A description of the problem and a discussion of the background.

The problem I try to answer is "what is the recommended location for a new cafe in New York". Suppose that a young entrepreneur, Jerry, wants to start his bussiness and he want to begin with a cafe. He is struggling with following questions: 1) Should I place my cafe in a place where there are a lot of existing cafes or in a place where there is none or should I try to find some kind of balance between these two extremes? 2) What is secret of a successful cafe? Can we get some tips or information based on the data provided by foursquare?

A description of the data and how it will be used to solve the problem.

For the problem one, I will: 1) find the high rating cafe in NY and identify them visually on a map; 2) create several "clusters" of these cafes to identfiy potential districts for Jerry to choose from. Jerry should choose one from these business district and locate his cafe in the center of it. The closer to the center, the high will be the rent. This is another choise that will be made by Jerry referring to his budget.

For the problem two, when Jerry has chosen a business district, researcher will collect the tips(customer reviews) for the cafe in that area and filter three types of tips - high ratings, low ratings and long comments because these tips will more likely include heeds for providing better services for the customers.

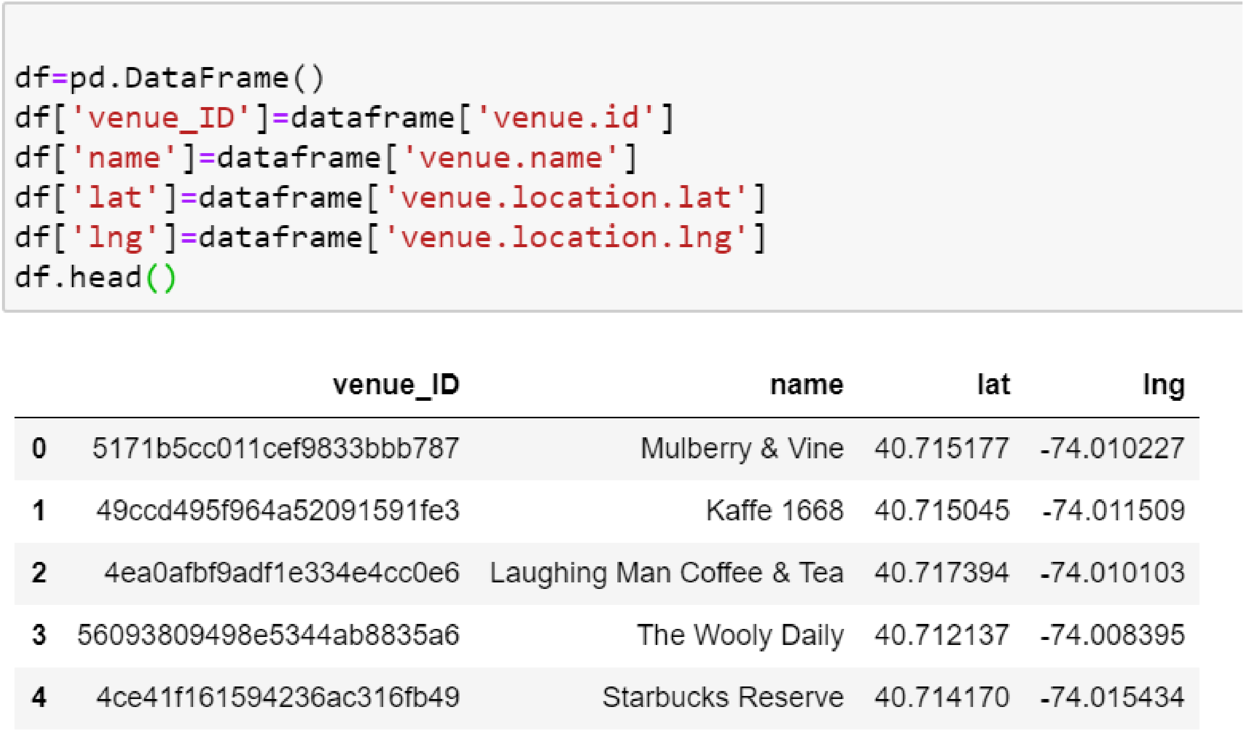
Data for the research

1. Search for top-rated "cafe" in New York city and their locations

We should use the "explore" endpoint. The request url is "https://api.foursquare.com/v2/venues/explore" According to the document, in the request, we should pass following parameters: "section=coffee" and "near=New York, NY". In the response, we are interested in groups.items.categories, groups.items.venue.name, group.items.venue.location. To simplify the problem, we will only use the geographic location to cluster the cafes.

2. Find the tips

We will recommend the "geographic mean"(the center of all cafes) as the location for Jerry. Based on this location, we use the search endpoint to get all the tips for coffee shops in the surrounding area (1km) . For this purpose, the Tips endpoint will be applied. The request url is "https://api.foursquare.com/v2/venues/VENUE_ID/tips" and VENUE_ID should be provided.



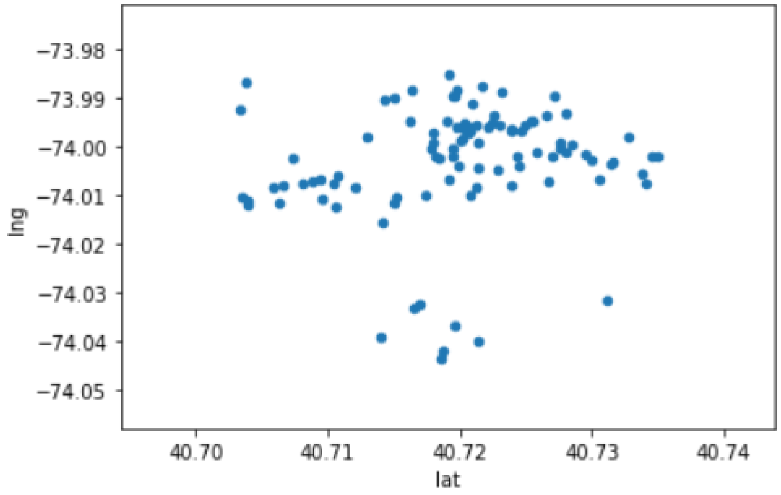
```
df=pd.DataFrame()
df['venue_ID']=dataframe['venue.id']
df['name']=dataframe['venue.name']
df['lat']=dataframe['venue.location.lat']
df['lng']=dataframe['venue.location.lng']
df.head()
```

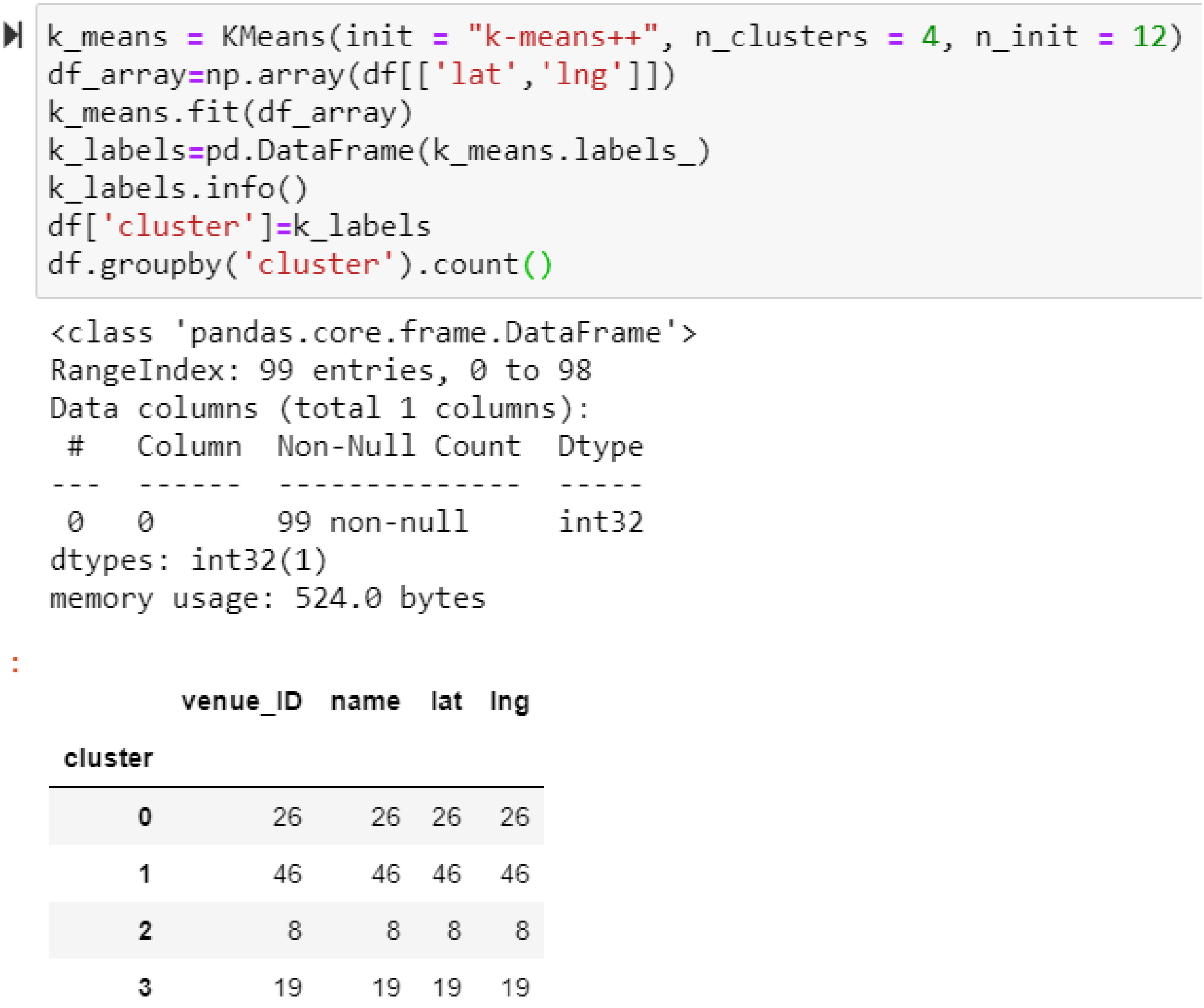
	venue_ID	name	lat	lng
0	5171b5cc011cef9833bbb787	Mulberry & Vine	40.715177	-74.010227
1	49ccd495f964a52091591fe3	Kaffe 1668	40.715045	-74.011509
2	4ea0afbf9adf1e334e4cc0e6	Laughing Man Coffee & Tea	40.717394	-74.010103
3	56093809498e5344ab8835a6	The Wooly Daily	40.712137	-74.008395
4	4ce41f161594236ac316fb49	Starbucks Reserve	40.714170	-74.015434

```
# Show the cafes on the map
map_newyork = folium.Map(location=[latitude, longitude], zoom_start=13)
```

```


# Cluster these Cafes based on their geographic locations

from sklearn.cluster import KMeans
import matplotlib.pyplot as plt # plotting library


%matplotlib inline

df.plot(x='lat',y='lng',kind='scatter')
```

```
]: <matplotlib.axes._subplots.AxesSubplot at 0x152d77e8d88>
```

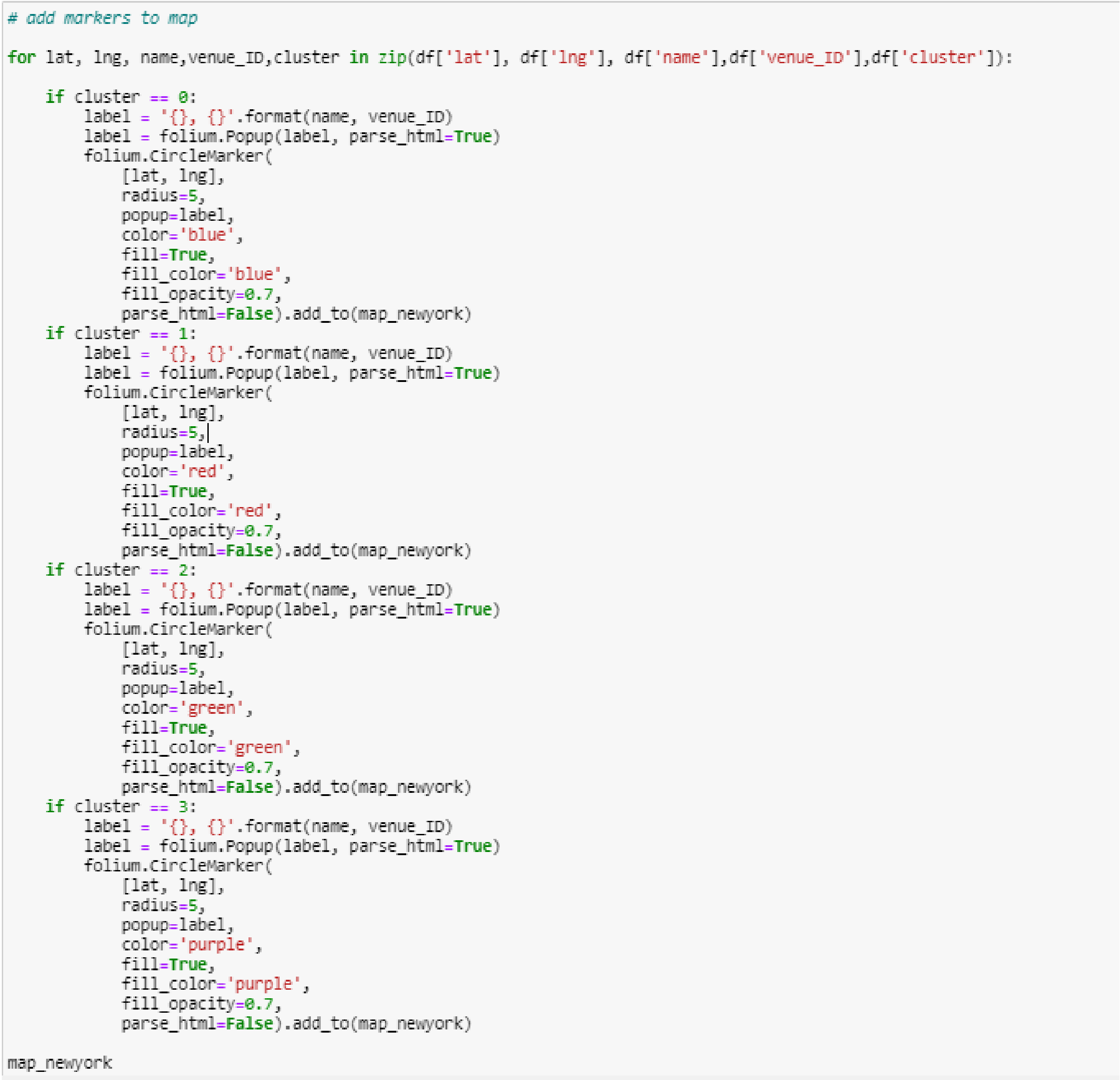




```
k_means = KMeans(init = "k-means++", n_clusters = 4, n_init = 12)
df_array=np.array(df[['lat','lng']])
k_means.fit(df_array)
k_labels=pd.DataFrame(k_means.labels_)
k_labels.info()
df['cluster']=k_labels
df.groupby('cluster').count()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 99 entries, 0 to 98
Data columns (total 1 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   0       99 non-null     int32
dtypes: int32(1)
memory usage: 524.0 bytes
```

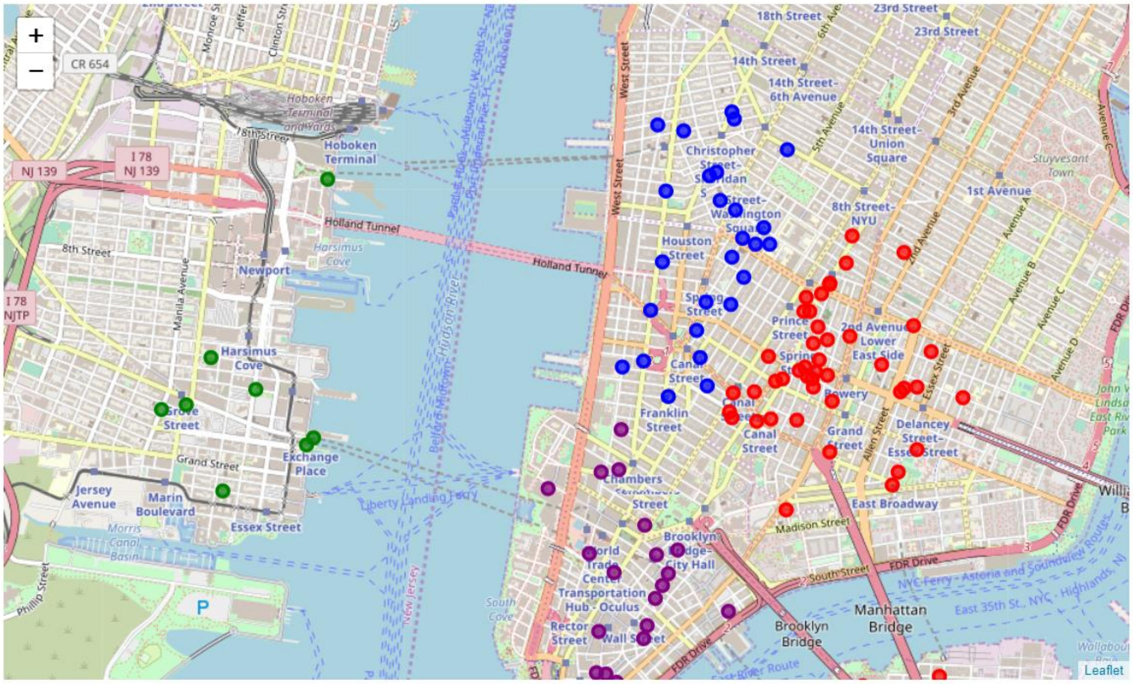
	venue_ID	name	lat	lng
cluster				
0	26	26	26	26
1	46	46	46	46
2	8	8	8	8
3	19	19	19	19

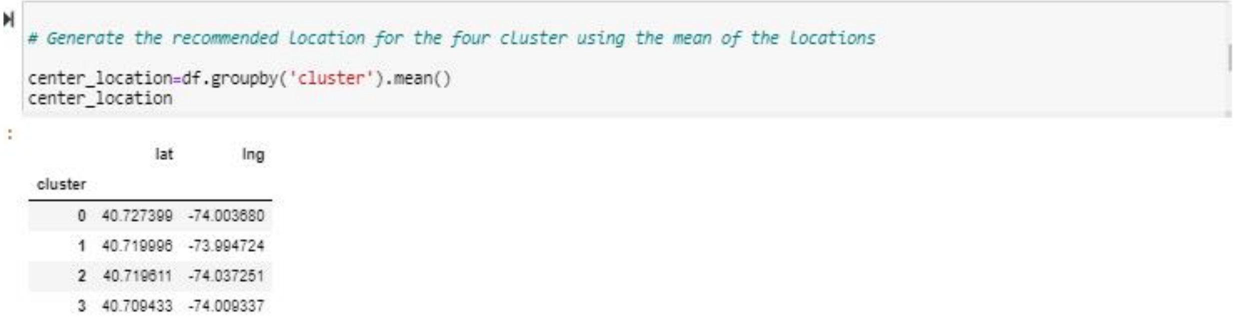


```python
# add markers to map

for lat, lng, name,venue_ID,cluster in zip(df['lat'], df['lng'], df['name'],df['venue_ID'],df['cluster']):

    if cluster == 0:
        label = '{}, {}'.format(name, venue_ID)
        label = folium.Popup(label, parse_html=True)
        folium.CircleMarker(
            [lat, lng],
            radius=5,
            popup=label,
            color='blue',
            fill=True,
            fill_color='blue',
            fill_opacity=0.7,
            parse_html=False).add_to(map_newyork)
    if cluster == 1:
        label = '{}, {}'.format(name, venue_ID)
        label = folium.Popup(label, parse_html=True)
        folium.CircleMarker(
            [lat, lng],
            radius=5,
            popup=label,
            color='red',
            fill=True,
            fill_color='red',
            fill_opacity=0.7,
            parse_html=False).add_to(map_newyork)
    if cluster == 2:
        label = '{}, {}'.format(name, venue_ID)
        label = folium.Popup(label, parse_html=True)
        folium.CircleMarker(
            [lat, lng],
            radius=5,
            popup=label,
            color='green',
            fill=True,
            fill_color='green',
            fill_opacity=0.7,
            parse_html=False).add_to(map_newyork)
    if cluster == 3:
        label = '{}, {}'.format(name, venue_ID)
        label = folium.Popup(label, parse_html=True)
        folium.CircleMarker(
            [lat, lng],
            radius=5,
            popup=label,
            color='purple',
            fill=True,
            fill_color='purple',
            fill_opacity=0.7,
            parse_html=False).add_to(map_newyork)

map_newyork
```

```
map_newyork
```

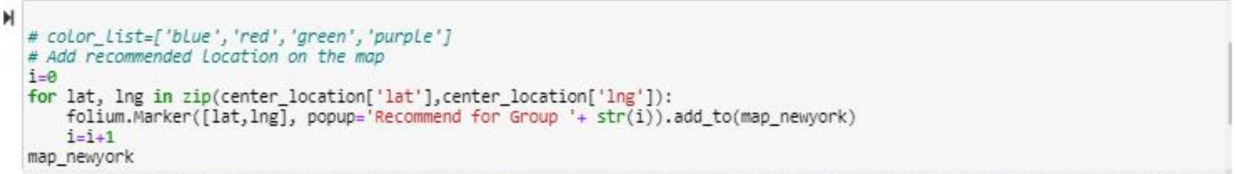
Out[13]:



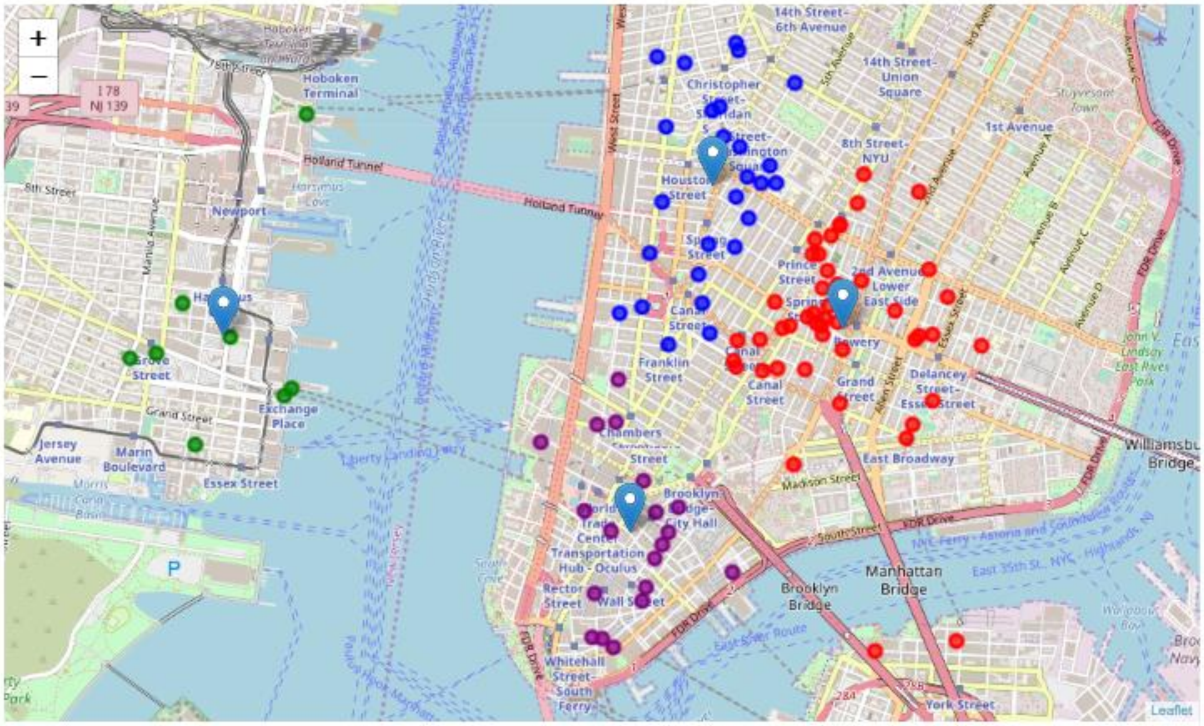


```
# Generate the recommended location for the four cluster using the mean of the locations

center_location=df.groupby('cluster').mean()
center_location
```

	lat	lng
cluster		
0	40.727399	-74.003680
1	40.719996	-73.994724
2	40.719611	-74.037251
3	40.709433	-74.009337



```
# color_list=['blue','red','green','purple']
# Add recommended location on the map
i=0
for lat, lng in zip(center_location['lat'],center_location['lng']):
    folium.Marker([lat,lng], popup='Recommend for Group '+ str(i)).add_to(map_newyork)
    i=i+1
map_newyork
```



Conclusion

Jerry can select from the above shown area by seeing the customer tips of the cafes around this area. This graph and data will help Jerry to make a decision and start a successful business.