Capstone Project Week 4+5

September 24, 2019

1 This notebook is created for Capstone Project-Battle of the Neighbors (Week1)

1.1 Battle of Neighborhoods

1.1.1 Introduction and Project Description

United State of America I recently had a chance to relocate to Orlando, Florida, and usually people tend to explore the places before moving to a new country, state, city or place for their work or to start a new life and that involves so many factors including neighborhood analysis. It would be easier if there was some sort of search algorithm that can returns the requested features such as population rate, median house price, school ratings, crime rates, weather conditions, recreational facilities etc. It'd be nice to have a hassle-free application that could spit out an extensive analysis of all these features for a neighborhood or a comparative analysis between neighborhoods with just sending out the names of the neighborhoods This Project tends to address this requirement as its main idea to help new movers and stakeholders to achieve the desired results, so that it helps them in saving their time and money in finding the right neighborhood and being keep trapped in an infinite loop of extensive search engines. This Project aims to help new movers and stakeholders to take a better decision on choosing the best neighborhood out of many neighborhoods and to build/buy their houses in USA based on the distribution of various facilities in and around that neighborhood. As an example, this project would compare 2 randomly picked neighborhoods and analyses the top 10 most common venues in each of those two neighborhoods based on the number of visits by people in each of those places. Also, this project will be using K-mean clustering unsupervised machine learning algorithm to cluster the venues based on the place category such as restaurants, park, coffee shop, gym etc. This will help in giving a better understanding of the similarities and dissimilarities between the two chosen neighborhoods to retrieve more insights and to conclude with ease which neighborhood wins over other.

[ ]:

1.1.2 Description of the data and how it will be used to solve the problem

Foursquare API:

This project will be using Four-square API as its prime data gathering source as it has a database of more than 105 million places, especially their places API which provides the ability to perform location search, location sharing and details about a business. Photos, tips and reviews jolted by Foursquare users can also be used in many productive ways to add value to the results.

Work Flow:

HTTP requests will be made to this Foursquare API server using zip codes of the USA states to pull the location information (Latitude and Longitude). Foursquare API search feature would be enabled to collect the nearby places of the neighborhoods. Due to http request limitations the number of places per neighborhood parameter would reasonably be set to 100 and the radius parameter would be set to 700. Folium- Python visualization library would be used to visualize the neighborhoods cluster distribution of Seattle city over an interactive leaflet map. Extensive comparative analysis of two randomly picked neighborhoods world be carried out to derive the desirable insights from the outcomes using python's scientific libraries Pandas, NumPy and Scikit-learn. Unsupervised machine learning algorithm K-mean clustering would be applied to form the clusters of different categories of places residing in and around the neighborhoods. These clusters from each of those two chosen neighborhoods would be analyzed individually collectively and comparatively to derive the conclusions.

Python packages and Dependencies which will be used:

• Pandas - Library for Data Analysis • NumPy – Library to handle data in a vectorized manner • JSON – Library to handle JSON files • Geopy – To retrieve Location Data • Requests – Library to handle http requests • Matplotlib – Python Plotting Module • Sklearn – Python machine learning Library • Folium – Map rendering Library

[ ]:

[2]:
```
#Importing all libraries
import numpy as np # library to handle data in a vectorized manner
import pandas as pd # library for data analsysis
pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', None)
import json # library to handle JSON files
!conda install -c conda-forge geopy --yes
from geopy.geocoders import Nominatim # convert an address into latitude and␣
 ↪longitude values
import requests # library to handle requests
from pandas.io.json import json_normalize # tranform JSON file into a pandas␣
 ↪dataframe
# Matplotlib and associated plotting modules
import matplotlib.cm as cm
import matplotlib.colors as colors
# import k-means from clustering stage
from sklearn.cluster import KMeans
from sklearn.datasets.samples_generator import make_blobs
!conda install -c conda-forge folium=0.5.0 --yes
import folium # map rendering library
!conda install -c anaconda beautifulsoup4 --yes
from bs4 import BeautifulSoup
!conda install -c anaconda lxml --yes
import lxml
import xml
print('Libraries imported.')
```

```
Solving environment: done


==> WARNING: A newer version of conda exists. <==
  current version: 4.5.11
  latest version: 4.7.12

Please update conda by running

    $ conda update -n base -c defaults conda



## Package Plan ##

  environment location: /home/jupyterlab/conda/envs/python

  added / updated specs:
    - geopy


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    geopy-1.20.0               |             py_0          57 KB  conda-forge
    certifi-2019.9.11          |           py36_0         147 KB  conda-forge
    geographiclib-1.49         |             py_0          32 KB  conda-forge
    ------------------------------------------------------------
                                           Total:         237 KB

The following NEW packages will be INSTALLED:

    geographiclib: 1.49-py_0        conda-forge
    geopy:         1.20.0-py_0      conda-forge

The following packages will be UPDATED:

    certifi:       2019.6.16-py36_1 conda-forge --> 2019.9.11-py36_0 conda-forge


Downloading and Extracting Packages
geopy-1.20.0         | 57 KB     | ##################################### | 100%
certifi-2019.9.11    | 147 KB    | ##################################### | 100%
geographiclib-1.49   | 32 KB     | ##################################### | 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
```

```
Solving environment: done


==> WARNING: A newer version of conda exists. <==
  current version: 4.5.11
  latest version: 4.7.12

Please update conda by running

    $ conda update -n base -c defaults conda



# All requested packages already installed.

Solving environment: done


==> WARNING: A newer version of conda exists. <==
  current version: 4.5.11
  latest version: 4.7.12

Please update conda by running

    $ conda update -n base -c defaults conda



## Package Plan ##

  environment location: /home/jupyterlab/conda/envs/python

  added / updated specs:
    - beautifulsoup4


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    soupsieve-1.9.2            |           py36_0          61 KB  anaconda
    openssl-1.1.1              |       h7b6447c_0         5.0 MB  anaconda
    beautifulsoup4-4.8.0       |           py36_0         147 KB  anaconda
    certifi-2019.6.16          |           py36_1         156 KB  anaconda
    ------------------------------------------------------------
                                           Total:         5.4 MB

The following NEW packages will be INSTALLED:
```

```
    soupsieve:      1.9.2-py36_0      anaconda

The following packages will be UPDATED:

    beautifulsoup4: 4.6.3-py37_0                  --> 4.8.0-py36_0     anaconda
    openssl:        1.1.1c-h516909a_0 conda-forge --> 1.1.1-h7b6447c_0 anaconda

The following packages will be DOWNGRADED:

    certifi:        2019.9.11-py36_0  conda-forge --> 2019.6.16-py36_1 anaconda


Downloading and Extracting Packages
soupsieve-1.9.2      | 61 KB     | ##################################### | 100%
openssl-1.1.1        | 5.0 MB    | ##################################### | 100%
beautifulsoup4-4.8.0 | 147 KB    | ##################################### | 100%
certifi-2019.6.16    | 156 KB    | ##################################### | 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
Solving environment: done


==> WARNING: A newer version of conda exists. <==
  current version: 4.5.11
  latest version: 4.7.12

Please update conda by running

    $ conda update -n base -c defaults conda



## Package Plan ##

  environment location: /home/jupyterlab/conda/envs/python

  added / updated specs:
    - lxml


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    lxml-4.3.0                 |   py36hefd8a0e_0         1.5 MB  anaconda
```

```
The following packages will be UPDATED:

    lxml: 4.2.5-py37hefd8a0e_0 --> 4.3.0-py36hefd8a0e_0 anaconda


Downloading and Extracting Packages
lxml-4.3.0           | 1.5 MB    | ##################################### | 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
Libraries imported.
```

[3]:
```
link = 'https://en.wikipedia.org/wiki/
 ↪List_of_United_States_cities_by_population'
page = requests.get(link)
soup = BeautifulSoup(page.text)
```

[13]:
```
table = soup.find_all('table')[4]
```

[ ]:
```
```

[14]:
```
table_rows = table.find_all('tr')
res = []
for tr in table_rows:
    td = tr.find_all('td')
    row = [tr.text.strip() for tr in td if tr.text.strip()]
    if row:
        res.append(row)
df = pd.DataFrame(res, columns=["Rank", "City", "State", "del1", "del2",␣
 ↪"del3", "Sq.Area", "del5", "population density in Sq Mi", "Population␣
 ↪density in Km2", "Location"])
df.head()
```

[14]:

	Rank	City	State	del1	del2	del3	Sq.Area	del5	population density in Sq Mi	Population density in Km2	\
0	1	New York[d]	New York	8,398,748	8,175,133	+2.74%	301.5 sq mi	780.9 km2	28,317/sq mi	10,933/km2	
1	2	Los Angeles	California	3,990,456	3,792,621	+5.22%	468.7 sq mi	1,213.9 km2	8,484/sq mi	3,276/km2	
2	3	Chicago	Illinois	2,705,994	2,695,598	+0.39%	227.3 sq mi	588.7 km2	11,900/sq mi	4,600/km2	
3	4	Houston[3]	Texas	2,325,502	2,100,263	+10.72%	637.5 sq mi	1,651.1 km2	3,613/sq mi	1,395/km2	
4	5	Phoenix	Arizona	1,660,272	1,445,632	+14.85%	517.6 sq mi	1,340.6 km2	3,120/sq mi	1,200/km2	

```
                                                Location
0   40°39 49 N 73°56 19 W / 40.6635°N 73.9387°W…
1   34°01 10 N 118°24 39 W / 34.0194°N 118.4108°…
2   41°50 15 N 87°40 54 W / 41.8376°N 87.6818°W…
3   29°47 12 N 95°23 27 W / 29.7866°N 95.3909°W…
4   33°34 20 N 112°05 24 W / 33.5722°N 112.0901°…
```

[15]:
```
new= df["Sq.Area"].str.split("s", n=1, expand = True)
new = new[0].str.replace(u'\xa0',u'')
df["Sq.Area"] = new.str.replace(',','')
df["Sq.Area"] = df["Sq.Area"].astype(float)
df["Radius"] = np.sqrt(df["Sq.Area"])
```

[16]:
```
df.drop(columns = ["Rank", "del1", "del2", "del3", "del5", "Sq.Area",␣
↪"population density in Sq Mi"], inplace = True)
df
```

[16]:

	City	State	Population density in Km2	\
0	New York[d]	New York	10,933/km2	
1	Los Angeles	California	3,276/km2	
2	Chicago	Illinois	4,600/km2	
3	Houston[3]	Texas	1,395/km2	
4	Phoenix	Arizona	1,200/km2	
5	Philadelphia[e]	Pennsylvania	4,511/km2	
6	San Antonio	Texas	1,250/km2	
7	San Diego	California	1,670/km2	
8	Dallas	Texas	1,493/km2	
9	San Jose	California	2,231/km2	
10	Austin	Texas	1,170/km2	
11	Jacksonville[f]	Florida	455/km2	
12	Fort Worth	Texas	962/km2	
13	Columbus	Ohio	1,520/km2	
14	San Francisco[g]	California	7,170/km2	
15	Charlotte	North Carolina	1,064/km2	
16	Indianapolis[h]	Indiana	914/km2	
17	Seattle	Washington	3,245/km2	
18	Denver[i]	Colorado	1,746/km2	
19	Washington[j]	District of Columbia	4,304/km2	
20	Boston	Massachusetts	5,381/km2	
21	El Paso	Texas	1,030/km2	
22	Detroit	Michigan	1,871/km2	
23	Nashville[k]	Tennessee	536/km2	
24	Portland	Oregon	1,851/km2	
25	Memphis	Tennessee	794/km2	
26	Oklahoma City	Oklahoma	407/km2	
27	Las Vegas	Nevada	1,818/km2	
28	Louisville[l]	Kentucky	903/km2	

29	Baltimore[m]	Maryland	2,934/km2
30	Milwaukee	Wisconsin	2,388/km2
31	Albuquerque	New Mexico	1,147/km2
32	Tucson	Arizona	888/km2
33	Fresno	California	1,762/km2
34	Mesa	Arizona	1,357/km2
35	Sacramento	California	1,953/km2
36	Atlanta	Georgia	1,366/km2
37	Kansas City	Missouri	590/km2
38	Colorado Springs	Colorado	918/km2
39	Miami	Florida	4,865/km2
40	Raleigh	North Carolina	1,221/km2
41	Omaha	Nebraska	1,296/km2
42	Long Beach	California	3,609/km2
43	Virginia Beach[m]	Virginia	710/km2
44	Oakland	California	2,901/km2
45	Minneapolis	Minnesota	2,960/km2
46	Tulsa	Oklahoma	791/km2
47	Arlington	Texas	1,600/km2
48	Tampa	Florida	1,284/km2
49	New Orleans[n]	Louisiana	892/km2
50	Wichita	Kansas	939/km2
51	Cleveland	Ohio	1,917/km2
52	Bakersfield	California	976/km2
53	Aurora	Colorado	910/km2
54	Anaheim	California	2,711/km2
55	Honolulu[b]	Hawaii	2,245/km2
56	Santa Ana	California	4,762/km2
57	Riverside	California	1,544/km2
58	Corpus Christi	Texas	720/km2
59	Lexington[o]	Kentucky	434/km2
60	Stockton	California	1,922/km2
61	Henderson	Nevada	1,080/km2
62	Saint Paul	Minnesota	2,245/km2
63	St. Louis[m]	Missouri	1,939/km2
64	Cincinnati	Ohio	1,490/km2
65	Pittsburgh	Pennsylvania	2,116/km2
66	Greensboro	North Carolina	864/km2
67	Anchorage[p]	Alaska	68/km2
68	Plano	Texas	1,540/km2
69	Lincoln	Nebraska	1,175/km2
70	Orlando	Florida	1,017/km2
71	Irvine	California	1,566/km2
72	Newark	New Jersey	4,514/km2
73	Toledo	Ohio	1,332/km2
74	Durham	North Carolina	925/km2
75	Chula Vista	California	2,080/km2

76	Fort Wayne	Indiana	923/km2
77	Jersey City	New Jersey	6,891/km2
78	St. Petersburg	Florida	1,631/km2
79	Laredo	Texas	982/km2
80	Madison	Wisconsin	1,270/km2
81	Chandler	Arizona	1,472/km2
82	Buffalo	New York	2,455/km2
83	Lubbock	Texas	783/km2
84	Scottsdale	Arizona	518/km2
85	Reno	Nevada	883/km2
86	Glendale	Arizona	1,607/km2
87	Gilbert[q]	Arizona	1,346/km2
88	Winston-Salem	North Carolina	706/km2
89	North Las Vegas	Nevada	941/km2
90	Norfolk[m]	Virginia	1,776/km2
91	Chesapeake[m]	Virginia	271/km2
92	Garland	Texas	1,592/km2
93	Irving	Texas	1,373/km2
94	Hialeah	Florida	4,245/km2
95	Fremont	California	1,161/km2
96	Boise[r]	Idaho	1,049/km2
97	Richmond[m]	Virginia	1,441/km2
98	Baton Rouge[s]	Louisiana	1,024/km2
99	Spokane	Washington	1,214/km2
100	Des Moines	Iowa	936/km2
101	Tacoma	Washington	1,641/km2
102	San Bernardino	California	1,358/km2
103	Modesto	California	1,905/km2
104	Fontana	California	1,883/km2
105	Santa Clarita	California	1,331/km2
106	Birmingham	Alabama	561/km2
107	Oxnard	California	2,984/km2
108	Fayetteville	North Carolina	535/km2
109	Moreno Valley	California	1,547/km2
110	Rochester	New York	2,253/km2
111	Glendale	California	2,551/km2
112	Huntington Beach	California	2,880/km2
113	Salt Lake City	Utah	673/km2
114	Grand Rapids	Michigan	1,708/km2
115	Amarillo	Texas	760/km2
116	Yonkers	New York	4,307/km2
117	Aurora	Illinois	1,729/km2
118	Montgomery	Alabama	483/km2
119	Akron	Ohio	1,231/km2
120	Little Rock	Arkansas	646/km2
121	Huntsville	Alabama	349/km2
122	Augusta[t]	Georgia	252/km2

123	Port St. Lucie	Florida	601/km2
124	Grand Prairie	Texas	1,018/km2
125	Columbus[u]	Georgia	353/km2
126	Tallahassee	Florida	734/km2
127	Overland Park	Kansas	971/km2
128	Tempe	Arizona	1,761/km2
129	McKinney	Texas	1,056/km2
130	Mobile	Alabama	534/km2
131	Cape Coral	Florida	658/km2
132	Shreveport	Louisiana	700/km2
133	Frisco	Texas	933/km2
134	Knoxville	Tennessee	730/km2
135	Worcester	Massachusetts	1,905/km2
136	Brownsville	Texas	536/km2
137	Vancouver	Washington	1,439/km2
138	Fort Lauderdale	Florida	1,995/km2
139	Sioux Falls	South Dakota	893/km2
140	Ontario	California	1,340/km2
141	Chattanooga	Tennessee	479/km2
142	Providence	Rhode Island	3,760/km2
143	Newport News[m]	Virginia	1,016/km2
144	Rancho Cucamonga	California	1,704/km2
145	Santa Rosa	California	1,637/km2
146	Oceanside	California	1,641/km2
147	Salem	Oregon	1,330/km2
148	Elk Grove	California	1,553/km2
149	Garden Grove	California	3,751/km2
150	Pembroke Pines	Florida	1,973/km2
151	Peoria	Arizona	361/km2
152	Eugene	Oregon	1,458/km2
153	Corona	California	1,630/km2
154	Cary[v]	North Carolina	1,109/km2
155	Springfield	Missouri	785/km2
156	Fort Collins	Colorado	1,136/km2
157	Jackson	Mississippi	588/km2
158	Alexandria[m]	Virginia	4,010/km2
159	Hayward	California	1,349/km2
160	Lancaster	California	656/km2
161	Lakewood	Colorado	1,390/km2
162	Clarksville	Tennessee	590/km2
163	Palmdale	California	573/km2
164	Salinas	California	2,572/km2
165	Springfield	Massachusetts	1,860/km2
166	Hollywood	Florida	2,150/km2
167	Pasadena	Texas	1,361/km2
168	Sunnyvale	California	2,681/km2
169	Macon[w]	Georgia	236/km2

170	Kansas City[x]	Kansas	470/km2
171	Pomona	California	2,560/km2
172	Escondido	California	1,578/km2
173	Killeen	Texas	1,030/km2
174	Naperville	Illinois	1,468/km2
175	Joliet	Illinois	889/km2
176	Bellevue	Washington	1,630/km2
177	Rockford	Illinois	898/km2
178	Savannah	Georgia	547/km2
179	Paterson	New Jersey	6,800/km2
180	Torrance	California	2,770/km2
181	Bridgeport	Connecticut	3,500/km2
182	McAllen	Texas	940/km2
183	Mesquite	Texas	1,176/km2
184	Syracuse	New York	2,214/km2
185	Midland	Texas	698/km2
186	Pasadena	California	2,385/km2
187	Murfreesboro	Tennessee	910/km2
188	Miramar	Florida	1,818/km2
189	Dayton	Ohio	974/km2
190	Fullerton	California	2,425/km2
191	Olathe	Kansas	859/km2
192	Orange	California	2,136/km2
193	Thornton	Colorado	1,478/km2
194	Roseville	California	1,191/km2
195	Denton	Texas	553/km2
196	Waco	Texas	580/km2
197	Surprise	Arizona	470/km2
198	Carrollton	Texas	1,419/km2
199	West Valley City	Utah	1,485/km2
200	Charleston	South Carolina	476/km2
201	Warren	Michigan	1,517/km2
202	Hampton[m]	Virginia	1,015/km2
203	Gainesville	Florida	815/km2
204	Visalia	California	1,349/km2
205	Coral Springs	Florida	2,110/km2
206	Columbia	South Carolina	388/km2
207	Cedar Rapids	Iowa	715/km2
208	Sterling Heights	Michigan	1,401/km2
209	New Haven	Connecticut	2,683/km2
210	Stamford	Connecticut	1,326/km2
211	Concord	California	1,630/km2
212	Kent	Washington	1,461/km2
213	Santa Clara	California	2,643/km2
214	Elizabeth	New Jersey	4,038/km2
215	Round Rock	Texas	1,311/km2
216	Thousand Oaks	California	902/km2

217	Lafayette[y]	Louisiana	916/km2
218	Athens[z]	Georgia	410/km2
219	Topeka	Kansas	796/km2
220	Simi Valley	California	1,175/km2
221	Fargo	North Dakota	950/km2
222	Norman	Oklahoma	264/km2
223	Columbia	Missouri	717/km2
224	Abilene	Texas	442/km2
225	Wilmington	North Carolina	880/km2
226	Hartford	Connecticut	2,735/km2
227	Victorville	California	644/km2
228	Pearland	Texas	947/km2
229	Vallejo	California	1,525/km2
230	Ann Arbor	Michigan	1,659/km2
231	Berkeley	California	4,458/km2
232	Allentown	Pennsylvania	2,657/km2
233	Richardson	Texas	1,530/km2
234	Odessa	Texas	1,007/km2
235	Arvada	Colorado	1,175/km2
236	Cambridge	Massachusetts	6,675/km2
237	Sugar Land	Texas	1,347/km2
238	Beaumont	Texas	556/km2
239	Lansing	Michigan	1,146/km2
240	Evansville	Indiana	975/km2
241	Rochester	Minnesota	806/km2
242	Independence	Missouri	581/km2
243	Fairfield	California	1,083/km2
244	Provo	Utah	1,082/km2
245	Clearwater	Florida	1,705/km2
246	College Station	Texas	849/km2
247	West Jordan	Utah	1,360/km2
248	Carlsbad	California	1,167/km2
249	El Monte	California	4,658/km2
250	Murrieta	California	1,283/km2
251	Temecula	California	1,170/km2
252	Springfield	Illinois	743/km2
253	Palm Bay	Florida	647/km2
254	Costa Mesa	California	2,775/km2
255	Westminster	Colorado	1,387/km2
256	North Charleston	South Carolina	573/km2
257	Miami Gardens	Florida	2,398/km2
258	Manchester	New Hampshire	1,289/km2
259	High Point	North Carolina	778/km2
260	Downey	California	3,527/km2
261	Clovis	California	1,700/km2
262	Pompano Beach	Florida	1,760/km2
263	Pueblo	Colorado	795/km2

264	Elgin	Illinois	1,158/km2
265	Lowell	Massachusetts	3,139/km2
266	Antioch	California	1,456/km2
267	West Palm Beach	Florida	758/km2
268	Peoria	Illinois	915/km2
269	Everett	Washington	1,264/km2
270	Ventura[aa]	California	1,941/km2
271	Centennial	Colorado	1,439/km2
272	Lakeland	Florida	624/km2
273	Gresham	Oregon	1,848/km2
274	Richmond	California	1,409/km2
275	Billings	Montana	975/km2
276	Inglewood	California	4,700/km2
277	Broken Arrow	Oklahoma	672/km2
278	Sandy Springs	Georgia	1,083/km2
279	Jurupa Valley	California	932/km2
280	Hillsboro	Oregon	1,624/km2
281	Waterbury	Connecticut	1,467/km2
282	Santa Maria	California	1,800/km2
283	Boulder	Colorado	1,683/km2
284	Greeley	Colorado	840/km2
285	Daly City	California	5,409/km2
286	Meridian	Idaho	1,384/km2
287	Lewisville	Texas	1,101/km2
288	Davie[ac]	Florida	1,127/km2
289	West Covina	California	2,600/km2
290	League City	Texas	769/km2
291	Tyler	Texas	715/km2
292	Norwalk	California	4,226/km2
293	San Mateo	California	3,317/km2
294	Green Bay	Wisconsin	894/km2
295	Wichita Falls	Texas	560/km2
296	Sparks	Nevada	1,080/km2
297	Lakewood[ad]	New Jersey	1,575/km2
298	Burbank	California	2,318/km2
299	Rialto	California	1,789/km2
300	Allen	Texas	1,434/km2
301	El Cajon	California	2,763/km2
302	Las Cruces	New Mexico	511/km2
303	Renton	Washington	1,666/km2
304	Davenport	Iowa	630/km2
305	South Bend	Indiana	949/km2
306	Vista	California	2,099/km2
307	Tuscaloosa	Alabama	540/km2
308	Clinton[ae]	Michigan	1,380/km2
309	Edison[ad]	New Jersey	1,309/km2
310	Woodbridge[ad]	New Jersey	1,680/km2

311	San Angelo	Texas	649/km2
312	Kenosha	Wisconsin	1,381/km2
313	Vacaville	California	1,332/km2

	Location	Radius
0	40°39 49 N 73°56 19 W / 40.6635°N 73.9387°W…	17.363755
1	34°01 10 N 118°24 39 W / 34.0194°N 118.4108°…	21.649480
2	41°50 15 N 87°40 54 W / 41.8376°N 87.6818°W…	15.076472
3	29°47 12 N 95°23 27 W / 29.7866°N 95.3909°W…	25.248762
4	33°34 20 N 112°05 24 W / 33.5722°N 112.0901°…	22.750824
5	40°00 34 N 75°08 00 W / 40.0094°N 75.1333°W…	11.584472
6	29°28 21 N 98°31 30 W / 29.4724°N 98.5251°W…	21.470911
7	32°48 55 N 117°08 06 W / 32.8153°N 117.1350°…	18.033303
8	32°47 36 N 96°45 59 W / 32.7933°N 96.7665°W…	18.463477
9	37°17 48 N 121°49 08 W / 37.2967°N 121.8189°…	13.322913
10	30°18 14 N 97°45 16 W / 30.3039°N 97.7544°W…	17.683325
11	30°20 13 N 81°39 42 W / 30.3369°N 81.6616°W…	27.338617
12	32°46 53 N 97°20 48 W / 32.7815°N 97.3467°W…	18.517559
13	39°59 07 N 82°59 05 W / 39.9852°N 82.9848°W…	14.781745
14	37°43 38 N 123°01 56 W / 37.7272°N 123.0322°…	6.848357
15	35°12 28 N 80°49 52 W / 35.2078°N 80.8310°W…	17.475697
16	39°46 36 N 86°08 45 W / 39.7767°N 86.1459°W…	19.013153
17	47°37 14 N 122°21 03 W / 47.6205°N 122.3509°…	9.154234
18	39°45 43 N 104°52 52 W / 39.7619°N 104.8811°…	12.381438
19	38°54 15 N 77°01 02 W / 38.9041°N 77.0172°W…	7.816649
20	42°19 55 N 71°01 13 W / 42.3320°N 71.0202°W…	6.949820
21	31°50 54 N 106°25 37 W / 31.8484°N 106.4270°…	16.024980
22	42°22 59 N 83°06 08 W / 42.3830°N 83.1022°W…	11.781341
23	36°10 18 N 86°47 06 W / 36.1718°N 86.7850°W…	21.815132
24	45°32 13 N 122°39 00 W / 45.5370°N 122.6500°…	11.554220
25	35°06 10 N 89°58 39 W / 35.1028°N 89.9774°W…	17.815723
26	35°28 02 N 97°30 49 W / 35.4671°N 97.5137°W…	24.623160
27	36°13 45 N 115°15 36 W / 36.2292°N 115.2601°…	11.593101
28	38°09 55 N 85°38 51 W / 38.1654°N 85.6474°W…	16.232683
29	39°18 00 N 76°36 38 W / 39.3000°N 76.6105°W…	8.994443
30	43°03 48 N 87°58 00 W / 43.0633°N 87.9667°W…	9.808160
31	35°06 20 N 106°38 51 W / 35.1056°N 106.6474°…	13.718601
32	32°09 11 N 110°52 14 W / 32.1531°N 110.8706°…	15.192103
33	36°47 01 N 119°47 36 W / 36.7836°N 119.7934°…	10.695794
34	33°24 07 N 111°43 03 W / 33.4019°N 111.7174°…	11.743083
35	38°34 00 N 121°28 07 W / 38.5666°N 121.4686°…	9.894443
36	33°45 46 N 84°25 22 W / 33.7629°N 84.4227°W…	11.554220
37	39°07 30 N 94°33 04 W / 39.1251°N 94.5510°W…	17.748239
38	38°52 02 N 104°45 39 W / 38.8673°N 104.7607°…	13.985707
39	25°46 31 N 80°12 31 W / 25.7752°N 80.2086°W…	6.000000
40	35°49 50 N 78°38 30 W / 35.8306°N 78.6418°W…	12.045746
41	41°15 52 N 96°02 42 W / 41.2644°N 96.0451°W…	11.541230

42	33°48 33 N 118°09 19 W / 33.8092°N 118.1553°…	7.092249
43	36°46 48 N 76°01 31 W / 36.7800°N 76.0252°W…	15.642890
44	37°46 11 N 122°13 33 W / 37.7698°N 122.2257°…	7.476630
45	44°57 48 N 93°16 06 W / 44.9633°N 93.2683°W…	7.348469
46	36°07 40 N 95°54 08 W / 36.1279°N 95.9023°W…	14.028542
47	32°42 03 N 97°07 29 W / 32.7007°N 97.1247°W…	9.787747
48	27°58 12 N 82°28 47 W / 27.9701°N 82.4797°W…	10.648944
49	30°03 12 N 89°56 04 W / 30.0534°N 89.9345°W…	13.015376
50	37°41 27 N 97°20 45 W / 37.6907°N 97.3459°W…	12.664912
51	41°28 43 N 81°40 46 W / 41.4785°N 81.6794°W…	8.814760
52	35°19 16 N 119°01 06 W / 35.3212°N 119.0183°…	12.198361
53	39°41 17 N 104°41 23 W / 39.6880°N 104.6897°…	12.389512
54	33°51 20 N 117°45 36 W / 33.8555°N 117.7601°…	7.071068
55	21°19 27 N 157°50 51 W / 21.3243°N 157.8476°…	7.778175
56	33°44 11 N 117°52 59 W / 33.7363°N 117.8830°…	5.205766
57	33°56 17 N 117°23 36 W / 33.9381°N 117.3932°…	9.011104
58	27°45 15 N 97°10 24 W / 27.7543°N 97.1734°W…	13.213629
59	38°02 27 N 84°27 30 W / 38.0407°N 84.4583°W…	16.840428
60	37°58 35 N 121°18 48 W / 37.9763°N 121.3133°…	7.854935
61	36°00 35 N 115°02 09 W / 36.0097°N 115.0357°…	10.232302
62	44°56 56 N 93°06 15 W / 44.9489°N 93.1041°W…	7.211103
63	38°38 09 N 90°14 41 W / 38.6357°N 90.2446°W…	7.874008
64	39°08 25 N 84°30 21 W / 39.1402°N 84.5058°W…	8.797727
65	40°26 23 N 79°58 36 W / 40.4398°N 79.9766°W…	7.443118
66	36°05 42 N 79°49 37 W / 36.0951°N 79.8270°W…	11.326959
67	61°10 27 N 149°17 03 W / 61.1743°N 149.2843°…	41.311015
68	33°03 03 N 96°44 52 W / 33.0508°N 96.7479°W…	8.467585
69	40°48 38 N 96°40 49 W / 40.8105°N 96.6803°W…	9.596874
70	28°25 00 N 81°16 25 W / 28.4166°N 81.2736°W…	10.256705
71	33°40 42 N 117°46 17 W / 33.6784°N 117.7713°…	8.099383
72	40°43 27 N 74°10 21 W / 40.7242°N 74.1726°W…	4.909175
73	41°39 51 N 83°34 55 W / 41.6641°N 83.5819°W…	8.983318
74	35°58 52 N 78°54 10 W / 35.9811°N 78.9029°W…	10.478550
75	32°37 40 N 117°00 55 W / 32.6277°N 117.0152°…	7.042727
76	41°05 18 N 85°08 38 W / 41.0882°N 85.1439°W…	10.516653
77	40°42 41 N 74°03 53 W / 40.7114°N 74.0648°W…	3.847077
78	27°45 43 N 82°38 39 W / 27.7620°N 82.6441°W…	7.861298
79	27°33 37 N 99°29 21 W / 27.5604°N 99.4892°W…	10.054850
80	43°05 16 N 89°25 48 W / 43.0878°N 89.4299°W…	8.774964
81	33°16 58 N 111°51 18 W / 33.2829°N 111.8549°…	8.056054
82	42°53 33 N 78°51 35 W / 42.8925°N 78.8597°W…	6.356099
83	33°33 56 N 101°53 12 W / 33.5656°N 101.8867°…	11.162437
84	33°41 03 N 111°51 40 W / 33.6843°N 111.8611°…	13.560973
85	39°32 57 N 119°51 00 W / 39.5491°N 119.8499°…	10.358571
86	33°31 59 N 112°11 24 W / 33.5331°N 112.1899°…	7.687652
87	33°18 37 N 111°44 35 W / 33.3103°N 111.7431°…	8.246211
88	36°06 10 N 80°15 40 W / 36.1027°N 80.2610°W…	11.510864

89	36°17 09 N 115°05 38 W / 36.2857°N 115.0939°…	9.899495
90	36°55 23 N 76°14 41 W / 36.9230°N 76.2446°W…	7.300685
91	36°40 46 N 76°18 06 W / 36.6794°N 76.3018°W…	18.398369
92	32°54 35 N 96°37 49 W / 32.9098°N 96.6303°W…	7.549834
93	32°51 28 N 96°58 12 W / 32.8577°N 96.9700°W…	8.185353
94	25°52 12 N 80°18 10 W / 25.8699°N 80.3029°W…	4.636809
95	37°29 40 N 121°56 28 W / 37.4945°N 121.9412°…	8.803408
96	43°36 01 N 116°13 54 W / 43.6002°N 116.2317°…	9.060905
97	37°31 53 N 77°28 34 W / 37.5314°N 77.4760°W…	7.733046
98	30°26 32 N 91°07 51 W / 30.4422°N 91.1309°W…	9.268225
99	47°40 01 N 117°26 00 W / 47.6669°N 117.4333°…	8.288546
100	41°34 21 N 93°36 37 W / 41.5726°N 93.6102°W…	9.428680
101	47°15 08 N 122°27 35 W / 47.2522°N 122.4598°…	7.049823
102	34°08 30 N 117°17 37 W / 34.1416°N 117.2936°…	7.842194
103	37°38 15 N 121°00 11 W / 37.6375°N 121.0030°…	6.557439
104	34°06 32 N 117°27 46 W / 34.1090°N 117.4629°…	6.557439
105	34°24 11 N 118°30 15 W / 34.4030°N 118.5042°…	7.266361
106	33°31 39 N 86°47 56 W / 33.5274°N 86.7990°W…	12.087183
107	34°12 08 N 119°12 17 W / 34.2023°N 119.2046°…	5.186521
108	35°04 58 N 78°58 25 W / 35.0828°N 78.9735°W…	12.153189
109	33°55 24 N 117°12 21 W / 33.9233°N 117.2057°…	7.162402
110	43°10 12 N 77°37 01 W / 43.1699°N 77.6169°W…	5.983310
111	34°10 53 N 118°14 45 W / 34.1814°N 118.2458°…	5.513620
112	33°41 26 N 118°00 33 W / 33.6906°N 118.0093°…	5.186521
113	40°46 37 N 111°55 52 W / 40.7769°N 111.9310°…	10.545141
114	42°57 40 N 85°39 20 W / 42.9612°N 85.6556°W…	6.663332
115	35°12 00 N 101°49 49 W / 35.1999°N 101.8302°…	10.069757
116	40°56 45 N 73°52 03 W / 40.9459°N 73.8674°W…	4.242641
117	41°45 49 N 88°17 24 W / 41.7635°N 88.2901°W…	6.700746
118	32°20 50 N 86°15 58 W / 32.3472°N 86.2661°W…	12.641202
119	41°04 50 N 81°31 17 W / 41.0805°N 81.5214°W…	7.874008
120	34°43 31 N 92°21 31 W / 34.7254°N 92.3586°W…	10.894953
121	34°41 56 N 86°40 23 W / 34.6990°N 86.6730°W…	14.608217
122	33°21 56 N 82°04 24 W / 33.3655°N 82.0734°W…	17.392527
123	27°16 50 N 80°23 18 W / 27.2806°N 80.3883°W…	10.904128
124	32°41 13 N 97°01 16 W / 32.6869°N 97.0211°W…	8.502941
125	32°30 37 N 84°52 30 W / 32.5102°N 84.8749°W…	14.710540
126	30°27 18 N 84°15 12 W / 30.4551°N 84.2534°W…	10.019980
127	38°53 20 N 94°41 26 W / 38.8890°N 94.6906°W…	8.666026
128	33°23 18 N 111°55 54 W / 33.3884°N 111.9318°…	6.324555
129	33°11 55 N 96°40 05 W / 33.1985°N 96.6680°W…	7.937254
130	30°40 06 N 88°06 01 W / 30.6684°N 88.1002°W…	11.806778
131	26°38 36 N 81°59 51 W / 26.6432°N 81.9974°W…	10.276186
132	32°28 01 N 93°47 32 W / 32.4669°N 93.7922°W…	10.348913
133	33°09 19 N 96°49 21 W / 33.1554°N 96.8226°W…	8.228001
134	35°58 15 N 83°56 57 W / 35.9707°N 83.9493°W…	9.924717
135	42°16 10 N 71°48 28 W / 42.2695°N 71.8078°W…	6.115554

136	25°59 57 N 97°27 18 W / 25.9991°N 97.4550°W…	11.510864
137	45°38 06 N 122°35 45 W / 45.6349°N 122.5957°…	6.848357
138	26°08 28 N 80°08 48 W / 26.1412°N 80.1467°W…	5.882176
139	43°32 18 N 96°43 55 W / 43.5383°N 96.7320°W…	8.683317
140	34°02 22 N 117°36 15 W / 34.0394°N 117.6042°…	7.063993
141	35°03 58 N 85°14 54 W / 35.0660°N 85.2484°W…	11.962441
142	41°49 23 N 71°25 08 W / 41.8231°N 71.4188°W…	4.289522
143	37°04 34 N 76°31 19 W / 37.0762°N 76.5220°W…	8.312641
144	34°07 24 N 117°33 51 W / 34.1233°N 117.5642°…	6.324555
145	38°26 48 N 122°42 22 W / 38.4468°N 122.7061°…	6.426508
146	33°13 28 N 117°18 22 W / 33.2245°N 117.3062°…	6.426508
147	44°55 25 N 123°01 24 W / 44.9237°N 123.0232°…	6.971370
148	38°24 53 N 121°23 06 W / 38.4146°N 121.3850°…	6.496153
149	33°46 44 N 117°57 38 W / 33.7788°N 117.9605°…	4.242641
150	26°01 16 N 80°20 25 W / 26.0210°N 80.3404°W…	5.744563
151	33°47 10 N 112°18 29 W / 33.7862°N 112.3080°…	13.255188
152	44°03 24 N 123°06 58 W / 44.0567°N 123.1162°…	6.640783
153	33°51 43 N 117°33 56 W / 33.8620°N 117.5655°…	6.284903
154	35°46 51 N 78°48 48 W / 35.7809°N 78.8133°W…	7.516648
155	37°11 39 N 93°17 29 W / 37.1942°N 93.2913°W…	9.071935
156	40°32 54 N 105°03 53 W / 40.5482°N 105.0648°…	7.469940
157	32°18 57 N 90°12 46 W / 32.3158°N 90.2128°W…	10.535654
158	38°49 12 N 77°05 03 W / 38.8201°N 77.0841°W…	3.872983
159	37°37 43 N 122°06 09 W / 37.6287°N 122.1024°…	6.745369
160	34°41 37 N 118°10 31 W / 34.6936°N 118.1753°…	9.710819
161	39°41 56 N 105°07 03 W / 39.6989°N 105.1176°…	6.549809
162	36°33 59 N 87°20 43 W / 36.5664°N 87.3452°W…	9.914636
163	34°35 28 N 118°06 19 W / 34.5910°N 118.1054°…	10.295630
164	36°41 25 N 121°38 01 W / 36.6902°N 121.6337°…	4.857983
165	42°06 56 N 72°32 24 W / 42.1155°N 72.5400°W…	5.648008
166	26°01 52 N 80°09 53 W / 26.0310°N 80.1646°W…	5.224940
167	29°39 31 N 95°09 02 W / 29.6586°N 95.1506°W…	6.595453
168	37°23 09 N 122°01 35 W / 37.3858°N 122.0263°…	4.690416
169	32°48 32 N 83°41 39 W / 32.8088°N 83.6942°W…	15.789237
170	39°07 21 N 94°44 30 W / 39.1225°N 94.7418°W…	11.171392
171	34°03 31 N 117°45 40 W / 34.0585°N 117.7611°…	4.795832
172	33°07 59 N 117°04 26 W / 33.1331°N 117.0740°…	6.090977
173	31°04 40 N 97°43 55 W / 31.0777°N 97.7320°W…	7.314369
174	41°44 57 N 88°09 43 W / 41.7492°N 88.1620°W…	6.220932
175	41°31 04 N 88°08 56 W / 41.5177°N 88.1488°W…	8.024961
176	47°35 52 N 122°09 23 W / 47.5979°N 122.1565°…	5.787918
177	42°15 32 N 89°03 53 W / 42.2588°N 89.0646°W…	7.968689
178	32°00 09 N 81°09 13 W / 32.0025°N 81.1536°W…	10.178409
179	40°54 53 N 74°09 46 W / 40.9148°N 74.1628°W…	2.898275
180	33°50 06 N 118°20 29 W / 33.8350°N 118.3414°…	4.527693
181	41°11 15 N 73°11 45 W / 41.1874°N 73.1958°W…	4.012481
182	26°13 56 N 98°14 47 W / 26.2322°N 98.2464°W…	7.641989

183	32°45 46 N 96°35 20 W / 32.7629°N 96.5888°W…	6.870226
184	43°02 28 N 76°08 37 W / 43.0410°N 76.1436°W…	5.000000
185	32°01 29 N 102°06 49 W / 32.0246°N 102.1135°…	8.625543
186	34°09 38 N 118°08 23 W / 34.1606°N 118.1396°…	4.795832
187	35°51 08 N 86°24 58 W / 35.8522°N 86.4160°W…	7.476630
188	25°58 37 N 80°20 09 W / 25.9770°N 80.3358°W…	5.422177
189	39°46 39 N 84°11 59 W / 39.7774°N 84.1996°W…	7.463243
190	33°53 09 N 117°55 41 W / 33.8857°N 117.9280°…	4.732864
191	38°53 03 N 94°49 10 W / 38.8843°N 94.8195°W…	7.803845
192	33°47 13 N 117°51 41 W / 33.7870°N 117.8613°…	5.039841
193	39°55 10 N 104°56 34 W / 39.9194°N 104.9428°…	5.974948
194	38°46 08 N 121°19 08 W / 38.7690°N 121.3189°…	6.557439
195	33°13 00 N 97°08 29 W / 33.2166°N 97.1414°W…	9.664368
196	31°33 36 N 97°11 10 W / 31.5601°N 97.1860°W…	9.433981
197	33°40 14 N 112°27 10 W / 33.6706°N 112.4527°…	10.387492
198	32°59 18 N 96°53 59 W / 32.9884°N 96.8998°W…	6.024948
199	40°41 19 N 112°00 42 W / 40.6885°N 112.0118°…	5.958188
200	32°49 04 N 79°57 32 W / 32.8179°N 79.9590°W…	10.440307
201	42°29 34 N 83°01 30 W / 42.4929°N 83.0250°W…	5.865151
202	37°02 53 N 76°17 50 W / 37.0480°N 76.2971°W…	7.176350
203	29°40 44 N 82°20 46 W / 29.6788°N 82.3461°W…	7.893035
204	36°19 38 N 119°19 44 W / 36.3273°N 119.3289°…	6.123724
205	26°16 15 N 80°15 33 W / 26.2707°N 80.2593°W…	4.878524
206	34°01 45 N 80°53 53 W / 34.0291°N 80.8980°W…	11.554220
207	41°58 01 N 91°40 40 W / 41.9670°N 91.6778°W…	8.414274
208	42°34 52 N 83°01 49 W / 42.5812°N 83.0303°W…	6.041523
209	41°18 39 N 72°55 30 W / 41.3108°N 72.9250°W…	4.324350
210	41°04 48 N 73°32 46 W / 41.0799°N 73.5460°W…	6.131884
211	37°58 20 N 122°00 06 W / 37.9722°N 122.0016°…	5.522681
212	47°23 17 N 122°12 46 W / 47.3880°N 122.2127°…	5.805170
213	37°21 53 N 121°58 04 W / 37.3646°N 121.9679°…	4.289522
214	40°39 59 N 74°11 37 W / 40.6664°N 74.1935°W…	3.507136
215	30°31 31 N 97°39 58 W / 30.5252°N 97.6660°W…	5.966574
216	34°11 36 N 118°52 27 W / 34.1933°N 118.8742°…	7.429670
217	30°12 27 N 92°01 43 W / 30.2074°N 92.0285°W…	7.334848
218	33°56 59 N 83°22 12 W / 33.9496°N 83.3701°W…	10.788883
219	39°02 05 N 95°41 46 W / 39.0347°N 95.6962°W…	7.842194
220	34°16 01 N 118°44 55 W / 34.2669°N 118.7485°…	6.442049
221	46°51 55 N 96°49 44 W / 46.8652°N 96.8290°W…	7.021396
222	35°14 26 N 97°20 43 W / 35.2406°N 97.3453°W…	13.371612
223	38°57 06 N 92°19 43 W / 38.951561°N 92.32863…	8.062258
224	32°27 16 N 99°44 17 W / 32.4545°N 99.7381°W…	10.329569
225	34°12 33 N 77°53 09 W / 34.2092°N 77.8858°W…	7.183314
226	41°45 57 N 72°40 54 W / 41.7659°N 72.6816°W…	4.171331
227	34°31 40 N 117°21 13 W / 34.5277°N 117.3536°…	8.561542
228	29°33 21 N 95°19 23 W / 29.5558°N 95.3231°W…	6.804410
229	38°06 28 N 122°15 50 W / 38.1079°N 122.2640°…	5.540758

230	42°16 34 N 83°43 51 W / 42.2761°N 83.7309°W…	5.300943
231	37°52 01 N 122°17 57 W / 37.8670°N 122.2991°…	3.240370
232	40°35 37 N 75°28 42 W / 40.5936°N 75.4784°W…	4.183300
233	32°58 20 N 96°42 29 W / 32.9723°N 96.7081°W…	5.347897
234	31°53 02 N 102°20 28 W / 31.8838°N 102.3411°…	6.723095
235	39°50 01 N 105°09 01 W / 39.8337°N 105.1503°…	6.212890
236	42°22 34 N 71°07 07 W / 42.3760°N 71.1187°W…	2.529822
237	29°35 58 N 95°36 51 W / 29.5994°N 95.6142°W…	5.830952
238	30°05 06 N 94°08 43 W / 30.0849°N 94.1453°W…	9.060905
239	42°42 51 N 84°33 33 W / 42.7143°N 84.5593°W…	6.252999
240	37°59 16 N 87°32 05 W / 37.9877°N 87.5347°W…	6.877500
241	44°00 55 N 92°28 38 W / 44.0154°N 92.4772°W…	7.389181
242	39°05 08 N 94°21 08 W / 39.0855°N 94.3521°W…	8.820431
243	38°15 33 N 122°01 56 W / 38.2593°N 122.0321°…	6.395311
244	40°14 43 N 111°38 41 W / 40.2453°N 111.6448°…	6.457554
245	27°58 44 N 82°46 00 W / 27.9789°N 82.7666°W…	5.089204
246	30°35 07 N 96°17 47 W / 30.5852°N 96.2964°W…	7.141428
247	40°36 09 N 112°00 03 W / 40.6024°N 112.0008°…	5.683309
248	33°07 26 N 117°16 58 W / 33.1239°N 117.2828°…	6.140033
249	34°04 29 N 118°01 45 W / 34.0746°N 118.0291°…	3.098387
250	33°34 20 N 117°11 25 W / 33.5721°N 117.1904°…	5.796551
251	33°29 35 N 117°07 54 W / 33.4931°N 117.1317°…	6.107373
252	39°47 28 N 89°38 41 W / 39.7911°N 89.6446°W…	7.752419
253	27°59 08 N 80°39 45 W / 27.9856°N 80.6626°W…	8.105554
254	33°39 57 N 117°54 44 W / 33.6659°N 117.9123°…	3.962323
255	39°52 56 N 105°03 52 W / 39.8822°N 105.0644°…	5.630275
256	32°55 04 N 80°03 54 W / 32.9178°N 80.0650°W…	8.584870
257	25°56 56 N 80°14 37 W / 25.9489°N 80.2436°W…	4.266146
258	42°59 06 N 71°26 39 W / 42.9849°N 71.4441°W…	5.753260
259	35°59 24 N 79°59 26 W / 35.9900°N 79.9905°W…	7.429670
260	33°56 18 N 118°07 51 W / 33.9382°N 118.1309°…	3.521363
261	36°49 42 N 119°41 06 W / 36.8282°N 119.6849°…	4.919350
262	26°14 30 N 80°08 02 W / 26.2416°N 80.1339°W…	4.898979
263	38°16 12 N 104°36 44 W / 38.2699°N 104.6123°…	7.321202
264	42°02 23 N 88°19 18 W / 42.0396°N 88.3217°W…	6.115554
265	42°38 20 N 71°19 16 W / 42.6390°N 71.3211°W…	3.687818
266	37°58 45 N 121°47 46 W / 37.9791°N 121.7962°…	5.422177
267	26°44 47 N 80°07 30 W / 26.7464°N 80.1251°W…	7.422937
268	40°45 05 N 89°37 03 W / 40.7515°N 89.6174°W…	6.942622
269	47°57 24 N 122°11 29 W / 47.9566°N 122.1914°…	5.770615
270	34°16 04 N 119°15 15 W / 34.2678°N 119.2542°…	4.669047
271	39°35 26 N 104°52 09 W / 39.5906°N 104.8691°…	5.431390
272	28°03 20 N 81°57 18 W / 28.0555°N 81.9549°W…	8.117881
273	45°30 08 N 122°26 30 W / 45.5023°N 122.4416°…	4.827007
274	37°57 08 N 122°21 38 W / 37.9523°N 122.3606°…	5.486347
275	45°47 19 N 108°33 00 W / 45.7885°N 108.5499°…	6.610598
276	33°57 22 N 118°20 39 W / 33.9561°N 118.3443°…	3.016621

```
277  36°02 11 N 95°46 52 W / 36.0365°N 95.7810°W…   7.854935
278  33°55 53 N 84°22 07 W / 33.9315°N 84.3687°W…   6.140033
279  34°00 09 N 117°28 03 W / 34.0026°N 117.4676°…  6.549809
280  45°31 41 N 122°56 09 W / 45.5280°N 122.9357°…  5.000000
281  41°33 31 N 73°02 12 W / 41.5585°N 73.0367°W…   5.338539
282  34°56 00 N 120°26 38 W / 34.9332°N 120.4438°…  4.774935
283  40°01 37 N 105°15 07 W / 40.0270°N 105.2519°…  4.979960
284  40°24 55 N 104°46 11 W / 40.4153°N 104.7697°…  6.913754
285  37°42 03 N 122°27 54 W / 37.7009°N 122.4650°…  2.756810
286  43°36 51 N 116°23 56 W / 43.6142°N 116.3989°…  5.458938
287  33°02 48 N 96°58 54 W / 33.0466°N 96.9818°W…   6.058052
288  26°04 45 N 80°17 06 W / 26.0791°N 80.2850°W…   5.907622
289  34°03 21 N 117°54 36 W / 34.0559°N 117.9099°…  4.000000
290  29°29 24 N 95°06 33 W / 29.4901°N 95.1091°W…   7.155418
291  32°19 02 N 95°18 21 W / 32.3173°N 95.3059°W…   7.523297
292  33°54 27 N 118°05 01 W / 33.9076°N 118.0835°…  3.114482
293  37°33 37 N 122°18 38 W / 37.5603°N 122.3106°…  3.478505
294  44°31 15 N 87°59 03 W / 44.5207°N 87.9842°W…   6.737952
295  33°54 24 N 98°31 33 W / 33.9067°N 98.5259°W…   8.497058
296  39°33 16 N 119°44 08 W / 39.5544°N 119.7356°…  5.991661
297  40°04 38 N 74°12 01 W / 40.0771°N 74.2004°W…   4.969909
298  34°11 24 N 118°19 35 W / 34.1901°N 118.3264°…  4.171331
299  34°06 42 N 117°23 18 W / 34.1118°N 117.3883°…  4.722288
300  33°05 59 N 96°39 47 W / 33.0997°N 96.6631°W…   5.205766
301  32°48 06 N 116°57 37 W / 32.8017°N 116.9604°…  3.807887
302  32°19 35 N 106°47 23 W / 32.3264°N 106.7897°…  8.769265
303  47°28 34 N 122°11 31 W / 47.4761°N 122.1920°…  4.837355
304  41°33 15 N 90°36 14 W / 41.5541°N 90.6040°W…   7.930952
305  41°40 37 N 86°16 08 W / 41.6769°N 86.2690°W…   6.434283
306  33°11 22 N 117°14 19 W / 33.1895°N 117.2386°…  4.324350
307  33°12 23 N 87°32 05 W / 33.2065°N 87.5346°W…   8.467585
308  42°35 25 N 82°55 01 W / 42.5903°N 82.9170°W…   5.300943
309  40°30 14 N 74°20 58 W / 40.5040°N 74.3494°W…   5.486347
310  40°33 39 N 74°17 34 W / 40.5607°N 74.2927°W…   4.827007
311  31°26 28 N 100°27 02 W / 31.4411°N 100.4505°…  7.739509
312  42°34 56 N 87°50 44 W / 42.5822°N 87.8456°W…   5.291503
313  38°21 14 N 121°58 22 W / 38.3539°N 121.9728°…  5.385165
```

[18]:
```
#Splitting the location into latitudes and longitudes
df["Location"]= df["Location"].str.split("/", n = 2, expand = True)[1]
df.head()
```

[18]:
```
           City       State Population density in Km2  \
0  New York[d]    New York                10,933/km2
1  Los Angeles  California                 3,276/km2
2      Chicago    Illinois                 4,600/km2
3   Houston[3]       Texas                 1,395/km2
```

```
4      Phoenix     Arizona                 1,200/km2

                   Location     Radius
0    40.6635°N 73.9387°W   17.363755
1   34.0194°N 118.4108°W   21.649480
2    41.8376°N 87.6818°W   15.076472
3    29.7866°N 95.3909°W   25.248762
4   33.5722°N 112.0901°W   22.750824
```

[19]:
```
new = df["Location"].str.split(" ", n = 0, expand = False)
k = df.copy(deep = True)
```

[20]:
```
Latitude = []
Longitude = []
for i in range(len(new)):
    Latitude.append(new[i][1][:-2])
    Longitude.append(new[i][2][:-3])

k["Latitude"] = Latitude
k["Longitude"] = Longitude
k["Latitude"] = k["Latitude"].str.replace(u'\ufeff',u'')
k.drop(columns = ["Location"], inplace = True)
k.head()
df = k.copy(deep = True)
```

[21]:
```
df['Longitude'] = -df['Longitude'].astype(float)
df['Latitude'] = df['Latitude'].astype(float)
df['Radius'] = df['Radius']* 1000
df.head()
```

[21]:
```
          City       State Population density in Km2        Radius  Latitude  \
0  New York[d]    New York                10,933/km2  17363.755354   40.6635
1  Los Angeles  California                 3,276/km2  21649.480363   34.0194
2      Chicago    Illinois                 4,600/km2  15076.471736   41.8376
3   Houston[3]       Texas                 1,395/km2  25248.762346   29.7866
4      Phoenix     Arizona                 1,200/km2  22750.824161   33.5722

   Longitude
0   -73.9387
1  -118.4108
2   -87.6818
3   -95.3909
4  -112.0901
```

[24]:
```
# create map of USA cities that we have using latitude and longitude values
map_tohood = folium.Map(location=[37.0902,-95.7129], zoom_start=3)
```

```
# add markers to map
for lat, lng, state, city in zip(df['Latitude'], df['Longitude'], df['State'],␣
 ↪df['City']):
    label = '{}, {}'.format(city, state)
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=3,
        popup=label,
        color='blue',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.3,
        parse_html=False).add_to(map_tohood)

map_tohood
```

[24]:
```
<folium.folium.Map at 0x7f8b135d35f8>
```

[26]:
```
CLIENT_ID = '152A02E2S4HV45V5XL4WL4YDCENCE45MUNCTEXVPNZ5VVMUY' # your␣
 ↪Foursquare ID
CLIENT_SECRET = 'SK3OWN14KEAN2KUHLXD1AGDYJ20YNEY1F4YRXV0HNNRQDFWR' # your␣
 ↪Foursquare Secret
VERSION = '20180604'
LIMIT = 20
print('Your credentails:')
print('CLIENT_ID: ' + CLIENT_ID)
print('CLIENT_SECRET:' + CLIENT_SECRET)
```

```
Your credentails:
CLIENT_ID: 152A02E2S4HV45V5XL4WL4YDCENCE45MUNCTEXVPNZ5VVMUY
CLIENT_SECRET:SK3OWN14KEAN2KUHLXD1AGDYJ20YNEY1F4YRXV0HNNRQDFWR
```

[27]:
```
def getNearbyVenues(names, latitudes, longitudes, radius):

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

        # 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']
       # print(results)
        # 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 = ['City',
                  'Latitude',
                  'Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']

    return(nearby_venues)
```

[37]:
```
df_venues = getNearbyVenues(names = df['City'], latitudes =␣
 ↪df['Latitude'],longitudes = df['Longitude'], radius = df['Radius'])
df_venues.head()
```

[37]:

	City	Latitude	Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	New York[d]	40.6635	-73.9387	Super Power	40.673952	-73.950184	Tiki Bar
1	New York[d]	40.6635	-73.9387	Brooklyn Botanic Garden	40.667622	-73.963191	Botanical Garden
2	New York[d]	40.6635	-73.9387	Covenhoven	40.675143	-73.960203	Beer Bar
3	New York[d]	40.6635	-73.9387	Brooklyn Museum	40.671521	-73.963677	Art Museum
4	New York[d]	40.6635	-73.9387	Kings Theatre	40.646110	-73.957175	Theater

[50]:
```
k = df_venues.copy(deep = True)
```

```
weights_dict={'Movie Theater':3,'Beach':3,'Concert Hall':2.5,'Playground':
↪3,'Coffee Shop':3.5,'Food Court':4,'Nightclub':4,'Toy / Game Store':4.
↪5,'Theme Park Ride / Attraction':4,'Pub':4}
data = df_venues['Venue Category']
```

[51]:
```
weights = []
for i in data:
    if i in weights_dict.keys():
        weights.append(weights_dict[i])
    else :
        weights.append(0)
df_venues['weights'] = weights;
df_venues.head()
```

[51]:

	City	Latitude	Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category	weights
0	New York[d]	40.6635	-73.9387	Super Power	40.673952	-73.950184	Tiki Bar	0.0
1	New York[d]	40.6635	-73.9387	Brooklyn Botanic Garden	40.667622	-73.963191	Botanical Garden	0.0
2	New York[d]	40.6635	-73.9387	Covenhoven	40.675143	-73.960203	Beer Bar	0.0
3	New York[d]	40.6635	-73.9387	Brooklyn Museum	40.671521	-73.963677	Art Museum	0.0
4	New York[d]	40.6635	-73.9387	Kings Theatre	40.646110	-73.957175	Theater	0.0

[ ]:
```

```

[52]:
```
# Dropping the rows that we are not giving any weight
df_venues.drop(df_venues[df_venues.weights < 1.0].index, inplace=True)
df_venues.head()
```

[52]:

	City	Latitude	Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category	weights
21	Los Angeles	34.0194	-118.4108	Blue Bottle Coffee	34.027115	-118.387637	Coffee Shop	3.5
28	Los Angeles	34.0194	-118.4108	Blue Bottle Coffee	34.059310	-118.419797	Coffee Shop	3.5
29	Los Angeles	34.0194	-118.4108	Blue Bottle Coffee	33.980027	-118.408020	Coffee Shop	3.5
39	Los Angeles	34.0194	-118.4108	iPic Theatres	34.059093	-118.441475	Movie Theater	3.0
57	Chicago	41.8376	-87.6818	Sawada Coffee	41.883730	-87.648726	Coffee Shop	3.5

[ ]:

[ ]:

[53]:
```
citywise_venues_weights = df_venues[['City','weights']].copy()
citywise_venues_weights_means = citywise_venues_weights.groupby(['City']).mean()
citywise_venues_weights_means = citywise_venues_weights_means.
 ↪reset_index(drop=False)
citywise_venues_weights_means.head()
```

[53]:

	City	weights
0	Abilene	3.5
1	Alexandria[m]	3.5
2	Allen	2.5
3	Amarillo	3.5
4	Anaheim	3.5

[ ]:

[54]:
```
city_selection = pd.merge(df, citywise_venues_weights_means, on='City')
city_selection = city_selection[['City','Population density in Km2','weights']].
 ↪copy()
city_selection.head()
```

[54]:

	City	Population density in Km2	weights
0	Los Angeles	3,276/km2	3.375
1	Chicago	4,600/km2	3.500
2	Houston[3]	1,395/km2	2.500
3	Phoenix	1,200/km2	3.500
4	Philadelphia[e]	4,511/km2	3.500

[ ]:

[55]:
```
# Preprocessing the population density in Km2 column as we have to normalize␣
 ↪these values
k = city_selection.copy(deep = True)
k['Population density in Km2'] = k['Population density in Km2'].str.split("/",␣
 ↪n = 0, expand = True)
k['Population density in Km2'] = k['Population density in Km2'].str.
 ↪replace(',','')
k['Population density in Km2'] = k['Population density in Km2'].astype(float)
city_selection = k.copy(deep = True)
city_selection.head()
```

[55]:

	City	Population density in Km2	weights
0	Los Angeles	3276.0	3.375
1	Chicago	4600.0	3.500

```
2       Houston[3]                    1395.0    2.500
3          Phoenix                    1200.0    3.500
4  Philadelphia[e]                    4511.0    3.500
```

[ ]:

[57]:
```
 ##Normalizing the data frame
from sklearn import preprocessing
column_names_to_normalize = ['Population density in Km2', 'weights']
x = city_selection[column_names_to_normalize].values #returns a numpy array
min_max_scaler = preprocessing.MinMaxScaler()
x_scaled = min_max_scaler.fit_transform(x)
city_selection[column_names_to_normalize] = pd.DataFrame(x_scaled)
city_selection.head()
```

[57]:

	City	Population density in Km2	weights
0	Los Angeles	0.470174	0.4375
1	Chicago	0.664224	0.5000
2	Houston[3]	0.194489	0.0000
3	Phoenix	0.165909	0.5000
4	Philadelphia[e]	0.651180	0.5000

[58]:
```
#calculating the sum of normalized columns to determine the city that has␣
 ↪maximum sum and conclude that one locality in that city would be the best␣
 ↪fit
city_selection['sum'] = city_selection['Population density in Km2'] +␣
 ↪city_selection['weights']
row_num = city_selection['sum'].argmax()
city_name = city_selection['City'].iloc[row_num]
city_name
```

```
/home/jupyterlab/conda/envs/python/lib/python3.6/site-
packages/ipykernel_launcher.py:3: FutureWarning:
The current behaviour of 'Series.argmax' is deprecated, use 'idxmax'
instead.
The behavior of 'argmax' will be corrected to return the positional
maximum in the future. For now, use 'series.values.argmax' or
'np.argmax(np.array(values))' to get the position of the maximum
row.
  This is separate from the ipykernel package so we can avoid doing imports
until
```

[58]:
```
'Jersey City'
```

[59]:
```
# Finding the state in which that city belongs
row = df.loc[df['City']== city_name].index[0]
state_name = df['State'].iloc[row]
```

```
state_name
```

[59]: 'New Jersey'

[61]:
```
# Getting coordinates of New Jersey
lat_newJercy = df['Latitude'].iloc[row]
long_newJercy = df['Longitude'].iloc[row]
print(lat_newJercy, long_newJercy)
```

```
40.7114 -74.0648
```

[62]:
```
# Getting the venues of New Jersey using four square API

def getNearbyVenues1(name, latitudes, longitudes, radius):

    LIMIT = 150
        # 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,
            latitudes,
            longitudes,
            radius,
            LIMIT)

        # make the GET request
    results = requests.get(url).json()["response"]['groups'][0]['items']
   # print(results)
    venues_list=[]
    venues_list.
 ↪append([(name,lat,lng,v['venue']['name'],v['venue']['location']['lat'],v['venue']['location
 ↪v in results])
    nearby_venues = pd.DataFrame([item for venue_list in venues_list for item␣
 ↪in venue_list])
    nearby_venues.columns = ['City', 'Latitude', 'Longitude', 'Venue', 'Venue␣
 ↪Latitude', 'Venue Longitude','Venue Category']
    return(nearby_venues)


new_jersey_venues = getNearbyVenues1(name = 'Jersey City', latitudes =␣
 ↪lat_newJercy ,longitudes = long_newJercy, radius = 2500)
new_jersey_venues.head()
```

[62]:

	City	Latitude	Longitude	\
0	Jersey City	38.3539	-121.9728	

```
1  Jersey City   38.3539  -121.9728
2  Jersey City   38.3539  -121.9728
3  Jersey City   38.3539  -121.9728
4  Jersey City   38.3539  -121.9728

                                      Venue  Venue Latitude  \
0                           The Grind Shop       40.711670
1                          Harry's Daughter       40.710904
2  Corgi Spirits at The Jersey City Distillery       40.708304
3                                Hooked JC       40.714709
4                     Liberty Science Center       40.707881

   Venue Longitude        Venue Category
0       -74.062872           Coffee Shop
1       -74.062071  Caribbean Restaurant
2       -74.064803            Distillery
3       -74.067009     Fish & Chips Shop
4       -74.055121        Science Museum
```

[63]:
```
venues_in_newjersey = new_jersey_venues.copy(deep = True)
venues_in_newjersey.shape
```

[63]:
```
(100, 7)
```

[64]:
```
#copying the data frame in and giving weights for each category
k = new_jersey_venues.copy(deep = True)
new_weightage_dict= {'Coffee Shop' : 3,
'Caribbean Restaurant':3,
'Distillery':2,
'Fish & Chips Shop':3,
'Science Museum':3,
'Latin American Restaurant':4,
'Restaurant':5,
'State / Provincial Park':1,
'Diner':1,
'Supermarket':1,
'Bar':1,
'Jazz Club':1,
'Golf Course':3,
'Park':2,
'Cajun / Creole Restaurant':2,
'Bakery':2,
'Go Kart Track':3,
'Taco Place':3,
'Hot Dog Joint':2,
'Food Truck':3,
'Beer Garden':3,
```

```
'Boutique':4,
'Café':5,
'Bagel Shop':1,
'Record Shop':1,
'Bakery':1,
'Pizza Place':1,
'Ramen Restaurant':1,
'Wine Bar':3,
'Middle Eastern Restaurant':2,
'French Restaurant':2,
'Theater':2,
'Lounge':3,
'Wine Shop':3,
'Cocktail Bar':2,
'New American Restaurant':3,
'Residential Building (Apartment / Condo)':3,
'Pool':4,
'Burger Joint':5,
'Cheese Shop':1,
'Coffee Shop':1,
'Bagel Shop':1,
'Vietnamese Restaurant':1,
'Portuguese Restaurant':1,
'Ice Cream Shop':3,
'Italian Restaurant':2,
'Gym':2,
'Farmers Market':2,
'Bar':3,
'Pizza Place':3,
'Bakery':2,
'Bookstore':3,
'Bar':3,
'Farmers Market':4,
'Asian Restaurant':5,
'Tea Room':1,
'Donut Shop':1,
'Historic Site':1,
'Gym / Fitness Center':1,
'Café':1,
'Mexican Restaurant':3,
'Plaza':2,
'Gay Bar':2,
'Bar':3,
'College Administrative Building':3,
'Mexican Restaurant':2,
'Bakery':3,
'American Restaurant':3,
```

```
'American Restaurant':4,
'American Restaurant':5,
'Café':1,
'New American Restaurant':1,
'Chocolate Shop':1,
'Gym':1,
'Grocery Store':1,
'Middle Eastern Restaurant':3,
'American Restaurant':2,
'Frozen Yogurt Shop':2,
'Japanese Restaurant':2,
'Bar':3,
'Liquor Store':3,
'Ice Cream Shop':2,
'Fish Market':3,
'Indie Movie Theater':3,
'Grocery Store':4,
'Modern European Restaurant':5,
'American Restaurant':1,
'Poke Place':1,
'Ramen Restaurant':1,
'Diner':1,
'Brewery':1,
'Burger Joint':3,
'Burger Joint':2,
'Café':2,
'Fried Chicken Joint':2,
'Beer Garden':3,
'Gym / Fitness Center':3,
'Vietnamese Restaurant':2,
'Italian Restaurant':3,
'Pet Store':3}
```

[65]:
```
import matplotlib.cm as cm
import matplotlib.colors as colors
import folium

# create map of the venues that we have using latitude and longitudes
venues_map = folium.Map(location=[lat_newJercy, long_newJercy], zoom_start=15)␣
 ↪# generate map centred around Jersey city


# add Jersey City as a red circle mark
folium.features.CircleMarker(
    [lat_newJercy, long_newJercy],
    radius=10,
    popup='Jersey city',
```

```
    fill=True,
    color='red',
    fill_color='red',
    fill_opacity=0.6
    ).add_to(venues_map)
```

[65]: <folium.features.CircleMarker at 0x7f8b12bd7f60>

[66]:
```
# add all the venuew of the Jersey city to the map as blue circle markers
for lat, lng, label in zip(venues_in_newjersey['Venue Latitude'],␣
 ↪venues_in_newjersey['Venue Longitude'], venues_in_newjersey['Venue']):
    label=folium.Popup(label,parse_html=True)
    folium.features.CircleMarker(
        [lat, lng],
        radius=5,
        popup=label,
        color='blue',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.6,
        parse_html = False).add_to(venues_map)
venues_map
```

[66]: <folium.folium.Map at 0x7f8b12bb2eb8>

[67]:
```
# Calculating new weights for our data frame as we have given weights for all␣
 ↪categories

allVenuesinCity1 = k['Venue Category']

f_weights1 = []
for i in allVenuesinCity1:
    if i in new_weightage_dict.keys():
        f_weights1.append(new_weightage_dict[i])
    else :
        f_weights1.append(0)
k['weights'] = f_weights1;
k.head()
```

[67]:

	City	Latitude	Longitude	\
0	Jersey City	38.3539	-121.9728	
1	Jersey City	38.3539	-121.9728	
2	Jersey City	38.3539	-121.9728	
3	Jersey City	38.3539	-121.9728	
4	Jersey City	38.3539	-121.9728	

	Venue	Venue Latitude	\

```
0                              The Grind Shop       40.711670
1                            Harry's Daughter       40.710904
2  Corgi Spirits at The Jersey City Distillery       40.708304
3                                  Hooked JC       40.714709
4                      Liberty Science Center       40.707881

   Venue Longitude        Venue Category  weights
0       -74.062872           Coffee Shop        1
1       -74.062071  Caribbean Restaurant        3
2       -74.064803            Distillery        2
3       -74.067009     Fish & Chips Shop        3
4       -74.055121        Science Museum        3
```

[69]:
```
# Dropping unnecessary columns

newframe = k[['City','Venue Category','weights']].copy()
newframe = k.groupby(['Venue Category']).mean()
newframe.drop(columns = ["Latitude", "Longitude"], inplace = True)
newframe
```

[69]:
```
                                      Venue Latitude  Venue Longitude  \
Venue Category
American Restaurant                        40.715969       -74.041594
Australian Restaurant                      40.717187       -74.044216
Bagel Shop                                 40.722990       -74.058068
Bakery                                     40.721297       -74.048836
Bar                                        40.718000       -74.056019
Beer Garden                                40.715149       -74.046633
Bookstore                                  40.719984       -74.043205
Boutique                                   40.717606       -74.044299
Brewery                                    40.720660       -74.040287
Burger Joint                               40.724874       -74.048082
Café                                       40.718949       -74.055604
Cajun / Creole Restaurant                  40.718230       -74.074402
Caribbean Restaurant                       40.710904       -74.062071
Cheese Shop                                40.721000       -74.046444
Chinese Restaurant                         40.719615       -74.043459
Chocolate Shop                             40.719766       -74.041235
Cocktail Bar                               40.721296       -74.045213
Coffee Shop                                40.715548       -74.049473
College Administrative Building            40.709881       -74.086332
Diner                                      40.707950       -74.057756
Distillery                                 40.708304       -74.064803
Donut Shop                                 40.718765       -74.041762
Dumpling Restaurant                        40.710330       -74.061160
Farmers Market                             40.718920       -74.044960
Fish & Chips Shop                          40.714709       -74.067009
```

Fish Market	40.720307	-74.046839
Food Truck	40.718258	-74.047446
French Restaurant	40.724861	-74.051592
Fried Chicken Joint	40.717814	-74.052620
Frozen Yogurt Shop	40.720100	-74.042963
Gay Bar	40.718950	-74.043687
Gift Shop	40.717548	-74.043721
Go Kart Track	40.700620	-74.071892
Golf Course	40.695965	-74.072566
Grocery Store	40.720139	-74.042991
Gym	40.716135	-74.043623
Gym / Fitness Center	40.719804	-74.042395
Hot Dog Joint	40.720389	-74.046651
Ice Cream Shop	40.721280	-74.042155
Indie Movie Theater	40.731967	-74.063941
Italian Restaurant	40.720086	-74.065450
Japanese Restaurant	40.721283	-74.046306
Jazz Club	40.721190	-74.070448
Korean Restaurant	40.717562	-74.052448
Latin American Restaurant	40.717446	-74.072934
Liquor Store	40.719562	-74.047235
Lounge	40.725010	-74.051419
Mexican Restaurant	40.720727	-74.042765
Middle Eastern Restaurant	40.718828	-74.045839
Modern European Restaurant	40.714431	-74.037769
New American Restaurant	40.719276	-74.044454
Park	40.720688	-74.064565
Pizza Place	40.718953	-74.043881
Plaza	40.720729	-74.044363
Pool	40.715670	-74.047576
Portuguese Restaurant	40.718589	-74.043527
Ramen Restaurant	40.722340	-74.046990
Record Shop	40.722842	-74.050888
Residential Building (Apartment / Condo)	40.720690	-74.050067
Restaurant	40.710312	-74.061406
Science Museum	40.707881	-74.055121
State / Provincial Park	40.702635	-74.050941
Supermarket	40.718434	-74.052564
Taco Place	40.716208	-74.044789
Tea Room	40.723066	-74.049044
Theater	40.724996	-74.051542
Vietnamese Restaurant	40.720328	-74.064457
Wine Bar	40.723776	-74.050355
Wine Shop	40.717339	-74.043989

Venue Category	weights

American Restaurant	1
Australian Restaurant	0
Bagel Shop	1
Bakery	3
Bar	3
Beer Garden	3
Bookstore	3
Boutique	4
Brewery	1
Burger Joint	2
Café	2
Cajun / Creole Restaurant	2
Caribbean Restaurant	3
Cheese Shop	1
Chinese Restaurant	0
Chocolate Shop	1
Cocktail Bar	2
Coffee Shop	1
College Administrative Building	3
Diner	1
Distillery	2
Donut Shop	1
Dumpling Restaurant	0
Farmers Market	4
Fish & Chips Shop	3
Fish Market	3
Food Truck	3
French Restaurant	2
Fried Chicken Joint	2
Frozen Yogurt Shop	2
Gay Bar	2
Gift Shop	0
Go Kart Track	3
Golf Course	3
Grocery Store	4
Gym	1
Gym / Fitness Center	3
Hot Dog Joint	2
Ice Cream Shop	2
Indie Movie Theater	3
Italian Restaurant	3
Japanese Restaurant	2
Jazz Club	1
Korean Restaurant	0
Latin American Restaurant	4
Liquor Store	3
Lounge	3

```
Mexican Restaurant                          2
Middle Eastern Restaurant                   3
Modern European Restaurant                  5
New American Restaurant                     1
Park                                        2
Pizza Place                                 3
Plaza                                       2
Pool                                        4
Portuguese Restaurant                       1
Ramen Restaurant                            1
Record Shop                                 1
Residential Building (Apartment / Condo)    3
Restaurant                                  5
Science Museum                              3
State / Provincial Park                     1
Supermarket                                 1
Taco Place                                  3
Tea Room                                    1
Theater                                     2
Vietnamese Restaurant                       2
Wine Bar                                    3
Wine Shop                                   3
```

[ ]:

[71]:
```
# Cluster them using K means algorithm
from scipy import stats
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
import seaborn as sns
#Standardize
clmns = ['weights','Venue Latitude', 'Venue Longitude']
df_tr_std = stats.zscore(newframe[clmns])
#Cluster the data
kmeans = KMeans(n_clusters=3, random_state=0).fit(df_tr_std)
labels = kmeans.labels_
newframe['clusters'] = labels
#Add the column into our list
clmns.extend(['clusters'])
#Lets analyze the clusters
kframe = newframe[clmns].groupby(['Venue Category']).mean()
kframe = kframe.reset_index(drop = False)
kframe.head()
```

[71]:

	Venue Category	weights	Venue Latitude	Venue Longitude	clusters
0	American Restaurant	1	40.715969	-74.041594	1
1	Australian Restaurant	0	40.717187	-74.044216	1

```
2              Bagel Shop        1       40.722990       -74.058068         1
3                  Bakery        3       40.721297       -74.048836         0
4                     Bar        3       40.718000       -74.056019         0
```

[ ]:

[ ]:

[72]:
```
#new group by clusters and add weights of each cluster
finalWeight = kframe.groupby(['clusters']).mean()
finalWeight
```

[72]:
```
            weights  Venue Latitude  Venue Longitude
clusters
0          3.272727       40.719636       -74.047997
1          1.323529       40.720169       -74.048436
2          2.538462       40.708859       -74.066030
```

[73]:
```
# Final coordinates of the place where we will be setting up an arcade is the␣
 ↪one that has maximum weight for, in the above data frame
lat1 = 40.720102
long1 = -74.048121
```

[74]:
```
# create map of the venues that we have using latitude and longitudes
final_map = folium.Map(location=[lat1, long1], zoom_start=15) # generate map␣
 ↪centred around Jersey city


# add prefered location in the City as a green circle mark
folium.features.CircleMarker(
    [lat1, long1],
    radius=50,
    popup='Gaming arcade can be installed within this circle',
    fill=True,
    color='green',
    fill_color='green',
    fill_opacity=0.6
    ).add_to(final_map)
final_map
```

[74]:
```
<folium.folium.Map at 0x7f8b129f23c8>
```

1.2 Conclusion

In conclusion, we managed to get a better place in the Jersey city which occurse between the Groove Street and the Grand Street, hence it would also have the best footfall and potential customers as well.However, this project can be enhanced by

considering many more attributes to define the weights and do the analysis and also by extending the LIMIT and Radius of the search that we are giving to extract the number of venues. As we have an API limit in the free trail of four square API we had to limit our search within a small Radius.

I hope this notebook would be beneficial for all who reviews and read as I personally going to make use of it when considering moving to a new place :).

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]: