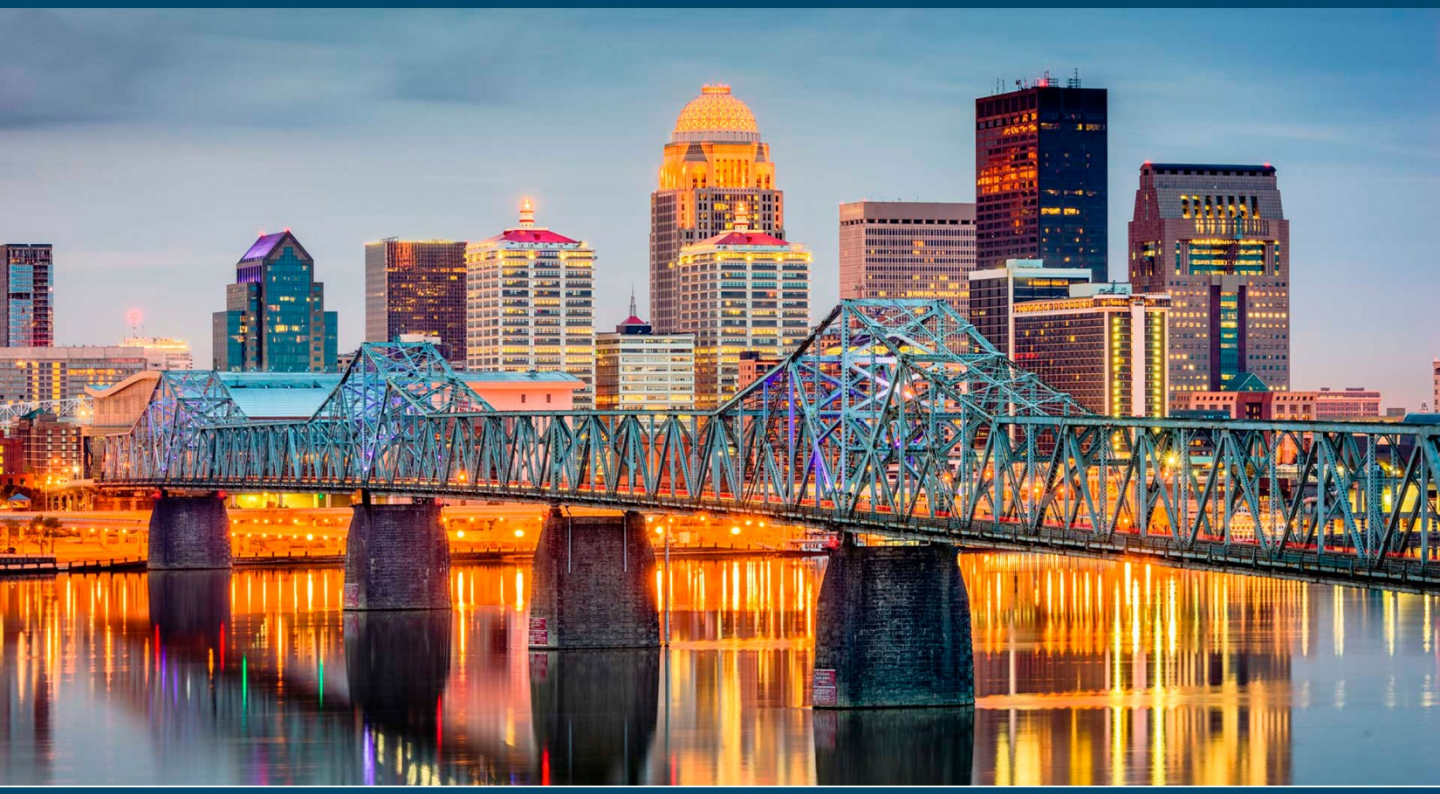
Capstone Project

The Battle of Neighborhoods



Find a Better Location for openning a Food and Beverage business in Louisville, Kentucky

By Vladimir Virguez

September, 2020

Table of Contents

1. Introduction

The purpose of this project is to help new entrepreneurs who want to make better decisions regarding a better location for openning any business place in the city of Louisville, Kentucky state in USA.

Before starting a business, it is necessary to carry out an analysis of features and facilities that the different neighborhoods of a city could offer. This project is for those people who are looking for safe and better neighborhoods in order to open a commercial establishments. Specifically, this report will be targeted to stakeholders interested in opening a food and beverage businesses in Louisville City, usa.

1.1 Background

Louisville Jefferson County is the largest city in the Commonwealth of Kentucky and the 29th most-populous city in the United States. It is one of two cities in Kentucky designated as first-class, is a significant manufacturing, with two major Ford Motor Company plants, and the headquarters and major home appliance factory of GE Appliances, is also a major center of the American Whiskey industry, with about one third of all bourbon whiskey comming from Louisville. This city prides itself in its large assortmen of small, independent businesses and restaurants, some of which have become known for their ingenuity and creativity. This city is a place that offers many opportunities to invest since the purchasing power of citizens is very favorable.

1.3 Target Audience

- ✓ A business entrepreneurs that want to open a new restaurant in Louisville City of Kentucky State.
- ✓ Someone curious about data have an idea, how to beneficial it is to open a restaurant and what area the pros and cons.
- ✓ Business Analyst or Data Scientists, who wish to analyze the neighborhoods of Louisville using some machine learning techniques

2. Data Section

Based on definition of the problem to solve, factors that may be relevants to make our decision are:

- ✓ Demographic information, e.g. population, density, education, age, income
- ✓ Louisville data containing neighborhoods and boroughs, latitudes, and longitudes, extracted from Wikipedia (Wikipedia) and Geodatos. In this section is necessary to clean the data in order to organize in a correct way
- ✓ Restaurants, shopping malls in every neighborhood
- ✓ Coordinate of Louisville obtained using https://geodatos.net
- ✓ It is necessary to use **Geopy** to get geological location by address name
- ✓ To get the most common venues of Borough of Louisville is necesary to use **Foursquare API**
- ✓ To get the venues' record we use Foursquare API
- ✓ **Folium:** this library is used to show Louisville boundary in the Folium map
- ✓ During the realization of the project, it is likely that it will be necessary to use other python libraries so that the data is adequate for better results.

3. Methodology Section

The purpose of this section is to develop the main component of the report where we discuss and describe any exploratory data analysis, and what machine learnings were used.

3.1 Download and Explore Dataset
3.2 Explore Neighborhoods in Louisville Ky
3.3 Analyze Each Neighborhoods
3.4 Cluster Neighborhoods

Before obtaining the data and starting to explore it was necessary to download all the python packages that are the tools that will help me perform the analysis.

3.1 Download and Explore Dataset

Fortunately the data can be collected from the Luisville project (LOJIC) of open Geospatial data.Louisville(LOJIC) Open GeoSpatial Data.

```
[25]: louisville_df = gpd.read_file('https://opendata.arcgis.com/datasets/6e3dea8bd9cf49e6a764f7baa9141a95_30.geojson')
      louisville_df.head()
```

[25]:

	OBJECTID	NH_CODE	NH_NAME	SHAPEAREA	SHAPELEN	geometry
0	1	99	REMAINDER OF CITY	2.558333e+07	23718.323690	POLYGON ((-85.70282 38.27046, -85.70268 38.270...
1	2	53	PORTLAND	7.009830e+07	43851.334658	POLYGON ((-85.82022 38.27686, -85.81992 38.276...
2	3	62	SHAWNEE	5.995385e+07	34088.057055	POLYGON ((-85.82022 38.27686, -85.82028 38.276...
3	4	12	BROWNSBORO ZORN	2.197732e+07	28060.141437	POLYGON ((-85.70282 38.27046, -85.70278 38.270...
4	5	23	CLIFTON HEIGHTS	1.785864e+07	19464.933453	POLYGON ((-85.71501 38.26404, -85.71394 38.263...

After obtaining the data, we proceeded to prepare and preprocess to extract the useful information to develop the analysis.

Data Preparation and preprocessing

```
[27]: louisville_N = louisville_df[['NH_CODE', 'NH_NAME', 'geometry']].copy()
      louisville_N.columns = ('Id', 'Neighborhood', 'Geometry')
      louisville_N['Neighborhood'] = louisville_N['Neighborhood'].str.title()
      louisville_N.tail()
```

[27]:

	Id	Neighborhood	Geometry
87	40	Iroquois	POLYGON ((-85.78123 38.16980, -85.78121 38.169...
88	66	Southland Park	POLYGON ((-85.75183 38.17000, -85.75189 38.169...
89	43	Kenwood Hill	POLYGON ((-85.77918 38.16163, -85.77915 38.161...
90	2	Auburndale	POLYGON ((-85.79126 38.15293, -85.79130 38.152...
91	99	Remainder Of City	POLYGON ((-85.77631 38.15065, -85.77630 38.150...

For the tasks that follow it was necessary to use a function to adapt two new columns of Latitude and longitude.

[29]:
```
#Extract the Longitude and Latitude
from shapely.wkt import loads as load_wkt

id_list = []

for polygon in louisville_df["geometry"]:
    box_str = str(polygon)
    p1 = load_wkt(box_str)
    point = p1.centroid
#     print(type(p1.centroid.x))
#     print(p1.centroid.y)
    id_list.append((p1.centroid.y, p1.centroid.x))

lat_centr, lon_centr = zip(*id_list)

louisville_N['Latitude'] = lat_centr
louisville_N['Longitude'] = lon_centr
louisville_N.head()
```

And as a result the following was obtained

[29]:

	Id	Neighborhood	Geometry	Latitude	Longitude
0	99	Remainder Of City	POLYGON ((-85.70282 38.27046, -85.70268 38.270...	38.272222	-85.708712
1	53	Portland	POLYGON ((-85.82022 38.27686, -85.81992 38.276...	38.268303	-85.792998
2	62	Shawnee	POLYGON ((-85.82022 38.27686, -85.82028 38.276...	38.261047	-85.818672
3	12	Brownsboro Zorn	POLYGON ((-85.70282 38.27046, -85.70278 38.270...	38.266630	-85.688639
4	23	Clifton Heights	POLYGON ((-85.71501 38.26404, -85.71394 38.263...	38.263401	-85.704026

3.2 Explore Neighborhoods in Louisville KY

To explore the data, I will use the "Folium" python Library to create a map of the Louisville City

```
address = 'Louisville, KY'

geolocator = Nominatim(user_agent="to_explorer")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print('The Geographical Coordinate of Louisville are {}, {}.'.format(latitude, longitude))
```

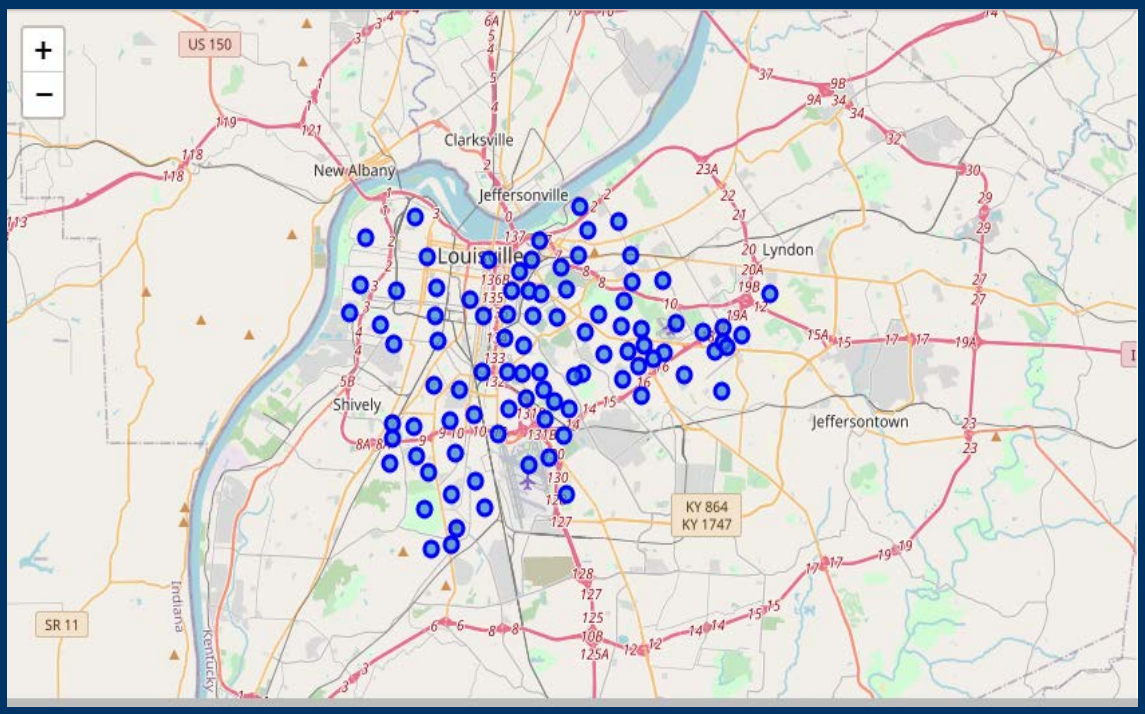
```
The Geographical Coordinate of Louisville are 38.2542376, -85.759407.
```

It is important to create a visualization to better understand the area

```
[6]: louisville_map = folium.Map(location=[latitude, longitude], zoom_start=12)

     # add markers to map
     for lat, lng, borough, neighborhood in zip(louisville_N['Latitude'],
                                                louisville_N['Longitude'],
                                                louisville_N['Id'],
                                                louisville_N['Neighborhood']):
         label = '{}, {}'.format(neighborhood, borough)
         label = folium.Popup(label, parse_html=True)
         folium.CircleMarker(
             [lat, lng],
             radius=5,
             popup=label,
             color='blue',
             fill=True,
             fill_color='#3186cc',
             fill_opacity=0.7,
             parse_html=False).add_to(louisville_map)

     louisville_map
```

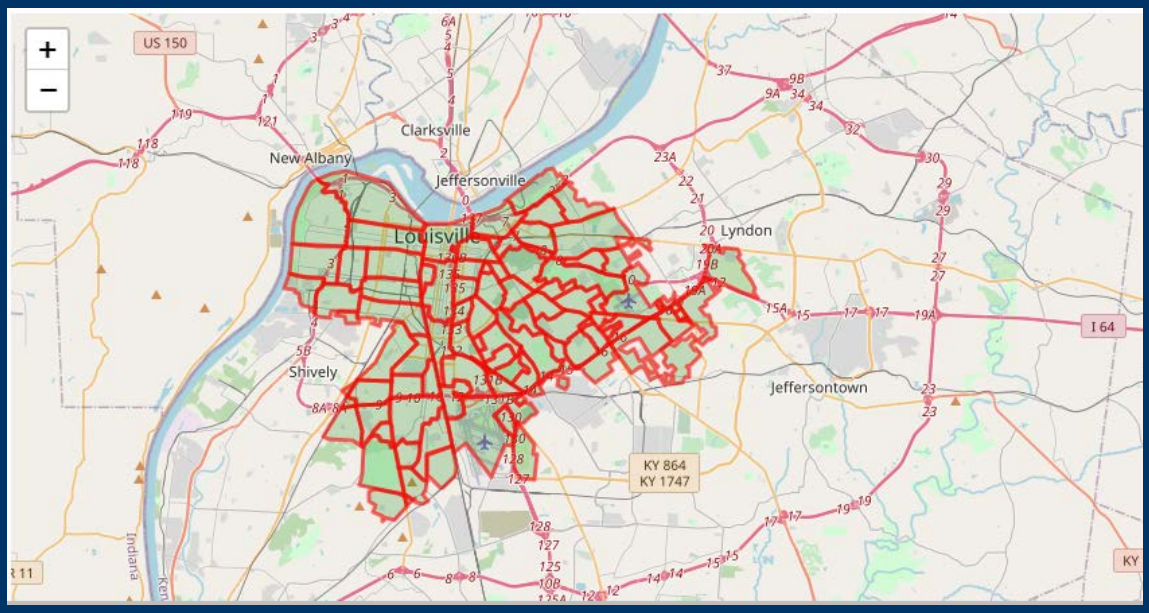


Another Map with GeoJson Data

```
lmap = folium.Map([latitude, longitude], zoom_start=12)

folium.GeoJson(louisville_df,
    style_function=lambda x: {
        'color' : 'red',
        'opacity': 0.6,
        'fillColor' : 'green',
        }).add_to(lmap)

lmap
```



Next, I have used the Foursquare API to explore the Neighborhoods and segment them

1. First I Define Foursquare Credentials and Version
2. Then I explore the first neighborhood in my dataframe

```
louisville_N.loc[0, 'Neighborhood']
```

```
'Remainder Of City'
```

3. Get the neighborhood's latitude and longitude values

```
neighborhood_latitude = louisville_N.loc[0, 'Latitude'] # neighborhood latitude value
neighborhood_longitude = louisville_N.loc[0, 'Longitude'] # neighborhood longitude value

neighborhood_name = louisville_N.loc[0, 'Neighborhood'] # neighborhood name

print('Latitude and longitude values of {} are {}, {}.'.format(neighborhood_name,
                                                               neighborhood_latitude,
                                                               neighborhood_longitude))
```

```
Latitude and longitude values of Remainder Of City are 38.27222193306387, -85.7087118431899.
```

4. Now let's Get venues in each Neighborhood using a function

```
[11]: def getNearbyVenues(names, latitudes, longitudes, radius=500):

          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)
```

5. Now I run the function on each neighborhood and create a new Dataframe .

```
Remainder Of City
Portland
Shawnee
Brownsboro Zorn
Clifton Heights
Butchertown
Crescent Hill
Central Business District
Russell
Clifton
Irish Hill
Remainder Of City
Phoenix Hill
Rockcreek Lexington Road
Chickasaw
Parkland
California
Cherokee Triangle
Remainder Of City
Cherokee Gardens
Smoketown Jackson
Old Louisville
```

```
Cherokee Seneca
Paristown Pointe
Highlands
Limerick
Bowman
Remainder Of City
Germantown
Park Hill
Tyler Park
Shelby Park
Bonnycastle
Park Duvalle
Remainder Of City
Hikes Point
Highlands Douglass
Deer Park
Algonquin
```

```
Bashford Manor
Remainder Of City
Fairgrounds
Camp Taylor
Remainder Of City
Wilder Park
Prestonia
Wyandotte/Oakdale
Jacobs
Remainder Of City
Highland Park
Remainder Of City
Remainder Of City
Beechmont
Prestonia
Hazelwood
Cloverleaf
Standiford
Southside
Iroquois
Edgewood
Iroquois Park
Iroquois
Southland Park
Kenwood Hill
Auburndale
```

```
print(louisville_venues.shape)
louisville_venues.head()
```

```
(655, 7)
```

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Remainder Of City	38.272222	-85.708712	Munson Business Interiors	38.273943	-85.708248	Home Service
1	Remainder Of City	38.272222	-85.708712	Tumbleweed Southwest Grill	38.272730	-85.708468	American Restaurant
2	Remainder Of City	38.272222	-85.708712	Champion Soccer Park	38.269385	-85.708292	Soccer Field
3	Remainder Of City	38.272222	-85.708712	Heuser Clinic	38.270264	-85.713112	Gym
4	Remainder Of City	38.272222	-85.708712	ProFormance Fitness & Training	38.270307	-85.713204	Gym

5. Now check how many venues were returned for neighborhood

```
louisville_venues.groupby('Neighborhood').count()
```

Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
	46	46	46	46	46	46
Algonquin	4	4	4	4	4	4
Auburndale	3	3	3	3	3	3
Audubon	1	1	1	1	1	1
Avondale Melbourne Heights	2	2	2	2	2	2
Beechmont	6	6	6	6	6	6
Belknap	4	4	4	4	4	4
Bon Air	2	2	2	2	2	2
Bonnycastle	4	4	4	4	4	4
Bowman	5	5	5	5	5	5
Brownsboro Zorn	1	1	1	1	1	1
Butchertown	16	16	16	16	16	16

6. How many unique categories can be curated from all the returned venues

```
print('There are {} uniques categories.'.format(len(louisville_venues['Venue Category'].unique())))
```

```
There are 169 uniques categories.
```

3.3 Analyze each Neighborhood in Louisville Kentucky

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

# add neighborhood column back to dataframe
louisville_onehot['Neighborhood'] = louisville_venues['Neighborhood']

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

louisville_onehot.head()
```

	Neighborhood	ATM	Advertising Agency	African Restaurant	Airport	Airport Service	Airport Terminal	American Restaurant	Art Gallery	Arts & Crafts Store	Asian Restaurant	Athletics & Sports	Auto Garage	Auto Workshop	Automotive Shop
0	Remainder Of City	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	Remainder Of City	0	0	0	0	0	0	1	0	0	0	0	0	0	0
2	Remainder Of City	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	Remainder Of City	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	Remainder Of City	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Now I print each Neighborhood along with the top 5 most common venues. For this report I show some of them only.

```
num_top_venues = 5

for hood in louisville_grouped['Neighborhood']:
    print("----"+hood+"----")
    temp = louisville_grouped[louisville_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')
```

---- ----

	venue	freq
0	Pizza Place	0.11
1	Pharmacy	0.07
2	Bank	0.04
3	Ice Cream Shop	0.04
4	Convenience Store	0.04

----Algonquin----

	venue	freq
0	Fast Food Restaurant	0.50
1	Convenience Store	0.25
2	Auto Garage	0.25
3	Restaurant	0.00
4	Pizza Place	0.00

----Auburndale----

	venue	freq
0	Daycare	0.33
1	Construction & Landscaping	0.33
2	Café	0.33
3	Pool	0.00
4	Paper / Office Supplies Store	0.00

----Audubon----

	venue	freq
0	Park	1.0
1	ATM	0.0
2	Playground	0.0
3	New American Restaurant	0.0
4	Nightclub	0.0

----Avondale Melbourne Heights----

	venue	freq
0	Donut Shop	0.5
1	Convenience Store	0.5
2	ATM	0.0
3	Playground	0.0
4	Nightclub	0.0

----Beechmont----

	venue	freq
0	Sandwich Place	0.33
1	Ice Cream Shop	0.17
2	Coffee Shop	0.17
3	Vietnamese Restaurant	0.17
4	Park	0.17

Now let's create the new Dataframe and display the top 10 venues for each neighborhood

```
num_top_venues = 10

indicators = ['st', 'nd', 'rd']

# create columns according to number of top venues
columns = ['Neighborhood']
for ind in np.arange(num_top_venues):
    try:
        columns.append('{}{} Most Common Venue'.format(ind+1, indicators[ind]))
    except:
        columns.append('{}th Most Common Venue'.format(ind+1))

# create a new dataframe
neighborhoods_venues_sorted = pd.DataFrame(columns=columns)
neighborhoods_venues_sorted['Neighborhood'] = louisville_grouped['Neighborhood']

for ind in np.arange(louisville_grouped.shape[0]):
    neighborhoods_venues_sorted.iloc[ind, 1:] = return_most_common_venues(louisville_grouped.iloc[ind, :], num_top_venues)

neighborhoods_venues_sorted.head()
```

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0		Pizza Place	Pharmacy	Sandwich Place	Ice Cream Shop	Bank	Deli / Bodega	Convenience Store	Bridal Shop	Business Service	Café
1	Algonquin	Fast Food Restaurant	Auto Garage	Convenience Store	Daycare	Deli / Bodega	Farmers Market	Farm	Falafel Restaurant	Event Service	Dry Cleaner
2	Auburndale	Daycare	Café	Construction & Landscaping	Yoga Studio	Doctor's Office	Farmers Market	Farm	Falafel Restaurant	Event Service	Dry Cleaner
3	Audubon	Park	Yoga Studio	Doctor's Office	Farmers Market	Farm	Falafel Restaurant	Event Service	Dry Cleaner	Donut Shop	Dog Run
4	Avondale Melbourne Heights	Convenience Store	Donut Shop	Yoga Studio	Dive Bar	Farmers Market	Farm	Falafel Restaurant	Event Service	Dry Cleaner	Dog Run

3.4 Cluster Neighborhoods

To analyze which neighborhood of Louisville is good to open a new restaurant, I will use a K-means clustering: a type of unsupervised learning, which is used when you have unlabeled data (i.e., data without defined categories or groups). The goal of this algorithm is to find groups in the data, with the number of groups represented by the variable K. The algorithm works iteratively to assign each data point to one of K groups based on the features that are provided. Data points are clustered based on feature similarity.

```
# set number of clusters
kclusters = 5

louisville_grouped_clustering = louisville_grouped.drop('Neighborhood', 1)

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

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

```
array([3, 3, 3, 0, 3, 3, 3, 2, 3, 3], dtype=int32)
```

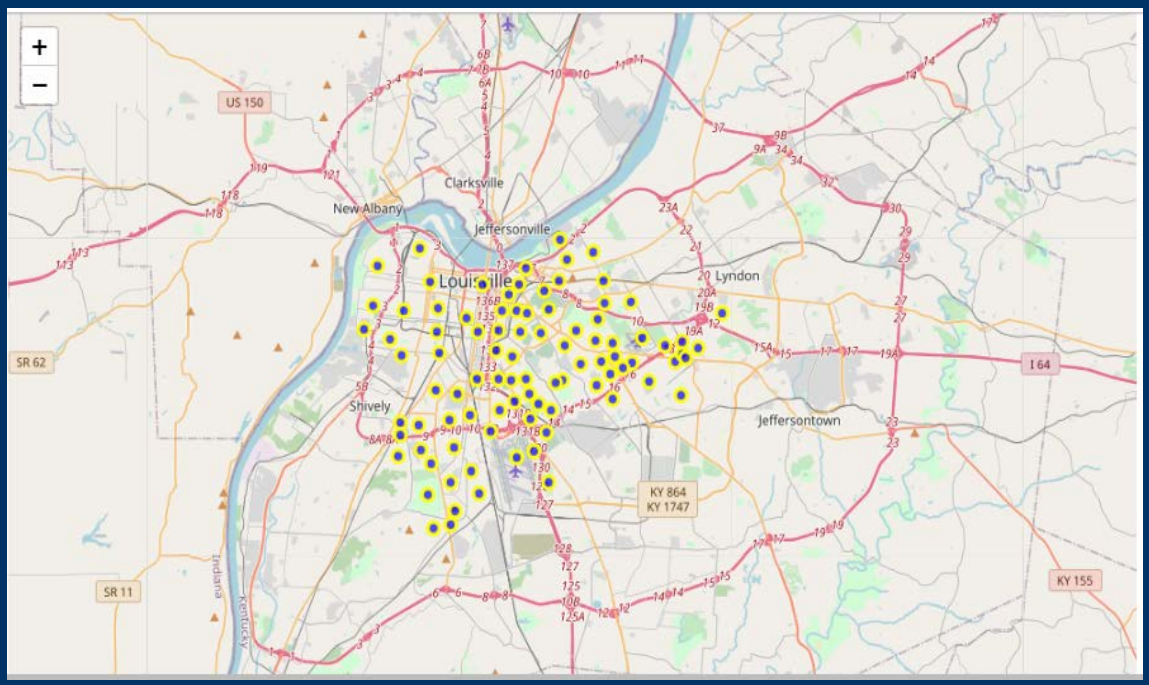
Finally, lets’s visualize the resulting cluster

```
# create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=11)

# set color scheme for the clusters
x = np.arange(kclusters)
ys = [i + x + (i*x)**2 for i in range(kclusters)]
colors_array = cm.rainbow(np.linspace(0, 1, len(ys)))
rainbow = [colors.rgb2hex(i) for i in colors_array]

# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(louisville_merged['Latitude'], louisville_merged['Longitude'], louisville_merged['Neighborhood'], louis
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    folium.CircleMarker(
        [lat, lon],
        radius=5,
        popup=label,
        color='yellow',
        fill=True,
        fill_color='blue',
        fill_opacity=0.7).add_to(map_clusters)

map_clusters
```



4. Examine Cluster

In this section I examine each cluster and determine the discriminating venue categories that distinguish each one. Based on the defining categories

Cluster 1

```
louisville_merged.loc[louisville_merged['Cluster Labels'] == 0, louisville_merged.columns[[1] + list(range(5, louisville_merged.shape[1])
```

	Neighborhood	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
57	Audubon	0.0	Park	Yoga Studio	Doctor's Office	Farmers Market	Farm	Falafel Restaurant	Event Service	Dry Cleaner	Donut Shop	Dog Run
63	Klondike	0.0	Park	Construction & Landscaping	Yoga Studio	Doctor's Office	Farmers Market	Farm	Falafel Restaurant	Event Service	Dry Cleaner	Donut Shop
89	Kenwood Hill	0.0	Construction & Landscaping	Historic Site	Discount Store	Farm	Falafel Restaurant	Event Service	Dry Cleaner	Donut Shop	Dog Run	Doctor's Office

Cluster 2

```
louisville_merged.loc[louisville_merged['Cluster Labels'] == 1, louisville_merged.columns[[1] + list(range(5, louisville_merged.shape[1])
```

	Neighborhood	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
1	Portland	1.0	Pizza Place	Clothing Store	Bank	Grocery Store	Dive Bar	Farm	Falafel Restaurant	Event Service	Dry Cleaner	Donut Shop
8	Russell	1.0	Pizza Place	Liquor Store	Dive Bar	Farm	Falafel Restaurant	Event Service	Dry Cleaner	Donut Shop	Dog Run	Doctor's Office
35	Hikes Point	1.0	Pizza Place	Ice Cream Shop	Bank	Discount Store	Dive Bar	Farm	Falafel Restaurant	Event Service	Dry Cleaner	Donut Shop
41	Merriwether	1.0	Pizza Place	Gas Station	Convenience Store	Karaoke Bar	Toy / Game Store	Gastropub	Ice Cream Shop	Department Store	Dive Bar	Falafel Restaurant
51	Hawthorne	1.0	Sushi Restaurant	Pizza Place	Yoga Studio	Dive Bar	Farm	Falafel Restaurant	Event Service	Dry Cleaner	Donut Shop	Dog Run
85	Edgewood	1.0	Ice Cream Shop	Yoga Studio	Dive Bar	Farmers Market	Farm	Falafel Restaurant	Event Service	Dry Cleaner	Donut Shop	Dog Run

Cluster 3

```
louisville_merged.loc[louisville_merged['Cluster Labels'] == 2, louisville_merged.columns[[1] + list(range(5, louisville_merged.shape[1])
```

	Neighborhood	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
28	Germantown	2.0	Bar	Coffee Shop	Thrift / Vintage Store	Record Shop	Farm	Falafel Restaurant	Event Service	Dry Cleaner	Donut Shop	Dog Run
56	Bon Air	2.0	Bar	Discount Store	Yoga Studio	Doctor's Office	Farmers Market	Farm	Falafel Restaurant	Event Service	Dry Cleaner	Donut Shop
68	Camp Taylor	2.0	Bar	Bank	Yoga Studio	Fish & Chips Shop	Farmers Market	Farm	Falafel Restaurant	Event Service	Dry Cleaner	Donut Shop

Cluster 4

```
louisville_merged.loc[louisville_merged['Cluster Labels'] == 3, louisville_merged.columns[[1] + list(range(5, louisville_merged.shape[1])
```

	Neighborhood	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0	Remainder Of City	3.0	Pizza Place	Sandwich Place	Fast Food Restaurant	Bank	Pharmacy	Bar	Furniture / Home Store	Gas Station	Automotive Shop	Chinese Restaurant
2	Shawnee	3.0	Sports Club	Sandwich Place	Food	Restaurant	Cosmetics Shop	Doctor's Office	Farm	Falafel Restaurant	Event Service	Dry Cleaner
3	Brownsboro Zorn	3.0	Gym / Fitness Center	Yoga Studio	Discount Store	Farm	Falafel Restaurant	Event Service	Dry Cleaner	Donut Shop	Dog Run	Doctor's Office
4	Clifton Heights	3.0	Convenience Store	Bar	Video Store	Art Gallery	Doctor's Office	Farmers Market	Farm	Falafel Restaurant	Event Service	Dry Cleaner
5	Butchertown	3.0	Park	Gay Bar	New American Restaurant	Food Truck	Shopping Mall	Soccer Stadium	Speakeasy	Athletics & Sports	Dessert Shop	Pizza Place
6	Crescent Hill	3.0	Boutique	Bakery	Lake	Music Store	Yoga Studio	Dog Run	Farmers Market	Farm	Falafel Restaurant	Event Service
7	Central Business District	3.0	Hotel	American Restaurant	Coffee Shop	Speakeasy	Restaurant	Sandwich Place	Mexican Restaurant	Pizza Place	Fast Food Restaurant	Whisky Bar
9	Clifton	3.0	Arts & Crafts Store	Asian Restaurant	Bakery	Sushi Restaurant	Sporting Goods Shop	New American Restaurant	Mediterranean Restaurant	Italian Restaurant	Gastropub	Sandwich Place
10	Irish Hill	3.0	Coffee Shop	American Restaurant	Park	Bagel Shop	Gastropub	Gas Station	Spa	Department Store	Farm	Falafel Restaurant
11	Remainder Of City	3.0	Pizza Place	Sandwich Place	Fast Food Restaurant	Bank	Pharmacy	Bar	Furniture / Home Store	Gas Station	Automotive Shop	Chinese Restaurant
12	Phoenix Hill	3.0	Fast Food Restaurant	Clothing Store	Sandwich Place	Grocery Store	Shipping Store	Brewery	Furniture / Home Store	Yoga Studio	Discount Store	Dry Cleaner
13	Rockcreek Lexington Road	3.0	Pool	Soccer Field	Trail	Lingerie Store	Athletics & Sports	Yoga Studio	Dive Bar	Falafel Restaurant	Event Service	Dry Cleaner
15	Parkland	3.0	Business Service	Food	Discount Store	Construction & Landscaping	Historic Site	Daycare	Dog Run	Farm	Falafel Restaurant	Cosmetics Shop
16	California	3.0	Bar	American Restaurant	Café	Park	Yoga Studio	Dog Run	Farmers Market	Farm	Falafel Restaurant	Event Service
17	Cherokee Triangle	3.0	Cemetery	Yoga Studio	Dive Bar	Farmers Market	Farm	Falafel Restaurant	Event Service	Dry Cleaner	Donut Shop	Dog Run
18	Remainder Of City	3.0	Pizza Place	Sandwich Place	Fast Food Restaurant	Bank	Pharmacy	Bar	Furniture / Home Store	Gas Station	Automotive Shop	Chinese Restaurant

Cluster 5

```
louisville_merged.loc[louisville_merged['Cluster Labels'] == 4, louisville_merged.columns[[1] + list(range(5, louisville_merged.shape[1])
```

	Neighborhood	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
60	Hayfield Dundee	4.0	Skating Rink	Yoga Studio	Farm	Falafel Restaurant	Event Service	Dry Cleaner	Donut Shop	Dog Run	Doctor's Office	Dive Bar

5. Result and Discussion Section

According to the objective of supplying information that allows orienting entrepreneurs who wish to invest in the city of Louisville, the results can be discussed from two main perspectives:

The first is that the city of Louisville is the city with the highest population density in the state of Kentucky, which is a state characterized by the economic activity of manufacturing industrialized products and tourism, the largest source of income comes from agricultural activity. Being the most populated city, it supposes a volume of inhabitants in certain neighborhoods that represent a business opportunity for commercial establishments of food and beverages.

Second we can analize our results according to the five clusters we have produced. Even though, all clusters could praise an optimal range of facilities, we have found one main pattern, this is Clusters 1,2 and 4 where there are the largest number of businesses with diverse economic activities and a low number of American food restaurants, which can represent an opportunity for entrepreneurs who want to invest in that sector.

6. Conclusion

As the analysis is performed on small set of data, we can achieve better results by increasing the neighborhood information. Anyway Louisville is a city with many different types of new restaurant business. According with the last ideas I think we have gone through the process of identifying information for the business problem, specifying the data required, clean the datasets, performing a machine learning algorithm using k-means clustering and providing some useful tips to our stakeholder.