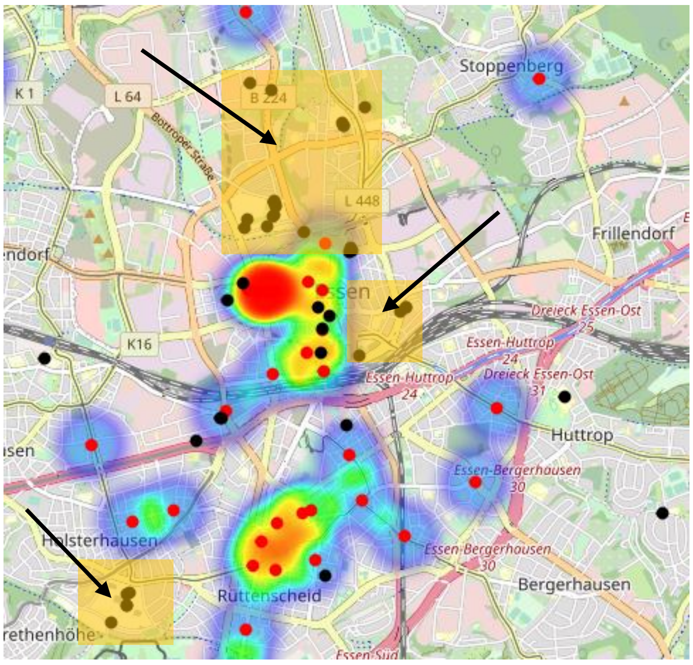


Figure 6: Colleges/ Universities, no Pizza Places ESSEN

Discussion

Based on the results, two regions can be identified in Duesseldorf where there are some colleges and universities, but (almost) no Pizza Places. These two regions are indicated above by the yellow shaded areas. In Essen, there are three areas that could be considered for opening a Pizza Place with the above target group - these are again highlighted in yellow shading. In both Duesseldorf and Essen, there seems to be no competition from other Pizza Places in these regions. As a next step, it is advisable to explore the local conditions on site: Are there any other establishments, offices, banks or businesses there that could be potential customers for the Pizza Place? Is there real estate available locally that could be considered for opening the Pizza Place? What about the local infrastructure? Based on the findings, a final decision can be made in which region it makes the most sense to open a Pizza Place.

Conclusion

Through the analysis regarding a suitable region to open a Pizza Place in the cities of Duesseldorf and Essen, which should appeal mainly to young people such as pupils and students, has identified potentially suitable regions. While two attractive regions can be identified in Duesseldorf, there are three areas in Essen that potentially represent a lucrative opportunity to open a Pizza Place.