In [1]:
```
# Put these at the top of every notebook, to get automatic reloading and inl
from IPython.core.display import display, HTML
import pandas as pd
import warnings
warnings.filterwarnings('ignore')

%reload_ext autoreload
%autoreload 1
%matplotlib inline

pd.set_option('display.max_rows', 500)
pd.set_option('display.max_columns', 500)
pd.set_option('display.width', 1000)

display(HTML("<style>.container { width:90% !important; }</style>"))
```

In [2]:
```
import os
import seaborn as sns
import pandas as pd
import math

from Utils.UtilsGeoViz import *
from Utils.UtilsViz import *
from Utils.DataUtils import *
```

In [6]:
```
US_coord = [37.0902, -102]
NY_COORD = [40.7128, -74.0060]
# data_path = os.path.join(os.getcwd(), "Data")
ny_datapath = "C:\\Users\\sriharis\\OneDrive\\UChicago\\DataMining\\project\
# ny_datapath = "C:\\Users\\SSrih\\OneDrive\\UChicago\\DataMining\\project\\
```

In [7]:
```
listings = pd.read_csv(os.path.join(ny_datapath, "listings.csv"))
```

In [8]:
```
listings['price'] = listings['price'].str.strip('').str.strip('$').str.repla
```

Listings per neighbourhood

In [94]:
```
nbhood_listings_data = listings[["id", "neighbourhood"]].groupby(by="neighbo
nbhood_listings_data = nbhood_listings_data.sort_values(by="id", ascending=F
nbhood_listings_data.reset_index(drop=True, inplace=True)
```

In [96]:
```
listings_loc_data = listings[["latitude", "longitude", "id"]]
listings_loc_group = listings_loc_data.groupby(by=["latitude", "longitude"],
                                               sort=False).count()

lats=list(listings_loc_group["latitude"].values)
lons=list(listings_loc_group["longitude"].values)
mag=list(listings_loc_group["id"].values)
```

In [97]:
```
neighbourhoods_geojson_path = os.path.join(ny_datapath, "neighbourhoods.geoj

th_scale = get_th_scale(data=nbhood_listings_data, col="id", n_steps=6) # ,

m1 = folium.Map(location=NY_COORD, tiles='OpenStreetMap', zoom_start=12)
# Add a choropleth map
m1 = add_choroplethmap(m1=m1, data=nbhood_listings_data,
                       json_path=neighbourhoods_geojson_path,
                       json_key='feature.properties.neighbourhood',
                       threshold_scale=th_scale,
                       name="Neighbourhoods with listings")
# m1
# m1 = addMarkerClusters(m1, listings_loc_data["latitude"].values, listings_
# open_map_in_browser(m1, os.path.join(ny_datapath, "geo_listings.html"))
```

Most expensive neighbourhoods

Let's make a price per listing for each neighbourhood

In [98]:
```
nbhood_price_data = listings[["price", "neighbourhood"]].groupby(by="neighbo
nbhood_price_data.head()
```

Out[98]:

	neighbourhood	price
0	Allerton	70.0
1	Alphabet City	140.0
2	Annadale	87.5
3	Arrochar	79.0
4	Astoria	84.0

In [99]:

```
neighbourhoods_geojson_path = os.path.join(ny_datapath, "neighbourhoods.geoj

th_scale = get_th_scale(data=nbhood_price_data, col="price", n_steps=6) # ,

m1 = folium.Map(location=NY_COORD, tiles='OpenStreetMap', zoom_start=12)
# Add a choropleth map
m1 = add_choroplethmap(m1=m1, data=nbhood_price_data,
                       json_path=neighbourhoods_geojson_path,
                       json_key='feature.properties.neighbourhood',
                       threshold_scale=th_scale,
                       name="Neighbourhoods Median Price",
                      color='Greens')

# https://stackoverflow.com/questions/44771779/adding-label-text-box-style-t

# open_map_in_browser(m1, os.path.join(ny_datapath, "geo_price.html"))
```

Neighbourhood groups

In [9]:
```
1 listings.head()
```

Out[9]:

	id	listing_url	scrape_id	last_scraped	name	summary
0	2454	https://www.airbnb.com/rooms/2454	20190201155637	2019-02-01	superCondo	Great light, exposed brick and 10 feet high ce...
1	2539	https://www.airbnb.com/rooms/2539	20190201155637	2019-02-02	Clean & quiet apt home by the park	Renovated apt home in elevator building.
2	2595	https://www.airbnb.com/rooms/2595	20190201155637	2019-02-02	Skylit Midtown Castle	Find your romantic getaway to this beautiful, ...
3	3330	https://www.airbnb.com/rooms/3330	20190201155637	2019-02-02	++ Brooklyn Penthouse Guestroom ++	This is a spacious, clean, furnished master be...
4	3647	https://www.airbnb.com/rooms/3647	20190201155637	2019-02-02	THE VILLAGE OF HARLEM....NEW YORK !	NaN

In [10]:
```
1 listings["neighbourhood_group_cleansed"].unique()
```

Out[10]:
```
array(['Manhattan', 'Brooklyn', 'Queens', 'Staten Island', 'Bronx'],
      dtype=object)
```

In [11]:
```
1 subset = listings[["id", "neighbourhood_cleansed", "neighbourhood_group_clea
2 subset.head()
```

Out[11]:

	id	neighbourhood_cleansed	neighbourhood_group_cleansed	price	host_is_superhost
0	2454	Midtown	Manhattan	137.0	f
1	2539	Kensington	Brooklyn	149.0	f
2	2595	Midtown	Manhattan	225.0	f
3	3330	Williamsburg	Brooklyn	70.0	f
4	3647	Harlem	Manhattan	150.0	f

Number of listings

In [12]:
```
nbhood_grp_grp = subset[["id", "neighbourhood_group_cleansed"]].groupby(by=[
nbhood_grp_grp
```

Out[12]:

	neighbourhood_group_cleansed	id
0	Bronx	977
1	Brooklyn	20500
2	Manhattan	22839
3	Queens	5565
4	Staten Island	347

In [13]:
```
def get_lcount(n):
    nbhgrp = subset[subset["neighbourhood_cleansed"]==n]["neighbourhood_grou
    return nbhood_grp_grp[nbhood_grp_grp["neighbourhood_group_cleansed"]==nb

subset1 = {"neighbourhood":[],
           "count":[]
          }
for n in subset["neighbourhood_cleansed"].unique():
    subset1["neighbourhood"].append(n)
    subset1["count"].append(get_lcount(n))

nbhood_listings_count = pd.DataFrame(subset1)
nbhood_listings_count = nbhood_listings_count[["neighbourhood", "count"]]
# nbhood_listings_count
```

In [14]:
```
neighbourhoods_geojson_path = os.path.join(ny_datapath, "neighbourhoods.geoj

# th_scale = get_th_scale(data=nbhood_listings_count, col="count", n_steps=6
th_scale = np.sort(nbhood_listings_count["count"].unique())
th_scale = np.append(th_scale, th_scale[-1]+1)

m1 = folium.Map(location=NY_COORD, tiles='OpenStreetMap', zoom_start=12)
# m1 = folium.Map(location=NY_COORD, tiles='Mapbox Bright', zoom_start=12)

# # Add a choropleth map
m1 = add_choroplethmap(m1=m1, data=nbhood_listings_count,
                       json_path=neighbourhoods_geojson_path,
                       json_key='feature.properties.neighbourhood',
                       threshold_scale=th_scale,
                       name="Neighbourhoods with listings",
                      color="Reds")
# m1
open_map_in_browser(m1, os.path.join(ny_datapath, "geo_listings_grp.html"))
```

Median price

In [15]:
```
nbhood_grp_grp = subset[["price", "neighbourhood_group_cleansed"]].groupby(b
nbhood_grp_grp
```

Out[15]:

	neighbourhood_group_cleansed	price
0	Bronx	86.308086
1	Brooklyn	121.573756
2	Manhattan	192.853102
3	Queens	96.848518
4	Staten Island	110.766571

In [16]:
```
def get_lcount(n):
    nbhgrp = subset[subset["neighbourhood_cleansed"]==n]["neighbourhood_grou
    return nbhood_grp_grp[nbhood_grp_grp["neighbourhood_group_cleansed"]==nb

subset1 = {"neighbourhood":[],
           "price":[]
          }
for n in subset["neighbourhood_cleansed"].unique():
    subset1["neighbourhood"].append(n)
    subset1["price"].append(get_lcount(n))

nbhood_listings_price = pd.DataFrame(subset1)
nbhood_listings_price = nbhood_listings_price[["neighbourhood", "price"]]
# nbhood_listings_price
```

In [17]:
```
neighbourhoods_geojson_path = os.path.join(ny_datapath, "neighbourhoods.geoj

# th_scale = get_th_scale(data=nbhood_listings_count, col="count", n_steps=6
th_scale = np.sort(nbhood_listings_price["price"].unique())
th_scale = np.append(th_scale, th_scale[-1]+1)

m1 = folium.Map(location=NY_COORD, tiles='OpenStreetMap', zoom_start=12)
# Add a choropleth map
m1 = add_choroplethmap(m1=m1, data=nbhood_listings_price,
                       json_path=neighbourhoods_geojson_path,
                       json_key='feature.properties.neighbourhood',
                       threshold_scale=th_scale,
                       name="Neighbourhoods with listings",
                      color="Greens")
# m1
# m1 = addMarkerClusters(m1, listings_loc_data["latitude"].values, listings_
open_map_in_browser(m1, os.path.join(ny_datapath, "geo_listings_price.html")
```

Super hosts

In [17]:
```
di = {"t": 1, "f": 0}
subset["host_is_superhost"].replace(di, inplace=True)
```

...

In [18]:
```
nbhood_grp_shost =\
    subset[["host_is_superhost", "neighbourhood_group_cleansed"]]
# nbhood_grp_shost.dropna(inplace=True)

nbhood_grp_shost =nbhood_grp_shost.groupby(by=["neighbourhood_group_cleansed
nbhood_grp_shost.reset_index(drop=False, inplace=True)
nbhood_grp_shost.columns = nbhood_grp_shost.columns.droplevel(0)
nbhood_grp_shost.columns = ["neighbourhood_group_cleansed", "num_listings",
# nbhood_grp_shost["shost_ratio"] = nbhood_grp_shost["shost_sum"] / nbhood_g
# nbhood_grp_shost["shost_ratio"] = 100 * nbhood_grp_shost["shost_sum"] / nb
nbhood_grp_shost["shost_ratio"] = 100 * nbhood_grp_shost["shost_sum"] / nbho
nbhood_grp_shost

```

Out[18]:

	neighbourhood_group_cleansed	num_listings	shost_sum	shost_ratio
0	Bronx	977	241.0	2.772665
1	Brooklyn	20496	3760.0	43.258168
2	Manhattan	22836	3500.0	40.266912
3	Queens	5564	1103.0	12.689830
4	Staten Island	347	88.0	1.012425

In [19]:
```
def get_sum(n):
    nbhgrp = subset[subset["neighbourhood_cleansed"]==n]["neighbourhood_grou
    return nbhood_grp_shost[nbhood_grp_shost["neighbourhood_group_cleansed"]

def get_ratio(n):
    nbhgrp = subset[subset["neighbourhood_cleansed"]==n]["neighbourhood_grou
    return nbhood_grp_shost[nbhood_grp_shost["neighbourhood_group_cleansed"]

subset1 = {"neighbourhood":[],
           "shost_ratio":[],
           "shost_sum":[]
          }
for n in subset["neighbourhood_cleansed"].unique():
    subset1["neighbourhood"].append(n)
    subset1["shost_ratio"].append(get_ratio(n))
    subset1["shost_sum"].append(get_sum(n))

nbhood_shost = pd.DataFrame(subset1)
nbhood_shost = nbhood_shost[["neighbourhood", "shost_ratio"]]
# nbhood_shost
```

In [22]:
```
neighbourhoods_geojson_path = os.path.join(ny_datapath, "neighbourhoods.geoj

# th_scale = get_th_scale(data=nbhood_listings_count, col="count", n_steps=6
th_scale = np.sort(nbhood_shost["shost_ratio"].unique())
th_scale = np.append(th_scale, th_scale[-1]+0.01)

m1 = folium.Map(location=NY_COORD, tiles='OpenStreetMap', zoom_start=12)
# Add a choropleth map
m1 = add_choroplethmap(m1=m1, data=nbhood_shost,
                       json_path=neighbourhoods_geojson_path,
                       json_key='feature.properties.neighbourhood',
                       threshold_scale=th_scale,
                       name="Neighbourhoods with super host (ratio)",
                       color="Blues")
# m1
# m1 = addMarkerClusters(m1, listings_loc_data["latitude"].values, listings_
open_map_in_browser(m1, os.path.join(ny_datapath, "geo_superhost_ratio.html"
```

In [21]:
```
# m1 = folium.Map(location=NY_COORD, tiles='OpenStreetMap', zoom_start=12)
# open_map_in_browser(m1, os.path.join(ny_datapath, "geo_superhost_ratio.htm
```