In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import folium
from folium import plugins
from sklearn.cluster import KMeans
import matplotlib
from scipy import stats
import math
from mpl_toolkits.mplot3d import Axes3D
```

In [2]:
```
df = pd.read_csv('AB_NYC_2019.csv')

# drops rows with price = 0
df = df[df.price != 0]

df = df[:]
print(df.shape)
df.head()
```

```
(48884, 16)
```

Out[2]:

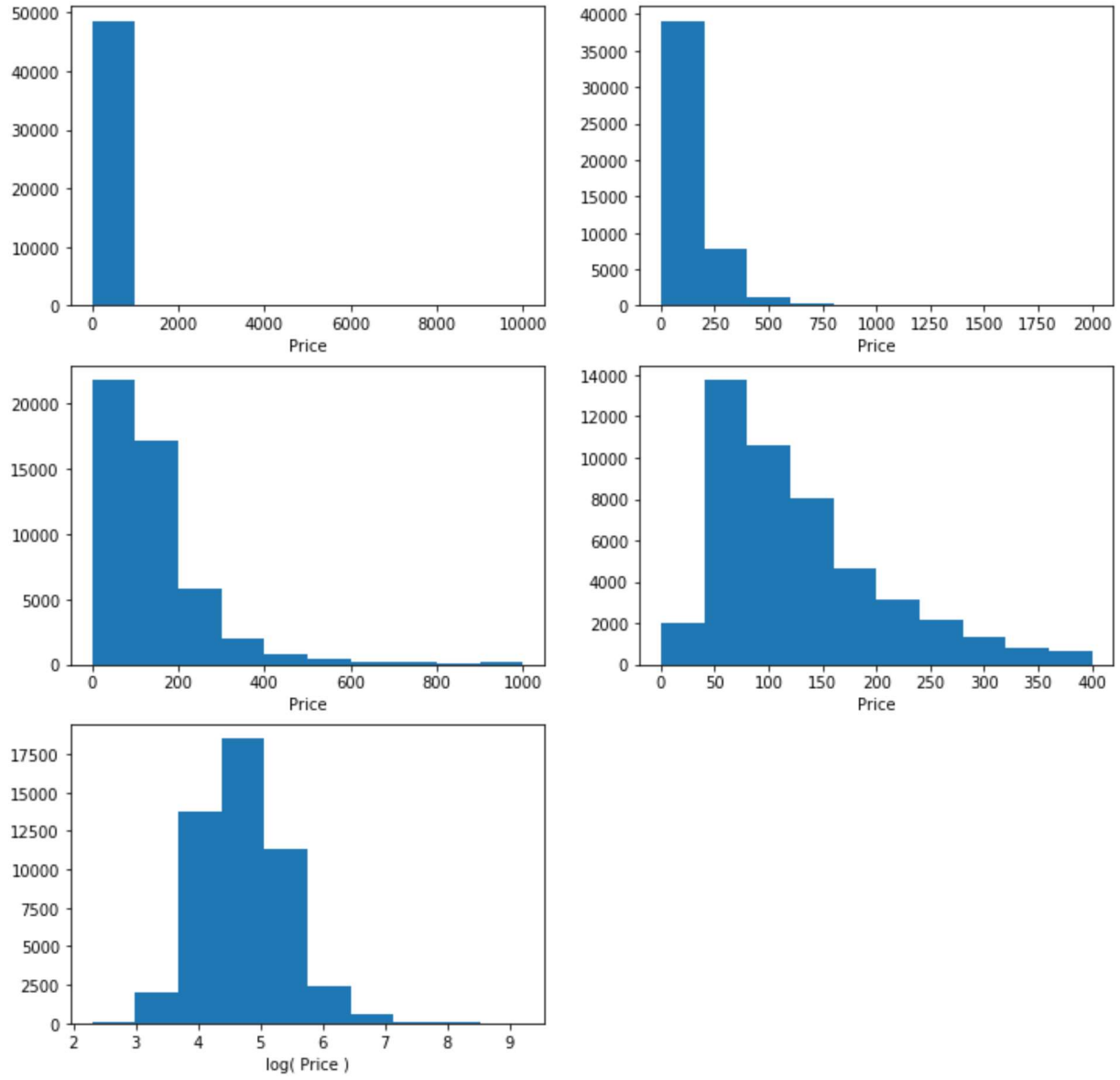
	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	lo
0	2539	Clean & quiet apt home by the park	2787	John	Brooklyn	Kensington	40.64749	-7
1	2595	Skylit Midtown Castle	2845	Jennifer	Manhattan	Midtown	40.75362	-7
2	3647	THE VILLAGE OF HARLEM....NEW YORK !	4632	Elisabeth	Manhattan	Harlem	40.80902	-7
3	3831	Cozy Entire Floor of Brownstone	4869	LisaRoxanne	Brooklyn	Clinton Hill	40.68514	-7
4	5022	Entire Apt: Spacious Studio/Loft by central park	7192	Laura	Manhattan	East Harlem	40.79851	-7

In [3]:
```
#useful to consider unique neigborhoods for the number of k clusters to do
print(df.neighbourhood_group.unique().size)

print(df.neighbourhood_group.unique())
```

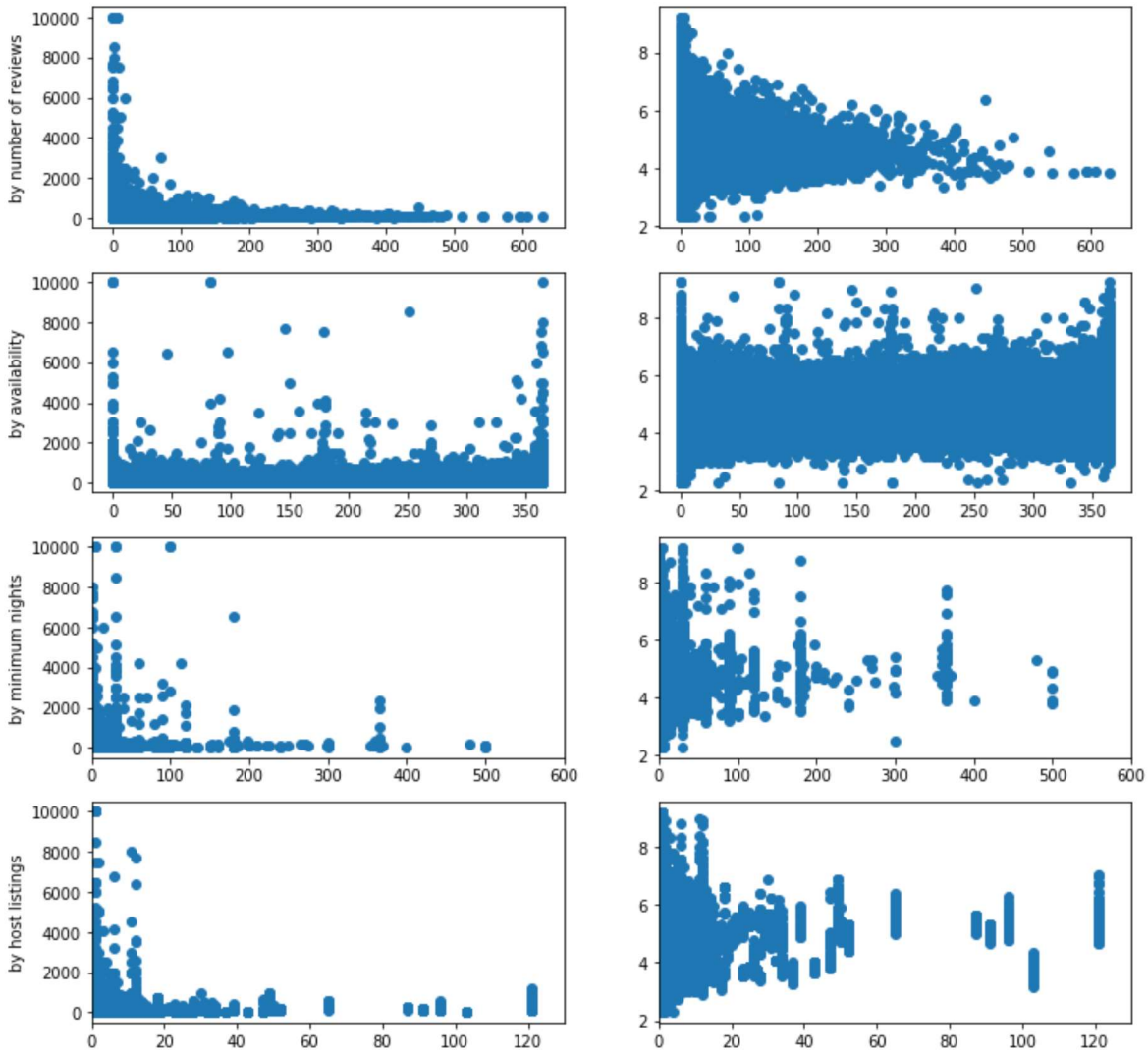
```
5
['Brooklyn' 'Manhattan' 'Queens' 'Staten Island' 'Bronx']
```

In [4]:
```
price = df['price'].to_numpy()
minimum_nights  = df['minimum_nights'].to_numpy()
number_of_reviews = df['number_of_reviews'].to_numpy()
reviews_per_month = df['reviews_per_month'].to_numpy()
host_listings = df['calculated_host_listings_count'].to_numpy()
availability = df['availability_365'].to_numpy()

log_price = np.log(price)
```

In [5]:
```
plt.figure(figsize=(12,12))
for i in enumerate([10000, 2000, 1000, 400]):
    plt.subplot(3,2, i[0]+1)
    plt.hist(price, range=(0,i[1]))
    plt.xlabel('Price')

plt.subplot(3,2,5)
plt.hist(log_price)
plt.xlabel('log( Price )')
plt.show()

#Shows histograms of price data here,
# see bottom historgram groups the log(price), approximates normal distribution
```



In [20]:
```
plt.figure(figsize=(12,12))

plt.subplot(4,2,1)
plt.scatter(number_of_reviews, price)
plt.ylabel('by number of reviews')

plt.subplot(4,2,2)
plt.scatter(number_of_reviews, log_price)

plt.subplot(4,2,3)
plt.scatter(availability, price)
plt.ylabel('by availability')

plt.subplot(4,2,4)
plt.scatter(availability, log_price)

plt.subplot(4,2,5)
plt.scatter(minimum_nights, price)
plt.xlim(0,600)
plt.ylabel('by minimum nights')

plt.subplot(4,2,6)
plt.scatter(minimum_nights, log_price)
plt.xlim(0,600)

plt.subplot(4,2,7)
plt.scatter(host_listings, price)
plt.xlim(0,130)
plt.ylabel('by host listings')

plt.subplot(4,2,8)
plt.scatter(host_listings, log_price)
plt.xlim(0,130)

plt.show()
```



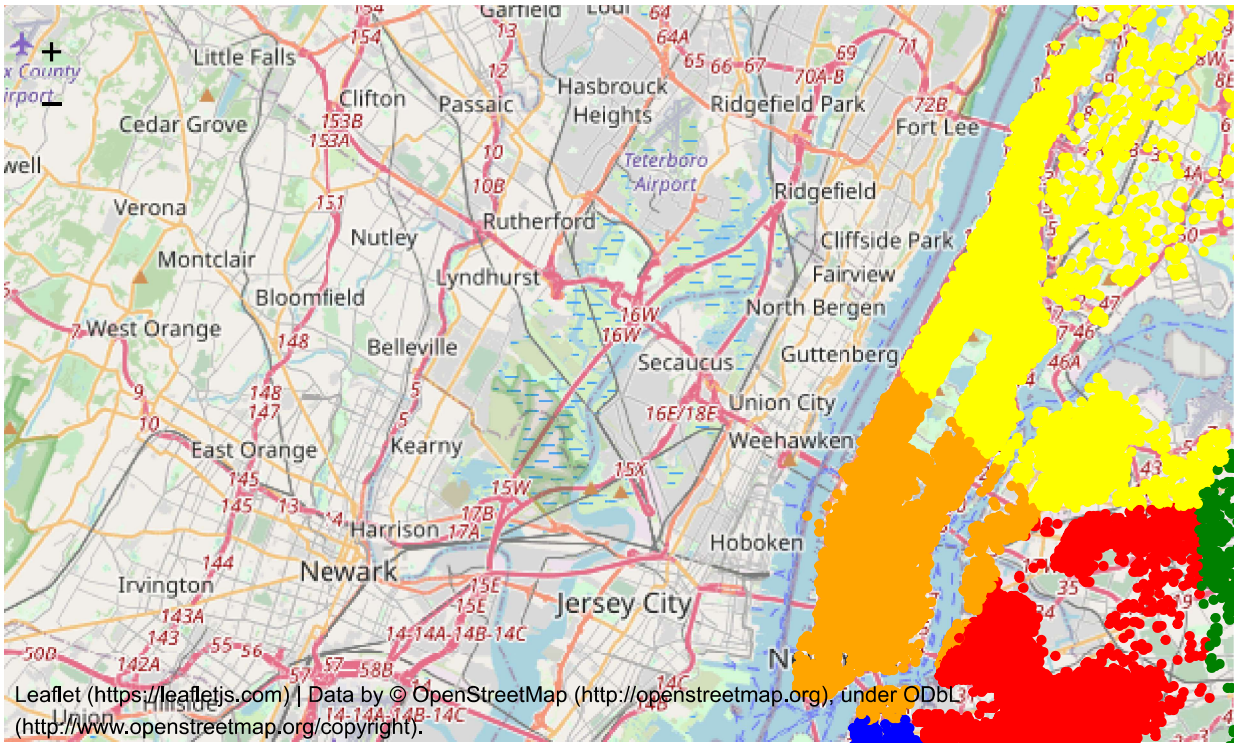
In [7]:
```
#create array of coordinates and a center to use for folium maps
coords = df[['latitude', 'longitude']].to_numpy()
center = np.mean(coords, axis=0)
print(np.max(coords, axis=0))
print(np.min(coords, axis=0))
```

```
[ 40.91306 -73.71299]
[ 40.49979 -74.24442]
```

In [8]:
```
color_set = ['red', 'orange', 'yellow', 'green', 'blue']

# k means clustering algoritm for inputted data, and maps coordinates onto folium
# color cooresponding to cluster label
def map_data(data, m, c=5):

    km = KMeans(n_clusters = c, random_state = 0).fit(data)

    for i in range(len(data)):
        folium.CircleMarker((coords[i,0], coords[i,1]), radius = 1,color = color_

    return m, km

# creates dataframe to display some stats for each cluster
def label_describe(labels):
    desc = pd.DataFrame(data={'avg price':['','','','',''],
                              'avg minimum nights':['','','','',''],
                              'avg number of reviews':['','','','',''],
                              'avg host listings':['','','','',''],
                              'avg availability':['','','','','']})
    desc.index = ['{}: {}'.format(i[0],i[1]) for i in enumerate(color_set)]
    for i in range(5):
        desc.iat[i, 0] = '{:.2f}, (std={:.1f})'.format(np.mean(price[:][labels==i
                                                       np.std(price[:][labels==i]
        desc.iat[i, 1] = '{:.2f}, (std={:.1f})'.format(np.mean(minimum_nights[:][
                                                       np.std(minimum_nights[:][]
        desc.iat[i, 2] = '{:.2f}, (std={:.1f})'.format(np.mean(number_of_reviews[
                                                       np.std(number_of_reviews[:
        desc.iat[i, 3] = '{:.2f}, (std={:.1f})'.format(np.mean(host_listings[:][]
                                                       np.std(host_listings[:][la
        desc.iat[i, 4] = '{:.2f}, (std={:.1f})'.format(np.mean(availability[:][la
                                                       np.std(availability[:][lab

    return desc
```

In [9]:
```
# k means cluster based on just the location data (coordinates)
m = folium.Map(center, zoom_start = 1)
m, km = map_data(coords[:], m)
m
```

Out[9]:



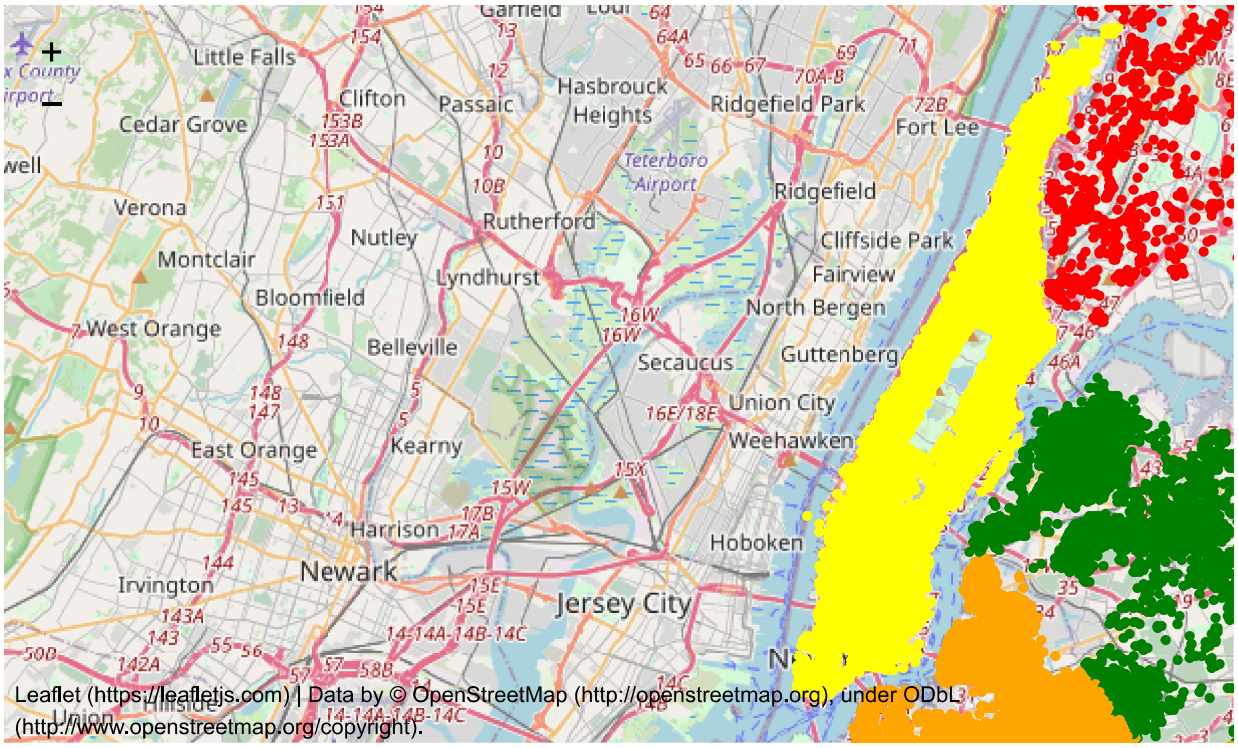
In [10]:
```
label_describe(km.labels_)
```

Out[10]:

	avg price	avg minimum nights	avg number of reviews	avg host listings	avg availability
0: red	114.75, (std=169.7)	6.02, (std=16.5)	24.38, (std=44.8)	2.86, (std=8.4)	103.75, (std=127.9)
1: orange	227.92, (std=327.6)	9.28, (std=26.4)	19.56, (std=41.1)	17.05, (std=57.1)	116.13, (std=134.9)
2: yellow	126.53, (std=210.2)	6.56, (std=18.5)	24.26, (std=46.2)	3.29, (std=12.4)	110.17, (std=129.8)
3: green	98.66, (std=117.4)	3.69, (std=11.1)	32.15, (std=56.0)	3.56, (std=8.1)	172.38, (std=133.7)
4: blue	130.81, (std=171.3)	6.21, (std=18.7)	24.19, (std=42.8)	1.85, (std=2.6)	106.70, (std=128.2)

In [11]:
```
#no k means clustering here, colors the points by the neigborhood group its assoc
m = folium.Map(center, zoom_start = 11)
color_set = ['red', 'orange', 'yellow', 'green', 'blue']
group_label = np.argmax(pd.get_dummies(df.neighbourhood_group).to_numpy(), axis=1
for i in range(len(coords)):
        folium.CircleMarker((coords[i,0], coords[i,1]), radius = 1,color = color_
m
```

Out[11]:



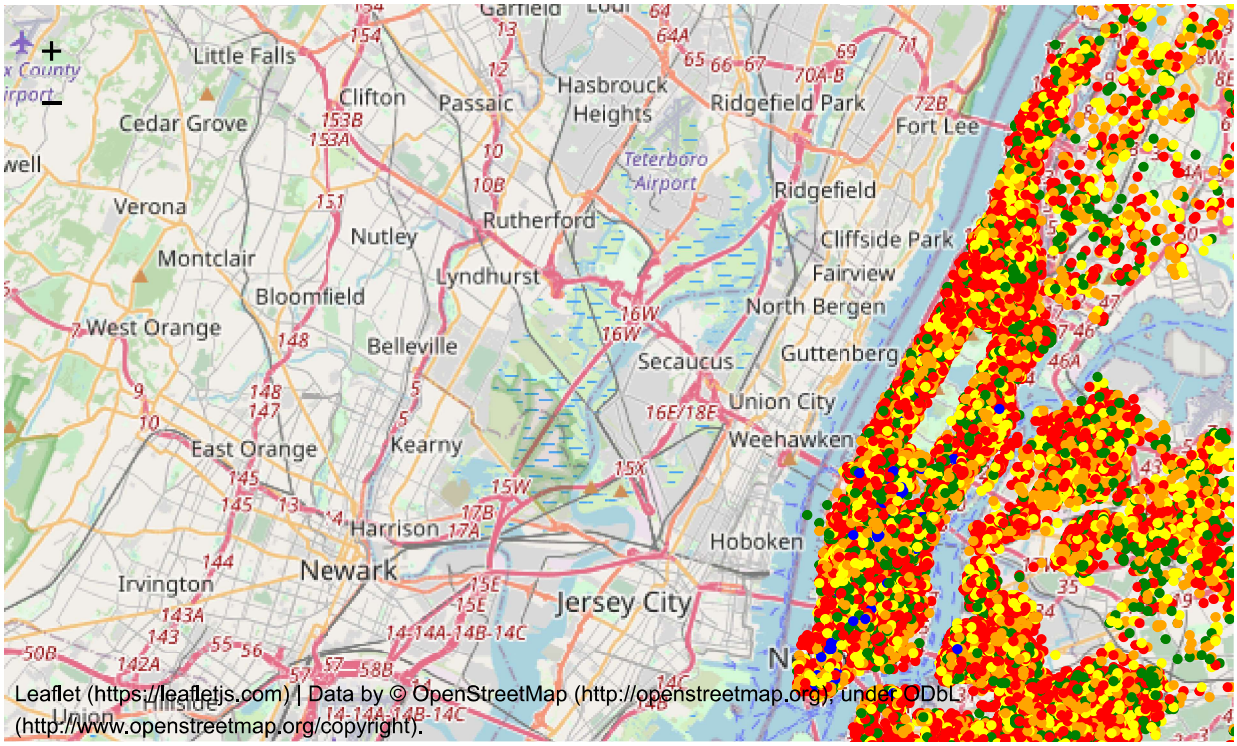
In [12]:
```
label_describe(group_label)
```

Out[12]:

	avg price	avg minimum nights	avg number of reviews	avg host listings	avg availability
0: red	87.58, (std=106.7)	4.56, (std=15.6)	25.98, (std=42.2)	2.23, (std=2.4)	165.79, (std=135.2)
1: orange	124.44, (std=186.9)	6.06, (std=17.6)	24.20, (std=44.3)	2.28, (std=5.3)	100.22, (std=126.3)
2: yellow	196.88, (std=291.4)	8.58, (std=24.1)	20.99, (std=42.6)	12.79, (std=48.2)	111.98, (std=132.7)
3: green	99.52, (std=167.1)	5.18, (std=15.0)	27.70, (std=52.0)	4.06, (std=12.4)	144.45, (std=135.5)
4: blue	114.81, (std=277.2)	4.83, (std=19.7)	30.94, (std=44.8)	2.32, (std=1.9)	199.68, (std=131.7)

In [13]:
```
#k means cluster by the other numerical features, not location
data = np.vstack((log_price, minimum_nights, number_of_reviews, host_listings, av

#scales each feature to between 0 and 1 for consistency in clustering
for i in range(5):
    data[:,i] = np.interp(data[:,i], (data[:,i].min(), data[:,i].max()), (0, 1))

m = folium.Map(center, zoom_start = 11)
m, km = map_data(data[:], m)
m
```

Out[13]:



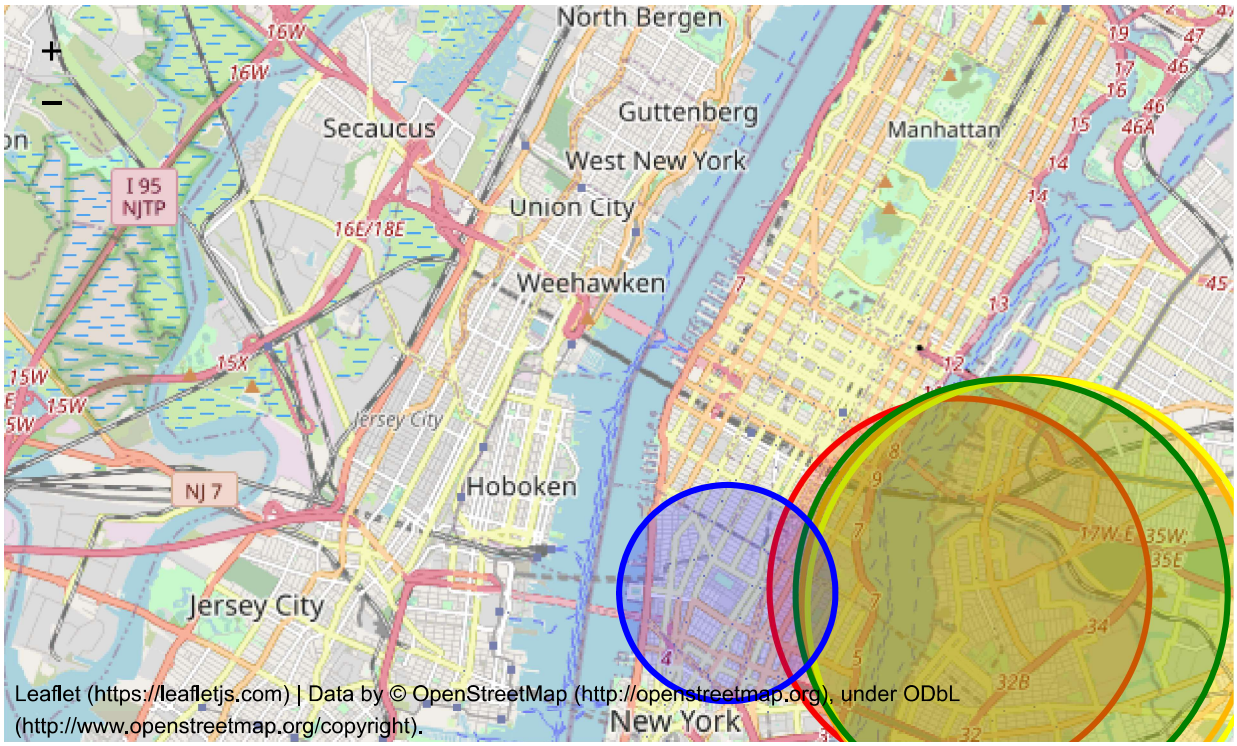
In [14]:
```
label_describe(km.labels_)
```

Out[14]:

	avg price	avg minimum nights	avg number of reviews	avg host listings	avg availability
0: red	136.73, (std=190.3)	4.83, (std=13.2)	12.28, (std=26.8)	1.66, (std=6.5)	5.20, (std=10.9)
1: orange	178.29, (std=311.3)	11.84, (std=33.4)	28.70, (std=49.9)	9.73, (std=21.3)	330.53, (std=29.8)
2: yellow	146.36, (std=251.1)	6.10, (std=16.8)	34.98, (std=51.2)	2.54, (std=6.3)	87.18, (std=26.9)
3: green	170.97, (std=271.2)	7.95, (std=20.4)	43.86, (std=64.6)	5.50, (std=16.3)	202.18, (std=36.1)
4: blue	273.57, (std=101.7)	20.96, (std=16.0)	2.39, (std=3.8)	288.79, (std=46.6)	287.76, (std=72.7)

In [15]:
```
#for each cluster identified above, maps the cluster center onto the map with a c
# cooresponding to how spread out the points in that cluster are
m = folium.Map(center, zoom_start = 12) #zoomed in a bit on the center of the cit

c_centers = np.array([np.mean(coords[:][km.labels_==i], axis=0) for i in range(5)
c_sizes = np.array([np.median(np.linalg.norm(c_centers[i] - coords[:][km.labels_=

for i in range(len(c_centers)):
    folium.CircleMarker((c_centers[i,0], c_centers[i,1]),
                        radius = c_sizes[i]*2000, color = color_set[i], fill_colo

m
```

Out[15]:



In [ ]:
```

```