Capstone Project - The Battle of the Neighborhoods (Week 2)

Applied Data Science Capstone by IBM/Coursera

Table of contents

Introduction: Business Problem

In this project, we will try to find an optimal location for a pizzeria. Specifically, this report will be targeted to stakeholders interested in opening a **Pizzeria** in **Central in Rio de Janeiro city**, Brazil.

Since there are lots of pizzeria in Central we will try to detect **locations that are not already crowded with a pizzeria**. We are also particularly interested in **areas without Pizzaria in the vicinity**. We would also prefer locations **as close to city center as possible**, assuming that the first two conditions are met.

We will use our data science powers to generate a few most promising neighborhoods based on these criteria. The advantages of each area will then be clearly expressed so that the best possible final location can be chosen by stakeholders.

Data

Based on definition of our problem, factors that will influence our decission are:

- Taxe of cargo
- number of and distance to pizzeria in the neighborhood, if any
- distance of neighborhood from city center

We decided to use regularly spaced grid of locations, centered around city center, to define our neighborhoods.

Following data sources will be needed to extract/generate the required information:

- centers of candidate areas will be generated algorithmically and approximate addresses of centers of those areas will be obtained using **Google Maps API reverse geocoding**.
- number of pizzeria and their type and location in every neighborhood will be obtained using **Foursquare API**
- coordinate of Copacabana center will be obtained using **Google Maps API geocoding** of well known Rio de Janeiro location Art Chopp, Taquara, Rio de Janeiro.

Neighborhood Candidates

Let's create latitude & longitude coordinates for centroids of our candidate neighborhoods. We will create a grid of cells covering our area of interest which is aprox. 12x12 killometers centered around Central - RJ city.

Let's first find the latitude & longitude of Central city RJ, using specific, well known address Google Maps geocoding API.

In [1]:

```
import requests # library to handle requests
import pandas as pd # library for data analsysis
import numpy as np # library to handle data in a vectorized manner
import random # library for random number generation

from geopy.geocoders import Nominatim # module to convert an address into latitude an

# libraries for displaying images
from IPython.display import Image
from IPython.core.display import HTML

# tranforming json file into a pandas dataframe library
from pandas.io.json import json_normalize

import folium # plotting library

print('Folium installed')
```

```
Folium installed
Libraries imported.
```

In [2]:

```
api_key = '***'
```

In [3]:

```
import requests

def get_coordinates(api_key, address, verbose=False):
    try:
        url = 'https://maps.googleapis.com/maps/api/geocode/json?key={}&address={}'.f

        response = requests.get(url).json()
        if verbose:
            print('Google Maps API JSON result =>', response)
        results = response['results']
        geographical_data = results[0]['geometry']['location'] # get geographical coo
        lat = geographical_data['lat']
        lon = geographical_data['lng']
        return [lat, lon]
    except:
        return [None, None]

rj_center = get_coordinates(api_key, address)
```

```
Coordinate of   Art Chopp, Taquara, Rio de Janeiro: [-22.9236275, -43.3775954]
```

Now let's create a grid of area candidates, equaly spaced, centered around city center and within ~6km from Hotel Atlantico. Our neighborhoods will be defined as circular areas with a radius of 300 meters, so our neighborhood centers will be 600 meters apart.

To accurately calculate distances we need to create our grid of locations in Cartesian 2D coordinate system which allows us to calculate distances in meters (not in latitude/longitude degrees). Then we'll project those coordinates back to latitude/longitude degrees to be shown on Folium map. So let's create functions to convert between WGS84 spherical coordinate system (latitude/longitude degrees) and UTM Cartesian coordinate system (X/Y coordinates in meters).

In [4]:

```
#!pip install shapely
import shapely.geometry

#!pip install pyproj
import pyproj

import math

def lonlat_to_xy(lon, lat):
    proj_latlon = pyproj.Proj(proj='latlong',datum='WGS84')
    proj_xy = pyproj.Proj(proj="utm", zone=33, datum='WGS84')
    xy = pyproj.transform(proj_latlon, proj_xy, lon, lat)
    return xy[0], xy[1]

def xy_to_lonlat(x, y):
    proj_latlon = pyproj.Proj(proj='latlong',datum='WGS84')
    proj_xy = pyproj.Proj(proj="utm", zone=33, datum='WGS84')
    lonlat = pyproj.transform(proj_xy, proj_latlon, x, y)
    return lonlat[0], lonlat[1]

def calc_xy_distance(x1, y1, x2, y2):
    dx = x2 - x1
    dy = y2 - y1
    return math.sqrt(dx*dx + dy*dy)

print('Coordinate transformation check')
print('-------------------------------')
print('Taquara longitude={}, latitude={}'.format(rj_center[1], rj_center[0]))
x, y = lonlat_to_xy(rj_center[1], rj_center[0])
print('Taquara UTM X={}, Y={}'.format(x, y))
lo, la = xy_to_lonlat(x, y)
print('Taquara longitude={}, latitude={}'.format(lo, la))
```

```
Coordinate transformation check
-------------------------------
Taquara longitude=-43.3775954, latitude=-22.9236275
Taquara UTM X=-6241503.61015519, Y=-4321215.150004552
Taquara longitude=-43.37759540000197, latitude=-22.923627499999675
```

Let's create a **hexagonal grid of cells**: we offset every other row, and adjust vertical row spacing so that **every cell center is equally distant from all it's neighbors**.

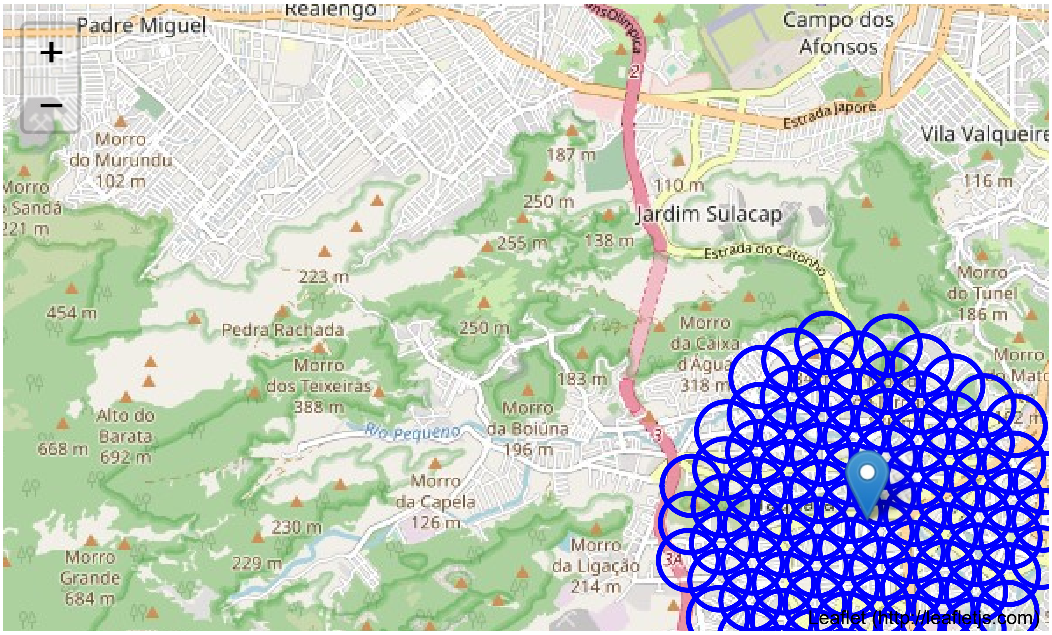
In [5]:
```python
rj_center_x, rj_center_y = lonlat_to_xy(rj_center[1], rj_center[0]) # City center in

k = math.sqrt(3) / 2 # Vertical offset for hexagonal grid cells
x_min = rj_center_x - 3000
x_step = 600
y_min = rj_center_y - 3000 - (int(21/k)*k*600 - 6000)/2
y_step = 600 * k

latitudes = []
longitudes = []
distances_from_center = []
xs = []
ys = []
for i in range(0, int(21/k)):
    y = y_min + i * y_step
    x_offset = 300 if i%2==0 else 0
    for j in range(0, 21):
        x = x_min + j * x_step + x_offset
        distance_from_center = calc_xy_distance(rj_center_x, rj_center_y, x, y)
        if (distance_from_center <= 3001):
            lon, lat = xy_to_lonlat(x, y)
            latitudes.append(lat)
            longitudes.append(lon)
            distances_from_center.append(distance_from_center)
            xs.append(x)
            ys.append(y)
```

```
92 candidate neighborhood centers generated.
```

In [6]:

In [7]:
```python
map_rj = folium.Map(location=rj_center, zoom_start=13)
folium.Marker(rj_center, popup='Art Chopp, Taquara, Rio de Janeiro').add_to(map_rj)
for lat, lon in zip(latitudes, longitudes):
    #folium.CircleMarker([lat, lon], radius=2, color='blue', fill=True, fill_color='b
    folium.Circle([lat, lon], radius=300, color='blue', fill=False).add_to(map_rj)
    #folium.Marker([lat, lon]).add_to(map_rj)
map_rj
```

Out[7]:



OK, we now have the coordinates of centers of neighborhoods/areas to be evaluated, equally spaced (distance from every point to it's neighbors is exactly the same) and within ~3km from Art Chopp.

Let's now use Google Maps API to get approximate addresses of those locations.

In [8]:
```python
def get_address(api_key, latitude, longitude, verbose=False):
    try:
        url = 'https://maps.googleapis.com/maps/api/geocode/json?key={}&latlng={},{}'
        response = requests.get(url).json()
        if verbose:
            print('Google Maps API JSON result =>', response)
        results = response['results']
        address = results[0]['formatted_address']
        return address
    except:
        return None

addr = get_address(api_key, rj_center[0], rj_center[1])
print('Reverse geocoding check')
print('-----------------------')
print('Address of [{}, {}] is: {}'.format(rj_center[0], rj_center[1], addr))
```

```
Reverse geocoding check
-----------------------
Address of [-22.9236275, -43.3775954] is: Estr. Macembu, 63 - Taquara, Rio de Jane
iro - RJ, 22710-241, Brazil
```

In [9]:
```
print('Obtaining location addresses: ', end='')
addresses = []
for lat, lon in zip(latitudes, longitudes):
    address = get_address(api_key, lat, lon)
    if address is None:
        address = 'NO ADDRESS'
    address = address.replace(', None', '') # We don't need country part of address
    addresses.append(address)
    print(' .', end='')
print(' done.')
```

```
Obtaining location addresses:  . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . done.
```

In [10]:
```
addresses[1:10]
```

Out[10]:
```
['R. Clodomir Lucas dos Reis, 66 - Jacarepaguá, Rio de Janeiro - RJ, 22713-562, Br
azil',
 'Tv. Gitahy, 4 - Jacarepaguá, Rio de Janeiro - RJ, 22713-566, Brazil',
 'Estrada do Guerenguê, 1992 - Taquara, Rio de Janeiro - RJ, 22713-001, Brazil',
 'R. Pôrto Vitória, 77a - Curicica, Rio de Janeiro - RJ, 22710-034, Brazil',
 'R. A, 624 - Jacarepaguá, Rio de Janeiro - RJ, 22743-846, Brazil',
 'Estr. do Outeiro Santo, 1168 - Taquara, Rio de Janeiro - RJ, 22713-169, Brazil',
 'R. Nadim Zeidam Ac Est Outeiro Santo, 2 - Taquara, Rio de Janeiro - RJ, 22713-16
9, Brazil',
 'Estr. do Outeiro Santo, 47 - Taquara, Rio de Janeiro - RJ, 22713-169, Brazil',
 'Estrada do Guerenguê próximo ao 1770 - Jacarepaguá, Rio de Janeiro - RJ, 22713-0
04, Brazil']
```

Looking good. Let's now place all this into a Pandas dataframe.

In [11]:
```
import pandas as pd

df_locations = pd.DataFrame({'Address': addresses,
                             'Latitude': latitudes,
                             'Longitude': longitudes,
                             'X': xs,
                             'Y': ys,
                             'Distance from center': distances_from_center})

df_locations.head(10)
```

Out[11]:

	Address	Latitude	Longitude	X	Y	Distance from center
0	R. Recanto do Outeiro, 102 - Jacarepaguá, Rio ...	-22.932262	-43.392147	-6.242704e+06	-4.323813e+06	2861.817604
1	R. Clodomir Lucas dos Reis, 66 - Jacarepaguá, ...	-22.934069	-43.389094	-6.242104e+06	-4.323813e+06	2666.458325
2	Tv. Gitahy, 4 - Jacarepaguá, Rio de Janeiro - ...	-22.935876	-43.386041	-6.241504e+06	-4.323813e+06	2598.076211
3	Estrada do Guerenguê, 1992 - Taquara, Rio de J...	-22.937683	-43.382987	-6.240904e+06	-4.323813e+06	2666.458325
4	R. Pôrto Vitória, 77a - Curicica, Rio de Janei...	-22.939490	-43.379933	-6.240304e+06	-4.323813e+06	2861.817604
5	R. A, 624 - Jacarepaguá, Rio de Janeiro - RJ, ...	-22.927103	-43.395037	-6.243604e+06	-4.323294e+06	2954.657341
6	Estr. do Outeiro Santo, 1168 - Taquara, Rio de...	-22.928909	-43.391985	-6.243004e+06	-4.323294e+06	2563.201124
7	R. Nadim Zeidam Ac Est Outeiro Santo, 2 - Taqu...	-22.930716	-43.388932	-6.242404e+06	-4.323294e+06	2264.950331
8	Estr. do Outeiro Santo, 47 - Taquara, Rio de J...	-22.932523	-43.385878	-6.241804e+06	-4.323294e+06	2100.000000
9	Estrada do Guerenguê próximo ao 1770 - Jacarep...	-22.934330	-43.382824	-6.241204e+06	-4.323294e+06	2100.000000

In [12]:
```
df_locations.to_pickle('./locations.pkl')
```

Foursquare

Now that we have our location candidates, let's use Foursquare API to get info on restaurants in each neighborhood.

We're interested in venues in 'pizza' category, but in Brazil same restaurant serves pizza too so we will look only those that are restaurants and pizzerias - coffe shops, bakeries etc. are not direct competitors so we don't care about those. So we will include in out list only venues that have 'restaurant' and 'pizza' in category name.

In [13]:
```
client_id = '***********' # your Foursquare ID
client_secret = '*************' # your Foursquare Secret
version = '20180604'
limit = 100
print('Your credentails:')
print('CLIENT_ID: ' + client_id)
print('CLIENT_SECRET:' + client_secret)
```

```
Your credentails:
CLIENT_ID: PB0VXS1VKOUZHYM4C0ODHSL4YKCINOHH0ZJ51LUSXZMMXENI
CLIENT_SECRET:5NEWWZ3WZXQGGKABUVT2SCUJ4ZACI3CGQE3KOLPBQUMZKXTY
```

In [14]:
```
# Category IDs corresponding to Pizzaria were taken from Foursquare web site (https:/

food_category = '4d4b7105d754a06374d81259' # 'Root' category for all food-related ven

pizza_restaurant_categories = ['4bf58dd8d48988d1ca941735']

def is_restaurant(categories, specific_filter=None):
    restaurant_words = ['pizza', 'massas','diner', 'jantar', 'refeição']
    restaurant = False
    specific = False
    for c in categories:
        category_name = c[0].lower()
        category_id = c[1]
        for r in restaurant_words:
            if r in category_name:
                restaurant = True
        if 'fast food' in category_name:
            restaurant = False
        if not(specific_filter is None) and (category_id in specific_filter):
            specific = True
            restaurant = True
    return restaurant, specific

def get_categories(categories):
    return [(cat['name'], cat['id']) for cat in categories]

def format_address(location):
    address = ', '.join(location['formattedAddress'])
    address = address.replace(', Rio de Janeiro', '')
    address = address.replace(', Brazil', '')
    return address

def get_venues_near_location(lat, lon, category, client_id, client_secret, radius=500
    version = '20180724'
    url = 'https://api.foursquare.com/v2/venues/explore?client_id={}&client_secret={}
        client_id, client_secret, version, lat, lon, category, radius, limit)
    try:
        results = requests.get(url).json()['response']['groups'][0]['items']
        venues = [(item['venue']['id'],
                   item['venue']['name'],
                   get_categories(item['venue']['categories']),
                   (item['venue']['location']['lat'], item['venue']['location']['lng'
                   format_address(item['venue']['location']),
                   item['venue']['location']['distance']) for item in results]
    except:
        venues = []
    return venues
```

In [15]:
```python
# Let's now go over our neighborhood locations and get nearby restaurants; we'll also

import pickle

def get_restaurants(lats, lons):
    restaurants = {}
    pizzaria_restaurants = {}
    location_restaurants = []

    print('Obtaining venues around candidate locations:', end='')
    for lat, lon in zip(lats, lons):
        # Using radius=350 to meke sure we have overlaps/full coverage so we don't mi
        venues = get_venues_near_location(lat, lon, food_category, client_id, client_
        area_restaurants = []
        for venue in venues:
            venue_id = venue[0]
            venue_name = venue[1]
            venue_categories = venue[2]
            venue_latlon = venue[3]
            venue_address = venue[4]
            venue_distance = venue[5]
            is_res, is_pizzaria = is_restaurant(venue_categories, specific_filter=piz
            if is_res:
                x, y = lonlat_to_xy(venue_latlon[1], venue_latlon[0])
                restaurant = (venue_id, venue_name, venue_latlon[0], venue_latlon[1],
                if venue_distance<=300:
                    area_restaurants.append(restaurant)
                restaurants[venue_id] = restaurant
                if is_pizzaria:
                    pizzaria_restaurants[venue_id] = restaurant
        location_restaurants.append(area_restaurants)
        print(' .', end='')
    print(' done.')
    return restaurants, pizzaria_restaurants, location_restaurants

# Try to load from local file system in case we did this before
restaurants = {}
pizzaria_restaurants = {}
location_restaurants = []
loaded = False
try:
    with open('restaurants_350.pkl', 'rb') as f:
        restaurants = pickle.load(f)
    with open('pizzaria_restaurants_350.pkl', 'rb') as f:
        pizzaria_restaurants = pickle.load(f)
    with open('location_restaurants_350.pkl', 'rb') as f:
        location_restaurants = pickle.load(f)
    print('Restaurant data loaded.')
    loaded = True
except:
    pass

# If load failed use the Foursquare API to get the data
if not loaded:
    restaurants, pizzaria_restaurants, location_restaurants = get_restaurants(latitud

    # Let's persists this in local file system
    with open('restaurants_350.pkl', 'wb') as f:
        pickle.dump(restaurants, f)
    with open('pizzaria_restaurants_350.pkl', 'wb') as f:
        pickle.dump(pizzaria_restaurants, f)
    with open('location_restaurants_350.pkl', 'wb') as f:
        pickle.dump(location_restaurants, f)
```

In [16]:
```
import numpy as np

print('Total number of restaurants:', len(restaurants))
print('Total number of restaurants:', len(pizzaria_restaurants))
print('Percentage of Pizzeria: {:.2f}%'.format(len(pizzaria_restaurants) / len(restau
```

```
Total number of restaurants: 47
Total number of restaurants: 40
Percentage of Pizzeria: 85.11%
Average number of restaurants in neighborhood: 0.41304347826086957
```

In [17]:
```
print('List of all restaurants')
print('-----------------------')
for r in list(restaurants.values())[:5]:
    print(r)
print('...')
```

```
List of all restaurants
-----------------------
('4f061b6bf790d4c1a356be37', 'Paulista Pizzaria', -22.94590085688614, -43.38408880
586431, 'R. Iperó, 6 - Curicica - RJ, 22710-200, RJ, 22710-200, Brasil', 167, Tru
e, 665682.3611210624, -2538442.049774494)
('583245fd8d8e99259a5ee443', 'pizzaria aquarius', -22.946113, -43.378662, 'Jurand
a, RJ, 22710-191, Brasil', 175, True, 666238.6262549148, -2538471.6714474596)
('509c6720e4b08f34f7639f57', 'Treile do gaguinho', -22.94576072692871, -43.3681869
50683594, 'Brasil', 295, False, 667313.2875366946, -2538444.556813935)
('560554d4498e98c349ae26d2', 'Toronto Grill', -22.941069, -43.391944, 'Brasil', 16
4, True, 664882.6601301269, -2537898.1826091968)
('4ea49a83be7ba4918f2b5fc3', 'Pizzaria Bambini', -22.9446276490718, -43.3853121843
28036, 'R. Guamaré, RJ, Brasil', 253, True, 665558.4519183682, -2538299.688479469)
...
Total: 47
```

In [18]:
```
print('List of all restaurants')
print('-----------------------')
for r in list(pizzaria_restaurants.values())[:5]:
    print(r)
print('...')
```

```
List of all restaurants
-----------------------
('4f061b6bf790d4c1a356be37', 'Paulista Pizzaria', -22.94590085688614, -43.38408880
586431, 'R. Iperó, 6 - Curicica - RJ, 22710-200, RJ, 22710-200, Brasil', 167, Tru
e, 665682.3611210624, -2538442.049774494)
('583245fd8d8e99259a5ee443', 'pizzaria aquarius', -22.946113, -43.378662, 'Jurand
a, RJ, 22710-191, Brasil', 175, True, 666238.6262549148, -2538471.6714474596)
('560554d4498e98c349ae26d2', 'Toronto Grill', -22.941069, -43.391944, 'Brasil', 16
4, True, 664882.6601301269, -2537898.1826091968)
('4ea49a83be7ba4918f2b5fc3', 'Pizzaria Bambini', -22.9446276490718, -43.3853121843
28036, 'R. Guamaré, RJ, Brasil', 253, True, 665558.4519183682, -2538299.688479469)
('540c872b498e634754d21fec', 'pizzaria juranda', -22.94353650893861, -43.382063402
297355, 'Brasil', 219, True, 665892.9510044158, -2538182.531654376)
...
Total: 40
```

In [19]:
```
print('List of all Pizzaria')
print('-----------------------')
for r in list(pizzaria_restaurants)[:10]:
    print(r)
print('...')
print('Total:', len(pizzaria_restaurants))
```

```
List of all Pizzaria
-----------------------
4f061b6bf790d4c1a356be37
583245fd8d8e99259a5ee443
560554d4498e98c349ae26d2
4ea49a83be7ba4918f2b5fc3
540c872b498e634754d21fec
4f5d52d7e4b01219ace674f7
522205e411d27ab2a65982b1
5897a05c266c115619e2c1b9
4d2b7cf38292236a636d34bb
520a71f911d23008f679f31b
...
Total: 40
```

In [20]:
```
location_restaurants
```

Out[20]:
```
[[],
 [('4f061b6bf790d4c1a356be37',
   'Paulista Pizzaria',
   -22.94590085688614,
   -43.38408880586431,
   'R. Iperó, 6 - Curicica - RJ, 22710-200, RJ, 22710-200, Brasil',
   167,
   True,
   665682.3611210624,
   -2538442.049774494)],
 [('583245fd8d8e99259a5ee443',
   'pizzaria aquarius',
   -22.946113,
   -43.378662,
   'Juranda, RJ, 22710-191, Brasil',
   175,
   True,
   666238.6262549148,
   -2538471.6714474596)],
```

Let's now see all the collected restaurants in our area of interest on map, and let's also show pizzeries in different color.

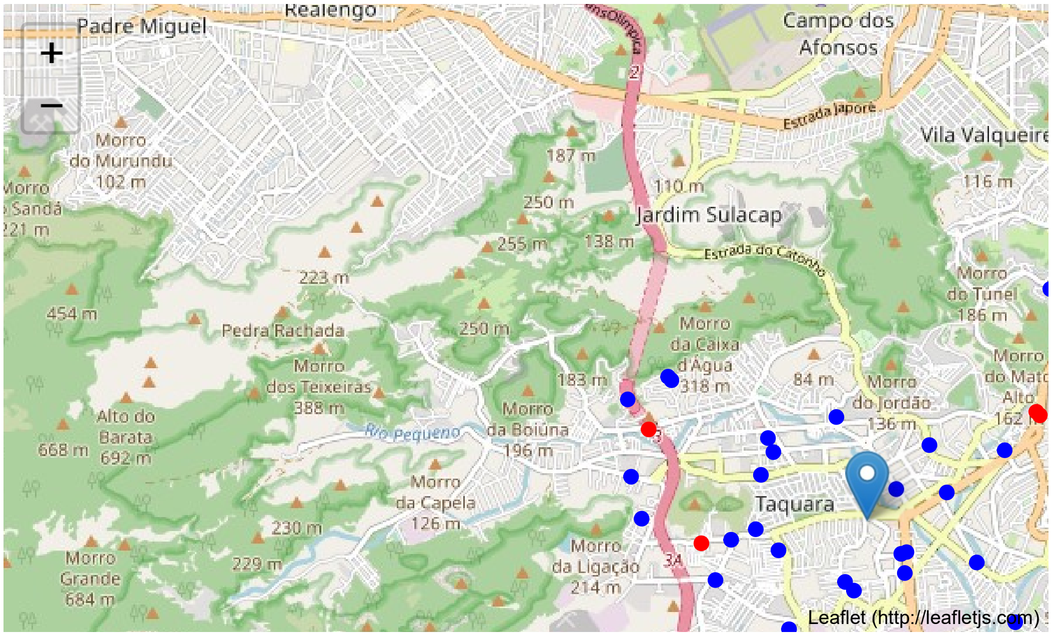
In [21]:
```
print('Pizzaria around location')
print('---------------------------')
for i in range(1, len(location_restaurants)):
    rs = location_restaurants[i][:1]
    names = ', '.join([r[1] for r in rs])
    print('Restaurants around location {}: {}'.format(i+1, names))
```

```
Pizzaria around location
---------------------------
Restaurants around location 2: Paulista Pizzaria
Restaurants around location 3: pizzaria aquarius
Restaurants around location 4:
Restaurants around location 5: Treile do gaguinho
Restaurants around location 6:
Restaurants around location 7: Toronto Grill
Restaurants around location 8: Pizzaria Bambini
Restaurants around location 9: pizzaria juranda
Restaurants around location 10:
Restaurants around location 11: Curtir Pizza
Restaurants around location 12: Mc' Lu Lanches
Restaurants around location 13:
Restaurants around location 14: Léo pizzas
Restaurants around location 15:
Restaurants around location 16:
Restaurants around location 17:
Restaurants around location 18:
Restaurants around location 19: Cavallino
Restaurants around location 20: Pizzaria 14
Restaurants around location 21:
Restaurants around location 22:
Restaurants around location 23:
Restaurants around location 24:
Restaurants around location 25:
Restaurants around location 26: Pizzaria N. S. Fátima
Restaurants around location 27:
Restaurants around location 28:
Restaurants around location 29: Lêpizzalá
Restaurants around location 30: Paulo Pizza
Restaurants around location 31:
Restaurants around location 32: Scuderia Bar Restaurante Pizzaria
Restaurants around location 33:
Restaurants around location 34: Bar e Restaurante Biroska
Restaurants around location 35:
Restaurants around location 36:
Restaurants around location 37: Jiló na Manteiga
Restaurants around location 38: Pizza Cone
Restaurants around location 39: Pizzaria RJ
Restaurants around location 40:
Restaurants around location 41: il Fornaccio
Restaurants around location 42:
Restaurants around location 43: Pizzaria Pizoni's
Restaurants around location 44: Léo Pizzas
Restaurants around location 45: Templo da Pizza
Restaurants around location 46:
Restaurants around location 47:
Restaurants around location 48:
Restaurants around location 49:
Restaurants around location 50:
Restaurants around location 51:
Restaurants around location 52: Porto das Pizzas
Restaurants around location 53:
Restaurants around location 54: Pizza do Valle
Restaurants around location 55:
Restaurants around location 56:
Restaurants around location 57: Pizzaria Gustosita
```

In [22]:

```
map_rj = folium.Map(location=rj_center, zoom_start=13)
folium.Marker(rj_center, popup='adress').add_to(map_rj)
for res in restaurants.values():
    lat = res[2]; lon = res[3]
    is_pizzaria = res[6]
    color = 'blue' if is_pizzaria else 'red'
    folium.CircleMarker([lat, lon], radius=3, color=color, fill=True, fill_color=colo
```

Out[22]:



Looking good. So now we have all the restaurants in area within few kilometers from Art shop, and we know which ones are pizzaries! We also know which restaurants exactly are in vicinity of every neighborhood candidate center.

This concludes the data gathering phase - we're now ready to use this data for analysis to produce the report on optimal locations for a new pizzaries!

Methodology

In this project we will direct our efforts on detecting areas of Taquara that have low restaurant density, particularly those with low number of restaurants that don't serves pizza. We will limit our analysis to area ~6km around city center.

In first step we have collected the required **data: location and type (category) of every restaurant within 6km from Taquara center** (Art Chopp). We have also **identified pizzaries** (according to Foursquare categorization).

Second step in our analysis will be calculation and exploration of '**restaurant density**' across different areas of Taquara - we will use **heatmaps** to identify a few promising areas close to center with low number of restaurants in general (*and* no pizzzaries in vicinity) and focus our attention on those areas.

In third and final step we will focus on most promising areas and within those create **clusters of locations that meet some basic requirements** established in discussion with stakeholders: we will take into consideration locations with **no more than two restaurants in radius of 250 meters**, and we want locations **without pizzaries in radius of 400 meters**. We will present map of all such locations but also create clusters (using **k-means clustering**) of those locations to identify general zones / neighborhoods / addresses which should be a starting point for final 'street level' exploration and search for optimal venue location by stakeholders.

Analysis

Let's perform some basic explanatory data analysis and derive some additional info from our raw data. First let's count the **number of restaurants in every area candidate**:

In [23]:
```
location_restaurants_count = [len(res) for res in location_restaurants]

df_locations['Restaurants in area'] = location_restaurants_count

print('Average number of restaurants in every area with radius=300m:', np.array(locat

df_locations.tail(10)
```

```
Average number of restaurants in every area with radius=300m: 0.41304347826086957
```

Out[23]:

	Address	Latitude	Longitude	X	Y	Distance from center	Restaurants in area
82	Rua São Calisto, 582 - Tanque, Rio de Janeiro ...	-22.912924	-43.372370	-6.241804e+06	-4.319137e+06	2100.000000	0
83	R. Atininga, 463 - Tanque, Rio de Janeiro - RJ...	-22.914729	-43.369316	-6.241204e+06	-4.319137e+06	2100.000000	0
84	R. Imbuí, 252 - Tanque, Rio de Janeiro - RJ, 2...	-22.916535	-43.366261	-6.240604e+06	-4.319137e+06	2264.950331	0
85	R. Lívio Barreto, 80 - Tanque, Rio de Janeiro ...	-22.918340	-43.363206	-6.240004e+06	-4.319137e+06	2563.201124	0
86	R. Meriti, 197 - Tanque, Rio de Janeiro - RJ, ...	-22.920145	-43.360151	-6.239404e+06	-4.319137e+06	2954.657341	2
87	Unnamed Road - Tanque, Rio de Janeiro - RJ, 22...	-22.907767	-43.375264	-6.242704e+06	-4.318617e+06	2861.817604	0
88	R. Tarso Coimbra, 35 - Tanque, Rio de Janeiro ...	-22.909571	-43.372210	-6.242104e+06	-4.318617e+06	2666.458325	0
89	R. Dr. Tomaz Rosas, 54 - Tanque, Rio de Janeir...	-22.911376	-43.369156	-6.241504e+06	-4.318617e+06	2598.076211	0
90	R. Atininga, 167 - Tanque, Rio de Janeiro - RJ...	-22.913181	-43.366101	-6.240904e+06	-4.318617e+06	2666.458325	0
91	Tv. Dalias Ac Candido Benicio 4168, 12 - Tanqu...	-22.914986	-43.363046	-6.240304e+06	-4.318617e+06	2861.817604	0

OK, now let's calculate the **distance to nearest pizzaria from every area candidate center** (not only those within 300m - we want distance to closest one, regardless of how distant it is).

In [24]:
```
distances_to_pizzaria_restaurant = []

for area_x, area_y in zip(xs, ys):
    min_distance = 10000
    for res in pizzaria_restaurants.values():
        res_x = res[7]
        res_y = res[8]
        d = calc_xy_distance(area_x, area_y, res_x, res_y)
        if d<min_distance:
            min_distance = d
    distances_to_pizzaria_restaurant.append(min_distance)

df_locations['Distance to Pizzaries restaurant'] = distances_to_pizzaria_restaurant
```

In [25]:
```
df_locations.head(10)
```

Out[25]:

	Address	Latitude	Longitude	X	Y	Distance from center	Restaurants in area	Distance to Pizzaries restaurant
0	R. Recanto do Outeiro, 102 - Jacarepaguá, Rio ...	-22.932262	-43.392147	-6.242704e+06	-4.323813e+06	2861.817604	0	10000
1	R. Clodomir Lucas dos Reis, 66 - Jacarepaguá, ...	-22.934069	-43.389094	-6.242104e+06	-4.323813e+06	2666.458325	1	10000
2	Tv. Gitahy, 4 - Jacarepaguá, Rio de Janeiro - ...	-22.935876	-43.386041	-6.241504e+06	-4.323813e+06	2598.076211	1	10000
3	Estrada do Guerenguê, 1992 - Taquara, Rio de J...	-22.937683	-43.382987	-6.240904e+06	-4.323813e+06	2666.458325	0	10000
4	R. Pôrto Vitória, 77a - Curicica, Rio de Janei...	-22.939490	-43.379933	-6.240304e+06	-4.323813e+06	2861.817604	1	10000
5	R. A, 624 - Jacarepaguá, Rio de Janeiro - RJ, ...	-22.927103	-43.395037	-6.243604e+06	-4.323294e+06	2954.657341	0	10000
6	Estr. do Outeiro Santo, 1168 - Taquara, Rio de...	-22.928909	-43.391985	-6.243004e+06	-4.323294e+06	2563.201124	1	10000
7	R. Nadim Zeidam Ac Est Outeiro Santo, 2 - Taqu...	-22.930716	-43.388932	-6.242404e+06	-4.323294e+06	2264.950331	1	10000
8	Estr. do Outeiro Santo, 47 - Taquara, Rio de J...	-22.932523	-43.385878	-6.241804e+06	-4.323294e+06	2100.000000	1	10000
9	Estrada do Guerenguê próximo ao 1770 - Jacarep...	-22.934330	-43.382824	-6.241204e+06	-4.323294e+06	2100.000000	0	10000

In [26]:
```
print('Average distance to closest pizzaria from each area center:', df_locations['Di
```

```
Average distance to closest pizzaria from each area center: 10000.0
```

OK, so **on average Pizzaries can be found within ~500m** from every area center candidate. That's fairly close, so we need to filter our areas carefully!

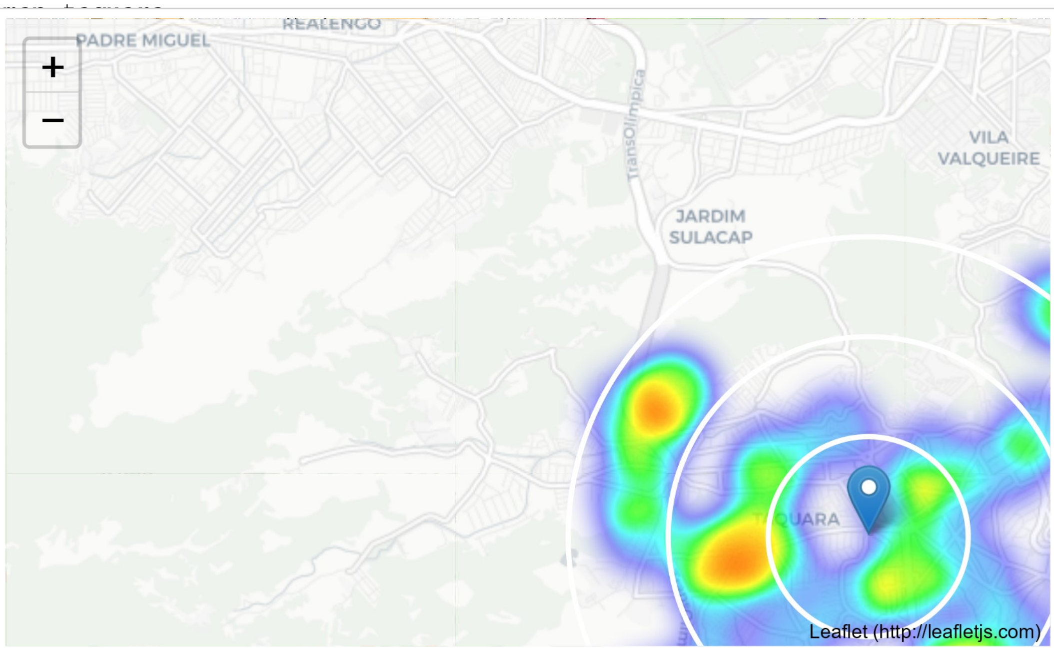
Let's crete a map showing **heatmap / density of restaurants** and try to extract some meaningfull info from that. Also, let's show **borders of Taquara boroughs** on our map and a few circles indicating distance of 1km, 2km and 3km from Art Shop.

In [27]:
```
taquara_boroughs_url = 'https://pgeo3.rio.rj.gov.br/arcgis/rest/services/Basicos/Muni
taquara_boroughs = requests.get(taquara_boroughs_url).json()

def boroughs_style(feature):
    return { 'color': 'blue', 'fill': False }
```

In [28]:
```
restaurant_latlons = [[res[2], res[3]] for res in restaurants.values()]
```

In [29]:
```
from folium import plugins
from folium.plugins import HeatMap

map_taquara= folium.Map(location=rj_center, zoom_start=13)
folium.TileLayer('cartodbpositron').add_to(map_taquara) #cartodbpositron cartodbdark_
HeatMap(restaurant_latlons).add_to(map_taquara)
folium.Marker(rj_center).add_to(map_taquara)
folium.Circle(rj_center, radius=1000, fill=False, color='white').add_to(map_taquara)
folium.Circle(rj_center, radius=2000, fill=False, color='white').add_to(map_taquara)
folium.Circle(rj_center, radius=3000, fill=False, color='white').add_to(map_taquara)
folium.GeoJson(taquara_boroughs, style_function=boroughs_style, name='geojson').add_t
```

Out[29]:



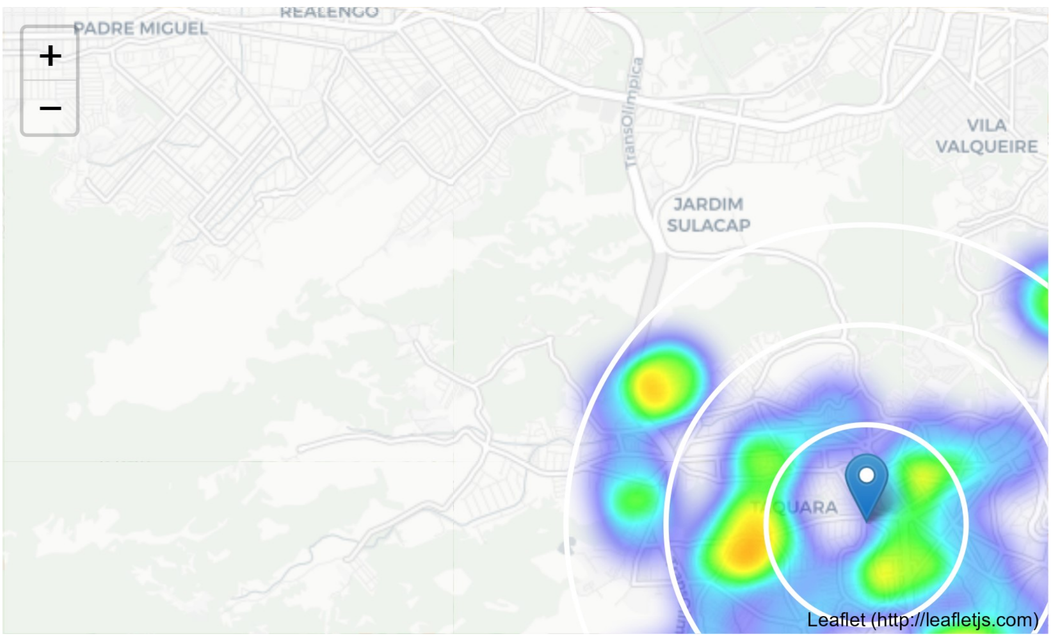
Looks like a few pockets of low restaurant density closest to city center can be found north, north-east and east from Art Shop.

Let's create another heatmap map showing heatmap/density of pizzaries only.

In [30]:
```
roi_x_min = rj_center_x - 1000
roi_y_max = rj_center_y + 500
roi_width = 2000
roi_height = 100
roi_center_x = roi_x_min + 1250
roi_center_y = roi_y_max - 1250
roi_center_lon, roi_center_lat = xy_to_lonlat(roi_center_x, roi_center_y)
roi_center = [roi_center_lat, roi_center_lon]
```

In [31]:
```
map_taquara= folium.Map(location=rj_center, zoom_start=13)
folium.TileLayer('cartodbpositron').add_to(map_taquara) #cartodbpositron cartodbdark_
HeatMap(pizza_latlons).add_to(map_taquara)
folium.Marker(rj_center).add_to(map_taquara)
folium.Circle(rj_center, radius=1000, fill=False, color='white').add_to(map_taquara)
folium.Circle(rj_center, radius=2000, fill=False, color='white').add_to(map_taquara)
folium.Circle(rj_center, radius=3000, fill=False, color='white').add_to(map_taquara)
folium.GeoJson(taquara_boroughs, style_function=boroughs_style, name='geojson').add_t
map_taquara
```

Out[31]:



This map is not so 'hot' (pizzaries represent a subset of ~87% of all restaurants in Taquara) but it also indicates higher density of existing pizzeries directly south-west, south-east and south from Art shop, with closest pockets of low pizzaries restaurant density positioned north and north-east from city center.

Based on this we will now focus our analysis on areas north and north-east from Taquara center - we will move the center of our area of interest and reduce it's size to have a radius of 2.5km. This places our location candidates mostly in boroughs Tanque and Pechincha (another potentially interesting borough is Noth Taquara with large low restaurant density north-east from city center, however this borough is less interesting to stakeholders as it's mostly residental and less popular with tourists).

Taquara south and Taquara North

Analysis of popular travel guides and web sites often mention Center Taquara as beautifull, interesting, and has more people density.

Tanque has more crime indice than Taquara and people that live in Taquara don´t like going in the Pechinche for make your lanch.

Let's define new, more narrow region of interest, which will include low-restaurant-count parts of Taquara south and Taquara north closest to Art Shop.

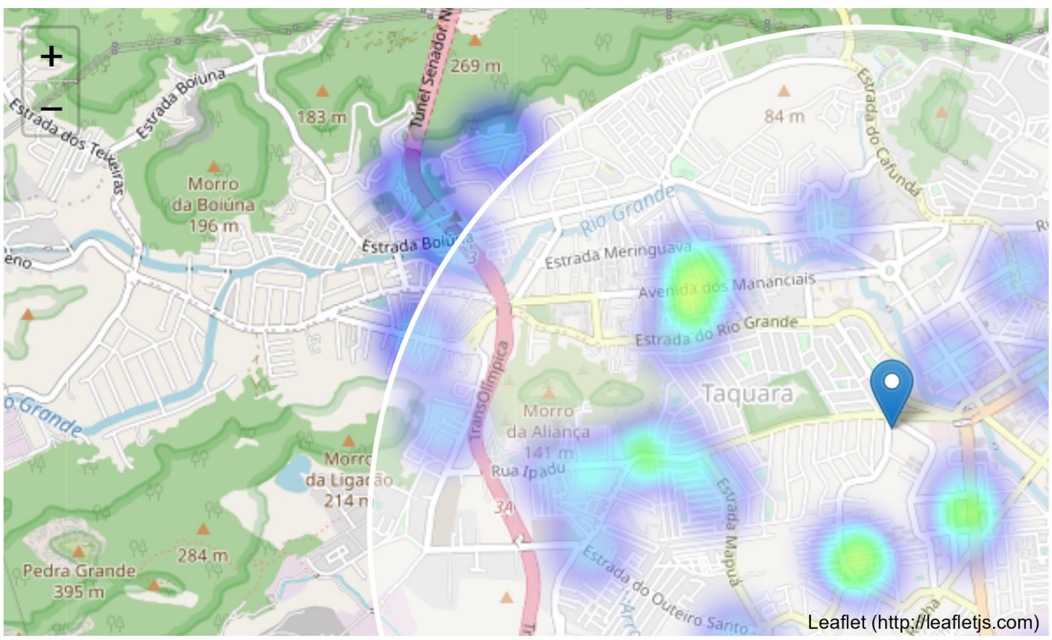
In [39]:

```
roi_x_min = rj_center_x - 2000
roi_y_max = rj_center_y + 2000
roi_width = 5000
roi_height = 5000
roi_center_x = roi_x_min + 2500
roi_center_y = roi_y_max - 2500
roi_center_lon, roi_center_lat = xy_to_lonlat(roi_center_x, roi_center_y)
roi_center = [roi_center_lat, roi_center_lon]
```

In [32]:

```
map_taquara = folium.Map(location=roi_center, zoom_start=14)
HeatMap(restaurant_latlons).add_to(map_taquara)
folium.Marker(rj_center).add_to(map_taquara)
folium.Circle(roi_center, radius=2500, color='white', fill=True, fill_opacity=0.4).ad
folium.GeoJson(taquara_boroughs, style_function=boroughs_style, name='geojson').add_t
map_taquara
```

Out[32]:




```
This nicely covers all the pockets of low pizzaries density in Taquara North and
Taquara south closest to Taquara center.

Let's also create new, more dense grid of location candidates restricted to our new
region of interest (let's make our location candidates 100m appart).
```

In [33]:
```
k = math.sqrt(3) / 2 # Vertical offset for hexagonal grid cells
x_step = 100
y_step = 100 * k
roi_y_min = roi_center_y - 1250

roi_latitudes = []
roi_longitudes = []
roi_xs = []
roi_ys = []
for i in range(0, int(51/k)):
    y = roi_y_min + i * y_step
    x_offset = 50 if i%2==0 else 0
    for j in range(0, 51):
        x = roi_x_min + j * x_step + x_offset
        d = calc_xy_distance(roi_center_x, roi_center_y, x, y)
        if (d <= 1250):
            lon, lat = xy_to_lonlat(x, y)
            roi_latitudes.append(lat)
            roi_longitudes.append(lon)
            roi_xs.append(x)
            roi_ys.append(y)

print(len(roi_latitudes), 'candidate neighborhood centers generated.')
```

```
567 candidate neighborhood centers generated.
```

OK. Now let's calculate two most important things for each location candidate: **number of restaurants in vicinity** (we'll use radius of **250 meters**) and **distance to closest pizzaries**.

In [34]:
```
def count_restaurants_nearby(x, y, restaurants, radius=250):
    count = 0
    for res in restaurants.values():
        res_x = res[7]; res_y = res[8]
        d = calc_xy_distance(x, y, res_x, res_y)
        if d<=radius:
            count += 1
    return count

def find_nearest_restaurant(x, y, restaurants):
    d_min = 100000
    for res in restaurants.values():
        res_x = res[7]; res_y = res[8]
        d = calc_xy_distance(x, y, res_x, res_y)
        if d<=d_min:
            d_min = d
    return d_min

roi_restaurant_counts = []
roi_pizzaria_distances = []

print('Generating data on location candidates... ', end='')
for x, y in zip(roi_xs, roi_ys):
    count = count_restaurants_nearby(x, y, restaurants, radius=250)
    roi_restaurant_counts.append(count)
    distance = find_nearest_restaurant(x, y, pizzaria_restaurants)
    roi_pizzaria_distances.append(distance)
print('done.')
```

```
Generating data on location candidates... done.
```

In [35]:
```
# Let's put this into dataframe
df_roi_locations = pd.DataFrame({'Latitude':roi_latitudes,
                                 'Longitude':roi_longitudes,
                                 'X':roi_xs,
                                 'Y':roi_ys,
                                 'Restaurants nearby':roi_restaurant_counts,
                                 'Distance to Pizzaria':roi_pizzaria_distances})

df_roi_locations.head(10)
```

Out[35]:

	Latitude	Longitude	X	Y	Restaurants nearby	Distance to Pizzaria
0	-22.933809	-43.382824	-6.241254e+06	-4.323215e+06	0	100000
1	-22.932046	-43.384833	-6.241704e+06	-4.323129e+06	0	100000
2	-22.932347	-43.384324	-6.241604e+06	-4.323129e+06	0	100000
3	-22.932648	-43.383815	-6.241504e+06	-4.323129e+06	0	100000
4	-22.932949	-43.383306	-6.241404e+06	-4.323129e+06	0	100000
5	-22.933251	-43.382797	-6.241304e+06	-4.323129e+06	0	100000
6	-22.933552	-43.382288	-6.241204e+06	-4.323129e+06	0	100000
7	-22.933853	-43.381779	-6.241104e+06	-4.323129e+06	0	100000
8	-22.934154	-43.381270	-6.241004e+06	-4.323129e+06	0	100000
9	-22.934455	-43.380761	-6.240904e+06	-4.323129e+06	0	100000

Let us now filter those locations: we're interested only in locations with no more than two restaurants in radius of 250 meters, and no pizzaries in radius of 400 meters.

In [36]:
```
good_res_count = np.array((df_roi_locations['Restaurants nearby']<=2))
print('Locations with no more than two restaurants nearby:', good_res_count.sum())

good_pizza_distance = np.array(df_roi_locations['Distance to Pizzaria']>=400)
print('Locations with no Pizzaria within 400m:', good_pizza_distance.sum())

good_locations = np.logical_and(good_res_count, good_pizza_distance)
print('Locations with both conditions met:', good_locations.sum())

df_good_locations = df_roi_locations[good_locations]
```

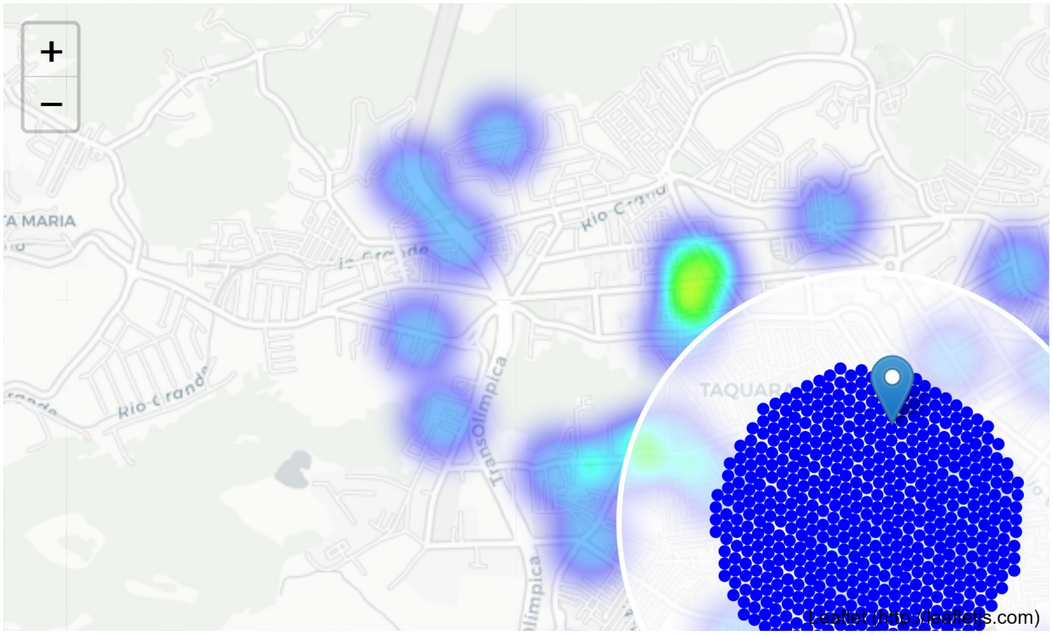
```
Locations with no more than two restaurants nearby: 567
Locations with no Pizzaria within 400m: 567
Locations with both conditions met: 567
```

Looks on a map.

In [37]:
```
good_latitudes = df_good_locations['Latitude'].values
good_longitudes = df_good_locations['Longitude'].values

good_locations = [[lat, lon] for lat, lon in zip(good_latitudes, good_longitudes)]
```

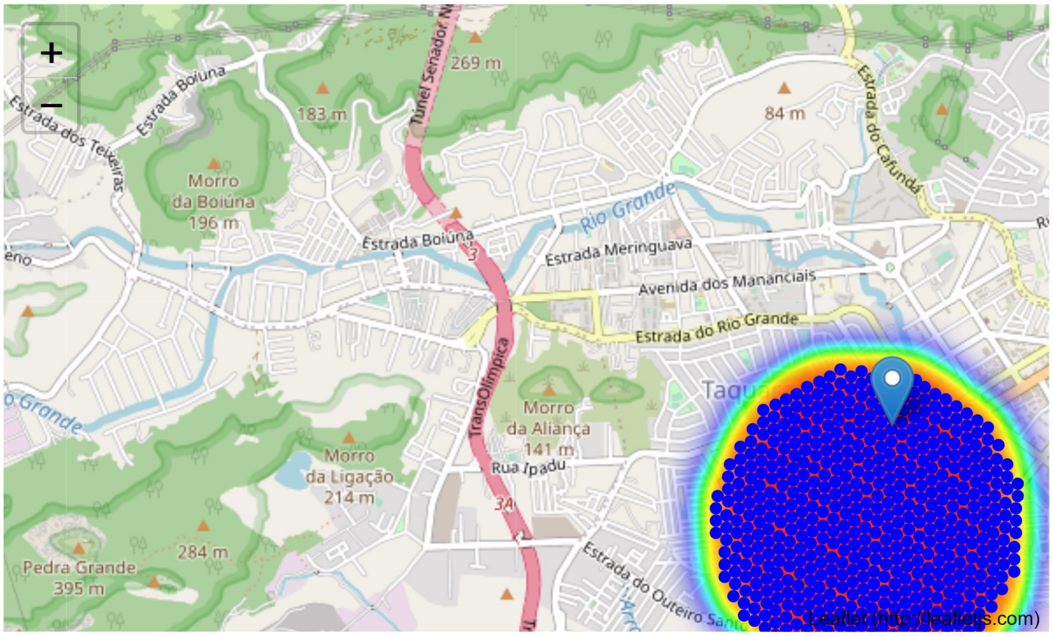
In [38]:
```
map_taquara = folium.Map(location=roi_center, zoom_start=14)
folium.TileLayer('cartodbpositron').add_to(map_taquara)
HeatMap(restaurant_latlons).add_to(map_taquara)
folium.Circle(roi_center, radius=1250, color='white', fill=True, fill_opacity=0.6).ad
folium.Marker(rj_center).add_to(map_taquara)
for lat, lon in zip(good_latitudes, good_longitudes):
    folium.CircleMarker([lat, lon], radius=2, color='blue', fill=True, fill_color='bl
folium.GeoJson(taquara_boroughs, style_function=boroughs_style, name='geojson').add_t
map_taquara
```

Out[38]:



In [39]:
```
map_taquara = folium.Map(location=roi_center, zoom_start=14)
HeatMap(good_locations, radius=25).add_to(map_taquara)
folium.Marker(rj_center).add_to(map_taquara)
for lat, lon in zip(good_latitudes, good_longitudes):
    folium.CircleMarker([lat, lon], radius=2, color='blue', fill=True, fill_color='bl
folium.GeoJson(taquara_boroughs, style_function=boroughs_style, name='geojson').add_t
map_taquara
```

Out[39]:

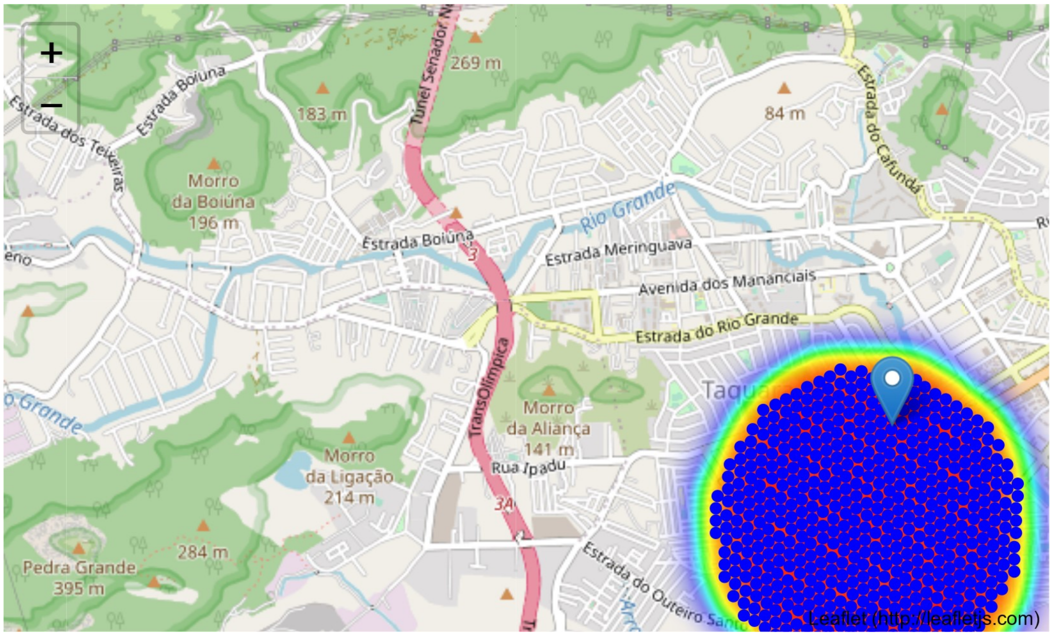


Looking good. We now have a bunch of locations fairly close to Art Shop (mostly in Taquara south and North), and we know that each of those locations has no more than two restaurants in radius of 250m, and no pizzaries closer than 400m. Any of those locations is a potential candidate for a new pizzaries, at least based on nearby competition.

Let's now show those good locations in a form of heatmap:

In [41]:
```
map_taquara = folium.Map(location=roi_center, zoom_start=14)
HeatMap(good_locations, radius=25).add_to(map_taquara)
folium.Marker(rj_center).add_to(map_taquara)
for lat, lon in zip(good_latitudes, good_longitudes):
    folium.CircleMarker([lat, lon], radius=2, color='blue', fill=True, fill_color='bl
folium.GeoJson(taquara_boroughs, style_function=boroughs_style, name='geojson').add_t
map_taquara
```

Out[41]:



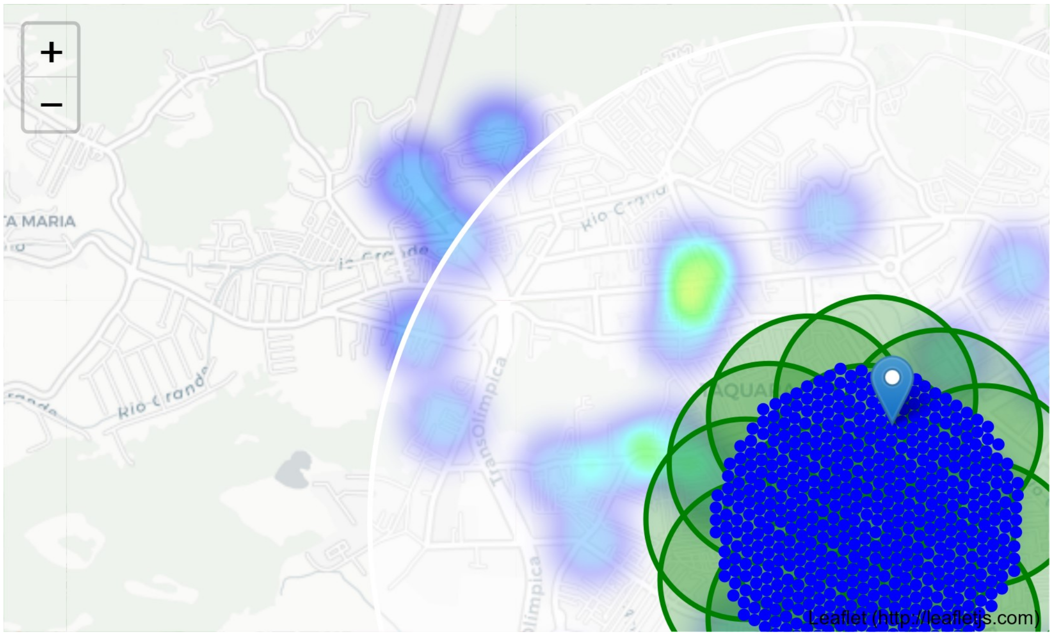
What we have now is a clear indication of zones with low number of restaurants in vicinity, and no pizzarean at all nearby.

Let us now cluster those locations to create centers of zones containing good locations. Those zones, their centers and addresses will be the final result of our analysis.

In [43]:
```
from sklearn.cluster import KMeans

number_of_clusters = 15

good_xys = df_good_locations[['X', 'Y']].values
kmeans = KMeans(n_clusters=number_of_clusters, random_state=0).fit(good_xys)

cluster_centers = [xy_to_lonlat(cc[0], cc[1]) for cc in kmeans.cluster_centers_]
```

In [44]:
```
map_taquara = folium.Map(location=roi_center, zoom_start=14)
folium.TileLayer('cartodbpositron').add_to(map_taquara)
HeatMap(restaurant_latlons).add_to(map_taquara)
folium.Circle(roi_center, radius=2500, color='white', fill=True, fill_opacity=0.4).ad
folium.Marker(rj_center).add_to(map_taquara)
for lon, lat in cluster_centers:
    folium.Circle([lat, lon], radius=500, color='green', fill=True, fill_opacity=0.25
for lat, lon in zip(good_latitudes, good_longitudes):
    folium.CircleMarker([lat, lon], radius=2, color='blue', fill=True, fill_color='bl
folium.GeoJson(taquara_boroughs, style_function=boroughs_style, name='geojson').add_t
map_taquara
```

Out[44]:



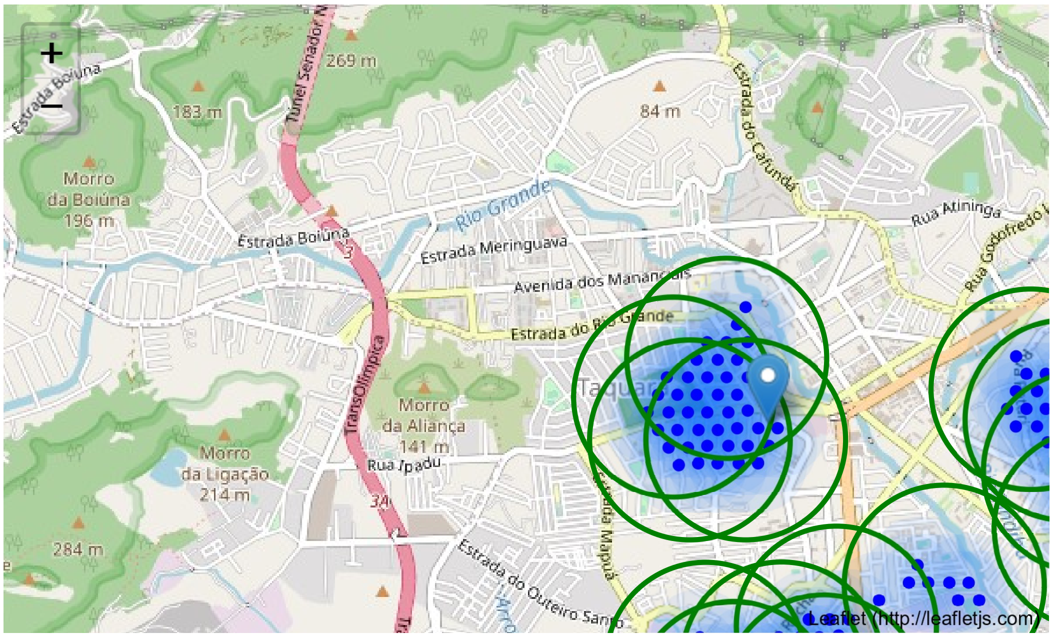
Our clusters represent groupings of most of the candidate locations and cluster centers are placed nicely in the middle of the zones 'rich' with location candidates.

Addresses of those cluster centers will be a good starting point for exploring the neighborhoods to find the best possible location based on neighborhood specifics.

Let's see those zones on a city map without heatmap, using shaded areas to indicate our clusters:

In [47]:
```
map_taquara = folium.Map(location=roi_center, zoom_start=14)
folium.Marker(rj_center).add_to(map_taquara)
for lat, lon in zip(good_latitudes, good_longitudes):
    folium.Circle([lat, lon], radius=250, color='#00000000', fill=True, fill_color='#
for lat, lon in zip(good_latitudes, good_longitudes):
    folium.CircleMarker([lat, lon], radius=2, color='blue', fill=True, fill_color='bl
for lon, lat in cluster_centers:
    folium.Circle([lat, lon], radius=500, color='green', fill=False).add_to(map_taqua
folium.GeoJson(taquara_boroughs, style_function=boroughs_style, name='geojson').add_t
map_taquara
```

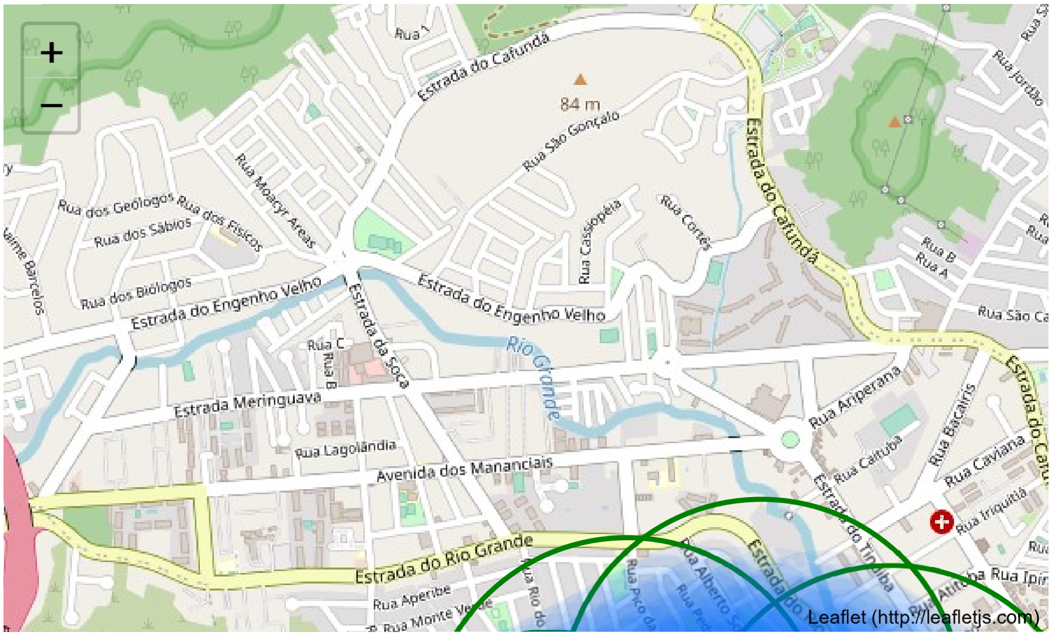
Out[47]:



In [45]:

```
map_taquara = folium.Map(location=[-22.918303, -43.375789], zoom_start=15)  # Sá ferr
folium.Marker(rj_center).add_to(map_taquara)
for lon, lat in cluster_centers:
    folium.Circle([lat, lon], radius=500, color='green', fill=False).add_to(map_taqua
for lat, lon in zip(good_latitudes, good_longitudes):
    folium.Circle([lat, lon], radius=250, color='#0000ff00', fill=True, fill_color='#
for lat, lon in zip(good_latitudes, good_longitudes):
    folium.CircleMarker([lat, lon], radius=2, color='blue', fill=True, fill_color='bl
folium.GeoJson(taquara_boroughs, style_function=boroughs_style, name='geojson').add_t
map_taquara
```

Out[45]:

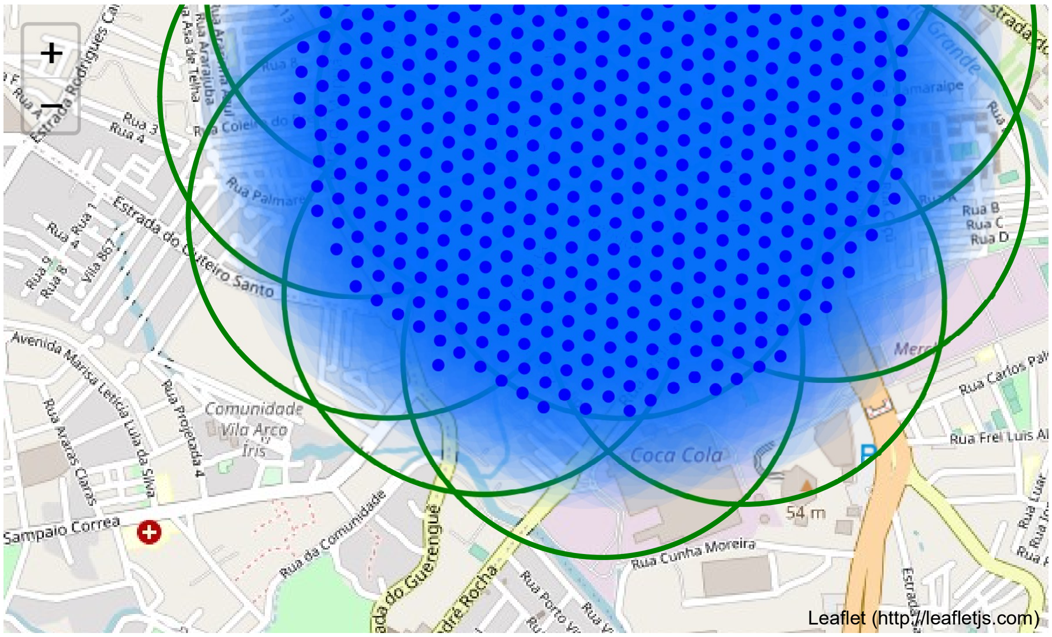


In [ ]:

```
Taquara South
```

In [46]:
```
map_taquara = folium.Map(location=[-22.937393, -43.372313], zoom_start=15)  # Sá ferr
folium.Marker(rj_center).add_to(map_taquara)
for lon, lat in cluster_centers:
    folium.Circle([lat, lon], radius=500, color='green', fill=False).add_to(map_taqua
for lat, lon in zip(good_latitudes, good_longitudes):
    folium.Circle([lat, lon], radius=250, color='#0000ff00', fill=True, fill_color='#
for lat, lon in zip(good_latitudes, good_longitudes):
    folium.CircleMarker([lat, lon], radius=2, color='blue', fill=True, fill_color='bl
folium.GeoJson(taquara_boroughs, style_function=boroughs_style, name='geojson').add_t
map_taquara
```

Out[46]:



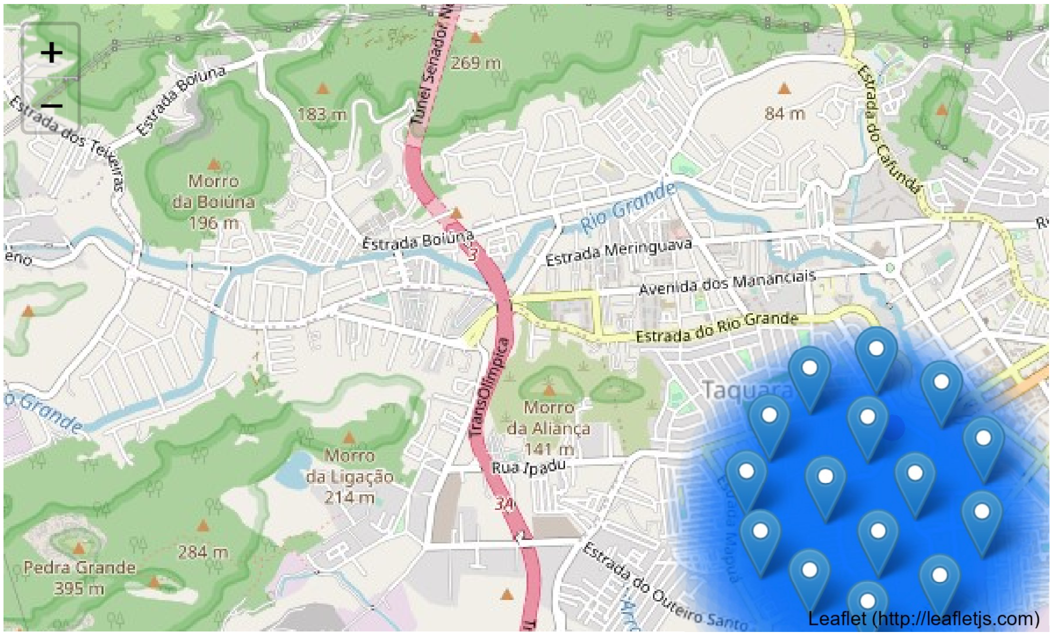
Finaly, let's **reverse geocode those candidate area centers to get the addresses** which can be presented to stakeholders.

In [47]:
```
candidate_area_addresses = []
print('==============================================================')
print('Addresses of centers of areas recommended for further analysis')
print('==============================================================\n')
for lon, lat in cluster_centers:
    addr = get_address(api_key, lat, lon).replace(', Brazil', '')
    candidate_area_addresses.append(addr)
    x, y = lonlat_to_xy(lon, lat)
    d = calc_xy_distance(x, y, rj_center_x, rj_center_y)
    print('{}{} => {:.1f}km from address'.format(addr, ' '*(50-len(addr)), d/1000))
```

```
==============================================================
Addresses of centers of areas recommended for further analysis
==============================================================

R. Pedra Branca, 60 - Taquara, Rio de Janeiro - RJ, 22715-330 => 0.3km from addres
s
Estrada do Guerenguê, 1272 - Taquara, Rio de Janeiro - RJ, 22713-004 => 1.7km from
address
R. Nacional, 539 - Taquara, Rio de Janeiro - RJ, 22710-093 => 0.9km from address
R. Correio do Rio, 658 - Taquara, Rio de Janeiro - RJ, 22715-010 => 1.4km from add
ress
R. Mapendi, 685 - Taquara, Rio de Janeiro - RJ, 22710-255 => 0.8km from address
Unnamed Road - Taquara, Rio de Janeiro - RJ, 22710-561 => 1.6km from address
Estr. Rodrigues Caldas, 751 - Taquara, Rio de Janeiro - RJ, 22713-372 => 0.7km fro
m address
Estr. Rodrigues Caldas, 127 - Taquara, Rio de Janeiro - RJ, 22713-372 => 0.4km fro
m address
R. Res. Macembu, 15 - Taquara, Rio de Janeiro - RJ, 22710-245 => 1.0km from addres
s
Estr. Mapuá, 791 - Taquara, Rio de Janeiro - RJ, 22713-320 => 1.6km from address
R. Triângulo Mineiro, 69 - Taquara, Rio de Janeiro - RJ, 22713-030 => 1.2km from a
ddress
R. M, 17 - Taquara, Rio de Janeiro - RJ, 22710-568 => 1.8km from address
Estr. dos Bandeirantes, 703 - Taquara, Rio de Janeiro - RJ, 22730-522 => 1.3km fro
m address
R. Aurora Fluminense, 204 - Taquara, Rio de Janeiro - RJ, 22715-180 => 0.4km from
address
R. Gazeta da Tarde, 189 - Taquara, Rio de Janeiro - RJ, 22715-100 => 1.0km from ad
dress
```

In [48]:
```
map_taquara = folium.Map(location=roi_center, zoom_start=14)
folium.Circle(rj_center, radius=50, color='red', fill=True, fill_color='red', fill_op
for lonlat, addr in zip(cluster_centers, candidate_area_addresses):
    folium.Marker([lonlat[1], lonlat[0]], popup=addr).add_to(map_taquara)
for lat, lon in zip(good_latitudes, good_longitudes):
    folium.Circle([lat, lon], radius=250, color='#0000ff00', fill=True, fill_color='#
map_taquara
```

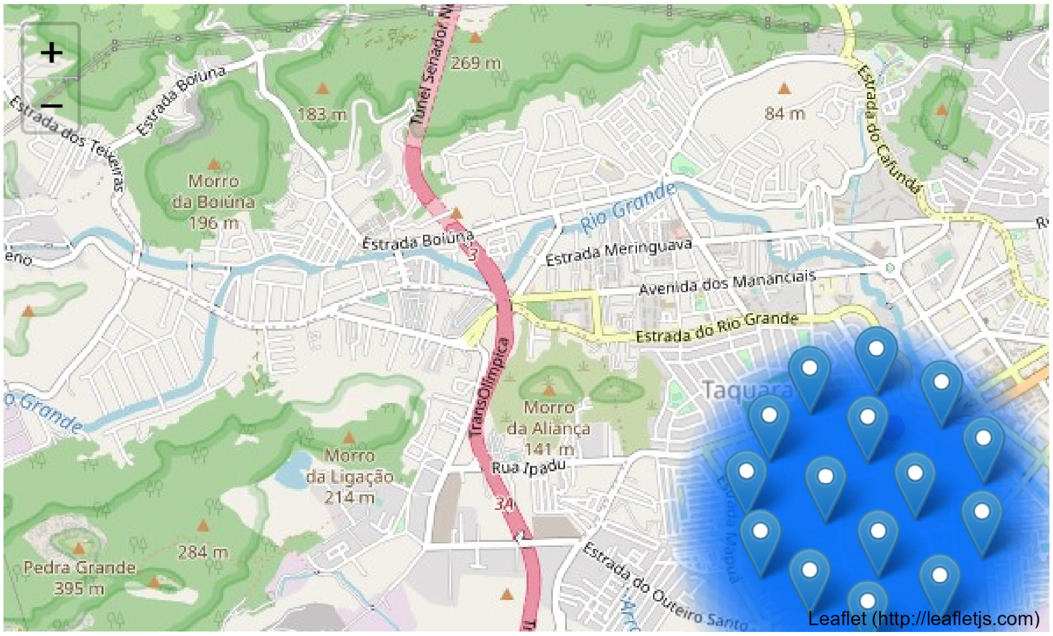
Out[48]:




```
This concludes our analysis. We have created 15 addresses representing centers of
zones containing locations with low number of restaurants and no pizzaries nearby, al
zones being fairly close to city center (all less than 4km from Art Shop, and about
half of those less than 2km from Art Shop). Although zones are shown on map with a
radius of ~500 meters (green circles), their shape is actually very irregular and
their centers/addresses should be considered only as a starting point for exploring
area neighborhoods in search for potential restaurant locations. Most of the zones ar
located in Taquara South and North, which we have identified as interesting due to
being popular more population, fairly close to city center and well connected by
```

In [49]:
```
map_taquara = folium.Map(location=roi_center, zoom_start=14)
folium.Circle(rj_center, radius=50, color='red', fill=True, fill_color='red', fill_op
for lonlat, addr in zip(cluster_centers, candidate_area_addresses):
    folium.Marker([lonlat[1], lonlat[0]], popup=addr).add_to(map_taquara)
for lat, lon in zip(good_latitudes, good_longitudes):
    folium.Circle([lat, lon], radius=250, color='#0000ff00', fill=True, fill_color='#
map_taquara
```

Out[49]:



Results and Discussion

Our analysis shows that although there is a great number of restaurants in Taquara (~2000 in our initial area of interest which was 12x12km around Art Shop), there are pockets of low restaurant density fairly close to city center. Highest concentration of restaurants was detected west and east from Art Shop, so we focused our attention to areas south and north, corresponding to Taquara south and Taquara north corner of central Taquara. Another borough was identified as potentially interesting (Tanque and Pechincha), but our attention was focused on TAquara which offer a combination of popularity, closeness to city center, strong socio-economic dynamics *and* a number of pockets of low restaurant density.

After directing our attention to this more narrow area of interest (covering approx. 5x5km south-north from Art Shop) we first created a dense grid of location candidates (spaced 100m appart); those locations were then filtered so that those with more than two restaurants in radius of 250m and those with an pizzaries closer than 400m were removed.

Those location candidates were then clustered to create zones of interest which contain greatest number of location candidates. Addresses of centers of those zones were also generated using reverse geocoding to be used as markers/starting points for more detailed local analysis based on other factors.

Result of all this is 15 zones containing largest number of potential new restaurant locations based on number of and distance to existing venues - both restaurants in general and pizzaries particularly. This, of course, does not imply that those zones are actually optimal locations for a new restaurant! Purpose of this analysis was to only provide info on areas close to Taquara center but not crowded with existing restaurants (particularly pizzaries) - it is entirely possible that there is a very good reason for small number of restaurants in any of those areas, reasons which would make them unsuitable for a new restaurant regardless of lack of competition in the area. Recommended zones should therefore be considered only as a starting point for more detailed analysis which could eventually result in location which has not only no nearby competition but also other factors taken into account and all other relevant conditions met.

Conclusion

Purpose of this project was to identify Taquara areas close to center with low number of restaurants (particularly pizzaries) in order to aid stakeholders in narrowing down the search for optimal location for a new pizzaries. By calculating restaurant density distribution from Foursquare data we have first identified general boroughs that justify further analysis (Taquara south and north), and then generated extensive collection of locations which satisfy some basic requirements regarding existing nearby restaurants. Clustering of those locations was then performed in order to create major zones of interest (containing greatest number of potential locations) and addresses of those zone centers were created to be used as starting points for final exploration by stakeholders.

Final decission on optimal restaurant location will be made by stakeholders based on specific characteristics of neighborhoods and locations in every recommended zone, taking into consideration additional factors like attractiveness of each location (proximity to park or water), levels of noise / proximity to major roads, real estate availability, prices, social and economic dynamics of every neighborhood etc.

```
In [ ]:
```