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 find optimal **coffee shops** in **Bratislava(Bratislavsky kraj) and Trnava (Trnavsky kraj) region in Slovakia**. Specifically, this report will be targeted to stakeholders - **specialty coffee roasters** - interested in finding potential partners for whom specialty coffee is/could be a unique selling point – a way to differentiate themselves from the masses of other coffee houses. Since there are lots of coffee shops in Slovakia, we will focus on coffee shops with high ranking or coffee shops which already sells specialty coffee.

Here are articles which explains specialty coffee in English and Slovak language:

- What is Specialty Coffee? https://sca.coffee/research/what-is-specialty-coffee

- Premiová, Specialty, Výberová – Čo to znamená? https://www.maliarik.sk/2019/09/27/premiova-specialty-vyberova-co-to-znamena/

As we can see specialty coffee roasters are determined to sell exceptional beans and roasts, refusing to compromise on quality in order to provide low-cost coffees. We need to use our data science knowledge to find not only top coffee shops, but also help with business trip planning by creating clusters of locations of potential partners. Business trips are needed for further negotiations with potential partners as in this market face-to-face communication is the must.

Data

The following factors will influence our decision:

- coffee shop rating by Foursquare

- coffee shop is considered as high quality coffee shop or coffee shop which already sells specialty coffee - based on list from web pages dedicated to this topic
- coffee shop is listed in one of webpages dedicated to high quality coffee shops or coffee shops already sell specialty coffee
- coffee shop location - must be within defined region

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

- Boundaries of territorial and administrative arrangement of the Slovak Republic - https://www.geoportal.sk/en/zbgis-smd/download-section/. Details can be found here https://www.geoportal.sk/en/zbgis/territorial-administrative-arrangement/. Municipality (slk: obec), District (slk: okres), Region (slk: kraj).
- Approximate addresses and centers of selected regions/districts will be obtained using **geocoding**
- Categories of Venues will be obtained using **Foursquare API**
- Venues of selected categories, their category and location in selected regions will be obtained using **Foursquare API**
- Rating of selected Venues will be obtained using **Foursquare API**
- List of high quality coffee shops or coffee shops already selling specialty coffee will be obtained using **web scraping** from these sources:
  - https://europeancoffeetrip.com/slovakia/
  - https://www.blackcheckguide.com/sk/kaviarne?nazov=&speciality=1
  - https://www.google.com/maps/d/u/0/viewer?mid=1tqMXO4_UfzKcZG0TEXTZ658pMKBcoig_&ll=48.737509009308354%2C19.91183545078126&z=10![image.png](attachment:96b27c9f-9e12-40fe-bb61-bcccd6283840.png)![image.png](attachment:08692074-17d3-47df-bb78-6148164875a5.png)![image.png](attachment:d9ac904a-a831-44b3-b258-3772255ca4b3.png)

Boundaries of territorial and administrative arrangement of the Slovak Republic

Let's download and unzip shapefiles from 'https://www.geoportal.sk/files/zbgis/na_stiahnutie/shp/ah_shp_3.zip'. This file is 3rd levels of generalization, which is recomended for maps with scale from 1:250 000 to 1:500 000. This is also smallest shapefile.

In [1]:

```
from io import BytesIO
from urllib.request import urlopen
from zipfile import ZipFile
zipurl = 'https://www.geoportal.sk/files/zbgis/na_stiahnutie/shp/ah_shp_3.zip'
with urlopen(zipurl) as zipresp:
    with ZipFile(BytesIO(zipresp.read())) as zfile:
        zfile.extractall('/zbgis')
```

Check that all is downladed and unzipped corectly

In [2]:

```
import glob
print(glob.glob("/zbgis/*"))
```

```
['/zbgis\\kraj_3.cpg', '/zbgis\\kraj_3.dbf', '/zbgis\\kraj_3.prj', '/zbgis\\kraj_3.sbn', '/zbgis\\kraj_3.sbx', '/zbgis\\kraj_3.shp', '/zbgis
\\kraj_3.shp.xml', '/zbgis\\kraj_3.shx', '/zbgis\\ku_3.cpg', '/zbgis\\ku_3.dbf', '/zbgis\\ku_3.prj', '/zbgis\\ku_3.sbn', '/zbgis\\ku_3.sbx',
'/zbgis\\ku_3.shp', '/zbgis\\ku_3.shp.xml', '/zbgis\\ku_3.shx', '/zbgis\\obec_3.cpg', '/zbgis\\obec_3.dbf', '/zbgis\\obec_3.prj', '/zbgis\\ob
ec_3.sbn', '/zbgis\\obec_3.sbx', '/zbgis\\obec_3.shp', '/zbgis\\obec_3.shp.xml', '/zbgis\\obec_3.shx', '/zbgis\\okres_3.cpg', '/zbgis\\okres_
3.dbf', '/zbgis\\okres_3.prj', '/zbgis\\okres_3.sbn', '/zbgis\\okres_3.sbx', '/zbgis\\okres_3.shp', '/zbgis\\okres_3.shp.xml', '/zbgis\\okres
_3.shx', '/zbgis\\sr_3.cpg', '/zbgis\\sr_3.dbf', '/zbgis\\sr_3.prj', '/zbgis\\sr_3.sbn', '/zbgis\\sr_3.sbx', '/zbgis\\sr_3.shp', '/zbgis\\sr_
3.shp.xml', '/zbgis\\sr_3.shx']
```

Now let's read **shape files to geopandas.**

- District (slk: okres) - okres_3.shp
- Region (slk: kraj) - kraj_3.shp

In [3]:

```python
import geopandas as gpd
shapefile_kraj = gpd.read_file('/zbgis\\kraj_3.shp',encoding='utf-8')
print(shapefile_kraj)
shapefile_okres = gpd.read_file('/zbgis\\okres_3.shp',encoding='utf-8')
print(shapefile_okres)
```

```
          DOW   FACC  IDN2              NM2        VYMERA      NUTS1  \
0  2021-02-26  FA002     1     Bratislavský  2.052618e+09  Slovensko
1  2021-02-26  FA002     2         Trnavský  4.146299e+09  Slovensko
2  2021-02-26  FA002     3      Trenčiansky  4.501807e+09  Slovensko
3  2021-02-26  FA002     4       Nitriansky  6.343731e+09  Slovensko
4  2021-02-26  FA002     5         Žilinský  6.808526e+09  Slovensko
5  2021-02-26  FA002     6  Banskobystrický  9.453987e+09  Slovensko
6  2021-02-26  FA002     7        Prešovský  8.972757e+09  Slovensko
7  2021-02-26  FA002     8          Košický  6.754322e+09  Slovensko

  NUTS1_CODE               NUTS2 NUTS2_CODE                 NUTS3 NUTS3_CODE  \
0        SK0   Bratislavský kraj       SK01     Bratislavský kraj      SK010
1        SK0   Západné Slovensko       SK02         Trnavský kraj      SK021
2        SK0   Západné Slovensko       SK02      Trenčiansky kraj      SK022
3        SK0   Západné Slovensko       SK02       Nitriansky kraj      SK023
4        SK0   Stredné Slovensko       SK03         Žilinský kraj      SK031
5        SK0   Stredné Slovensko       SK03  Banskobystrický kraj      SK032
6        SK0  Východné Slovensko       SK04        Prešovský kraj      SK041
7        SK0  Východné Slovensko       SK04          Košický kraj      SK042

      Shape_Leng    Shape_Area  \
0  292741.931518  2.051231e+09
1  555215.618184  4.145886e+09
2  460054.318819  4.501965e+09
3  544773.494028  6.341979e+09
4  552295.385282  6.805971e+09
5  618655.306703  9.451640e+09
6  768482.502636  8.970032e+09
```

```
7  653756.645170  6.750783e+09

                                          geometry
0  POLYGON ((-551128.940 -1226442.410, -551410.87...
1  POLYGON ((-558618.700 -1200307.840, -557530.09...
2  POLYGON ((-470197.700 -1159177.350, -469775.95...
3  POLYGON ((-501581.440 -1224109.550, -501086.41...
4  POLYGON ((-388708.420 -1132697.420, -388267.22...
5  POLYGON ((-386801.680 -1207727.440, -384887.38...
6  POLYGON ((-257535.460 -1157385.430, -255453.95...
7  POLYGON ((-314980.480 -1203550.400, -314866.33...
           DOW   FACC  IDN3               NM3  IDN2           NM2  \
0   2021-02-26  FA003   101      Bratislava I     1  Bratislavský
1   2021-02-26  FA003   102     Bratislava II     1  Bratislavský
2   2021-02-26  FA003   103    Bratislava III     1  Bratislavský
3   2021-02-26  FA003   104     Bratislava IV     1  Bratislavský
4   2021-02-26  FA003   105      Bratislava V     1  Bratislavský
..         ...    ...   ...               ...   ...           ...
74  2021-02-26  FA003   807        Michalovce     8       Košický
75  2021-02-26  FA003   808           Rožňava     8       Košický
76  2021-02-26  FA003   809          Sobrance     8       Košický
77  2021-02-26  FA003   810  Spišská Nová Ves     8       Košický
78  2021-02-26  FA003   811          Trebišov     8       Košický

          VYMERA      NUTS1 NUTS1_CODE               NUTS2 NUTS2_CODE  \
0   9.590124e+06  Slovensko        SK0   Bratislavský kraj       SK01
1   9.249007e+07  Slovensko        SK0   Bratislavský kraj       SK01
2   7.467488e+07  Slovensko        SK0   Bratislavský kraj       SK01
3   9.666503e+07  Slovensko        SK0   Bratislavský kraj       SK01
4   9.420708e+07  Slovensko        SK0   Bratislavský kraj       SK01
..           ...        ...        ...                ...        ...
74  1.019236e+09  Slovensko        SK0  Východné Slovensko       SK04
75  1.173348e+09  Slovensko        SK0  Východné Slovensko       SK04
76  5.381599e+08  Slovensko        SK0  Východné Slovensko       SK04
77  5.874591e+08  Slovensko        SK0  Východné Slovensko       SK04
78  1.073475e+09  Slovensko        SK0  Východné Slovensko       SK04

                NUTS3 NUTS3_CODE              LAU1 LAU1_CODE     Shape_Leng  \
0   Bratislavský kraj      SK010      Bratislava I    SK0101   15530.304966
1   Bratislavský kraj      SK010     Bratislava II    SK0102   51785.183572
2   Bratislavský kraj      SK010    Bratislava III    SK0103   37952.539692
3   Bratislavský kraj      SK010     Bratislava IV    SK0104   54711.669045
4   Bratislavský kraj      SK010      Bratislava V    SK0105   57951.270082
..                ...        ...               ...       ...            ...
74       Košický kraj      SK042        Michalovce    SK0427  226065.347604
75       Košický kraj      SK042           Rožňava    SK0428  210420.526385
76       Košický kraj      SK042          Sobrance    SK0429  123123.808874
77       Košický kraj      SK042  Spišská Nová Ves    SK042A  195583.684472
78       Košický kraj      SK042          Trebišov    SK042B  252613.287699
```

```
      Shape_Area                                           geometry
0   9.515509e+06  POLYGON ((-575145.790 -1278340.870, -574102.68...
1   9.236431e+07  POLYGON ((-564493.360 -1279174.560, -564669.22...
2   7.455699e+07  POLYGON ((-570000.500 -1270060.390, -569868.70...
3   9.708500e+07  POLYGON ((-583175.970 -1266380.590, -582462.86...
4   9.451473e+07  POLYGON ((-577488.960 -1280696.340, -576304.95...
..           ...                                                ...
74  1.021117e+09  POLYGON ((-220620.170 -1221506.660, -220299.72...
75  1.173146e+09  POLYGON ((-327299.360 -1214116.650, -326381.65...
76  5.375512e+08  POLYGON ((-191740.540 -1219163.220, -191349.47...
77  5.870782e+08  POLYGON ((-314980.480 -1203550.400, -314866.33...
78  1.072653e+09  POLYGON ((-241349.730 -1232456.770, -241005.52...

[79 rows x 18 columns]
```

Now we will **select all districts from Bratislava(Bratislavsky kraj) and Trnava (Trnavsky kraj) region.** In shapefile_kraj we can see that **NUTS3_CODE** for **Bratislava is SK010** and for **Trnava is SK021**

In [4]:

```
#NUTS2_CODE SK01 and SK02#
cafe_gpd = shapefile_okres[shapefile_okres['NUTS3_CODE'] == 'SK010']
cafe_gpd = cafe_gpd.append(shapefile_okres[shapefile_okres['NUTS3_CODE'] == 'SK021'])
cafe_gpd
```

Out[4]:

	DOW	FACC	IDN3	NM3	IDN2	NM2	VYMERA	NUTS1	NUTS1_CODE	NUTS2	NUTS2_CODE	NUTS3	NUTS3_CODE	LAU1	LAU1_CO
0	2021-02-26	FA003	101	Bratislava I	1	Bratislavský	9.590124e+06	Slovensko	SK0	Bratislavský kraj	SK01	Bratislavský kraj	SK010	Bratislava I	SK0
1	2021-02-26	FA003	102	Bratislava II	1	Bratislavský	9.249007e+07	Slovensko	SK0	Bratislavský kraj	SK01	Bratislavský kraj	SK010	Bratislava II	SK0
2	2021-02-26	FA003	103	Bratislava III	1	Bratislavský	7.467488e+07	Slovensko	SK0	Bratislavský kraj	SK01	Bratislavský kraj	SK010	Bratislava III	SK0
3	2021-02-26	FA003	104	Bratislava IV	1	Bratislavský	9.666503e+07	Slovensko	SK0	Bratislavský kraj	SK01	Bratislavský kraj	SK010	Bratislava IV	SK0

	DOW	FACC	IDN3	NM3	IDN2	NM2	VYMERA	NUTS1	NUTS1_CODE	NUTS2	NUTS2_CODE	NUTS3	NUTS3_CODE	LAU1	LAU1_CO
4	2021-02-26	FA003	105	Bratislava V	1	Bratislavský	9.420708e+07	Slovensko	SK0	Bratislavský kraj	SK01	Bratislavský kraj	SK010	Bratislava V	SK0
5	2021-02-26	FA003	106	Malacky	1	Bratislavský	9.495646e+08	Slovensko	SK0	Bratislavský kraj	SK01	Bratislavský kraj	SK010	Malacky	SK0
6	2021-02-26	FA003	107	Pezinok	1	Bratislavský	3.755380e+08	Slovensko	SK0	Bratislavský kraj	SK01	Bratislavský kraj	SK010	Pezinok	SK0
7	2021-02-26	FA003	108	Senec	1	Bratislavský	3.598880e+08	Slovensko	SK0	Bratislavský kraj	SK01	Bratislavský kraj	SK010	Senec	SK0
8	2021-02-26	FA003	201	Dunajská Streda	2	Trnavský	1.074589e+09	Slovensko	SK0	Západné Slovensko	SK02	Trnavský kraj	SK021	Dunajská Streda	SK0
9	2021-02-26	FA003	202	Galanta	2	Trnavský	6.417120e+08	Slovensko	SK0	Západné Slovensko	SK02	Trnavský kraj	SK021	Galanta	SK0
10	2021-02-26	FA003	203	Hlohovec	2	Trnavský	2.672267e+08	Slovensko	SK0	Západné Slovensko	SK02	Trnavský kraj	SK021	Hlohovec	SK0
11	2021-02-26	FA003	204	Piešťany	2	Trnavský	3.811156e+08	Slovensko	SK0	Západné Slovensko	SK02	Trnavský kraj	SK021	Piešťany	SK0
12	2021-02-26	FA003	205	Senica	2	Trnavský	6.832569e+08	Slovensko	SK0	Západné Slovensko	SK02	Trnavský kraj	SK021	Senica	SK0

	DOW	FACC	IDN3	NM3	IDN2	NM2	VYMERA	NUTS1	NUTS1_CODE	NUTS2	NUTS2_CODE	NUTS3	NUTS3_CODE	LAU1	LAU1_CO
13	2021-02-26	FA003	206	Skalica	2	Trnavský	3.570820e+08	Slovensko	SK0	Západné Slovensko	SK02	Trnavský kraj	SK021	Skalica	SK0
14	2021-02-26	FA003	207	Trnava	2	Trnavský	7.413163e+08	Slovensko	SK0	Západné Slovensko	SK02	Trnavský kraj	SK021	Trnava	SK0

Geographical center of these **two regions** is close to vilage **Bahon**. We will use **geocoding to find latitude & longitude**.

In [5]:

```
from geopy.geocoders import Nominatim # convert an address into latitude and longitude values

#find coordinnates
address = 'Bahon, Slovakia'
geolocator = Nominatim(user_agent="region_explorer")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print('The geograpical coordinate are {}, {}.'.format(latitude, longitude))
location
```

```
The geograpical coordinate are 48.3087562, 17.4451279.
```

Out[5]:

```
Location(Báhoň, okres Pezinok, Bratislavský kraj, Slovensko, (48.3087562, 17.4451279, 0.0))
```

Now lets create **map of selected regions and districts**

In [71]:

```
import folium
from folium import ColorMap, LinearColormap, StepColormap
# create map of BK and TK using latitude and longitude values
f = folium.Figure(width=800, height=400)
m = folium.Map(location=[latitude, longitude], zoom_start=8)
choro = folium.Choropleth(cafe_gpd, data=cafe_gpd,
                  name='Regional boundaries',
                  key_on='feature.properties.NM3',
                  threshold_scale=[0,100, 200,300],
                  columns=['NM3', 'IDN3'],
                  #fill_color='YlOrBr'
                  #fill_color='YlGn'
                  #fill_color='PuBu'
```

```
                    fill_color='YlGnBu'
                    #fill_color='YlOrBr', # ‘BuGn’, ‘BuPu’, ‘GnBu’, ‘OrRd’, ‘PuBu’, ‘PuBuGn’, ‘PuRd’, ‘RdPu’, ‘YlGn’, ‘YlGnBu’, ‘YlOrBr’, and ‘
                    )
# remove legend from map
for key in choro._children:
    if key.startswith('color_map'):
        del(choro._children[key])

choro.add_to(m)
folium.LayerControl().add_to(m)
f.add_child(m)
f
```

Out[71]:

Approximate addresses and centers of selected regions/districts will be obtained using **geocoding**

we will create **cafe_regions** dataframe with **address and latitude, longitude for each region**

In [7]:
```
cafe_regions = cafe_gpd['NM3']
cafe_regions
```

Out[7]:
```
0        Bratislava I
1       Bratislava II
2      Bratislava III
3       Bratislava IV
4        Bratislava V
5             Malacky
6             Pezinok
7               Senec
8     Dunajská Streda
9             Galanta
10           Hlohovec
11           Piešťany
12             Senica
13            Skalica
14             Trnava
Name: NM3, dtype: object
```

Let's **find address, latitude, longitude** for each **cafe region**

In [8]:
```
import pandas as pd
locs = [ geolocator.geocode(cr+', Slovensko') for cr in cafe_regions ]
cafe_region_geo_info = pd.DataFrame(
    #[ (addr.address, addr.latitude, addr.longitude) for addr in locs[1:-1] ],
    [ (addr.address, addr.latitude, addr.longitude) for addr in locs ],
    columns=['address', 'latitude', 'longitude'])
cafe_region_geo_info
```

Out[8]:

	address	latitude	longitude
0	okres Bratislava I, Bratislava, Bratislavský k...	48.155137	17.101021
1	okres Bratislava II, Bratislava, Bratislavský ...	48.122740	17.210335
2	okres Bratislava III, Bratislava, Bratislavský...	48.200147	17.148075
3	okres Bratislava IV, Bratislava, Bratislavský ...	48.202306	17.017936
4	okres Bratislava V, Bratislava, Bratislavský k...	48.074586	17.115642
5	Malacky, okres Malacky, Bratislavský kraj, Slo...	48.434750	17.020348
6	Pezinok, okres Pezinok, Bratislavský kraj, Slo...	48.285454	17.270194
7	Senec, okres Senec, Bratislavský kraj, Slovensko	48.219947	17.396990

	address	latitude	longitude
8	Dunajská Streda, okres Dunajská Streda, Trnavs...	47.989375	17.620259
9	Galanta, okres Galanta, Trnavský kraj, Západné...	48.191050	17.727063
10	Hlohovec, okres Hlohovec, Trnavský kraj, Západ...	48.427884	17.798782
11	Piešťany, okres Piešťany, Trnavský kraj, Západ...	48.589525	17.821385
12	Senica, okres Senica, Trnavský kraj, Západné S...	48.678756	17.366153
13	Skalica, okres Skalica, Trnavský kraj, Západné...	48.845208	17.227913
14	Trnava, okres Trnava, Trnavský kraj, Západné S...	48.376765	17.585818

Let's **finalize cafe_regions** data frame

In [9]:

```
cafe_regions = pd.DataFrame(cafe_regions).join(cafe_region_geo_info)
cafe_regions['region_name'] = cafe_regions['NM3'].map(lambda x: str('Okres ' + x + ', Slovensko'))
cafe_regions
```

Out[9]:

	NM3	address	latitude	longitude	region_name
0	Bratislava I	okres Bratislava I, Bratislava, Bratislavský k...	48.155137	17.101021	Okres Bratislava I, Slovensko
1	Bratislava II	okres Bratislava II, Bratislava, Bratislavský ...	48.122740	17.210335	Okres Bratislava II, Slovensko
2	Bratislava III	okres Bratislava III, Bratislava, Bratislavský...	48.200147	17.148075	Okres Bratislava III, Slovensko
3	Bratislava IV	okres Bratislava IV, Bratislava, Bratislavský ...	48.202306	17.017936	Okres Bratislava IV, Slovensko
4	Bratislava V	okres Bratislava V, Bratislava, Bratislavský k...	48.074586	17.115642	Okres Bratislava V, Slovensko
5	Malacky	Malacky, okres Malacky, Bratislavský kraj, Slo...	48.434750	17.020348	Okres Malacky, Slovensko
6	Pezinok	Pezinok, okres Pezinok, Bratislavský kraj, Slo...	48.285454	17.270194	Okres Pezinok, Slovensko
7	Senec	Senec, okres Senec, Bratislavský kraj, Slovensko	48.219947	17.396990	Okres Senec, Slovensko
8	Dunajská Streda	Dunajská Streda, okres Dunajská Streda, Trnavs...	47.989375	17.620259	Okres Dunajská Streda, Slovensko
9	Galanta	Galanta, okres Galanta, Trnavský kraj, Západné...	48.191050	17.727063	Okres Galanta, Slovensko
10	Hlohovec	Hlohovec, okres Hlohovec, Trnavský kraj, Západ...	48.427884	17.798782	Okres Hlohovec, Slovensko
11	Piešťany	Piešťany, okres Piešťany, Trnavský kraj, Západ...	48.589525	17.821385	Okres Piešťany, Slovensko
12	Senica	Senica, okres Senica, Trnavský kraj, Západné S...	48.678756	17.366153	Okres Senica, Slovensko

	NM3	address	latitude	longitude	region_name
13	Skalica	Skalica, okres Skalica, Trnavský kraj, Západné...	48.845208	17.227913	Okres Skalica, Slovensko
14	Trnava	Trnava, okres Trnava, Trnavský kraj, Západné S...	48.376765	17.585818	Okres Trnava, Slovensko

Now we can put all info on **map**

In [72]:

```
f = folium.Figure(width=800, height=400)
m = folium.Map(location=[latitude, longitude], zoom_start=8)

choro = folium.Choropleth(cafe_gpd, data=cafe_gpd,
                  name='Regional boundaries',
                  key_on='feature.properties.NM3',
                  threshold_scale=[0,100, 200,300],
                  columns=['NM3', 'IDN3'],
                  fill_color='YlGnBu'
                  )
# remove legend from map
for key in choro._children:
    if key.startswith('color_map'):
        del(choro._children[key])

choro.add_to(m)


fg_districts = folium.FeatureGroup(name='Districts centers')
for lat, lon, name , address in zip(cafe_regions['latitude'], cafe_regions['longitude'], cafe_regions['NM3'], cafe_regions['address']):
    pp= folium.Html('<h3>' + str(name) + '</h3>' + '<p>' + str(address) + '</p>', script=True)
    label = folium.Popup(pp, max_width=200)
    folium.CircleMarker(
        [lat, lon],
        radius=2,
        popup=label,
        color='darkred',
        fill=True,
        fill_color='darkred',
        fill_opacity=0.7).add_to(fg_districts)

fg_districts.add_to(m)
folium.LayerControl().add_to(m)
f.add_child(m)

f
```

Out[72]:

Categories of Venues will be obtained using **Foursquare API**

Define Foursquare Credentials and Version

In [70]:

```
CLIENT_ID = 'your-client-ID' # your Foursquare ID
CLIENT_SECRET = 'your-client-secret' # your Foursquare Secret

CLIENT_ID = 'FSQID' # your Foursquare ID
CLIENT_SECRET = 'FSQ SEC' # your Foursquare Secret

VERSION = '20180605' # Foursquare API version
LIMIT = 100 # A default Foursquare API limit value

print('Your credentails:')
print('CLIENT_ID: ' + CLIENT_ID)
print('CLIENT_SECRET:' + CLIENT_SECRET)
```

```
Your credentails:
```

```
CLIENT_ID: FSQID
CLIENT_SECRET:FSQ SEC
```

Venue Categories

We will use Venue Categories API to get list of categeries - we want to find categories of venues with search string "caf" in category name, pluralname or shortname.

Description from https://developer.foursquare.com/docs/api-reference/venues/categories/ :

Returns a hierarchical list of categories applied to venues. This list is also available on our categories page. Request - GET https://api.foursquare.com/v2/venues/categories

In [12]:

```
import requests
VERSION = '20180605' # Foursquare API version
url = 'https://api.foursquare.com/v2/venues/categories?client_id={}&client_secret={}&v={}'.format(CLIENT_ID, CLIENT_SECRET, VERSION)
results = requests.get(url).json()
```

Response is hierarchical list with maximum depth = 5. We will put this list to dataframe category_data.

In [13]:

```
#pd.json_normalize(results['response'])

#works_data =
category_data = pd.DataFrame()
category_data = category_data.append(pd.json_normalize(results['response'],errors='ignore', record_path=['categories'])[['id','name','pluralN
category_data = category_data.append(pd.json_normalize(results['response'],errors='ignore', record_path=['categories', 'categories'])[['id','
category_data = category_data.append(pd.json_normalize(results['response'],errors='ignore', record_path=['categories', 'categories', 'categor
category_data = category_data.append(pd.json_normalize(results['response'],errors='ignore', record_path=['categories', 'categories', 'categor
category_data = category_data.append(pd.json_normalize(results['response'],errors='ignore', record_path=['categories', 'categories', 'categor
category_data.shape
```

Out[13]: (970, 4)

Let's find all categories which contains string "caf" in columns name, pluralName or shortName. As we can see in some categories is used word Cafe in others Café.

In [14]:

```
search_string ='caf'
caf_ids = category_data[category_data["name"].str.contains(search_string, case=False)][['id']]
caf_ids.append(category_data[category_data["pluralName"].str.contains(search_string, case=False)][['id']])
caf_ids.append(category_data[category_data["shortName"].str.contains(search_string, case=False)][['id']])
```

```
print(caf_ids)
category_data[category_data["id"].isin(caf_ids.astype(str).values.flatten().tolist())]
```

```
                          id
35   5f2c14a5b6d05514c7042eb7
42   4bf58dd8d48988d1a1941735
90   4bf58dd8d48988d128941735
91   4bf58dd8d48988d16d941735
139  56aa371be4b08b9a8d573508
344  4bf58dd8d48988d18d941735
357  4bf58dd8d48988d1f0941735
129  54135bf5e4b08f3d2429dfe7
297  54f4ba06498e2cf5561da814
```

Out[14]:

	id	name	pluralName	shortName
35	5f2c14a5b6d05514c7042eb7	VR Cafe	VR Cafe	VR Cafe
42	4bf58dd8d48988d1a1941735	College Cafeteria	College Cafeterias	Cafeteria
90	4bf58dd8d48988d128941735	Cafeteria	Cafeterias	Cafeteria
91	4bf58dd8d48988d16d941735	Café	Cafés	Café
139	56aa371be4b08b9a8d573508	Pet Café	Pet Cafés	Pet Café
344	4bf58dd8d48988d18d941735	Gaming Cafe	Gaming Cafes	Gaming Cafe
357	4bf58dd8d48988d1f0941735	Internet Cafe	Internet Cafes	Internet Cafe
129	54135bf5e4b08f3d2429dfe7	Irani Cafe	Irani Cafes	Irani
297	54f4ba06498e2cf5561da814	Corporate Cafeteria	Corporate Cafeterias	Corporate Cafeteria

Lets create string with ids of selected categories that we will use for searching using **Foursquare API**

In [15]:

```
caf_ids_string = ','.join(caf_ids.astype(str).values.flatten().tolist())
caf_ids_string
```

Out[15]:

```
'5f2c14a5b6d05514c7042eb7,4bf58dd8d48988d1a1941735,4bf58dd8d48988d128941735,4bf58dd8d48988d16d941735,56aa371be4b08b9a8d573508,4bf58dd8d48988d
18d941735,4bf58dd8d48988d1f0941735,54135bf5e4b08f3d2429dfe7,54f4ba06498e2cf5561da814'
```

Venues of selected catagories, their type and location in selected regions will be obtained using **Foursquare API**

Define function that extracts the category of the venue and find candidates

In [16]:

```
# function that extracts the category of the venue
def get_category_type(row):
    try:
        categories_list = row['categories']
    except:
        categories_list = row['venue.categories']

    if len(categories_list) == 0:
        return None
    else:
        return categories_list[0]['name']
```

We will prepare venue candidates max 50 venues from every region

We will find candidates using FSQ explore API using region name for near parameter and list of selected categories from previous step for category id parameter . We will us also radius 20 km for radius and limit returned result for maximum 50 venues.

We are using radius to select venues in region so we have to check if all venues in result are from region. We will drop all venues outside of region

We will store venue candidates in pickle. We can also load venue candidates from pickle - to speedup process , in case of need to rerun this part again.

In [17]:

```
radius = 20000
LIMIT = 50
VERSION = '20180605' # Foursquare API version
venue_candidates = pd.DataFrame()

import pickle
cofee_rating = []
loaded = False
save_new_version = False

# find rating for venue_candidates with rating = 0
find_candidates = False
#find_candidates = True
if find_candidates == False:
    try:
        with open('venue_candidates.pkl', 'rb') as f:
            venue_candidates = pickle.load(f)
        venue_candidates = venue_candidates[~venue_candidates.index.duplicated(keep='first')]
        print('venue_candidates data loaded.')
        print(len(venue_candidates.index))
        loaded = True
    except:
```

```
        pass

# If load failed use the Foursquare API to get the data
if not loaded:
    for index, row in cafe_regions.iterrows():
        near_region = row['region_name']
        NM3 = row['NM3']
        # explore  url
        url = 'https://api.foursquare.com/v2/venues/explore?client_id={}&client_secret={}&near={}&categoryId={}&v={}&radius={}&limit={}&sortB

        results = requests.get(url).json()


        venues = results['response']['groups'][0]['items']
        nearby_venues = pd.json_normalize(venues) # flatten JSON

        # filter the category for each row
        nearby_venues['venue.categories'] = nearby_venues.apply(get_category_type, axis=1)
        nearby_venues_det = nearby_venues

        # detailed info about venue
        filtered_columns = ['venue.id','venue.name', 'venue.categories', 'venue.location.lat', 'venue.location.lng',
                            'venue.location.postalCode','venue.location.cc', 'venue.location.city', 'venue.location.state',
                            'venue.location.country', 'venue.location.formattedAddress']
        new_row  = nearby_venues.loc[:, filtered_columns]
        new_row['NM3'] = NM3
        new_row['search_name'] = new_row['venue.name'].str.normalize('NFKD').str.encode('ascii', errors='ignore').str.decode('utf-8').str.str

        # we need to check if all venues are from region
        poly = cafe_gpd[cafe_gpd['NM3'] == NM3]
        points = gpd.GeoDataFrame(new_row, geometry= gpd.points_from_xy(new_row["venue.location.lng"], new_row["venue.location.lat"])).set_cr
        poly = poly.to_crs(epsg =4326)
        #print(gdf.shape)
        #    venue_candidates = venue_candidates.append(new_row)
        venue_candidates = venue_candidates.append(gpd.sjoin(poly, points, op='contains'))

    # new empty columnn with rating
    venue_candidates['FSQ_rating'] = 0
    save_new_version = True

venue_candidates.head(10)

if save_new_version:
    print('new venue_candidates size')
    print(len(venue_candidates.index))
    # Let's persists this in local file system
```

```
    with open('venue_candidates.pkl', 'wb') as f:
        pickle.dump(venue_candidates, f)
```

```
venue_candidates data loaded.
342
```

Rating of selected Venues will be obtained using **Foursquare API**

Define function that extracts the rating of the venue_candidates

We can run max 50 premium queries per day.

In [18]:

```
def get_cofee_rating(venues):
    rating_list = []
    VERSION = '20180605' # Foursquare API version

    VENUE_ID = '78fbfc6d03f26080c39'
    url = 'https://api.foursquare.com/v2/venues/{}&lient_id={}&client_secret={}&v={}'.format(VENUE_ID,CLIENT_ID, CLIENT_SECRET, VERSION)
    # https://api.foursquare.com/v2/venues/VENUE_ID

    print('Obtaining venues rating:', end='')
    max_queries = 50
    end_index = int(min(max_queries,venues.size))

    for VENUE_ID in venues[0:end_index+1]['venue.id']:
        url = 'https://api.foursquare.com/v2/venues/{}?client_id={}&client_secret={}&v={}'.format(VENUE_ID,CLIENT_ID, CLIENT_SECRET, VERSION)
        results = requests.get(url).json()         #['response']['venue']['rating']
        if results['meta']['code'] == 200:
            venue_id = VENUE_ID
            try:
                FSQ_rating = int(results['response']['venue']['rating'])
                rating = (venue_id,FSQ_rating)
                rating_list.append(rating)
            except:
                print('error')
                #print(results)
                # if venue doesn't have rating - there will be -1 to not try find it again
                rating = (venue_id,-1)
                rating_list.append(rating)
        else:
            print('error')
            print(results)

    print(' .', end='')
```

```
    print(' done.')
    return pd.DataFrame(rating_list, columns = ['venue.id','FSQ_rating'])
```

In this part we are reusinge information about rating that we already have stored in pickle from previous runs

Imported rating is merged with venue candidates. If there are some venue candidates without rating (rating = 0) we are trying to find rating for them using function get_cofee_rating. If FSQ dont have rating for venue, function get_cofee_rating returns -1 for rating. This way we can mark that venue don't have rating and we are not trying to find rating for it in repeated run. All found rating from pickle and from function get_cofee_rating is again stored in pickle for later reuse.

In [19]:

```
import pickle
cofee_rating = []
loaded = False

# find rating for venue_candidates with rating = 0
find_rating = False
find_rating = True

try:
    with open('cofee_rating.pkl', 'rb') as f:
        cofee_rating = pickle.load(f)
    cofee_rating = cofee_rating[~cofee_rating.index.duplicated(keep='first')]
    print('cofee_rating data loaded.')
    print(len(cofee_rating.index))
    loaded = True
except:
    pass

# If load failed use the Foursquare API to get the data
if not loaded:
    venue_rating = pd.DataFrame (venue_candidates[venue_candidates['FSQ_rating'] == 0][['venue.id','FSQ_rating']])
    cofee_rating = get_cofee_rating(venue_rating)

#lets map rating to venue_candidates using ['venue.id']
cofee_rating = cofee_rating.set_index('venue.id')
venue_candidates = venue_candidates.set_index('venue.id')
venue_candidates = venue_candidates[~venue_candidates.index.duplicated(keep='first')]
cofee_rating = cofee_rating[~cofee_rating.index.duplicated(keep='first')]
venue_candidates.update(cofee_rating)
venue_candidates.reset_index(inplace=True)
cofee_rating.reset_index(inplace=True)

#if there is missing rating (rating = 0) - try to find it
venue_rating = pd.DataFrame (venue_candidates[venue_candidates['FSQ_rating'] == 0][['venue.id','FSQ_rating']])
if venue_rating.size > 0 and find_rating:
```

```
    print("VR")
    #print(venue_rating)
    cofee_rating = get_cofee_rating(venue_rating)
    cofee_rating = cofee_rating.set_index('venue.id')
    venue_candidates = venue_candidates.set_index('venue.id')
    venue_candidates.update(cofee_rating)
    venue_candidates.reset_index(inplace=True)
    cofee_rating.reset_index(inplace=True)

# lets select id and rating from venue_candidates and store it
cofee_rating_n = pd.DataFrame(venue_candidates[venue_candidates['FSQ_rating']!= 0][[ 'venue.id','FSQ_rating']])
if cofee_rating_n.size > 0 :
    print('new rating size')
    print(len(cofee_rating_n.index))
    # Let's persists this in local file system
    with open('cofee_rating.pkl', 'wb') as f:
        pickle.dump(cofee_rating_n, f)
```

```
cofee_rating data loaded.
323
VR
Obtaining venues rating:error
error
error
error
error
error
error
error
error
error
 . done.
new rating size
342
```

In [20]:

```
## check if we have venue_candidates with rating != 0
venue_candidates[venue_candidates['FSQ_rating']!= 0][[ 'venue.name','FSQ_rating']]
```

Out[20]:

	venue.name	FSQ_rating
0	.klub pod lampou	8.0
1	Kafe Scherz	7.0
2	Mačkafé klub	8.0
3	Kafe Nervosa	6.0

	venue.name	FSQ_rating
4	Tepláreň Café	6.0
...	...	...
337	Baterkaren	-1.0
338	Paleta Cafe & Wine Bar	6.0
339	Bufet UCM v JAME	-1.0
340	Caffe Pascucci	5.0
341	Daisy Café	-1.0

342 rows × 2 columns

List of high quality coffee shops or coffee shops already selling specialty coffee will be obtained using **web scraping** from these sources:

- https://www.google.com/maps/d/u/0/viewer?mid=1tqMXO4_UfzKcZG0TEXTZ658pMKBcoig_&ll=48.737509009308354%2C19.91183545078126&z=10![image.png](attachment:96b27c9f-9e12-40fe-bb61-bcccd6283840.png)![image.png](attachment:08692074-17d3-47df-bb78-6148164875a5.png)![image.png](attachment:d9ac904a-a831-44b3-b258-3772255ca4b3.png)
- https://www.blackcheckguide.com/sk/kaviarne?nazov=&speciality=1
- https://europeancoffeetrip.com/slovakia/

google my maps scraper

We will scrape cafe names from https://www.google.com/maps/d/u/0/viewer?mid=1tqMXO4_UfzKcZG0TEXTZ658pMKBcoig_&ll=48.737509009308354%2C19.91183545078126&z=10 This source has two types of coffe shop - specialty and comodity. Beacuse this resource lists coffee shops open during Covid restrictions, we will mark this resource with columns cnames_covid_sp and cnames_covid_co.

In [21]:

```
from selenium import webdriver
from selenium.webdriver.chrome.options import Options
from bs4 import BeautifulSoup
import time
import io
import pandas as pd
```

```python
options = Options()
options.add_argument('--headless')
CHROMEDRIVER_PATH = "d://chromredriver//chromedriver.exe"
browser = webdriver.Chrome(executable_path=CHROMEDRIVER_PATH, options=options)

url = "https://www.google.com/maps/d/u/0/viewer?mid=1tqMXO4_UfzKcZG0TEXTZ658pMKBcoig_&ll=48.73750900930835 4%2C19.91183545078126&z=10"
browser.get(url)
time.sleep(3)

html_source = browser.page_source.encode('utf-8')
soup = BeautifulSoup(html_source,"html.parser")


mydivs = soup.findAll("div", {"class": "suEOdc"})
counter = 0
l = []
for a in mydivs:
    l.append(a.text)
    counter = counter + 1

print("Total cafes scraped:"+str(counter))
print(l)


# first row is header, specialty cafes are only in first 63 rows
cnames_covid_sp = pd.DataFrame (l,columns=['C_Name'])[1:64]
cnames_covid_co =  pd.DataFrame (l,columns=['C_Name'])[64:84]
print("-------------------------------------")
print("Selected specialty cafes :")
print(cnames_covid_sp)
print("Selected comodity cafes :")
print(cnames_covid_co)
```

```
Total cafes scraped:88
['Kaviarne počas II. vlny COVID-19', 'Antikvariát Kaviareň', 'Aurelica coffee', 'Barista Kaviareň', 'Barovňa Čarovňa', 'Bezkydov', 'black.',
'Bomba barista - coffee laboratory & shop', 'BON BON Nitra', 'BOTTOVA • Coffee • Brunch • Gineria', 'BØLGE.', 'Brixton House', 'Cafe Bar Cent
ral', 'Cafe Casa Mia', 'CAFFETTINO', 'Central Café Levice', 'Ciao Papa', 'COBURG coffee & burger', 'Coffeeshopa', 'Coffia Cafe (specialty cof
fee)', "Daddy's café", 'Depo Café Telgárt', 'Dobrá Nálada Kaviareň', 'Eg Cafe', 'En Bloc Café’', 'Fefe cafe', 'FELKA café & brew bar', 'Foxfo
rd - Cubicon Bratislava', 'Foxford - Martin', 'Foxford - Obchodná ulica Bratislava', 'Foxford OC Galéria Košice', 'Foxford - OC Forum Popra
d', 'Habesh Coffee Shop', 'HISTORY CAFFE & BAKERY', 'Chocolateria BON BON', 'Karma Coffee', 'Kaviareň Pacas', 'KONTAKT - Mládežnícke centrum
a kaviareň', 'Moja Malá Budapešť', 'Moment', 'NICO CAFFÉ Prešov', 'NICO CAFFÉ Košice', 'Oto&Oto Fitness', 'Pán Králiček Priestor', 'Pražiareň
kávy Ollivier coffee', 'PROFILE COFFEE', 'Reštaurácia Trio', 'Rodinná kaviareň JUST LOVELY', 'Stará škola', 'Street Cafe', 'Sweet Beans Coffe
e', 'Sweet Beans Coffee Roastery', 'SWEET SPOT café', 'Triplefive coffee roasters - cafe', 'The Fleck coffee roasters', 'Valéria coffee & te
a', 'Verticcio coffee & tea | Bratislava | Špecializovaná predajňa káva, čaj, čokoláda', 'Verticcio coffee & tea | Trnava | Špecializovaná pr
edajňa káva, čaj, čokoláda', 'Verticcio coffee & tea | Žilina | Špecializovaná predajňa káva, čaj, čokoláda', 'Výberofka', 'U Baristu', 'Un c
affé', 'Urban Bistro', 'ZOY Coffee & Chocolate', 'Abstract cafe', 'Elis caffe', 'Hotel Panorama', 'Kaviaren pod brezičkami', 'Kúpeľná kaviarn
ička', 'Lodenica Caffe', 'Meadow - FLOWERS & COFFEE', 'PIZZA TAXI', 'Spusta Avion', 'Spusta cafe ,,Vazka,,', 'Spusta Freshmarket', 'Spusta Ka
```

```
viareň & Pekáreň', 'Spusta kaviareň a pekáreň Rača', 'Spusta OD Slimák', 'Streč Caffe', "Yogi's Avion", "Yogi's Bory Mall", 'Yogi´s Eperia',
"Yogi's Eurovea", 'Yogi ´s Mirage', 'Kaviareň - Slovensko v srdci', 'Kaviareň - Slovensko v srdci', 'ME GUSTA café - bar', 'Montmartre Café G
allery']
-------------------------------------
Selected specialty cafes :
                    C_Name
1     Antikvariát Kaviareň
2          Aurelica coffee
3         Barista Kaviareň
4          Barovňa Čarovňa
5                 Bezkydov
..                     ...
59               Výberofka
60               U Baristu
61                Un caffé
62            Urban Bistro
63  ZOY Coffee & Chocolate

[63 rows x 1 columns]
Selected comodity cafes :
                            C_Name
64                   Abstract cafe
65                      Elis caffe
66                  Hotel Panorama
67         Kaviaren pod brezičkami
68             Kúpeľná kaviarnička
69                  Lodenica Caffe
70       Meadow - FLOWERS & COFFEE
71                      PIZZA TAXI
72                    Spusta Avion
73           Spusta cafe ,,Vazka,,
74              Spusta Freshmarket
75       Spusta Kaviareň & Pekáreň
76  Spusta kaviareň a pekáreň Rača
77                Spusta OD Slimák
78                     Streč Caffe
79                    Yogi's Avion
80                Yogi's Bory Mall
81                   Yogi´s Eperia
82                  Yogi's Eurovea
83                  Yogi ´s Mirage
```

www.blackcheckguide.com scraper

We will scrape cafe names from https://www.blackcheckguide.com/sk/kaviarne?nazov=&speciality=1

In [22]:

```
url = 'https://www.blackcheckguide.com/sk/kaviarne?nazov=&speciality=1'
```

```
browser.get(url)
time.sleep(3)

html_source = browser.page_source.encode('utf-8')
soup = BeautifulSoup(html_source,"html.parser")
#print(soup.prettify())
mydivs = soup.findAll("div", {"class": "inner"})

counter = 0
l = []
#scrape_data ={'Reviewer Name':[],'Reviewer Rating':[],'Reviewer Profile URL':[],'Review':[],'Time':[]}
for a in mydivs:
    l.append(a.find("h2").text.rstrip().lstrip())
    counter = counter + 1
print("Total cafes scraped:"+str(counter))
#pd.DataFrame(l, columns="Name")
cnames_bc = pd.DataFrame (l,columns=['C_Name'])
cnames_bc
```

```
Total cafes scraped:387
```

Out[22]:

	C_Name
0	Rosetta / coffee shop / kaviareň
1	Kolt
2	Kaviareň Pacas
3	Moment Liptov
4	.BLACK - Coffee & Wine
...	...
382	Porta Café
383	Matsu premium tea & coffee
384	Metropola Café & Bistro
385	7edmička - coffee place
386	The Caffe

387 rows × 1 columns

europeancoffeetrip.com scraper

We will scrape cafe names from https://europeancoffeetrip.com/slovakia/

In [23]:

```
url = 'https://europeancoffeetrip.com/slovakia/'

browser.get(url)
time.sleep(3)

html_source = browser.page_source.encode('utf-8')
soup = BeautifulSoup(html_source,"html.parser")
#print(soup.prettify())

mydivs = soup.findAll("a")
#image src defines if entry is roaster or cofee shop
cafe_src = 'https://mk0europeancoffmnbn2.kinstacdn.com/wp-content/themes/15zine-child/city-guides/images/cup_white.svg'

counter = 0
l = []

for a in mydivs:
    u = a.find("div", {"class": "icon-cafe-roaster"})
    if (u != None):
        v = a.find('img')['src']
        if (v == cafe_src):
            t = a.find("h3").text
            l.append(t.rstrip().lstrip())
            counter = counter + 1

print("Total cafes scraped:"+str(counter))

cnames_ect = pd.DataFrame (l,columns=['C_Name'])
cnames_ect
```

```
Total cafes scraped:67
```

Out[23]:

	C_Name
0	Záhir Cafe
1	Výberofka
2	Urban House
3	Urban Bistro
4	True Brew Bar
...	...

	C_Name
62	Stará Škola
63	MONO café
64	Espresso Bar
65	Coffeein Specialty Coffee Shop
66	Minimal Coffeeshop

67 rows × 1 columns

End of data colection

Now we have all required data - coffe shops from selected region and also coofee shops names from web pages with special focus on cofee shops.

This concludes the data gathering phase - we're now ready to use this data for analysis to produce the report on top coffee shops in region.

Methodology

In this project we will find top **coffee shops** in **Bratislava(Bratislavsky kraj) and Trnava (Trnavsky kraj) region in Slovakia**.

In first step we have collected the required **data:

- Boundaries of territorial and administrative arrangement of the Slovak Republic, Approximate addresses and centers of selected regions/districts
- Categories of Venues
- Venues of selected categories
- Rating of selected Venues
- List of high quality coffee shops or coffee shops already selling specialty coffee

Second step in our analysis will be calculation of rating for all selected coffee shops. This rating will be based on weighted rating of data sources where was coffee shop located. After identifying coffee shops with highest rating, we will prepare clusters (using **DBSCAN clustering**), which will help with business trip planning.

In third and final step we will focus on preparation of final report - on map and in a table. Both output will contain detailed information about coffee shop to help stakeholders with final decisions.

Analysis

Let's perform some basic explanatory data analysis and derive some additional info from our raw data.

We will realize this steps:

- we need to remove duplicates from sources (webscraped)
- we need to put all info to one data_frame with attributes representing source of coffe shop name
- we need to create new column without accents, special characters and spaces - all lowercase - this column will be used for merging data sources by coffee shop name
- we need to merge web scarped data sources with venue_candidates from FSQ
- we need to compare if all webscraped names are in venue candidates, if not we will try to find this venues by name in FSQ
- we need to calculate total rating for merged venue_candidates
- we need to select top 100 merged venue_candidates
- we need to prepare clusters using DB clustering
- we need to prepare final report map and table

Lets remove duplicity from data sources

In [24]:

```
#remove duplicity
cnames_covid_sp = cnames_covid_sp.groupby(['C_Name']).sum()
cnames_covid_co = cnames_covid_co.groupby(['C_Name']).sum()
cnames_ect = cnames_ect.groupby(['C_Name']).sum()
cnames_bc = cnames_bc.groupby(['C_Name']).sum()
```

Now we can put all scraped cafe names to one dataframe with source atribute

In [25]:

```
# all cafe names to one dataframe
cnames = cnames_covid_sp.assign(covid_sp = 1,covid_co = 0, bc = 0, ect = 0)
cnames = cnames_covid_co.assign(covid_sp = 0,covid_co = 1, bc = 0, ect = 0)
cnames = cnames.append(cnames_ect.assign(covid_sp = 0,covid_co = 0, bc = 0, ect = 1))
cnames = cnames.append(cnames_bc.assign(covid_sp = 0,covid_co = 0, bc = 1, ect = 0))
cnames = cnames.groupby(['C_Name']).sum().reset_index()
cnames
```

Out[25]:

	C_Name	covid_sp	covid_co	bc	ect
0	.BLACK - Coffee & Wine	0	0	1	0

	C_Name	covid_sp	covid_co	bc	ect
1	7edmička - coffee place	0	0	1	0
2	ARCH CAFFE	0	0	1	0
3	About coffee	0	0	1	0
4	Abstract cafe	0	1	0	0
...	...	...	...	...	...
251	Čukoláda	0	0	1	0
252	Čáry-Máry (Martin)	0	0	1	0
253	Čáry-Máry (Žilina)	0	0	1	0
254	Špacírka Bistro	0	0	1	0
255	Žufaňa	0	0	1	0

256 rows × 5 columns

Now we will create new column '**search_name**' - name without accents, special characters and spaces, all lowercase

Column will be used for merging with FSQ venue_candidates

In [26]:

```
cnames['search_name'] = cnames['C_Name'].str.normalize('NFKD').str.encode('ascii', errors='ignore').str.decode('utf-8').str.strip('').str.low
#cnames =cnames.groupby(['search_name']).sum()
cnames.columns
```

Out[26]:

```
Index(['C_Name', 'covid_sp', 'covid_co', 'bc', 'ect', 'search_name'], dtype='object')
```

In [27]:

```
#remove duplicates
cnames  = cnames.groupby('search_name').agg({'C_Name': '-'.join,
                                        'covid_sp': ['sum'],
                                        'covid_co': ['sum'],
                                        'bc': ['sum'],
                                        'ect': ['sum']
                                    }).reset_index()
```

```
cnames.columns =('search_name', 'C_Name', 'covid_sp', 'covid_co', 'bc', 'ect')
cnames
```

Out[27]:

	search_name	C_Name	covid_sp	covid_co	bc	ect
0	aboutcoffee	About coffee	0	0	1	0
1	abstractcafe	Abstract cafe	0	1	0	0
2	adelcafe	Adel cafe	0	0	1	0
3	amnesiatrnava	Amnesia Trnava	0	0	1	0
4	antikvariatkaviarensenec	Antikvariát-kaviareň Senec	0	0	1	0
...	...	...	...	...	...	...
234	yogismirage	Yogi ´s Mirage	0	1	0	0
235	zahircafe	Záhir Cafe	0	0	0	1
236	zahircoffeedrinks	Záhir Coffee & Drinks	0	0	1	0
237	zuckmannvilla	Zuckmann VILLA	0	0	1	0
238	zufana	Žufaňa	0	0	1	0

239 rows × 6 columns

In [ ]:

In [28]:

```
print(venue_candidates.shape)
print(cnames.shape)
```

```
(342, 33)
(239, 6)
```

merge web scarped data sources with venue_candidates fromFSQ

prepare temporary dataframe to check what can be merged and what web scraped cafes are not in venue_candidates

In [29]:

```
# prepare temporary df to check what can be merged and what web scraped cafes are not in venue_candidates
```

```
print(venue_candidates.shape)
print(cnames.shape)

df=pd.merge(venue_candidates,cnames,on=['search_name','search_name'],how="outer",indicator=True)
print(df.shape)
df
```

```
(342, 33)
(239, 6)
(561, 39)
```

Out[29]:

	venue.id	DOW	FACC	IDN3	NM3_left	IDN2	NM2	VYMERA	NUTS1	NUTS1_CODE	...	venue.location.formattedAddress	NM3_righ
0	5a1d478fbfc6d03f26080c39	2021-02-26	FA003	101.0	Bratislava I	1.0	Bratislavský	9590124.0	Slovensko	SK0	...	[Partizánska 2, 811 03 Bratislava, Slovensko]	Bratislava
1	4bca1c7868f976b0feab5e83	2021-02-26	FA003	101.0	Bratislava I	1.0	Bratislavský	9590124.0	Slovensko	SK0	...	[Partizánska 2 (Palisády), 811 03 Bratislava, ...	Bratislava
2	55689447498e0fb52a2c1b6d	2021-02-26	FA003	101.0	Bratislava I	1.0	Bratislavský	9590124.0	Slovensko	SK0	...	[Zámocká 7327/24, 811 01 Bratislava, Slovensko]	Bratislava
3	4c40c026af052d7f734b7c79	2021-02-26	FA003	101.0	Bratislava I	1.0	Bratislavský	9590124.0	Slovensko	SK0	...	[Zámocká 30, 811 01 Bratislava, Slovensko]	Bratislava
4	55830c83498e2ca875974b43	2021-02-26	FA003	101.0	Bratislava I	1.0	Bratislavský	9590124.0	Slovensko	SK0	...	[Zámocká 30, Bratislava, Slovensko]	Bratislava
...	...	...	...	...	...	...	...	...	...	...	...	...	..
556	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN
557	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN
558	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN
559	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN
560	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN

561 rows × 39 columns

In [30]:
```
# cafes only from FSQ datasource
df[df['_merge']=='left_only']
```

Out[30]:

	venue.id	DOW	FACC	IDN3	NM3_left	IDN2	NM2	VYMERA	NUTS1	NUTS1_CODE	...	venue.location.formattedAddress	NM3_ri
0	5a1d478fbfc6d03f26080c39	2021-02-26	FA003	101.0	Bratislava I	1.0	Bratislavský	9590124.0	Slovensko	SK0	...	[Partizánska 2, 811 03 Bratislava, Slovensko]	Bratisla
1	4bca1c7868f976b0feab5e83	2021-02-26	FA003	101.0	Bratislava I	1.0	Bratislavský	9590124.0	Slovensko	SK0	...	[Partizánska 2 (Palisády), 811 03 Bratislava, ...	Bratisla
2	55689447498e0fb52a2c1b6d	2021-02-26	FA003	101.0	Bratislava I	1.0	Bratislavský	9590124.0	Slovensko	SK0	...	[Zámocká 7327/24, 811 01 Bratislava, Slovensko]	Bratisla
3	4c40c026af052d7f734b7c79	2021-02-26	FA003	101.0	Bratislava I	1.0	Bratislavský	9590124.0	Slovensko	SK0	...	[Zámocká 30, 811 01 Bratislava, Slovensko]	Bratisla
4	55830c83498e2ca875974b43	2021-02-26	FA003	101.0	Bratislava I	1.0	Bratislavský	9590124.0	Slovensko	SK0	...	[Zámocká 30, Bratislava, Slovensko]	Bratisla
...	...	...	...	...	...	...	...	...	...	...	...	...	
337	5e62552dbadf1d0008c8c204	2021-02-26	FA003	207.0	Trnava	2.0	Trnavský	741316337.0	Slovensko	SK0	...	[Starohájska 1, 917 01 Trnava, Slovensko]	Trn
338	555614f2498e41d3e8a0e966	2021-02-26	FA003	207.0	Trnava	2.0	Trnavský	741316337.0	Slovensko	SK0	...	[PALETA cafe & win bar (Starohájska 9), 917 0...	Trn
339	534cfda1498e1bdd444491d0	2021-02-26	FA003	207.0	Trnava	2.0	Trnavský	741316337.0	Slovensko	SK0	...	[V Jame 3, Trnava, Slovensko]	Trn
340	4c680ef19cedd13a287f79a1	2021-02-26	FA003	207.0	Trnava	2.0	Trnavský	741316337.0	Slovensko	SK0	...	[Veterná 40/A (Arkadia Shopping Park), 917 01 ...	Trn
341	4de73f5d7d8b1f2dd2938c40	2021-02-26	FA003	207.0	Trnava	2.0	Trnavský	741316337.0	Slovensko	SK0	...	[Veterná, 917 01 Trnava, Slovensko]	Trn

320 rows × 39 columns

In [31]:
```
# cafes only from webscrape data source
df[df['_merge']=='right_only']
```

Out[31]:

	venue.id	DOW	FACC	IDN3	NM3_left	IDN2	NM2	VYMERA	NUTS1	NUTS1_CODE	...	venue.location.formattedAddress	NM3_right	search_name	F

	venue.id	DOW	FACC	IDN3	NM3_left	IDN2	NM2	VYMERA	NUTS1	NUTS1_CODE	...	venue.location.formattedAddress	NM3_right	search_name	F
342	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	aboutcoffee	
343	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	abstractcafe	
344	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	adelcafe	
345	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	amnesiatrnava	
346	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	antikvariatkaviarensenec	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
556	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	yogiseurovea	
557	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	yogismirage	
558	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	zahircafe	
559	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	zahircoffeedrinks	
560	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	zufana	

219 rows × 39 columns

In [32]:

```
# cafes that are mergeable
df[df['_merge']=='both']
```

Out[32]:

	venue.id	DOW	FACC	IDN3	NM3_left	IDN2	NM2	VYMERA	NUTS1	NUTS1_CODE	...	venue.location.formattedAddress	NM3_ri
7	5249b447498e7cc671c96118	2021-02-26	FA003	101.0	Bratislava I	1.0	Bratislavský	9590124.0	Slovensko	SK0	...	[Skalná 1 (Zámocká), 811 03 Bratislava, Sloven...	Bratisla
19	55c11bdb498ec2a4cbe19a84	2021-02-26	FA003	101.0	Bratislava I	1.0	Bratislavský	9590124.0	Slovensko	SK0	...	[Michalská 5, 811 01 Bratislava, Slovensko]	Bratisla

	venue.id	DOW	FACC	IDN3	NM3_left	IDN2	NM2	VYMERA	NUTS1	NUTS1_CODE	...	venue.location.formattedAddress	NM3_ri
20	57c6bae2498e8cb651ee4ec1	2021-02-26	FA003	101.0	Bratislava I	1.0	Bratislavský	9590124.0	Slovensko	SK0	...	[Panská 23, 811 03 Bratislava, Slovensko]	Bratisla
38	5735f63f498ecb3c24d62247	2021-02-26	FA003	101.0	Bratislava I	1.0	Bratislavský	9590124.0	Slovensko	SK0	...	[Rybárska Brána 9 (Hviezdoslavovo nám.), 811 0...	Bratisla
39	4bf3a81d6a31d13aaeda942e	2021-02-26	FA003	102.0	Bratislava II	1.0	Bratislavský	92490067.0	Slovensko	SK0	...	[Galvaniho 15/B (HP), 831 04 Bratislava, Slove...	Bratisla
59	54109fb0498e2c760a38a4f0	2021-02-26	FA003	101.0	Bratislava I	1.0	Bratislavský	9590124.0	Slovensko	SK0	...	[Špitálska 4, Bratislava, Slovensko]	Bratisla
61	5da3206ee202ec0008c3f764	2021-02-26	FA003	102.0	Bratislava II	1.0	Bratislavský	92490067.0	Slovensko	SK0	...	[Miletičova 90, 821 08 Bratislava, Slovensko]	Bratisla
76	51a38870498e456048c53056	2021-02-26	FA003	102.0	Bratislava II	1.0	Bratislavský	92490067.0	Slovensko	SK0	...	[Rožňavská 1A, 831 04 Bratislava, Slovensko]	Bratisla
101	4d0bb1d65f86f04d2663cda0	2021-02-26	FA003	102.0	Bratislava II	1.0	Bratislavský	92490067.0	Slovensko	SK0	...	[Ivanská cesta 16, 821 04 Bratislava, Slovensko]	Bratisla
102	583afede610528301a23ee78	2021-02-26	FA003	207.0	Trnava	2.0	Trnavský	741316337.0	Slovensko	SK0	...	[Trstínska cesta, Trnava, Slovensko]	Trn
148	592e76dac9f9074856fcd2f4	2021-02-26	FA003	104.0	Bratislava IV	1.0	Bratislavský	96665027.0	Slovensko	SK0	...	[Bridlicová 17, 841 97 Bratislava, Slovensko]	Bratis
210	5682a6c438fa6298b1938efc	2021-02-26	FA003	106.0	Malacky	1.0	Bratislavský	949564634.0	Slovensko	SK0	...	[Sasinkova 2, 901 01 Malacky, Slovensko]	Mala
235	5ce11f60b8fd9d002c565f2d	2021-02-26	FA003	107.0	Pezinok	1.0	Bratislavský	375538031.0	Slovensko	SK0	...	[Harmónia 3001 (Okružná), 900 01 Modra, Sloven...	Pezi
271	549ead0a498e3fb730cd17ee	2021-02-26	FA003	204.0	Piešťany	2.0	Trnavský	381115603.0	Slovensko	SK0	...	[The CUP cafe & home (Námestie J. Murgaša 3), ...	Piešt
282	548ac4ef498e812f8430e169	2021-02-26	FA003	204.0	Piešťany	2.0	Trnavský	381115603.0	Slovensko	SK0	...	[Winterova 56, 921 01 Piešťany, Slovensko]	Piešt
299	524ae60411d2a6739e6a3986	2021-02-26	FA003	205.0	Senica	2.0	Trnavský	683256902.0	Slovensko	SK0	...	[Hviezdoslavova 484/6, 905 01 Senica, Slovensko]	Se

	venue.id	DOW	FACC	IDN3	NM3_left	IDN2	NM2	VYMERA	NUTS1	NUTS1_CODE	...	venue.location.formattedAddress	NM3_ri
316	53a412d5498e56d991609bbe	2021-02-26	FA003	207.0	Trnava	2.0	Trnavský	741316337.0	Slovensko	SK0	...	[Kalinčiakova 14, 917 01 Trnava, Slovensko]	Trn
317	59e215234ce06668f3d5107e	2021-02-26	FA003	207.0	Trnava	2.0	Trnavský	741316337.0	Slovensko	SK0	...	[Andreja Hlinku, 64, 917 01 Trnava, Slovensko]	Trn
318	52cc561f498e832b858040d3	2021-02-26	FA003	207.0	Trnava	2.0	Trnavský	741316337.0	Slovensko	SK0	...	[Františkánska 18, 917 01 Trnava, Slovensko]	Trn
320	5a9013fe8c812a7fe68dab83	2021-02-26	FA003	207.0	Trnava	2.0	Trnavský	741316337.0	Slovensko	SK0	...	[Divadelna 6, 917 01 Trnava, Slovensko]	Trn
323	4e6a5afcd22d0e4cf6024e78	2021-02-26	FA003	207.0	Trnava	2.0	Trnavský	741316337.0	Slovensko	SK0	...	[Trojičné nám. 4, 917 01 Trnava, Slovensko]	Trn
333	54aedd3d498e892d44a56c96	2021-02-26	FA003	207.0	Trnava	2.0	Trnavský	741316337.0	Slovensko	SK0	...	[Haulíková 2, 917 01 Trnava, Slovensko]	Trn

22 rows × 39 columns

Now we will try to find non-merged (right-only) cafe shops names from webscrape data source in FSQ by name

lets prepare dataframe ws_names

In [33]:

```
# cafes only from webscrape data source
ws_names = df[df['_merge']=='right_only'][['C_Name']]
ws_names['FSQ_search_name'] = ws_names['C_Name'].str.normalize('NFKD').str.encode('ascii', errors='ignore').str.decode('utf-8').str.strip('')
ws_names['search_name'] = ws_names['C_Name'].str.normalize('NFKD').str.encode('ascii', errors='ignore').str.decode('utf-8').str.strip('').str
#.groupby(['C_Name']).sum().reset_index()
ws_names['venue.id'] = 0
ws_names
```

Out[33]:

	C_Name	FSQ_search_name	search_name	venue.id
342	About coffee	about_coffee	aboutcoffee	0
343	Abstract cafe	abstract_cafe	abstractcafe	0

	C_Name	FSQ_search_name	search_name	venue.id
344	Adel cafe	adel_cafe	adelcafe	0
345	Amnesia Trnava	amnesia_trnava	amnesiatrnava	0
346	Antikvariát-kaviareň Senec	antikvariat_kaviaren_senec	antikvariatkaviarensenec	0
...	...	...	...	...
556	Yogi's Eurovea	yogi_s_eurovea	yogiseurovea	0
557	Yogi ´s Mirage	yogi__s_mirage	yogismirage	0
558	Záhir Cafe	zahir_cafe	zahircafe	0
559	Záhir Coffee & Drinks	zahir_coffee___drinks	zahircoffeedrinks	0
560	Žufaňa	zufana	zufana	0

219 rows × 4 columns

We are going to search for venues by name using radius parameter

We will use geographical center of these two regions - vilage Bahon as ll parameter and radius 67000 m as radius parameter. Let's see it on map.

As we can see, both regions fit to defined radius.

In [73]:

```
radius_meters = 66000
f = folium.Figure(width=800, height=400)
m = folium.Map(location=[latitude, longitude], zoom_start=8)

choro = folium.Choropleth(cafe_gpd, data=cafe_gpd,
                  name='Regional boundaries',
                  key_on='feature.properties.NM3',
                  threshold_scale=[0,100, 200,300],
                  columns=['NM3', 'IDN3'],
                  fill_color='YlGnBu'
                  )
# remove legend from map
for key in choro._children:
    if key.startswith('color_map'):
        del(choro._children[key])

choro.add_to(m)
```

```
fg_radius = folium.FeatureGroup(name='Search radius')
folium.Circle([latitude, longitude], radius=radius_meters, color='blue', fill=False).add_to(fg_radius)

fg_districts = folium.FeatureGroup(name='Districts centers')
for lat, lon, name , address in zip(cafe_regions['latitude'], cafe_regions['longitude'], cafe_regions['NM3'], cafe_regions['address']):
    pp= folium.Html('<h3>' + str(name) + '</h3>' + '<p>' + str(address) + '</p>', script=True)
    label = folium.Popup(pp, max_width=200)
    folium.CircleMarker(
        [lat, lon],
        radius=2,
        popup=label,
        #color=rainbow[int(cluster-1)],
        color='darkred',
        fill=True,
        #fill_color=rainbow[int(cluster-1)],
        fill_color='darkred',
        fill_opacity=0.7).add_to(fg_districts)

fg_districts.add_to(m)
fg_radius.add_to(m)

folium.LayerControl().add_to(m)
f.add_child(m)

f
```

Out[73]:

Define function that finds venue_candidates by name using FSQ API

Function will use FSQ search API with parameters : latitude,longitude, cafe name as search string, list of selected venue categories for categoryID and 66km for radius parameter.

In [35]:

```
def get_venues_by_name(venue_names):

    venues_by_name_r = pd.DataFrame()
    VERSION = '20180605' # Foursquare API version
    LIMIT = 3
    print('Obtaining venues by name:', end='')
    version = '20180724'
    max_queries = 500
    end_index = int(min(max_queries,venue_names.size))

    for venue_name in venue_names[0:end_index+1]['FSQ_search_name']:
        venues_by_name_l = pd.DataFrame()
        #find by name
```

```
        url = 'https://api.foursquare.com/v2/venues/search?client_id={}&client_secret={}&ll={},{}&query={}&categoryId={}&v={}&radius={}&limit
        results = requests.get(url).json()
        if results['meta']['code'] == 200:
            try:
                ven_res = results['response']['venues']
                venues_result = pd.json_normalize(ven_res) # flatten JSON
                if venues_result.size > 0:
                    #
                    venues_result = venues_result.add_prefix('venue.')

                    # filter the category for each row
                    venues_result['venue.categories'] = venues_result.apply(get_category_type, axis=1)

                    # detailed info about venue
                    filtered_columns = ['venue.id','venue.name', 'venue.categories', 'venue.location.lat', 'venue.location.lng',
                        'venue.location.postalCode','venue.location.cc', 'venue.location.city', 'venue.location.state',
                        'venue.location.country', 'venue.location.formattedAddress']
                    # fill missing columns wit NaN
                    columns = list(venues_result.columns.values)
                    for col in filtered_columns:
                        if col not in columns:
                            venues_result[col] = np.nan

                    new_row  = venues_result.loc[:, filtered_columns]
                    new_row['NM3'] = ""
                    new_row['FSQ_rating'] = 0
                    new_row['FSQ_search_name'] = venue_name
                    new_row['search_name'] = new_row['venue.name'].str.normalize('NFKD').str.encode('ascii', errors='ignore').str.decode('utf
                    venues_by_name_l = venues_by_name_l.append(new_row)

            except:
                import traceback
                print("Exception: ")
                traceback.print_exc()
                print(url)
                #print(results)
        else:
            print('error')
            print(url)
            print(results)

        if venues_by_name_l.size > 0:
            venues_by_name_r = venues_by_name_r.append(pd.DataFrame(venues_by_name_l))

    print(' .', end='')
    print(' done.')
    return venues_by_name_r
```

Find candidates by name or load them from pickle

In this part we are again using pickle to speed up process. If there is need to find venues in FSQ by name, we are using function get_venues_by_name to find them, check if they are in region, merge them with existing venue candidates and store them in pickle.

In [36]:

```
#import pickle
loaded = False
import numpy as np
# find venues_by_name with venue.id = 0
find_by_name = False
find_by_name = True

try:
    with open('venue_candidates.pkl', 'rb') as f:
        venue_candidates = pickle.load(f)
    print('venue_candidates data loaded.')
    print(len(venue_candidates.index))
    loaded = True
except:
    pass

# If load failed or we want find by name, then use the Foursquare API to get the data

if not loaded or find_by_name:
    ws_names_sel = pd.DataFrame (ws_names[ws_names['venue.id'] == 0])
    candidates_by_name = get_venues_by_name(ws_names_sel)


    # we need to check if all venues are from region
    poly = cafe_gpd
    points = gpd.GeoDataFrame(candidates_by_name, geometry= gpd.points_from_xy(candidates_by_name["venue.location.lng"], candidates_by_name['
    poly = poly.to_crs(epsg =4326)
    candidates_by_name = pd.DataFrame(gpd.sjoin(poly, points, op='contains'))
    #candidates_by_name = pd.DataFrame(gpd.sjoin(poly, points, op='contains'))
    #candidates_by_name

    #remove duplicates and select columns
    candidates_by_name = candidates_by_name.drop_duplicates(subset = ["venue.id"])
    filtered_columns = ['venue.id', 'DOW', 'FACC', 'IDN3', 'NM3_left', 'IDN2', 'NM2', 'VYMERA',
           'NUTS1', 'NUTS1_CODE', 'NUTS2', 'NUTS2_CODE', 'NUTS3', 'NUTS3_CODE',
           'LAU1', 'LAU1_CODE', 'Shape_Leng', 'Shape_Area', 'geometry',
           'index_right', 'venue.name', 'venue.categories', 'venue.location.lat',
           'venue.location.lng', 'venue.location.postalCode', 'venue.location.cc',
```

```
        'venue.location.city', 'venue.location.state', 'venue.location.country',
        'venue.location.formattedAddress', 'NM3_right', 'search_name',
        'FSQ_rating']
    candidates_by_name  = candidates_by_name.loc[:, filtered_columns]


    # prepare temporary merge_df to select found candidates_by_name
    merge_df = pd.merge(candidates_by_name,cnames,on=['search_name','search_name'],how="outer",indicator=True)

    merge_df2 = merge_df[merge_df['_merge']=='both'].drop_duplicates(subset = ["venue.id"])


    filtered_columns = ['venue.id', 'DOW', 'FACC', 'IDN3', 'NM3_left', 'IDN2', 'NM2', 'VYMERA',
        'NUTS1', 'NUTS1_CODE', 'NUTS2', 'NUTS2_CODE', 'NUTS3', 'NUTS3_CODE',
        'LAU1', 'LAU1_CODE', 'Shape_Leng', 'Shape_Area', 'geometry',
        'index_right', 'venue.name', 'venue.categories', 'venue.location.lat',
        'venue.location.lng', 'venue.location.postalCode', 'venue.location.cc',
        'venue.location.city', 'venue.location.state', 'venue.location.country',
        'venue.location.formattedAddress', 'NM3_right', 'search_name',
        'FSQ_rating']
    merge_df2  = merge_df2.loc[:, filtered_columns]
    #merge_df2
    venue_candidates_n = venue_candidates.copy()

    venue_candidates_n = venue_candidates_n.append(merge_df2)

    # lets put venue_candidates_n in pickle
    if venue_candidates_n.size > 0:
        print('new venue_candidates size')
        print(len(venue_candidates_n.index))
        #if everything is ok , use this:
        #venue_candidates = venue_candidates_n.copy()
        #with open('venue_candidates.pkl', 'wb') as f:

        # Let's persists this in local file system
        with open('venue_candidates.pkl', 'wb') as f:
            pickle.dump(venue_candidates_n, f)
        venue_candidates = venue_candidates_n.copy()
```

```
venue_candidates data loaded.
342
Obtaining venues by name: . done.
new venue_candidates size
368
```

In [37]:

```
venue_candidates_n
```

Out[37]:

	venue.id	DOW	FACC	IDN3	NM3_left	IDN2	NM2	VYMERA	NUTS1	NUTS1_CODE	...	venue.location.lng	venue.location.postalC
0	5a1d478fbfc6d03f26080c39	2021-02-26	FA003	101.0	Bratislava I	1.0	Bratislavský	9590124.0	Slovensko	SK0	...	17.099158	81
1	4bca1c7868f976b0feab5e83	2021-02-26	FA003	101.0	Bratislava I	1.0	Bratislavský	9590124.0	Slovensko	SK0	...	17.099320	81
2	55689447498e0fb52a2c1b6d	2021-02-26	FA003	101.0	Bratislava I	1.0	Bratislavský	9590124.0	Slovensko	SK0	...	17.100038	81
3	4c40c026af052d7f734b7c79	2021-02-26	FA003	101.0	Bratislava I	1.0	Bratislavský	9590124.0	Slovensko	SK0	...	17.101435	81
4	55830c83498e2ca875974b43	2021-02-26	FA003	101.0	Bratislava I	1.0	Bratislavský	9590124.0	Slovensko	SK0	...	17.101443	
...	...	...	...	...	...	...	...	...	...	...	...	...	
107	5d9e6c529766eb00080fe372	2021-02-26	FA003	207.0	Trnava	2.0	Trnavský	741316337.0	Slovensko	SK0	...	17.585556	91
108	53a412d5498e56d991609bbe	2021-02-26	FA003	207.0	Trnava	2.0	Trnavský	741316337.0	Slovensko	SK0	...	17.580152	91
109	4e6a5afcd22d0e4cf6024e78	2021-02-26	FA003	207.0	Trnava	2.0	Trnavský	741316337.0	Slovensko	SK0	...	17.586010	91
113	59edff90c4df1d5067d394bb	2021-02-26	FA003	207.0	Trnava	2.0	Trnavský	741316337.0	Slovensko	SK0	...	17.583695	91
119	59e215234ce06668f3d5107e	2021-02-26	FA003	207.0	Trnava	2.0	Trnavský	741316337.0	Slovensko	SK0	...	17.581095	91

368 rows × 33 columns

Final merge

This is final merge of venue candidates with information from web scraping

In [40]:

```
# prepare temporary df to check what can be merged and what web scraped cafes are not in venue_candidates
print("venue_candidates.shape {} ".format(venue_candidates.shape))
print("cnames.shape {} ".format(cnames.shape))
dff=pd.merge(venue_candidates,cnames,on=['search_name','search_name'],how="outer",indicator=True)
print("dff.shape {} ".format(dff.shape))
```

```
venue_candidates.shape (368, 33)
cnames.shape (239, 6)
dff.shape (567, 39)
```

In [42]:

```python
#372 - 2 duplicates
# select all FSQ candidiates - merge type both nad left
merged_venues = dff[dff['_merge'] !='right_only'].drop_duplicates(subset = ["venue.id"])

filtered_columns = ['venue.id', 'IDN3', 'NM3_left', 'IDN2', 'NM2',
       'NUTS1', 'NUTS1_CODE', 'NUTS2', 'NUTS2_CODE', 'NUTS3', 'NUTS3_CODE',
       'LAU1', 'LAU1_CODE','venue.name', 'venue.categories', 'venue.location.lat',
       'venue.location.lng', 'venue.location.postalCode', 'venue.location.cc',
       'venue.location.city', 'venue.location.state', 'venue.location.country',
       'venue.location.formattedAddress', 'NM3_right', 'search_name',
       'FSQ_rating', 'C_Name', 'covid_sp', 'covid_co', 'bc', 'ect', '_merge']
merged_venues  = merged_venues.loc[:, filtered_columns]
merged_venues
```

Out[42]:

	venue.id	IDN3	NM3_left	IDN2	NM2	NUTS1	NUTS1_CODE	NUTS2	NUTS2_CODE	NUTS3	...	venue.location.formattedAddre
0	5a1d478fbfc6d03f26080c39	101.0	Bratislava I	1.0	Bratislavský	Slovensko	SK0	Bratislavský kraj	SK01	Bratislavský kraj	...	[Partizánska 2, 811 03 Bratislav Slovensk
1	4bca1c7868f976b0feab5e83	101.0	Bratislava I	1.0	Bratislavský	Slovensko	SK0	Bratislavský kraj	SK01	Bratislavský kraj	...	[Partizánska 2 (Palisády), 811 0 Bratislava,
2	55689447498e0fb52a2c1b6d	101.0	Bratislava I	1.0	Bratislavský	Slovensko	SK0	Bratislavský kraj	SK01	Bratislavský kraj	...	[Zámocká 7327/24, 811 0 Bratislava, Slovensk
3	4c40c026af052d7f734b7c79	101.0	Bratislava I	1.0	Bratislavský	Slovensko	SK0	Bratislavský kraj	SK01	Bratislavský kraj	...	[Zámocká 30, 811 01 Bratislav Slovensk
4	55830c83498e2ca875974b43	101.0	Bratislava I	1.0	Bratislavský	Slovensko	SK0	Bratislavský kraj	SK01	Bratislavský kraj	...	[Zámocká 30, Bratislava, Slovensk
...	...	...	...	...	...	...	...	...	...	...	...	
363	5e1b30b62a96260008fdc896	108.0	Senec	1.0	Bratislavský	Slovensko	SK0	Bratislavský kraj	SK01	Bratislavský kraj	...	[900 42 Dunajská Lužná, Slovensk
364	5662cae9498efe11ead5115b	204.0	Piešťany	2.0	Trnavský	Slovensko	SK0	Západné Slovensko	SK02	Trnavský kraj	...	[Slovensk
365	5d691a2a228cfc00081c978d	206.0	Skalica	2.0	Trnavský	Slovensko	SK0	Západné Slovensko	SK02	Trnavský kraj	...	[Potočná 50, 909 01 Skalic Slovensk

	venue.id	IDN3	NM3_left	IDN2	NM2	NUTS1	NUTS1_CODE	NUTS2	NUTS2_CODE	NUTS3	...	venue.location.formattedAddre
366	5d9e6c529766eb00080fe372	207.0	Trnava	2.0	Trnavský	Slovensko	SK0	Západné Slovensko	SK02	Trnavský kraj	...	[2 Trojičné námestie, 917 01 Trnav Slovensk
367	59edff90c4df1d5067d394bb	207.0	Trnava	2.0	Trnavský	Slovensko	SK0	Západné Slovensko	SK02	Trnavský kraj	...	[Štafánikova 26, 917 01 Trnav Slovensk

362 rows × 32 columns

In [43]:

```
# Replace NaN in ratings
subset_cols = ['covid_sp', 'covid_co', 'bc', 'ect']
[merged_venues[col].fillna(0, inplace=True) for col in subset_cols]
merged_venues
```

Out[43]:

	venue.id	IDN3	NM3_left	IDN2	NM2	NUTS1	NUTS1_CODE	NUTS2	NUTS2_CODE	NUTS3	...	venue.location.formattedAddre
0	5a1d478fbfc6d03f26080c39	101.0	Bratislava I	1.0	Bratislavský	Slovensko	SK0	Bratislavský kraj	SK01	Bratislavský kraj	...	[Partizánska 2, 811 03 Bratislav Slovensk
1	4bca1c7868f976b0feab5e83	101.0	Bratislava I	1.0	Bratislavský	Slovensko	SK0	Bratislavský kraj	SK01	Bratislavský kraj	...	[Partizánska 2 (Palisády), 811 0 Bratislava,
2	55689447498e0fb52a2c1b6d	101.0	Bratislava I	1.0	Bratislavský	Slovensko	SK0	Bratislavský kraj	SK01	Bratislavský kraj	...	[Zámocká 7327/24, 811 0 Bratislava, Slovensk
3	4c40c026af052d7f734b7c79	101.0	Bratislava I	1.0	Bratislavský	Slovensko	SK0	Bratislavský kraj	SK01	Bratislavský kraj	...	[Zámocká 30, 811 01 Bratislav Slovensk
4	55830c83498e2ca875974b43	101.0	Bratislava I	1.0	Bratislavský	Slovensko	SK0	Bratislavský kraj	SK01	Bratislavský kraj	...	[Zámocká 30, Bratislava, Slovensk
...	...	...	...	...	...	...	...	...	...	...	...	
363	5e1b30b62a96260008fdc896	108.0	Senec	1.0	Bratislavský	Slovensko	SK0	Bratislavský kraj	SK01	Bratislavský kraj	...	[900 42 Dunajská Lužná, Slovensk
364	5662cae9498efe11ead5115b	204.0	Piešťany	2.0	Trnavský	Slovensko	SK0	Západné Slovensko	SK02	Trnavský kraj	...	[Slovensk
365	5d691a2a228cfc00081c978d	206.0	Skalica	2.0	Trnavský	Slovensko	SK0	Západné Slovensko	SK02	Trnavský kraj	...	[Potočná 50, 909 01 Skalic Slovensk
366	5d9e6c529766eb00080fe372	207.0	Trnava	2.0	Trnavský	Slovensko	SK0	Západné Slovensko	SK02	Trnavský kraj	...	[2 Trojičné námestie, 917 01 Trnav Slovensk

	venue.id	IDN3	NM3_left	IDN2	NM2	NUTS1	NUTS1_CODE	NUTS2	NUTS2_CODE	NUTS3	...	venue.location.formattedAddre
367	59edff90c4df1d5067d394bb	207.0	Trnava	2.0	Trnavský	Slovensko	SK0	Západné Slovensko	SK02	Trnavský kraj	...	[Štafánikova 26, 917 01 Trnav Slovensk

362 rows × 32 columns

We will add FSQ rating from pickle

For new venue candidates we will try to merge rating from pickle. If there is missing rating we will try to find it using function get_cofee_rating. Updated rating is stored again in pickle.

In [44]:

```
# add rating from pickle - we added some new venues
import pickle
cofee_rating = []
loaded = False

# find rating for merged_venues with rating = 0
find_rating = False
#find_rating = True

try:
    with open('cofee_rating.pkl', 'rb') as f:
        cofee_rating = pickle.load(f)
    print('cofee_rating data loaded.')
    print(len(cofee_rating.index))
    loaded = True
except:
    pass

# If load failed use the Foursquare API to get the data
if not loaded:
    venue_rating = pd.DataFrame (merged_venues[merged_venues['FSQ_rating'] == 0][['venue.id','FSQ_rating']])
    cofee_rating = get_cofee_rating(venue_rating)

#lets map rating to merged_venues using ['venue.id']
cofee_rating = cofee_rating.set_index('venue.id')
merged_venues = merged_venues.set_index('venue.id')
merged_venues.update(cofee_rating)
merged_venues.reset_index(inplace=True)
cofee_rating.reset_index(inplace=True)
```

```
#if there is missing rating (rating = 0) - try to find it
venue_rating = pd.DataFrame (merged_venues[merged_venues['FSQ_rating'] == 0][['venue.id','FSQ_rating']])
if venue_rating.size > 0 and find_rating:
    print("VR")
    print(venue_rating)
    cofee_rating = get_cofee_rating(venue_rating)
    cofee_rating = cofee_rating.set_index('venue.id')
    merged_venues = merged_venues.set_index('venue.id')
    merged_venues.update(cofee_rating)
    merged_venues.reset_index(inplace=True)
    cofee_rating.reset_index(inplace=True)

# lets select id and rating from merged_venues and store it
cofee_rating_n = pd.DataFrame(merged_venues[merged_venues['FSQ_rating']!= 0][[ 'venue.id','FSQ_rating']])
if cofee_rating_n.size > 0 :
    print('new rating size')
    print(len(cofee_rating_n.index))
    # Let's persists this in local file system
    with open('cofee_rating.pkl', 'wb') as f:
        pickle.dump(cofee_rating_n, f)
```

```
cofee_rating data loaded.
342
new rating size
342
```

Final check that all venues are from region

In [46]:

```
# we need to check if all venues are from region
filtered_columns = ['venue.id', 'IDN3', 'NM3_left', 'IDN2', 'NM2',
       'NUTS1', 'NUTS1_CODE', 'NUTS2', 'NUTS2_CODE', 'NUTS3', 'NUTS3_CODE',
       'LAU1', 'LAU1_CODE','venue.name', 'venue.categories', 'venue.location.lat',
       'venue.location.lng', 'venue.location.postalCode', 'venue.location.cc',
       'venue.location.city', 'venue.location.state', 'venue.location.country',
       'venue.location.formattedAddress', 'NM3_right', 'search_name',
       'FSQ_rating', 'C_Name', 'covid_sp', 'covid_co', 'bc', 'ect', '_merge']
merged_venues  = merged_venues.loc[:, filtered_columns]

poly = cafe_gpd
points = gpd.GeoDataFrame(merged_venues, geometry= gpd.points_from_xy(merged_venues["venue.location.lng"], merged_venues["venue.location.lat'
poly = poly.to_crs(epsg =4326)
merged_venues = pd.DataFrame(gpd.sjoin(poly, points, op='contains'))
merged_venues
```

Out[46]:

DOW	FACC	IDN3_left	NM3	IDN2_left	NM2_left	VYMERA	NUTS1_left	NUTS1_CODE_left	NUTS2_left	...	venue.location.formattedAddress	NM3_right

	DOW	FACC	IDN3_left	NM3	IDN2_left	NM2_left	VYMERA	NUTS1_left	NUTS1_CODE_left	NUTS2_left	...	venue.location.formattedAddress	NM3_right
0	2021-02-26	FA003	101	Bratislava I	1	Bratislavský	9590124.0	Slovensko	SK0	Bratislavský kraj	...	[Hviedzoslavovo nám. 11 (Rybné nám.), 811 02 B...	Bratislava I
0	2021-02-26	FA003	101	Bratislava I	1	Bratislavský	9590124.0	Slovensko	SK0	Bratislavský kraj	...	[Medená 4 (Tobrucká), 841 06 Bratislava, Slove...	Bratislava I
0	2021-02-26	FA003	101	Bratislava I	1	Bratislavský	9590124.0	Slovensko	SK0	Bratislavský kraj	...	[Palackého 6, 811 02 Bratislava, Slovensko]	
0	2021-02-26	FA003	101	Bratislava I	1	Bratislavský	9590124.0	Slovensko	SK0	Bratislavský kraj	...	[Hviezdoslavovo nám. 19, 811 02 Bratislava, Sl...	Bratislava I
0	2021-02-26	FA003	101	Bratislava I	1	Bratislavský	9590124.0	Slovensko	SK0	Bratislavský kraj	...	[Panská 27 (Straková), 811 01 Bratislava, Slov...	Bratislava I
...	...	...	...	...	...	...	...	...	...	...	...	...	...
14	2021-02-26	FA003	207	Trnava	2	Trnavský	741316337.0	Slovensko	SK0	Západné Slovensko	...	[Slovensko]	Trnava
14	2021-02-26	FA003	207	Trnava	2	Trnavský	741316337.0	Slovensko	SK0	Západné Slovensko	...	[F. Urbanka, Trnava, Slovensko]	Trnava
14	2021-02-26	FA003	207	Trnava	2	Trnavský	741316337.0	Slovensko	SK0	Západné Slovensko	...	[J. Bottu 25, 917 01 Trnava, Slovensko]	Trnava
14	2021-02-26	FA003	207	Trnava	2	Trnavský	741316337.0	Slovensko	SK0	Západné Slovensko	...	[Andreja Hlinku, 64, 917 01 Trnava, Slovensko]	Trnava
14	2021-02-26	FA003	207	Trnava	2	Trnavský	741316337.0	Slovensko	SK0	Západné Slovensko	...	[Slovensko]	Trnava

362 rows × 51 columns

Replace/explain -1 in FSQ rating

As we already mentioned -1 in rating means that there is no information about rating in FSQ. We can replace this value with 0 for computing of total rating using commented code in next cells.

We decided to not replace this value and use -1 to lower total score of venue. This way if there is 0 in FSQ rating (because of 50 request premium limit per day) this venues with unknow rating will be higher in order than venues without rating.

In [66]:

```
#code for update FSQ rating for not found rating (-1) - comemented , we are not going to use it
#merged_venues.loc[merged_venues['FSQ_rating'] == -1, 'FSQ_rating'] = 0
```

Lets compute total rating

total_rating is computed using weights defined for each source

weights for sources:

- W_FSQ = 1 - we will use same score asi in FSQ
- W_covid_sp = 12 - this source have listed caffe shops which are open during COVID restrictions - sp - means that coffe shop allready sells specialty coffe
- W_covid_co = 9 - this source have listed caffe shops which are open during COVID restrictions - co - means that coffe shop sells comodity coffe - good candidate to sell something new :-)
- W_bc = 8.5 - this source have listed best caffe shops
- W_ect = 8.5 - this source have listed best caffe shops

In [48]:

```
# define weights for each source
W_FSQ = 1
W_covid_sp = 12
W_covid_co = 9
W_bc = 8.5
W_ect = 8.5
#nearby_venues_det["Total_rating"] = nearby_venues_det.apply(lambda row: (row['FSQ_rating']*W_FSQ + row['covid']*W_covid + row['bc']*W_bc + r
merged_venues["total_rating"] = merged_venues['FSQ_rating']*W_FSQ + merged_venues['covid_sp']*W_covid_sp + merged_venues['covid_co']*W_covid_

merged_venues
```

Out[48]:

	DOW	FACC	IDN3_left	NM3	IDN2_left	NM2_left	VYMERA	NUTS1_left	NUTS1_CODE_left	NUTS2_left	...	NM3_right	search_name	FSQ_rating	C
0	2021-02-26	FA003	101	Bratislava I	1	Bratislavský	9590124.0	Slovensko	SK0	Bratislavský kraj	...	Bratislava I	moods	6.0	
0	2021-02-26	FA003	101	Bratislava I	1	Bratislavský	9590124.0	Slovensko	SK0	Bratislavský kraj	...	Bratislava I	skodovka	8.0	
0	2021-02-26	FA003	101	Bratislava I	1	Bratislavský	9590124.0	Slovensko	SK0	Bratislavský kraj	...		kafehausbarzzuz	0.0	Ka

	DOW	FACC	IDN3_left	NM3	IDN2_left	NM2_left	VYMERA	NUTS1_left	NUTS1_CODE_left	NUTS2_left	...	NM3_right	search_name	FSQ_rating	C
0	2021-02-26	FA003	101	Bratislava I	1	Bratislavský	9590124.0	Slovensko	SK0	Bratislavský kraj	...	Bratislava I	zylinder	6.0	
0	2021-02-26	FA003	101	Bratislava I	1	Bratislavský	9590124.0	Slovensko	SK0	Bratislavský kraj	...	Bratislava I	mondieu	7.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
14	2021-02-26	FA003	207	Trnava	2	Trnavský	741316337.0	Slovensko	SK0	Západné Slovensko	...	Trnava	funcafe	-1.0	
14	2021-02-26	FA003	207	Trnava	2	Trnavský	741316337.0	Slovensko	SK0	Západné Slovensko	...	Trnava	lokalka	-1.0	
14	2021-02-26	FA003	207	Trnava	2	Trnavský	741316337.0	Slovensko	SK0	Západné Slovensko	...	Trnava	mtftbufetjedalen	-1.0	
14	2021-02-26	FA003	207	Trnava	2	Trnavský	741316337.0	Slovensko	SK0	Západné Slovensko	...	Trnava	oneespressobar	8.0	ESI BA e
14	2021-02-26	FA003	207	Trnava	2	Trnavský	741316337.0	Slovensko	SK0	Západné Slovensko	...	Trnava	caffemanuel	-1.0	

362 rows × 52 columns

prepare clickable URL links for report

We decided to prepare URL links to FSQ detail abpout venue which will be used in reports - for map labels and in table. This way can stakeholder check all details about cafe shop to decide if it interesting candidate or not.

In [49]:

```
merged_venues["url"] = merged_venues.apply(lambda row: 'https://foursquare.com/v/{}/{}'.format(row['venue.name'],row['venue.id']), axis=1)
merged_venues["clickable_url"] = merged_venues.apply(lambda row: "<a href='{}' target='_blank'>{}</a>".format(row['url'], row['venue.name']),
merged_venues
```

Out[49]:

	DOW	FACC	IDN3_left	NM3	IDN2_left	NM2_left	VYMERA	NUTS1_left	NUTS1_CODE_left	NUTS2_left	...	FSQ_rating	C_Name	covid_sp	covid_co
0	2021-02-26	FA003	101	Bratislava I	1	Bratislavský	9590124.0	Slovensko	SK0	Bratislavský kraj	...	6.0	NaN	0.0	0.0

	DOW	FACC	IDN3_left	NM3	IDN2_left	NM2_left	VYMERA	NUTS1_left	NUTS1_CODE_left	NUTS2_left	...	FSQ_rating	C_Name	covid_sp	covid_co
0	2021-02-26	FA003	101	Bratislava I	1	Bratislavský	9590124.0	Slovensko	SK0	Bratislavský kraj	...	8.0	NaN	0.0	0.0
0	2021-02-26	FA003	101	Bratislava I	1	Bratislavský	9590124.0	Slovensko	SK0	Bratislavský kraj	...	0.0	KafeHaus Barzzuz	0.0	0.0
0	2021-02-26	FA003	101	Bratislava I	1	Bratislavský	9590124.0	Slovensko	SK0	Bratislavský kraj	...	6.0	NaN	0.0	0.0
0	2021-02-26	FA003	101	Bratislava I	1	Bratislavský	9590124.0	Slovensko	SK0	Bratislavský kraj	...	7.0	NaN	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
14	2021-02-26	FA003	207	Trnava	2	Trnavský	741316337.0	Slovensko	SK0	Západné Slovensko	...	-1.0	NaN	0.0	0.0
14	2021-02-26	FA003	207	Trnava	2	Trnavský	741316337.0	Slovensko	SK0	Západné Slovensko	...	-1.0	NaN	0.0	0.0
14	2021-02-26	FA003	207	Trnava	2	Trnavský	741316337.0	Slovensko	SK0	Západné Slovensko	...	-1.0	NaN	0.0	0.0
14	2021-02-26	FA003	207	Trnava	2	Trnavský	741316337.0	Slovensko	SK0	Západné Slovensko	...	8.0	ONÉ ESPRESSO BAR-ONé espresso bar	0.0	0.0
14	2021-02-26	FA003	207	Trnava	2	Trnavský	741316337.0	Slovensko	SK0	Západné Slovensko	...	-1.0	NaN	0.0	0.0

362 rows × 54 columns

Lets select top 100 merged venue_candidates

We are going to select top 100 candidates sorted by total rating, these candidates will be on final report. We are using pickle again to store data. In case that stakeholder wants to see another top coffe shops, we don't have to run all analysis again, we will just load data from pickle and change n - number of candidates.

Save merged venues to pickle to be able choose another TOP n coffe shops

In [50]:

```
## Save output to pickle for reuse
```

```
save_merged_venues = True
load_merged_venues = False
loaded = False

if merged_venues.size > 0 and save_merged_venues :
    print('merged_venues size')
    print(len(merged_venues.index))
    # Let's persists this in local file system
    with open('merged_venues.pkl', 'wb') as f:
        pickle.dump(merged_venues, f)

if load_merged_venues :
    try:
        with open('merged_venues.pkl', 'rb') as f:
            merged_venues = pickle.load(f)
        print('merged_venues data loaded.')
        print(len(merged_venues.index))
        loaded = True
    except:
        pass
```

```
merged_venues size
362
```

Select TOP n merged venues by total rating

In [52]:

```
n = 100
venues_top_n = merged_venues.sort_values(['total_rating'], ascending=[False]).head(n)
venues_top_n
```

Out[52]:

	DOW	FACC	IDN3_left	NM3	IDN2_left	NM2_left	VYMERA	NUTS1_left	NUTS1_CODE_left	NUTS2_left	...	FSQ_rating	C_Name	covid_sp	covid_co
1	2021-02-26	FA003	102	Bratislava II	1	Bratislavský	92490067.0	Slovensko	SK0	Bratislavský kraj	...	9.0	Brew Bar Café	0.0	0.0
14	2021-02-26	FA003	207	Trnava	2	Trnavský	741316337.0	Slovensko	SK0	Západné Slovensko	...	8.0	Synagóga Café	0.0	0.0
14	2021-02-26	FA003	207	Trnava	2	Trnavský	741316337.0	Slovensko	SK0	Západné Slovensko	...	8.0	ONÉ ESPRESSO BAR-ONé espresso bar	0.0	0.0

	DOW	FACC	IDN3_left	NM3	IDN2_left	NM2_left	VYMERA	NUTS1_left	NUTS1_CODE_left	NUTS2_left	...	FSQ_rating	C_Name	covid_sp	covid_co
0	2021-02-26	FA003	101	Bratislava I	1	Bratislavský	9590124.0	Slovensko	SK0	Bratislavský kraj	...	8.0	Urban Bistro	0.0	0.0
6	2021-02-26	FA003	107	Pezinok	1	Bratislavský	375538031.0	Slovensko	SK0	Bratislavský kraj	...	7.0	Hollerung káva & koloniál	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
0	2021-02-26	FA003	101	Bratislava I	1	Bratislavský	9590124.0	Slovensko	SK0	Bratislavský kraj	...	7.0	NaN	0.0	0.0
0	2021-02-26	FA003	101	Bratislava I	1	Bratislavský	9590124.0	Slovensko	SK0	Bratislavský kraj	...	7.0	NaN	0.0	0.0
3	2021-02-26	FA003	104	Bratislava IV	1	Bratislavský	96665027.0	Slovensko	SK0	Bratislavský kraj	...	7.0	NaN	0.0	0.0
6	2021-02-26	FA003	107	Pezinok	1	Bratislavský	375538031.0	Slovensko	SK0	Bratislavský kraj	...	7.0	NaN	0.0	0.0
14	2021-02-26	FA003	207	Trnava	2	Trnavský	741316337.0	Slovensko	SK0	Západné Slovensko	...	7.0	NaN	0.0	0.0

100 rows × 54 columns

Now we will prepare cluster using DBSCAN to help organize busines trips

DBSCAN is specially very good for tasks like class identification on a spatial context. The wonderful attribute of DBSCAN algorithm is that it can find out any arbitrary shape cluster without getting affected by noise.

Clustering of cafes based on their location i.e. Lat & Lon

In [53]:

```
from sklearn.cluster import DBSCAN
import sklearn.utils
import numpy as np
from sklearn.preprocessing import StandardScaler
sklearn.utils.check_random_state(42)
Clus_dataSet = venues_top_n[['venue.location.lng','venue.location.lat']]
Clus_dataSet = np.nan_to_num(Clus_dataSet)
Clus_dataSet = StandardScaler().fit_transform(Clus_dataSet)
```

In [54]:

```
# Compute DBSCAN
#db = DBSCAN(eps=0.15, min_samples=3).fit(Clus_dataSet)
db = DBSCAN(eps=0.3, min_samples=2).fit(Clus_dataSet)
core_samples_mask = np.zeros_like(db.labels_, dtype=bool)
core_samples_mask[db.core_sample_indices_] = True
labels = db.labels_
venues_top_n["cluster"]=labels

realClusterNum=len(set(labels)) - (1 if -1 in labels else 0)
clusterNum = len(set(labels))
print(clusterNum)
# A sample of clusters
venues_top_n.head(5)
```

9

Out[54]:

	DOW	FACC	IDN3_left	NM3	IDN2_left	NM2_left	VYMERA	NUTS1_left	NUTS1_CODE_left	NUTS2_left	...	C_Name	covid_sp	covid_co	bc	ect	_m
1	2021-02-26	FA003	102	Bratislava II	1	Bratislavský	92490067.0	Slovensko	SK0	Bratislavský kraj	...	Brew Bar Café	0.0	0.0	1.0	1.0	
14	2021-02-26	FA003	207	Trnava	2	Trnavský	741316337.0	Slovensko	SK0	Západné Slovensko	...	Synagóga Café	0.0	0.0	1.0	1.0	
14	2021-02-26	FA003	207	Trnava	2	Trnavský	741316337.0	Slovensko	SK0	Západné Slovensko	...	ONÉ ESPRESSO BAR-ONé espresso bar	0.0	0.0	1.0	1.0	
0	2021-02-26	FA003	101	Bratislava I	1	Bratislavský	9590124.0	Slovensko	SK0	Bratislavský kraj	...	Urban Bistro	0.0	0.0	1.0	1.0	
6	2021-02-26	FA003	107	Pezinok	1	Bratislavský	375538031.0	Slovensko	SK0	Bratislavský kraj	...	Hollerung káva & koloniál	0.0	0.0	1.0	1.0	

5 rows × 55 columns

Visualize clusters on map

Red color is used for outliers (-1)

In [55]:

```
# Create a color map
color_list = ['red', 'blue', 'green', 'purple', 'orange', 'darkred',
       'lightred', 'beige', 'darkblue', 'darkgreen', 'cadetblue',
       'darkpurple', 'pink',  'lightblue', 'lightgreen','white',
       'gray', 'black', 'lightgray']

venues_top_n['marker_color'] = pd.cut(venues_top_n['cluster'], bins=clusterNum,labels=color_list[0:clusterNum])
```

In [74]:

```
from folium.features import DivIcon
f = folium.Figure(width=800, height=400)
m = folium.Map(location=[latitude, longitude], zoom_start=8)

choro = folium.Choropleth(cafe_gpd, data=cafe_gpd,
                  name='Regional boundaries',
                  key_on='feature.properties.NM3',
                  threshold_scale=[0,100, 200,300],
                  columns=['NM3', 'IDN3'],
                  #fill_color='YlOrBr'
                  #fill_color='YlGn'
                  #fill_color='PuBu'
                  fill_color='YlGnBu'
                  #fill_color='YlOrBr', # ‘BuGn’, ‘BuPu’, ‘GnBu’, ‘OrRd’, ‘PuBu’, ‘PuBuGn’, ‘PuRd’, ‘RdPu’, ‘YlGn’, ‘YlGnBu’, ‘YlOrBr’, and ‘
                  )
# remove legend from map
for key in choro._children:
    if key.startswith('color_map'):
        del(choro._children[key])
choro.add_to(m)

fg_districts = folium.FeatureGroup(name='Districts centers')
for lat, lon, name , address in zip(cafe_regions['latitude'], cafe_regions['longitude'], cafe_regions['NM3'], cafe_regions['address']):
    pp= folium.Html('<h3>' + str(name) + '</h3>' + '<p>' + str(address) + '</p>', script=True)
    label = folium.Popup(pp, max_width=200)
    folium.CircleMarker(
        [lat, lon],
        radius=2,
        popup=label,
        #color=rainbow[int(cluster-1)],
        color='darkred',
        fill=True,
        #fill_color=rainbow[int(cluster-1)],
        fill_color='darkred',
        fill_opacity=0.7).add_to(fg_districts)

# add markers to the map
```

```
fg_districts.add_to(m)

fg_clusters = folium.FeatureGroup(name='Clusters')

for clust_number in set(labels):
    clust_set = venues_top_n[venues_top_n.cluster == clust_number]
    for index,clust_member in clust_set.iterrows():
        pp= folium.Html('<h2>' + str(clust_number) + '</h2>', script=True)
        label = folium.Popup(pp, max_width=100)
        folium.CircleMarker(
            [clust_member['venue.location.lat'], clust_member['venue.location.lng']],
            radius=8,
            popup=label,
            color = clust_member['marker_color'],
            fill=True,
            fill_color=clust_member['marker_color'],
            opacity=0.2,
            fill_opacity=0.2).add_to(fg_clusters)

fg_clusters.add_to(m)

folium.LayerControl().add_to(m)
f.add_child(m)

f
```

Out[74]:

Lets prepare final report in table

Sort top n venues by cluster,total rating and district (NM3), create output in sorted table and save it

In [57]:

```
venues_top_n_sorted = venues_top_n.sort_values(['cluster', 'total_rating','NM3'], ascending=[True, False, True])
```

In [58]:

```
html = venues_top_n_sorted[['cluster','clickable_url','venue.categories','venue.location.formattedAddress','NM3','covid_sp','covid_co','bc',
# write html to file
file = open("output\\coffe_shops_table.html", "w", encoding="utf-8")
file.write(html)
file.close()
```

In [59]:

```
from IPython.core.display import display, HTML
html_output = HTML(html)
display(html_output)
```

	cluster	clickable_url	venue.categories	venue.location.formattedAddress	NM3	covid_sp	covid_co	bc	ect	FSQ_rating	total_rating
12	-1	Positivo - café & bar	Café	[Hviezdoslavova 484/6, 905 01 Senica, Slovensko]	Senica	0.0	0.0	1.0	0.0	8.0	16.5
5	-1	Tma v Hrnku	Café	[28 Hlavná, 900 31 Stupava, Slovensko]	Malacky	0.0	0.0	0.0	1.0	0.0	8.5
7	-1	Coffee Me	Coffee Shop	[900 42 Dunajská Lužná, Slovensko]	Senec	0.0	0.0	0.0	1.0	0.0	8.5
13	-1	Kafé Knižnica	Café	[Potočná 50, 909 01 Skalica, Slovensko]	Skalica	0.0	0.0	1.0	0.0	0.0	8.5
7	-1	Café Štefánik	Café	[Bernolákovská 18, 900 28 Ivanka pri Dunaji, Slovensko]	Senec	0.0	0.0	0.0	0.0	8.0	8.0
13	-1	Presto	Café	[Hollého 860/35, 908 51 Holíč, Slovensko]	Skalica	0.0	0.0	0.0	0.0	8.0	8.0
9	-1	Albero Rosso - restaurant & bar	Café	[M.R.Štefaníka 2156, 926 01 Sereď, Slovensko]	Galanta	0.0	0.0	0.0	0.0	7.0	7.0
10	-1	Coffee Berry	Café	[Kapitána nálepku 4, 920 01 Hlohovec, Slovensko]	Hlohovec	0.0	0.0	0.0	0.0	7.0	7.0
1	0	Brew Bar Café	Café	[Rožňavská 1A, 831 04 Bratislava, Slovensko]	Bratislava II	0.0	0.0	1.0	1.0	9.0	26.0
0	0	Urban Bistro	Café	[Michalská 5, 811 01 Bratislava, Slovensko]	Bratislava I	0.0	0.0	1.0	1.0	8.0	25.0
1	0	Street Cafe	Café	[Ivanská cesta 16, 821 04 Bratislava, Slovensko]	Bratislava II	0.0	0.0	1.0	1.0	5.0	22.0
1	0	Pán Králiček Priestor	Café	[Budovateľská 2, 821 08 Bratislava, Slovensko]	Bratislava II	0.0	0.0	1.0	1.0	0.0	17.0
1	0	MONO café	Coffee Shop	[Budovateľská 25, 821 08 Bratislava, Slovensko]	Bratislava II	0.0	0.0	1.0	1.0	0.0	17.0
0	0	Five Points	Café	[Panská 23, 811 03 Bratislava, Slovensko]	Bratislava I	0.0	0.0	0.0	1.0	8.0	16.5
1	0	Sweet Spot Café	Café	[Miletičova 90, 821 08 Bratislava, Slovensko]	Bratislava II	0.0	0.0	1.0	1.0	-1.0	16.0
0	0	Kava.Bar	Café	[Skalná 1 (Zámocká), 811 03 Bratislava, Slovensko]	Bratislava I	0.0	0.0	1.0	0.0	7.0	15.5
0	0	Kafe Haus Lumière	Café	[Špitálska 4, Bratislava, Slovensko]	Bratislava I	0.0	0.0	1.0	0.0	7.0	15.5
0	0	cafe-cafe	Café	[Rybárska Brána 9 (Hviezdoslavovo nám.), 811 01 Bratislava, Slovensko]	Bratislava I	0.0	0.0	1.0	0.0	7.0	15.5
1	0	Café & Café	Café	[Galvaniho 15/B (HP), 831 04 Bratislava, Slovensko]	Bratislava II	0.0	0.0	1.0	0.0	5.0	13.5
0	0	Martinus	Bookstore	[Obchodná 26 (Poštová), 811 06 Bratislava, Slovensko]	Bratislava I	0.0	0.0	0.0	0.0	9.0	9.0
0	0	W cafe	Café	[Grösslingová 62 (Karadžičova), 811 09 Bratislava, Slovensko]	Bratislava I	0.0	0.0	1.0	0.0	0.0	8.5

	cluster	clickable_url	venue.categories	venue.location.formattedAddress	NM3	covid_sp	covid_co	bc	ect	FSQ_rating	total_rating
0	0	Matsu premium tea & coffee	Café	[Špitálska 51, 811 08 Bratislava, Slovensko]	Bratislava I	0.0	0.0	1.0	0.0	0.0	8.5
0	0	Eleven Books & Coffee	Café	[Baštová 9 (Klariská), 811 03 Bratislava, Slovensko]	Bratislava I	0.0	0.0	1.0	0.0	0.0	8.5
0	0	Blue mondays	Café	[811 07 Bratislava, Slovensko]	Bratislava I	0.0	0.0	0.0	1.0	0.0	8.5
0	0	Green Cafe	Café	[Jozefská 17, 811 06 Bratislava, Slovensko]	Bratislava I	0.0	0.0	1.0	0.0	0.0	8.5
0	0	Bianco Café Bar	Café	[811 07 Bratislava, Slovensko]	Bratislava I	0.0	0.0	1.0	0.0	0.0	8.5
0	0	Good Times coffee roasters	Café	[Legionárska 9 (Jiskrova), Bratislava, Slovensko]	Bratislava I	0.0	0.0	1.0	0.0	0.0	8.5
0	0	KafeHaus BARZZUZ	Café	[Palackého 6, 811 02 Bratislava, Slovensko]	Bratislava I	0.0	0.0	1.0	0.0	0.0	8.5
1	0	Xocolat	Café	[Pribinova 8, 821 09 Bratislava, Slovensko]	Bratislava II	0.0	0.0	1.0	0.0	0.0	8.5
0	0	Škodovka	Café	[Medená 4 (Tobrucká), 841 06 Bratislava, Slovensko]	Bratislava I	0.0	0.0	0.0	0.0	8.0	8.0
0	0	Artforum	Bookstore	[Kozia 20 (Panenská), 811 03 Bratislava, Slovensko]	Bratislava I	0.0	0.0	0.0	0.0	8.0	8.0
0	0	Vespa Caffeteria	Café	[Laurinská 11, 811 01 Bratislava, Slovensko]	Bratislava I	0.0	0.0	0.0	0.0	8.0	8.0
0	0	Mačkafé klub	Pet Café	[Zámocká 7327/24, 811 01 Bratislava, Slovensko]	Bratislava I	0.0	0.0	0.0	0.0	8.0	8.0
0	0	Pinot u Bruna	Café	[Rudnayovo nám. 2, 811 01 Bratislava, Slovensko]	Bratislava I	0.0	0.0	0.0	0.0	8.0	8.0
0	0	Funki Punki Cafe	Café	[Klariská 12 (Kapucínska), 811 03 Bratislava, Slovensko]	Bratislava I	0.0	0.0	0.0	0.0	8.0	8.0
0	0	Pollito	Café	[Laurinská, 811 01 Bratislava, Slovensko]	Bratislava I	0.0	0.0	0.0	0.0	8.0	8.0
0	0	.klub pod lampou	Social Club	[Partizánska 2, 811 03 Bratislava, Slovensko]	Bratislava I	0.0	0.0	0.0	0.0	8.0	8.0
0	0	Foxford	Café	[Obchodná 26, 811 06 Bratislava, Slovensko]	Bratislava I	0.0	0.0	0.0	0.0	8.0	8.0
0	0	Caffe4U	Café	[Obchodna 42, 811 06 Bratislava, Slovensko]	Bratislava I	0.0	0.0	0.0	0.0	8.0	8.0
0	0	La Putika	Café	[Panská 237/12, 811 01 Bratislava, Slovensko]	Bratislava I	0.0	0.0	0.0	0.0	8.0	8.0
0	0	Next Apache	Café	[Panenská 28, 811 03 Bratislava, Slovensko]	Bratislava I	0.0	0.0	0.0	0.0	8.0	8.0
0	0	La Putika	Café	[Klobučnícka 442/4 (Nedbalova), 811 01 Bratislava, Slovensko]	Bratislava I	0.0	0.0	0.0	0.0	8.0	8.0

	cluster	clickable_url	venue.categories	venue.location.formattedAddress	NM3	covid_sp	covid_co	bc	ect	FSQ_rating	total_rating
0	0	Mondieu	Café	[Laurinská 7 (Uršulínska), 811 01 Bratislava, Slovensko]	Bratislava I	0.0	0.0	0.0	0.0	8.0	8.0
1	0	Porto	Café	[Kaštieľska 4, 821 05 Bratislava, Slovensko]	Bratislava II	0.0	0.0	0.0	0.0	8.0	8.0
1	0	La Putika	Café	[Trnavská 82 (Tomášikova), 821 02 Bratislava, Slovensko]	Bratislava II	0.0	0.0	0.0	0.0	8.0	8.0
1	0	Bar Zuz (KafeHaus Barzzuz)	Café	[Rožňavská 1A, 831 04 Bratislava, Slovensko]	Bratislava II	0.0	0.0	0.0	0.0	8.0	8.0
1	0	Centrum Rafael	Playground	[Narcisová 5, 821 03 Bratislava, Slovensko]	Bratislava II	0.0	0.0	0.0	0.0	8.0	8.0
2	0	Dobrodruh	Café	[Vajnorská 3, Bratislava, Slovensko]	Bratislava III	0.0	0.0	0.0	0.0	8.0	8.0
3	0	Foxford+Martinus	Bookstore	[Staré grunty 24 (Cubicon), 841 04 Bratislava, Slovensko]	Bratislava IV	0.0	0.0	0.0	0.0	8.0	8.0
4	0	Goio freshgallery cafe	Café	[Lužná 1, 851 04 Bratislava, Slovensko]	Bratislava V	0.0	0.0	0.0	0.0	8.0	8.0
4	0	Zmrzlina u Bajrama	Café	[Fedinova 16/A, 851 01 Bratislava, Slovensko]	Bratislava V	0.0	0.0	0.0	0.0	8.0	8.0
4	0	Kafé Lampy	Café	[Černyševského 3761, 851 01 Bratislava, Slovensko]	Bratislava V	0.0	0.0	0.0	0.0	8.0	8.0
0	0	Steinplatz	Café	[Kamenné nám.1 (Dunajská), 811 07 Bratislava, Slovensko]	Bratislava I	0.0	0.0	0.0	0.0	7.0	7.0
0	0	Cafe Del Via	Café	[Židovská 19, Bratislava, Slovensko]	Bratislava I	0.0	0.0	0.0	0.0	7.0	7.0
0	0	í Nonní | Cremeria	Café	[Laurinská 5, 811 01 Bratislava, Slovensko]	Bratislava I	0.0	0.0	0.0	0.0	7.0	7.0
0	0	ENJOY Coffee	Café	[Michalská 3, 811 01 Bratislava, Slovensko]	Bratislava I	0.0	0.0	0.0	0.0	7.0	7.0
0	0	10 prstov	Café	[Laurinská 10 (Gorkého 7), 811 01 Bratislava, Slovensko]	Bratislava I	0.0	0.0	0.0	0.0	7.0	7.0
0	0	Zeppelin Café & Souvenirs	Café	[Sedlárska 10, 811 06 Bratislava, Slovensko]	Bratislava I	0.0	0.0	0.0	0.0	7.0	7.0
0	0	Štúr Cafe	Café	[Štúrova 8, 811 02 Bratislava, Slovensko]	Bratislava I	0.0	0.0	0.0	0.0	7.0	7.0
0	0	Savage Garden	Café	[Nám. Slobody, 811 06 Bratislava, Slovensko]	Bratislava I	0.0	0.0	0.0	0.0	7.0	7.0
2	0	Montmartre Café Gallery	Café	[Halkova 1/A, 831 04 Bratislava, Slovensko]	Bratislava III	0.0	0.0	0.0	0.0	7.0	7.0

	cluster	clickable_url	venue.categories	venue.location.formattedAddress	NM3	covid_sp	covid_co	bc	ect	FSQ_rating	total_rating
2	0	Cafe Marcus	Café	[Pri starej pracharni, Bratislava, Slovensko]	Bratislava III	0.0	0.0	0.0	0.0	7.0	7.0
3	0	Kontajner Riviéra	Café	[Na Riviére 2, Bratislava, Slovensko]	Bratislava IV	0.0	0.0	0.0	0.0	7.0	7.0
14	1	Synagóga Café	Café	[Haulíková 2, 917 01 Trnava, Slovensko]	Trnava	0.0	0.0	1.0	1.0	8.0	25.0
14	1	ONé - espresso bar	Café	[Andreja Hlinku, 64, 917 01 Trnava, Slovensko]	Trnava	0.0	0.0	1.0	1.0	8.0	25.0
14	1	Leháro	Café	[Františkánska 18, 917 01 Trnava, Slovensko]	Trnava	0.0	0.0	1.0	0.0	9.0	17.5
14	1	Thalmeiner	Café	[Trojičné nám. 4, 917 01 Trnava, Slovensko]	Trnava	0.0	0.0	0.0	1.0	9.0	17.5
14	1	Rotunda Spiegelsaal	Café	[Kalinčiakova 14, 917 01 Trnava, Slovensko]	Trnava	0.0	0.0	1.0	0.0	8.0	16.5
14	1	Street Cafe	Café	[Trstínska cesta, Trnava, Slovensko]	Trnava	0.0	0.0	1.0	1.0	-1.0	16.0
14	1	Bábovka	Café	[Divadelna 6, 917 01 Trnava, Slovensko]	Trnava	0.0	0.0	1.0	0.0	7.0	15.5
14	1	Amnesia Trnava	Café	[Štafánikova 26, 917 01 Trnava, Slovensko]	Trnava	0.0	0.0	1.0	0.0	0.0	8.5
14	1	Bezkydov	Café	[2 Trojičné námestie, 917 01 Trnava, Slovensko]	Trnava	0.0	0.0	1.0	0.0	0.0	8.5
14	1	Betonka kaviareň	Café	[Kollárova 20 (Hlboká), 917 01 Trnava, Slovensko]	Trnava	0.0	0.0	0.0	0.0	8.0	8.0
14	1	Amnesia	Café	[Štefanikova 26, 917 01 Trnava, Slovensko]	Trnava	0.0	0.0	0.0	0.0	7.0	7.0
6	2	HOLLERUNG káva & koloniál	Café	[Harmónia 3001 (Okružná), 900 01 Modra, Slovensko]	Pezinok	0.0	0.0	1.0	1.0	7.0	24.0
6	2	Cukráreň Roma	Café	[Štúrova 80, 900 01 Modra, Slovensko]	Pezinok	0.0	0.0	0.0	0.0	7.0	7.0
6	3	.BLACK - Coffee & Wine	Café	[Holubyho 16, 902 01 Pezinok, Slovensko]	Pezinok	0.0	0.0	1.0	1.0	0.0	17.0
6	3	Pezinská Kaviareň	Café	[902 01 Pezinok, Slovensko]	Pezinok	0.0	0.0	1.0	0.0	0.0	8.5
6	3	CoffeeShopa	Café	[SNP 570, 900 91 Limbach, Slovensko]	Pezinok	0.0	0.0	1.0	0.0	0.0	8.5
6	3	Čokokafé	Café	[Horné predmestie 214/12, 900 21 Svätý Jur, Slovensko]	Pezinok	0.0	0.0	0.0	0.0	8.0	8.0
6	3	Mlsná Emma	Chocolate Shop	[M.R.Štefánika 2, 902 01 Pezinok, Slovensko]	Pezinok	0.0	0.0	0.0	0.0	8.0	8.0
6	3	Topper Club	Café	[Myslenická 2/C, 902 01 Pezinok, Slovensko]	Pezinok	0.0	0.0	0.0	0.0	7.0	7.0
6	3	Cafe PORTA	Café	[Rulandska, Svätý Jur, Slovensko]	Pezinok	0.0	0.0	0.0	0.0	7.0	7.0
11	4	Zuckmann villa	Café	[Winterova 56, 921 01 Piešťany, Slovensko]	Piešťany	0.0	0.0	1.0	0.0	7.0	15.5

	cluster	clickable_url	venue.categories	venue.location.formattedAddress	NM3	covid_sp	covid_co	bc	ect	FSQ_rating	total_rating
11	4	The CUP cafe & home	Café	[The CUP cafe & home (Námestie J. Murgaša 3), 921 01 Piešťany, Slovensko]	Piešťany	0.0	0.0	1.0	0.0	7.0	15.5
11	4	Colonial café	Café	[Slovensko]	Piešťany	0.0	0.0	1.0	0.0	0.0	8.5
11	4	Classic Coffee - Pražiareň	Café	[Námestie 1.mája 1, 921 01 Piešťany, Slovensko]	Piešťany	0.0	0.0	0.0	0.0	8.0	8.0
11	4	Heaven Caffe&Bar	Café	[Sad Andreja Kmeťa 76, 921 01 Piešťany, Slovensko]	Piešťany	0.0	0.0	0.0	0.0	8.0	8.0
11	4	Monsalvy	Café	[Teplická 6, 921 01 Piešťany, Slovensko]	Piešťany	0.0	0.0	0.0	0.0	7.0	7.0
11	4	Glacio	Café	[Nitrianska 7555/18, 921 01 Piešťany, Slovensko]	Piešťany	0.0	0.0	0.0	0.0	7.0	7.0
3	5	Favor.it	Café	[Bridlicová 17, 841 97 Bratislava, Slovensko]	Bratislava IV	0.0	0.0	1.0	0.0	7.0	15.5
3	5	Ahoy Cafe	Café	[Slovanské nábr.46, Bratislava, Slovensko]	Bratislava IV	0.0	0.0	0.0	0.0	8.0	8.0
3	5	McDonald's & McCafé	Café	[Lamačská brána 541/664, 841 03 Bratislava, Slovensko]	Bratislava IV	0.0	0.0	0.0	0.0	7.0	7.0
3	5	Mondieu	Café	[Lamačská brána 541/664, 841 03 Bratislava, Slovensko]	Bratislava IV	0.0	0.0	0.0	0.0	7.0	7.0
3	5	Cafe Eden	Café	[Hradná 5, 841 10 Devín, Slovensko]	Bratislava IV	0.0	0.0	0.0	0.0	7.0	7.0
5	6	KRALIKOVA Café & Cakes	Café	[Sasinkova 2, 901 01 Malacky, Slovensko]	Malacky	0.0	0.0	1.0	0.0	7.0	15.5
5	6	Eis Cafe Delikana	Café	[Pezinská 5647/11, 901 01 Malacky, Slovensko]	Malacky	0.0	0.0	0.0	0.0	7.0	7.0
5	6	Break Caffe	Café	[Klaštorné nám. 2, 901 01 Malacky, Slovensko]	Malacky	0.0	0.0	0.0	0.0	7.0	7.0
8	7	Wagner	Café	[Hlavná 28/7, 929 01 Dunajská Streda, Slovensko]	Dunajská Streda	0.0	0.0	0.0	0.0	7.0	7.0
8	7	Buena Cafe	Café	[Dunajská Streda, Slovensko]	Dunajská Streda	0.0	0.0	0.0	0.0	7.0	7.0

Lets prepare final report on map

Prepare map with feature groups that will let user choose which layer/feature group turn on or off. Prepare customized label with informations about coffe shop. Save map.

In [75]:

```
from folium.features import DivIcon
f = folium.Figure(width=800, height=400)
m = folium.Map(location=[latitude, longitude], zoom_start=8)

choro = folium.Choropleth(cafe_gpd, data=cafe_gpd,
                  name='Regional boundaries',
                  key_on='feature.properties.NM3',
                  threshold_scale=[0,100, 200,300],
                  columns=['NM3', 'IDN3'],
                  fill_color='YlGnBu'
                  )
# remove legend from map
for key in choro._children:
    if key.startswith('color_map'):
        del(choro._children[key])
choro.add_to(m)

fg_districts = folium.FeatureGroup(name='Districts centers')
for lat, lon, name , address in zip(cafe_regions['latitude'], cafe_regions['longitude'], cafe_regions['NM3'], cafe_regions['address']):
    pp= folium.Html('<h3>' + str(name) + '</h3>' + '<p>' + str(address) + '</p>', script=True)
    label = folium.Popup(pp, max_width=200)
    folium.CircleMarker(
        [lat, lon],
        radius=2,
        popup=label,
        color='darkred',
        fill=True,
        fill_color='darkred',
        fill_opacity=0.7).add_to(fg_districts)

# add markers to the map
fg_districts.add_to(m)

fg_clusters = folium.FeatureGroup(name='Clusters')

for clust_number in set(labels):
    clust_set = venues_top_n[venues_top_n.cluster == clust_number]
    for index,clust_member in clust_set.iterrows():
        pp= folium.Html('<h2>' + str(clust_number) + '</h2>', script=True)
        label = folium.Popup(pp, max_width=100)
        folium.CircleMarker(
            [clust_member['venue.location.lat'], clust_member['venue.location.lng']],
```

```
            radius=8,
            popup=label,
            color = clust_member['marker_color'],
            fill=True,
            fill_color=clust_member['marker_color'],
            opacity=0.2,
            fill_opacity=0.2).add_to(fg_clusters)

fg_clusters.add_to(m)


fg_cafes = folium.FeatureGroup(name='Cafes')
for lat, lon, cat, poi, address, vid, total_rating, covid_sp, covid_co, bc, ect, FSQ_rating in zip(venues_top_n['venue.location.lat'], venues
                                                                , venues_top_n['venue.location.formattedAddress'],venues_top_n['venue.id'],venues_t
                                                                 ,venues_top_n['covid_sp'] ,venues_top_n['covid_co'] ,venues_top_n['bc'] ,venues_to

                                                                                                                                                  ):

    link = str('https://foursquare.com/v/' + poi + '/'  + vid)
    pp= folium.Html('<h2>' + '<a href="'+ link +'"target="_blank">'+ str(poi) + '</a>' + '</h2>' + '<p>' + 'Category :' + '<b>' + str(cat) +
                    '<br>' + 'Total rating :' + '<b>' + str(total_rating) + '</b>'+
                    '<br>' + 'FSQ rating :' + str(FSQ_rating) +
                    '<br>' + 'COVID map specialty :' + str(covid_sp) +
                    '<br>' + 'COVID map comodity :' + str(covid_co) +
                    '<br>' + 'black check guide:' + str(bc) +
                    '<br>' + 'european coffee trip :' + str(ect) +

                    '</p>', script=True)
    label = folium.Popup(pp, max_width=200)
    #folium.CircleMarker(
    folium.Marker(
        [lat, lon],
        radius=5,
        popup=label,
       # icon=folium.Icon(color='black',icon_color='#FFFF00'),
        icon=folium.Icon(color='black',icon_color='#8B4513')).add_to(fg_cafes)

fg_cafes.add_to(m)


folium.LayerControl().add_to(m)
f.add_child(m)

f
```

Out[75]:

In [61]:

```
# Save map to html
f.save('output\\coffe_shops_map.html')
```

End of analysis

This concludes our analysis. We prepared two reports (table and map) with Top 100 venue candidates and details about each candidate. Data for this reports are stored in pickles which speeds up analysis and also give chance for stakeholders to fine tune weights for final scoring, parameters for clustering or change number of selected final candidates. URL Link for detailed info about venue also helps stakeholder with final decisions which venues are good candidates for starting communication or do some business trip.

Results and Discussion

Our analysis shows that although there are lots of coffee shops in selected region, not all of them have FSQ rating or have high rating. Most of selected candidates are located in Bratislava region which is capital city of Slovakia. There are also some candidates with high total rating in Trnava region, but number of

this candidates is much lower. In this analysis we were focused on Trnava and Bratislava region. There were two reason for this - one of stakeholders interested in output of this analysis - coffee roaster is located near this region, second - there are restriction in number of calls of FSQ API per day. If there will be more stakeholders, it will make sense to pay for premium API calls and do this analysis on whole Slovakia. All code in this analysis is prepared that way that this kind of analysis can be done . We think that output of this analysis is good starting point for stakeholder to choose protentional clients and to start to communicate with them .

Conclusion

Purpose of this project was to identify top coffee shops in Bratislava(Bratislavsky kraj) and Trnava (Trnavsky kraj) region in Slovakia in order to help stakeholders - specialty coffee roasters - in narrowing down the search for potential partners for whom specialty coffee is/could be a unique selling point. By merging data from many resources and calculating weighted score we identified top candidates. Using clustering we prepared groups of candidates based on location which helps stakeholders with business trip planning. Business trip are needed for further negotiations with potential partners as in this market face-to-face communication is the must.

Prepared reports can be used as starting points for final exploration by stakeholders.

Final decision will be made by stakeholders based on their preferences and additional factors like distance from roastery/travel time, social and economic dynamics of region where is coffee shop located.