Coursera Capstone Project

The Battle of Neighborhoods

Author:Roman Mayerson

(1)Introduction/Business Problem ¶

A stakeholder is interested in starting a new buisness in the City of Riga(Latvia). He thinks about two options:an 'Irish Pub' or 'Italian Restaraunt'.The city has a number of Neighbourhoods So he also want to find a best possible location for his pub\restaraunt.He decides to rely on Data Analysis Methodologies to choose a specific type of buisness and it's location.

(2)Data

We will need a data regarding the venues ,geographic locations for every Neighbourhood in order to apply our methodologies for Descriptive Analysis and Segmentation Web scraping and Foursquare API will be used to generate this Data We will work with resulted data sets and will perform different manipulations on data in order to make conclusions for our Final Report

Importing packages and installing Libraries

In [3]:
```
import pandas as pd
import requests
from pandas.io.json import json_normalize # tranform JSON file into a pandas dataframe
from geopy.geocoders import Nominatim
!conda install -c conda-forge folium=0.5.0 --yes
import folium # map rendering library
```

```
Solving environment: done

## Package Plan ##

  environment location: /opt/conda/envs/Python36

  added / updated specs:
    - folium=0.5.0


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    vincent-0.4.4              |             py_1          28 KB  conda-forge
    certifi-2019.6.16          |           py36_1         149 KB  conda-forge
    ca-certificates-2019.6.16  |       hecc5488_0         145 KB  conda-forge
    folium-0.5.0               |             py_0          45 KB  conda-forge
    altair-3.2.0               |           py36_0         770 KB  conda-forge
    openssl-1.1.1c             |       h516909a_0         2.1 MB  conda-forge
    branca-0.3.1               |             py_0          25 KB  conda-forge
    ------------------------------------------------------------
                                           Total:         3.3 MB

The following NEW packages will be INSTALLED:

    altair:          3.2.0-py36_0      conda-forge
    branca:          0.3.1-py_0        conda-forge
    folium:          0.5.0-py_0        conda-forge
    vincent:         0.4.4-py_1        conda-forge

The following packages will be UPDATED:

    ca-certificates: 2019.5.15-0                 --> 2019.6.16-hecc5488_0 conda-forge
    certifi:         2019.6.16-py36_1            --> 2019.6.16-py36_1     conda-forge

The following packages will be DOWNGRADED:

    openssl:         1.1.1c-h7b6447c_1           --> 1.1.1c-h516909a_0    conda-forge


Downloading and Extracting Packages
vincent-0.4.4        | 28 KB     | ##################################### | 100%
certifi-2019.6.16    | 149 KB    | ##################################### | 100%
ca-certificates-2019 | 145 KB    | ##################################### | 100%
folium-0.5.0         | 45 KB     | ##################################### | 100%
altair-3.2.0         | 770 KB    | ##################################### | 100%
openssl-1.1.1c       | 2.1 MB    | ##################################### | 100%
branca-0.3.1         | 25 KB     | ##################################### | 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
```

Importing a dataset of Riga's Boroughs,obtained through Web Scraping from Wikipedia Page

https://en.wikipedia.org/wiki/Administrative_divisions_of_Riga

In [4]:
```
import types
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
# You might want to remove those credentials before you share your notebook.
client_b8a1b766aa0744c8895891e0a15f83dc = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx',
    ibm_auth_endpoint="https://iam.eu-gb.bluemix.net/oidc/token",
```

```
    endpoint_url='https://s3.eu-geo.objectstorage.service.networklayer.com')

body = client_b8a1b766aa0744c8895891e0a15f83dc.get_object(Bucket='segmentingandclusteringneighborho-donotdelete-pr-hforf89holr8ns',Key='Riga_Neighborhoods.csv')['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__, body )

riga_boroughs = pd.read_csv(body)
riga_boroughs.head()
```

Out[4]:

	Neighbourhood	lat	lng
0	Central District	56.951100	24.118300
1	Kurzeme District	56.996623	24.029045
2	Latgale Suburb	56.920000	24.197200
3	Northern District	56.948889	24.106389
4	Vidzeme Suburb	56.966700	24.216700

Foursquare API related info

In [45]:
```
CLIENT_ID = 'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx' # your Foursquare ID
CLIENT_SECRET = 'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx' # your Foursquare Secret
VERSION = '20180605' # Foursquare API version

print('Your credentails:')
print('CLIENT_ID: ' + CLIENT_ID)
print('CLIENT_SECRET:' + CLIENT_SECRET)
# type your answer here

LIMIT = 10000 # limit of number of venues returned by Foursquare API

radius = 100000 # define radius
```

```
Your credentails:
CLIENT_ID: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CLIENT_SECRET:xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

Exploring Neighborhoods in Riga

Creating a function to repeat the same process to all the neighborhoods in Riga

In [6]:
```
def getNearbyVenues(names, latitudes, longitudes, radius=10000):

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

        # create the API request URL
        url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={}&v={}&ll={},{}&radius={}&limit={}'.format(
            CLIENT_ID,
            CLIENT_SECRET,
            VERSION,
            lat,
            lng,
            radius,
            LIMIT)

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

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

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

    return(nearby_venues)
```

Now let's write the code to run the above function on each borough and create a new dataframe called riga_venues

In [7]:
```
riga_venues = getNearbyVenues(names=riga_boroughs['Neighbourhood'],
                              latitudes=riga_boroughs['lat'],
                              longitudes=riga_boroughs['lng']
                             )
```

```
Central District
Kurzeme District
Latgale Suburb
Northern District
Vidzeme Suburb
Zemgale Suburb
```

In [8]:
```
print(riga_venues.shape)
riga_venues.head()
```

```
(600, 7)
```

Out[8]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Central District	56.9511	24.1183	Vērmanes dārzs	56.951254	24.118834	Park
1	Central District	56.9511	24.1183	ESPA	56.955375	24.117737	Spa
2	Central District	56.9511	24.1183	Radisson Blu Elizabete	56.952151	24.120593	Hotel

2	Central District	56.9511	24.1183	Hotel	56.952151	24.120383	Hotel
3	Central District	56.9511	24.1183	Bastejkalns	56.951900	24.110685	Park
4	Central District	56.9511	24.1183	Splendid Palace	56.953591	24.119222	Movie Theater

Let's check how many venues were returned for each borough

In [9]:
```
riga_venues.groupby('Neighborhood').count()
```

Out[9]:

	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
Neighborhood						
Central District	100	100	100	100	100	100
Kurzeme District	100	100	100	100	100	100
Latgale Suburb	100	100	100	100	100	100
Northern District	100	100	100	100	100	100
Vidzeme Suburb	100	100	100	100	100	100
Zemgale Suburb	100	100	100	100	100	100

Let's find out how many unique categories can be curated from all the returned venues

In [10]:
```
print('There are {} uniques categories.'.format(len(riga_venues['Venue Category'].unique())))
```

```
There are 106 uniques categories.
```

Analyze Each Borough

In [11]:
```
# one hot encoding
riga_onehot = pd.get_dummies(riga_venues[['Venue Category']], prefix="", prefix_sep="")

# add neighborhood column back to dataframe
riga_onehot['Neighborhood'] = riga_venues['Neighborhood']

# move neighborhood column to the first column
fixed_columns = [riga_onehot.columns[-1]] + list(riga_onehot.columns[:-1])
riga_onehot = riga_onehot[fixed_columns]
riga_grouped = riga_onehot.groupby('Neighborhood').mean().reset_index()
riga_grouped
```

Out[11]:

	Neighborhood	Adult Boutique	Art Gallery	Art Museum	Asian Restaurant	Athletics & Sports	BBQ Joint	Bagel Shop	Bakery	Bar	...	Theater	Tourist Information Center	Toy / Game Store	Turkish Restaurant	Vegeta / V Restau
0	Central District	0.01	0.00	0.01	0.01	0.01	0.00	0.01	0.01	0.07	...	0.03	0.00	0.00	0.01	0.02
1	Kurzeme District	0.00	0.00	0.01	0.01	0.01	0.00	0.00	0.02	0.02	...	0.02	0.00	0.00	0.02	0.00
2	Latgale Suburb	0.00	0.00	0.01	0.01	0.01	0.00	0.00	0.01	0.05	...	0.03	0.01	0.00	0.00	0.01
3	Northern District	0.01	0.02	0.02	0.01	0.01	0.00	0.00	0.01	0.06	...	0.03	0.00	0.00	0.01	0.00
4	Vidzeme Suburb	0.01	0.00	0.01	0.01	0.00	0.00	0.00	0.01	0.05	...	0.02	0.01	0.01	0.01	0.01
5	Zemgale Suburb	0.00	0.01	0.01	0.00	0.01	0.01	0.00	0.03	0.04	...	0.02	0.00	0.00	0.00	0.00

6 rows × 107 columns

Let's print each neighborhood along with the top 5 most common venues

In [12]:
```
num_top_venues = 5

for hood in riga_grouped['Neighborhood']:
    print("----"+hood+"----")
    temp = riga_grouped[riga_grouped['Neighborhood'] == hood].T.reset_index()
    temp.columns = ['venue','freq']
    temp = temp.iloc[1:]
    temp['freq'] = temp['freq'].astype(float)
    temp = temp.round({'freq': 2})
    print(temp.sort_values('freq', ascending=False).reset_index(drop=True).head(num_top_venues))
    print('\n')
```

```
----Central District----
                       venue  freq
0                      Hotel  0.07
1                        Bar  0.07
2                       Park  0.06
3                 Restaurant  0.06
4  Eastern European Restaurant  0.04


----Kurzeme District----
      venue  freq
0      Park  0.08
1     Hotel  0.06
2      Café  0.05
3     Beach  0.04
4  Restaurant  0.04


----Latgale Suburb----
                venue  freq
0                Park  0.08
1  Gym / Fitness Center  0.06
2                 Bar  0.05
3                 Gym  0.05
4               Hotel  0.04


----Northern District----
                       venue  freq
0                      Hotel  0.08
1                        Bar  0.06
```

```
2                  Restaurant  0.06
3                        Park  0.06
4  Eastern European Restaurant  0.05


----Vidzeme Suburb----
                venue  freq
0                Park  0.07
1  Gym / Fitness Center  0.05
2                 Bar  0.05
3                 Gym  0.04
4          Restaurant  0.04


----Zemgale Suburb----
                        venue  freq
0                        Park  0.09
1                       Hotel  0.07
2  Eastern European Restaurant  0.05
3                  Restaurant  0.04
4                         Bar  0.04
```

From this Quick Summary we already can make a quick assumption that perfect location for our Pub\Restaraunt will be 'Central' or 'Northern' Districts

So at first let's explore these Districts to see how many Pubs\Italian Restaraunts they have

In [13]:
```
central_venues=riga_venues[riga_venues['Neighborhood']=='Central District']
northern_venues=riga_venues[riga_venues['Neighborhood']=='Northern District']
```

In [14]:
```
central_venues=central_venues[['Venue','Venue Latitude','Venue Longitude','Venue Category']]
central_venues.head()
northern_venues=northern_venues[['Venue','Venue Latitude','Venue Longitude','Venue Category']]
northern_venues.head()
```

Out[14]:

	Venue	Venue Latitude	Venue Longitude	Venue Category
300	Doma laukums	56.949445	24.105763	Plaza
301	Rātslaukums	56.947552	24.106681	Plaza
302	Bastejkalns	56.951900	24.110685	Park
303	11. novembra krastmalā starp Akmens un Vanšu	56.948814	24.101963	Athletics & Sports
304	Neiburgs	56.948525	24.105513	Restaurant

In [35]:
```
central_bars_pubs=central_venues[central_venues["Venue Category"].str.contains("Bar")]

central_bars_pubs=central_bars_pubs[central_bars_pubs["Venue Category"] != "Salon / Barbershop"]
```

In [36]:
```
northern_bars_pubs=northern_venues[northern_venues["Venue Category"].str.contains("Bar")]

northern_bars_pubs=northern_bars_pubs[northern_bars_pubs["Venue Category"] != "Salon / Barbershop"]
```

In [37]:
```
central_bars_pubs.append(central_bars_pubs[central_bars_pubs["Venue Category"].str.contains("Pub")])
```

Out[37]:

	Venue	Venue Latitude	Venue Longitude	Venue Category
16	Aleponija	56.949359	24.130570	Wine Bar
17	Easy Wine	56.947118	24.110989	Wine Bar
27	Alus Muiža	56.953490	24.129706	Bar
38	Miezis un kompānija. Vecrīga	56.945309	24.112040	Bar
39	ON AIR Café	56.952189	24.125598	Bar
50	Folkklubs Ala Pagrabs	56.946582	24.107327	Bar
51	Gauja	56.956065	24.128593	Bar
56	B-bārs Restorāns	56.949189	24.103901	Cocktail Bar
69	Balzambārs	56.951636	24.106230	Cocktail Bar
74	Vīna Studija	56.957943	24.110697	Wine Bar
79	Garāža	56.951040	24.106400	Bar
82	Č	56.951414	24.123301	Bar

In [38]:
```
northern_bars_pubs.append(northern_venues[northern_venues["Venue Category"].str.contains("Pub")])
```

Out[38]:

	Venue	Venue Latitude	Venue Longitude	Venue Category
305	Easy Wine	56.947118	24.110989	Wine Bar
306	B-bārs Restorāns	56.949189	24.103901	Cocktail Bar
310	Folkklubs Ala Pagrabs	56.946582	24.107327	Bar
323	Miezis un kompānija. Vecrīga	56.945309	24.112040	Bar
327	Balzambārs	56.951636	24.106230	Cocktail Bar
330	Garāža	56.951040	24.106400	Bar
366	Aleponija	56.949359	24.130570	Wine Bar
372	Vīna Studija	56.957943	24.110697	Wine Bar
383	Alus Muiža	56.953490	24.129706	Bar
389	ON AIR Café	56.952189	24.125598	Bar
396	Gauja	56.956065	24.128593	Bar
349	Aussie Backpackers Pub	56.946791	24.113965	Pub

In [23]:
```
central_italian=central_venues[central_venues["Venue Category"].str.contains("Italian")]

central_italian
```

Out[23]:

	Venue	Venue Latitude	Venue Longitude	Venue Category

36	Italissimo	56.957697	24.120253	Italian Restaurant

In [24]:
```
northern_italian=northern_venues[northern_venues["Venue Category"].str.contains("Italian")]

northern_italian
```

Out[24]:

	Venue	Venue Latitude	Venue Longitude	Venue Category
377	Portofino Rīga	56.961185	24.100187	Italian Restaurant
378	Italissimo	56.957697	24.120253	Italian Restaurant

By analyzing the frequencies of occurrences of Italian restaurants and Pubs\Bars in Riga , we can see that the Riga has in average more Pubs\Bars than Italian Restaurants in total.

In [25]:
```
northern_italian.count()
```

Out[25]:
```
Venue              2
Venue Latitude     2
Venue Longitude    2
Venue Category     2
dtype: int64
```

In [39]:
```
northern_bars_pubs.count()
```

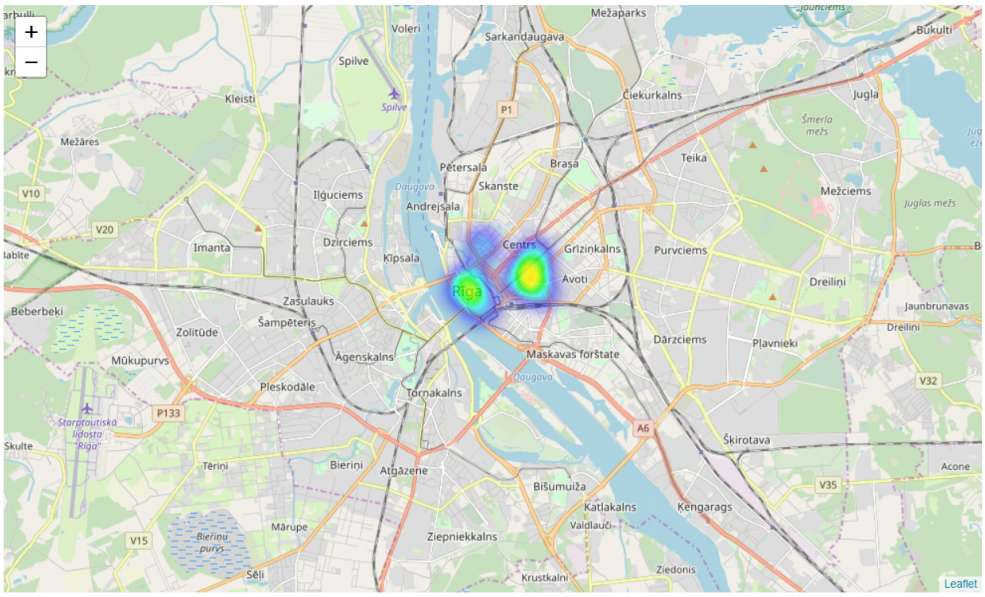
Out[39]:
```
Venue              11
Venue Latitude     11
Venue Longitude    11
Venue Category     11
dtype: int64
```

As we can see from map below Riga have high concentration of "drinking" Areas in Central Part in "Old City" Area.This can be Perfect Place for a brand new Irish Pub

In [40]:
```
bars=central_bars_pubs.append(northern_bars_pubs)
```

In [43]:
```
import numpy as np # linear algebra
from folium.plugins import HeatMap
pubs_map = folium.Map(location=[56.9479542,24.1141862], zoom_start=12)
data = [[x[0], x[1], 1] for x in np.array(bars[['Venue Latitude', 'Venue Longitude']])]
HeatMap(data, radius = 20).add_to(pubs_map)
pubs_map
```

Out[43]:



Discussion:

In General Pubs\Bars are much more popular in the city of Riga than Italian Restaurants.So there are already a large client base for such type of buisness. There almost no "Irish" Pubs.There are a couple but Foursquare engine was unable to find them at all.So i belive "Irish" style Pub can be very popular in this area because it is very unique It should provide a very high quality service of course

Conclusion：

A stakeholder should open an "Irish Pub" in the Central\North\Old City Area of Riga. It should have a Big Success.

In [ ]:
```
```