The Battle of Neighborhoods - Week 2

June 25, 2020

1 Coursera Capstone Project

1.1 The Battle of Neighborhoods - Final Report (Week 1 and 2)

1.2 Sai Anirudh Somanchi

1.2.1 Upload Libraries Required

[1]:
```
import numpy as np # library to handle data in a vectorized manner
import time
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
import requests # library to handle requests
from pandas.io.json import json_normalize # tranform JSON file into a pandas␣
 ↪dataframe



!conda install -c conda-forge geopy --yes # uncomment this line if you haven't␣
 ↪completed the Foursquare API lab
from geopy.geocoders import Nominatim # convert an address into latitude and␣
 ↪longitude values

!conda install -c conda-forge folium=0.5.0 --yes # uncomment this line if you␣
 ↪haven't completed the Foursquare API lab
import folium # map rendering library
import folium # map rendering library
from folium import plugins

# Matplotlib and associated plotting modules
import matplotlib.cm as cm
import matplotlib.colors as colors

import seaborn as sns

# import k-means from clustering stage
```

```
from sklearn.cluster import KMeans



print('Libraries imported.')
```

```
Collecting package metadata (current_repodata.json): done
Solving environment: done

# All requested packages already installed.

Collecting package metadata (current_repodata.json): done
Solving environment: done

# All requested packages already installed.

Libraries imported.
```

2 Coursera Capstone - REPORT

2.0.1 Content

1. Introduction Section :

   1.1 Discussion of the “backgroung situation” leading to the problem at hand:

   1.2 Problem to be resolved

   1.3 Audience for this project.

2. Data Section:

   2.1 Data of Current Situation (current residence place)

   2.2 Data required to resolve the problem

   2.3 Data sources and data manipulation

3. Methodology section :

   3.1 Process steps and strategy to resolve the problem

   3.2 Data Science Methods, machine learing, mapping tools and exploratory data analysis.

4. Results section

   Discussion of the results and how they help to take a decision.

5. Discussion section

   Elaboration and discussion on any observations and/or recommendations for improvement.

6. Conclusion section

   Desicison taken and Report Conclusion.

3 1. Introduction Section :

3.0.1 Discussion of the business problem and the audience who would be interested in this project.

3.0.2 1.1 Scenario and Background

I am a data scientist currently residing in Downtown Singapore. I currently live within walking distance to Downtown "Telok Ayer MRT metro station" therefore I have access to good public transportation to work. Likewise, I enjoy many ammenities in the neighborhood , such as international cousine restaurants, cafes, food shops and entertainment. I have been offered a great opportunity to work in Manhattan, NY. Although, I am very excited about it, I am a bit stress toward the process to secure a comparable place to live in Manhattan. Therefore, I decided to apply the learned skills during the Coursera course to explore ways to make sure my decision is factual and rewarding. Of course, there are alternatives to achieve the answer using available Google and Social media tools, but it rewarding doing it myself with learned tools.

3.0.3 1.2 Problem to be resolved:

The challenge to resolve is being able to find a rental apartment unit in Manhattan NY that offers similar characteristics and benefits to my current situation. Therefore, in order to set a basis for comparison, I want to find a renta unit subject to the following conditions: - Apartment with min 2 bedrooms with monthly rent not to exceed US$7000/month - Unit located within walking distance (<=1.0 mile, 1.6 km) from a subway metro station in Manhattan - Area with ammenities and venues similar to the ones described for current location (See item 2.1)

3.0.4 1.3 Interested Audience

I believe this is a relevant project for a person or entity considering moving to a major city in Europe, US or Asia, since the approach and methodologies used here are applicable in all cases. The use of FourSquare data and mapping techniques combined with data analysis will help resolve the key questions arisen. Lastly, this project is a good practical case toward the development of Data Science skills.

4 2. Data Section:

4.0.1 Description of the data and its sources that will be used to solve the problem

4.0.2 2.1 Data of Current Situation

I Currently reside in the neighborhood of 'Mccallum Street' in Downtonw Singapore. I use Foursquare to identify the venues around the area of residence which are then shown in the Singapore map shown in methodology and execution in section 3.0 . It serves as a reference for comparison with the desired future location in Manhattan NY

4.0.3 2.2 Data Required to resolve the problem

In order to make a good choice of a similar apartment in Manhattan NY, the following data is required: List/Information on neighborhoods form Manhattan with their Geodata (latitud and longitud. List/Information about the subway metro stations in Manhattan with geodata. Listed

apartments for rent in Manhattan area with descriptions (how many beds, price, location, address) Venues and ammenities in the Manhattan neighborhoods (e.g. top 10) 2.3 sources and manipulation The list of Manhattan neighborhoods is worked out during LAb exercise during the course. A csv file was created which will be read in order to create a dataframe and its mapping. The csv file 'mh_neigh_data.csv' has the following below data structure. The file will be directly read to the Jupiter Notebook for convenience and space savings. The clustering of neighborhoods and mapping will be shown however. An algorythm was used to determine the geodata from Nominatim . The actual algorythm coding may be shown in 'markdown' mode becasues it takes time to run.

mh_neigh_data.tail():

	Borough	Neighborhood	Latitude	Longitude
35	Manhattan	Turtle Bay	40.752042	-73.967708
36	Manhattan	Tudor City	40.746917	-73.971219
37	Manhattan	Stuyvesant Town	40.731000	-73.974052
38	Manhattan	Flatiron	40.739673	-73.990947
39	Manhattan	Hudson Yards	40.756658	-74.000111

A list of Manhattan subway metro stops was complied in Numbers (Apple excel) and it was complemeted with wikipedia data (https://en.wikipedia.org/wiki/List_of_New_York_City_Subway_stations_in_Manhattan) and information from NY Transit authority and Google maps (https://www.google.com/maps/search/manhattan+subway+metro+stations/@40.7837297,-74.1033043,11z/data=!3m1!4b1) for a final consolidated list of subway stops names and their address. The geolocation was obtained via an algorythm using Nominatim. Details will be shown in the execution of methodolody in section 3.0. The subway csv file is ''MH_subway.csv'' and the data structure is: mhsub.tail(): sub_station sub_address lat long

17	190 Street Subway Station	Bennett Ave, New York, NY 10040, USA	40.858113	-73.9
18	59 St-Lexington Av Station	E 60th St, New York, NY 10065, USA	40.762259	-7
19	57 Street Station	New York, NY 10019, United States	40.764250	-73
20	14 Street / 8 Av	New York, NY 10014, United States	40.730862	-73.9
21	MTA New York City	525 11th Ave, New York, NY 10018, USA	40.759809	-73

A list of places for rent was collected by web-browsing real estate companies in Manhattan : http://www.rentmanhattan.com/index.cfm?page=search&state=results https://www.nestpick.com/search?city=new-york&page=1&order=relevance&district=manhattan&gclid=CjwKC cPxjZYkURqQEswQK2jKQEpv_MvKcrIhRWRzNkc_r-fGi0lxoCA7cQAvD_BwE&type=apartment&display=lis https://www.realtor.com/apartments/Manhattan_NY A csv file was compiled with the rental place that indicated: areas of Manhattan, address, number of beds, area and monthly rental price. The csv file "nnnn.csv" had the following below structure. An algorythm was used to create all the

geodata using Nominatim, as shown in section 3.0. The actual algorythm coding may be shown in 'markdown' mode becasues it takes time to run. With the use of geolocator = Nominatim() , it was possible to determine the latitude and longiude for the subway metro locations as well as for the geodata for each rental place listed. The loop algorythms used are shown in the execution of data in section 3.0 "Great_circle" function from geolocator was used to calculate distances between two points , as in the case to calculate average rent price for units around each subway station and at 1.6 km radius. Foursquare is used to find the avenues at Manhattan neighborhoods in general and a cluster is created to later be able to search for the venues depending of the location shown.

4.0.4 2.4 How the data will be used to solve the problem

The data will be used as follows: Use Foursquare and geopy data to map top 10 venues for all Manhattan neighborhoods and clustered in groups (as per Course LAB) Use foursquare and geopy data to map the location of subway metro stations , separately and on top of the above clustered map in order to be able to identify the venues and ammenities near each metro station, or explore each subway location separately Use Foursquare and geopy data to map the location of rental places, in some form, linked to the subway locations. create a map that depicts, for instance, the average rental price per square ft, around a radious of 1.0 mile (1.6 km) around each subway station - or a similar metrics. I will be able to quickly point to the popups to know the relative price per subway area. Addresses from rental locations will be converted to geodata(lat, long) using Geopy-distance and Nominatim. Data will be searched in open data sources if available, from real estate sites if open to reading, libraries or other government agencies such as Metro New York MTA, etc.

4.0.5 2.5 Mapping of Data

The following maps were created to facilitate the analysis and the choice of the palace to live. Manhattan map of Neighborhoods manhattan subway metro locations Manhattan map of places for rent Manhattan map of clustered venues and neighborhoods Combined maps of Manhattan rent places with subway locations Combined maps of Manhattan rent places with subway locations and venues clusters

4.1 3. Methodology section:

This section represents the main component of the report where the data is gathered, prepared for analysis. The tools described are used here and the Notebook cells indicates the execution of steps.

4.1.1 The analysis and the stragegy:

The strategy is based on mapping the above described data in section 2.0, in order to facilitate the choice of at least two candidate places for rent. The choice is made based on the demands imposed : location near a subway, rental price and similar venues to Singapore. This visual approach and maps with popups labels allow quick identification of location, price and feature, thus making the selection very easy.

The procesing of these DATA and its mapping will allow to answer the key questions to make a decision: - what is the cost of available rental places that meet the demands? - what is the cost of rent around a mile radius from each subway metro station? - what is the area of Manhattan with best rental pricing that meets criteria established? - What is the distance from work place (Park Ave and 53 rd St) and the tentative future rental home? - What are the venues of the two best places to live? How the prices compare? - How venues distribute among Manhattan neighborhoods

and around metro stations? - Are there tradeoffs between size and price and location? - Any other interesting statistical data findings of the real estate and overall data.

5 METHODOLOY EXECUTION - Mapping Data

5.1 Singapore Map - Current residence and venues in neighborhood

for comparison to future Manhattan renting place

[2]:
```
# Shenton Way, District 01, Singapore
address = 'Mccallum Street, Singapore'
geolocator = Nominatim()
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print('The geograpical coordinate of Singapore home are {}, {}.'.
 ↪format(latitude, longitude))
```

```
/home/jupyterlab/conda/envs/python/lib/python3.6/site-
packages/ipykernel_launcher.py:3: DeprecationWarning: Using Nominatim with the
default "geopy/1.22.0" `user_agent` is strongly discouraged, as it violates
Nominatim's ToS https://operations.osmfoundation.org/policies/nominatim/ and may
possibly cause 403 and 429 HTTP errors. Please specify a custom `user_agent`
with `Nominatim(user_agent="my-application")` or by overriding the default
`user_agent`: `geopy.geocoders.options.default_user_agent = "my-application"`.
In geopy 2.0 this will become an exception.
  This is separate from the ipykernel package so we can avoid doing imports
until
```

```
The geograpical coordinate of Singapore home are 1.2784801, 103.8493717.
```

[3]:
```
neighborhood_latitude=1.2792655
neighborhood_longitude=103.8480938
```

5.2 Dial FourSquare to find venues around current residence in Singapore

[6]:
```
LIMIT = 100 # limit of number of venues returned by Foursquare API
radius = 500 # define radius
# create URL
CLIENT_ID = 'UR5QJRSP3BIOFEFBLJCJLKIZJELRLKIX2WXUPTLCERTMHA3V'
CLIENT_SECRET = 'J5KBOYYL5PUYASWYKX4Y3K5LD1DBVBOXTIHWACGO2W4JHMQ1'
VERSION = '20180605' # Foursquare API version
url = 'https://api.foursquare.com/v2/venues/explore?
 ↪&client_id={}&client_secret={}&v={}&ll={},{}&radius={}&limit={}'.format(
    CLIENT_ID,
    CLIENT_SECRET,
    VERSION,
    neighborhood_latitude,
```

```
    neighborhood_longitude,
    radius,
    LIMIT)
url # display URL
```

[6]:
```
'https://api.foursquare.com/v2/venues/explore?&client_id=UR5QJRSP3BI0FEFBLJCJLKI
ZJELRLKIX2WXUPTLCERTMHA3V&client_secret=J5KB0YYL5PUYASWYKX4Y3K5LD1DBVBOXTIHWACGO
2W4JHMQ1&v=20180605&ll=1.2792655,103.8480938&radius=500&limit=100'
```

[7]:
```
# results display is hidden for report simplification
results = requests.get(url).json()
#results
```

function that extracts the category of the venue - borrow from the Foursquare lab.

[8]:
```
def get_category_type(row):
    try:
        categories_list = row['categories']
    except:
        categories_list = row['venue.categories']

    if len(categories_list) == 0:
        return None
    else:
        return categories_list[0]['name']
```

[9]:
```
venues = results['response']['groups'][0]['items']
SGnearby_venues = json_normalize(venues) # flatten JSON
# filter columns
filtered_columns = ['venue.name', 'venue.categories', 'venue.location.lat',␣
 ↪'venue.location.lng']
SGnearby_venues =SGnearby_venues.loc[:, filtered_columns]
# filter the category for each row
SGnearby_venues['venue.categories'] = SGnearby_venues.apply(get_category_type,␣
 ↪axis=1)
# clean columns
SGnearby_venues.columns = [col.split(".")[-1] for col in SGnearby_venues.
 ↪columns]

SGnearby_venues.shape
```

```
/home/jupyterlab/conda/envs/python/lib/python3.6/site-
packages/ipykernel_launcher.py:2: FutureWarning: pandas.io.json.json_normalize
is deprecated, use pandas.json_normalize instead
```

[9]:
```
(100, 4)
```

[10]:
```
# Venues near current Singapore residence place
SGnearby_venues.head(10)
```

[10]:

	name	categories	lat	lng
0	Napoleon Food & Wine Bar	Wine Bar	1.279925	103.847333
1	Pepper Bowl	Asian Restaurant	1.279371	103.846710
2	Sofitel So Singapore	Hotel	1.280017	103.849813
3	Park Bench Deli	Deli / Bodega	1.279872	103.847287
4	Meat Smith	Southern / Soul Food Restaurant	1.280205	103.847410
5	Native	Cocktail Bar	1.280135	103.846844
6	Freehouse	Beer Garden	1.281254	103.848513
7	Lau Pa Sat Satay Street	Street Food Gathering	1.280261	103.850235
8	Mellower Coffee	Café	1.277814	103.848188
9	PS.Cafe	Café	1.280468	103.846264

5.3 Map of Singapore residence place with venues in Neighborhood - for reference

[11]:
```
latitude=1.2792655
longitude=103.8480938
# create map of Singapore place  using latitude and longitude values
map_sg = folium.Map(location=[latitude, longitude], zoom_start=18)
# add markers to map
for lat, lng, label in zip(SGnearby_venues['lat'], SGnearby_venues['lng'],␣
 ↪SGnearby_venues['name']):
    label = folium.Popup(label, parse_html=True)
    folium.RegularPolygonMarker(
        [lat, lng],
        number_of_sides=30,
        radius=7,
        popup=label,
        color='blue',
        fill_color='#0f0f0f',
        fill_opacity=0.6,
    ).add_to(map_sg)
```

```
map_sg
```

[11]: <folium.folium.Map at 0x7f3e8f184cf8>

6 MANHATTAN NEIGHBORHOODS - DATA AND MAPPING

6.1 Cluster neighborhood data was produced with Foursquare during course lab work. A csv file was produced containing the neighborhoods around the 40 Boroughs. Now, the csv file is just read for convenience and consolidation of report.

[12]:
```
# Read csv file with clustered neighborhoods with geodata
manhattan_data  = pd.read_csv('mh_neigh_data.csv')
manhattan_data.head()
```

[12]:

	Borough	Neighborhood	Latitude	Longitude	Cluster Labels
0	Manhattan	Marble Hill	40.876551	-73.910660	2
1	Manhattan	Chinatown	40.715618	-73.994279	2
2	Manhattan	Washington Heights	40.851903	-73.936900	4
3	Manhattan	Inwood	40.867684	-73.921210	3
4	Manhattan	Hamilton Heights	40.823604	-73.949688	0

[13]:
```
manhattan_data.tail()
```

[13]:

	Borough	Neighborhood	Latitude	Longitude	Cluster Labels
35	Manhattan	Turtle Bay	40.752042	-73.967708	3
36	Manhattan	Tudor City	40.746917	-73.971219	3
37	Manhattan	Stuyvesant Town	40.731000	-73.974052	4
38	Manhattan	Flatiron	40.739673	-73.990947	3
39	Manhattan	Hudson Yards	40.756658	-74.000111	2

7 Manhattan Borough neighborhoods - data with top 10 clustered venues

[14]:
```
manhattan_merged = pd.read_csv('manhattan_merged.csv')
manhattan_merged.head()
```

[14]:

	Borough	Neighborhood	Latitude	Longitude	Cluster Labels	\
0	Manhattan	Marble Hill	40.876551	-73.910660	2	
1	Manhattan	Chinatown	40.715618	-73.994279	2	
2	Manhattan	Washington Heights	40.851903	-73.936900	4	
3	Manhattan	Inwood	40.867684	-73.921210	3	
4	Manhattan	Hamilton Heights	40.823604	-73.949688	0	

```
  1st Most Common Venue 2nd Most Common Venue 3rd Most Common Venue  \
0           Coffee Shop        Discount Store           Yoga Studio
1    Chinese Restaurant          Cocktail Bar    Dim Sum Restaurant
2                 Café                Bakery     Mobile Phone Shop
3    Mexican Restaurant                Lounge           Pizza Place
4    Mexican Restaurant           Coffee Shop                 Café

  4th Most Common Venue  5th Most Common Venue 6th Most Common Venue  \
0            Steakhouse       Supplement Shop        Tennis Stadium
1   American Restaurant  Vietnamese Restaurant    Salon / Barbershop
2           Pizza Place        Sandwich Place                  Park
3                 Café             Wine Bar                Bakery
4         Deli / Bodega           Pizza Place          Liquor Store

  7th Most Common Venue      8th Most Common Venue 9th Most Common Venue  \
0            Shoe Store                        Gym                  Bank
1          Noodle House                     Bakery       Bubble Tea Shop
2                   Gym  Latin American Restaurant      Tapas Restaurant
3   American Restaurant                       Park    Frozen Yogurt Shop
4     Indian Restaurant          Sushi Restaurant        Sandwich Place

  10th Most Common Venue
0     Seafood Restaurant
1         Ice Cream Shop
2     Mexican Restaurant
3     Spanish Restaurant
4            Yoga Studio
```

8 Map of Manhattan neighborhoods with top 10 clustered venues

8.1 popus allow to identify each neighborhood and the cluster of venues around it in order to proceed to examine in more detail in the next cell

[15]:
```
# create map of Manhattan using latitude and longitude values from Nominatim
latitude= 40.7308619
longitude= -73.9871558

kclusters=5
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=13)

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

```
# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(manhattan_merged['Latitude'],␣
 ↪manhattan_merged['Longitude'], manhattan_merged['Neighborhood'],␣
 ↪manhattan_merged['Cluster Labels']):
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    folium.CircleMarker(
        [lat, lon],
        radius=20,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7).add_to(map_clusters)
  # add markers for rental places to map
for lat, lng, label in zip(manhattan_data['Latitude'],␣
 ↪manhattan_data['Longitude'], manhattan_data['Neighborhood']):
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=5,
        popup=label,
        color='blue',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.7,
        parse_html=False).add_to(map_clusters)


map_clusters
```

[15]: <folium.folium.Map at 0x7f3e87930240>

8.2 Examine a paticular Cluster - print venues

8.2.1 After examining several cluster data , I concluded that cluster # 2 resembles closer the Singapore place, therefore providing guidance as to where to look for the future apartment .

8.2.2 Assign a value to 'kk' to explore a given cluster.

[16]:
```
## kk is the cluster number to explore
kk = 2
manhattan_merged.loc[manhattan_merged['Cluster Labels'] == kk, manhattan_merged.
 ↪columns[[1] + list(range(5, manhattan_merged.shape[1]))]]
```

[16]:

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0	Marble Hill	Coffee Shop	Discount Store	Yoga Studio	Steakhouse	Supplement Shop	Tennis Stadium	Shoe Store	Gym	Bank	Seafood Restaurant
1	Chinatown	Chinese Restaurant	Cocktail Bar	Dim Sum Restaurant	American Restaurant	Vietnamese Restaurant	Salon / Barbershop	Noodle House	Bakery	Bubble Tea Shop	Ice Cream Shop
6	Central Harlem	African Restaurant	Seafood Restaurant	French Restaurant	American Restaurant	Cosmetics Shop	Chinese Restaurant	Event Space	Liquor Store	Beer Bar	Gym / Fitness Center
9	Yorkville	Coffee Shop	Gym	Bar	Italian Restaurant	Sushi Restaurant	Pizza Place	Mexican Restaurant	Deli / Bodega	Japanese Restaurant	Pub
14	Clinton	Theater	Italian Restaurant	Coffee Shop	American Restaurant	Gym / Fitness Center	Hotel	Wine Shop	Spa	Gym	Indie Theater
23	Soho	Clothing Store	Boutique	Women's Store	Shoe Store	Men's Store	Furniture / Home Store	Italian Restaurant	Mediterranean Restaurant	Art Gallery	Design Studio
26	Morningside Heights	Coffee Shop	American Restaurant	Park	Bookstore	Pizza Place	Sandwich Place	Burger Joint	Café	Deli / Bodega	Tennis Court
34	Sutton Place	Gym / Fitness Center	Italian Restaurant	Furniture / Home Store	Indian Restaurant	Dessert Shop	American Restaurant	Bakery	Juice Bar	Boutique	Sushi Restaurant
39	Hudson Yards	Coffee Shop	Italian Restaurant	Hotel	Theater	American Restaurant	Café	Gym / Fitness Center	Thai Restaurant	Restaurant	Gym

9 Map of Manhattan places for rent

9.1 Several Manhattan real estate webs were webscrapped to collect rental data, as mentioned in section 2.0 . The resut was summarized in a csv file for direct reading, in order to consolidate the proces.

9.1.1 The initial data for 144 apartment did not have the latitude and longitude data (NaN) but the information was established in the following cell using an algorythm and Nominatim.

[18]:
```
# csv files with rental places with basic data but still wihtout geodata (
 ↪latitude and longitude)
# pd.read_csv(' le.csv', header=None, nrows=5)
mh_rent=pd.read_csv('MH_flats_price.csv')
mh_rent.head()
```

[18]:

	Address	Area	Price_per_ft2	Rooms	Area-ft2	\
0	West 105th Street	Upper West Side	2.94	5.0	3400	
1	East 97th Street	Upper East Side	3.57	3.0	2100	
2	West 105th Street	Upper West Side	1.89	4.0	2800	
3	CARMINE ST.	West Village	3.03	2.0	1650	
4	171 W 23RD ST.	Chelsea	3.45	2.0	1450	

	Rent_Price	Lat	Long
0	10000	NaN	NaN
1	7500	NaN	NaN
2	5300	NaN	NaN
3	5000	NaN	NaN
4	5000	NaN	NaN

[19]:
```
mh_rent.tail()
```

[19]:

	Address	Area	Price_per_ft2	\
139	200 East 72nd Street	Rental in Lenox Hill	5.15	
140	50 Murray Street	No fee rental in Tribeca	7.11	
141	300 East 56th Street	No fee rental in Midtown East	3.87	
142	1930 Broadway	No fee rental in Central Park West	5.06	
143	33 West 9th Street	Rental in Greenwich Village	6.67	

	Rooms	Area-ft2	Rent_Price	Lat	Long
139	3.0	1700	8750	NaN	NaN
140	2.0	1223	8700	NaN	NaN
141	3.0	2100	8118	NaN	NaN
142	2.0	1600	8095	NaN	NaN
143	2.0	1500	10000	NaN	NaN

9.2 Obtain geodata (lat,long) for each rental place in Manhattan with Nominatim

9.2.1 Data was stored in a csv file for simplifaction report purposes and saving code processing time in future.

[22]:
```
## This section may be 'markedown' for the report because its execution takes␣
 ↪few minutes .
## Terefore, the csv previusly made may be just read directly.

for n in range(len(mh_rent)):
    address= mh_rent['Address'][n]
    address=(mh_rent['Address'][n]+ '  , '+' Manhattan NY ')
    geolocator = Nominatim()
    location = geolocator.geocode(address)
    latitude = location.latitude
    longitude = location.longitude
    mh_rent['Lat'][n]=latitude
    mh_rent['Long'][n]=longitude
    #print(n,latitude,longitude)
    time.sleep(2)

print('Geodata completed')
# save dataframe to csv file
mh_rent.to_csv('MH_rent_latlong.csv',index=False)
mh_rent.shape
```

```
/home/jupyterlab/conda/envs/python/lib/python3.6/site-
packages/ipykernel_launcher.py:7: DeprecationWarning: Using Nominatim with the
default "geopy/1.22.0" `user_agent` is strongly discouraged, as it violates
Nominatim's ToS https://operations.osmfoundation.org/policies/nominatim/ and may
possibly cause 403 and 429 HTTP errors. Please specify a custom `user_agent`
with `Nominatim(user_agent="my-application")` or by overriding the default
`user_agent`: `geopy.geocoders.options.default_user_agent = "my-application"`.
In geopy 2.0 this will become an exception.
  import sys
/home/jupyterlab/conda/envs/python/lib/python3.6/site-
packages/ipykernel_launcher.py:11: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  # This is added back by InteractiveShellApp.init_path()
/home/jupyterlab/conda/envs/python/lib/python3.6/site-
packages/ipykernel_launcher.py:12: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
```

```
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  if sys.path[0] == '':

Geodata completed
```

[22]:
```
(144, 8)
```

[23]:
```
mh_rent=pd.read_csv('MH_rent_latlong.csv')
mh_rent.head()
```

[23]:

	Address	Area	Price_per_ft2	Rooms	Area-ft2	Rent_Price	Lat	Long
0	West 105th Street	Upper West Side	2.94	5.0	3400	10000	40.799771	-73.966213
1	East 97th Street	Upper East Side	3.57	3.0	2100	7500	40.788517	-73.955118
2	West 105th Street	Upper West Side	1.89	4.0	2800	5300	40.799771	-73.966213
3	CARMINE ST.	West Village	3.03	2.0	1650	5000	40.730337	-74.002476
4	171 W 23RD ST.	Chelsea	3.45	2.0	1450	5000	40.744118	-73.995299

[24]:
```
mh_rent.tail()
```

[24]:

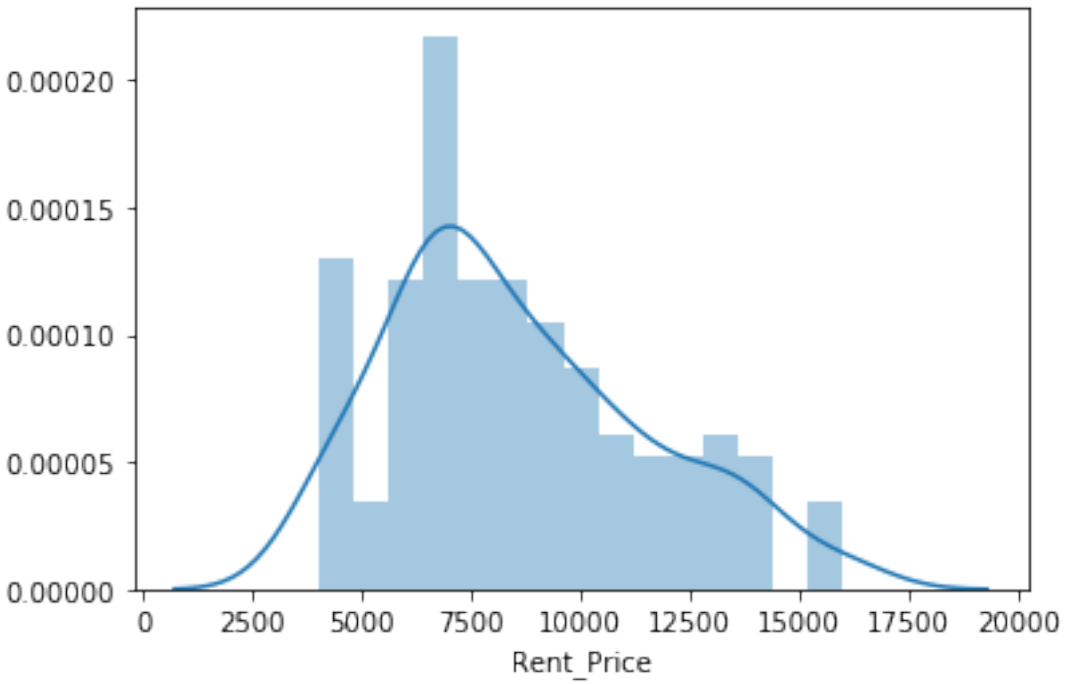
	Address	Area	Price_per_ft2	Rooms	Area-ft2	Rent_Price	Lat	Long
139	200 East 72nd Street	Rental in Lenox Hill	5.15	3.0	1700	8750	40.769465	-73.960339
140	50 Murray Street	No fee rental in Tribeca	7.11	2.0	1223	8700	40.714051	-74.009608
141	300 East 56th Street	No fee rental in Midtown East	3.87	3.0	2100	8118	40.758216	-73.965190
142	1930 Broadway	No fee rental in Central Park West	5.06	2.0	1600	8095	40.772433	-73.981705
143	33 West 9th Street	Rental in Greenwich Village	6.67	2.0	1500	10000	40.733691	-73.997323

10 Manhattan apartment rent price statistics

10.0.1 A US 7000 Dollar per month rent is actually around the mean value - similar to Singapore! wow!

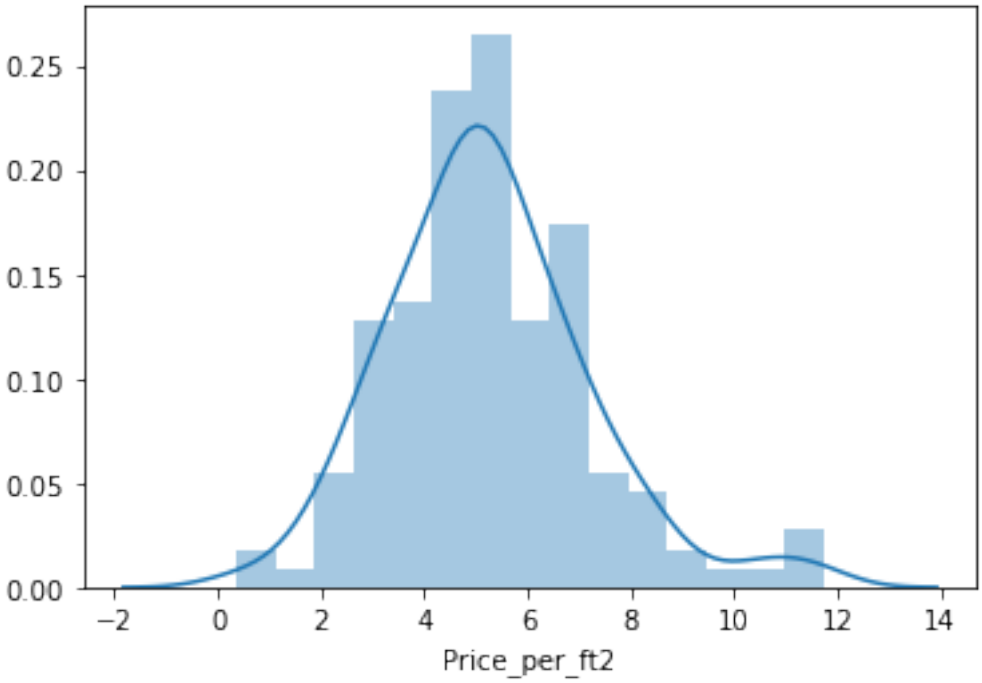
[25]:
```
import seaborn as sns
sns.distplot(mh_rent['Rent_Price'],bins=15)
```

[25]: <matplotlib.axes._subplots.AxesSubplot at 0x7f3e6fe3a4e0>



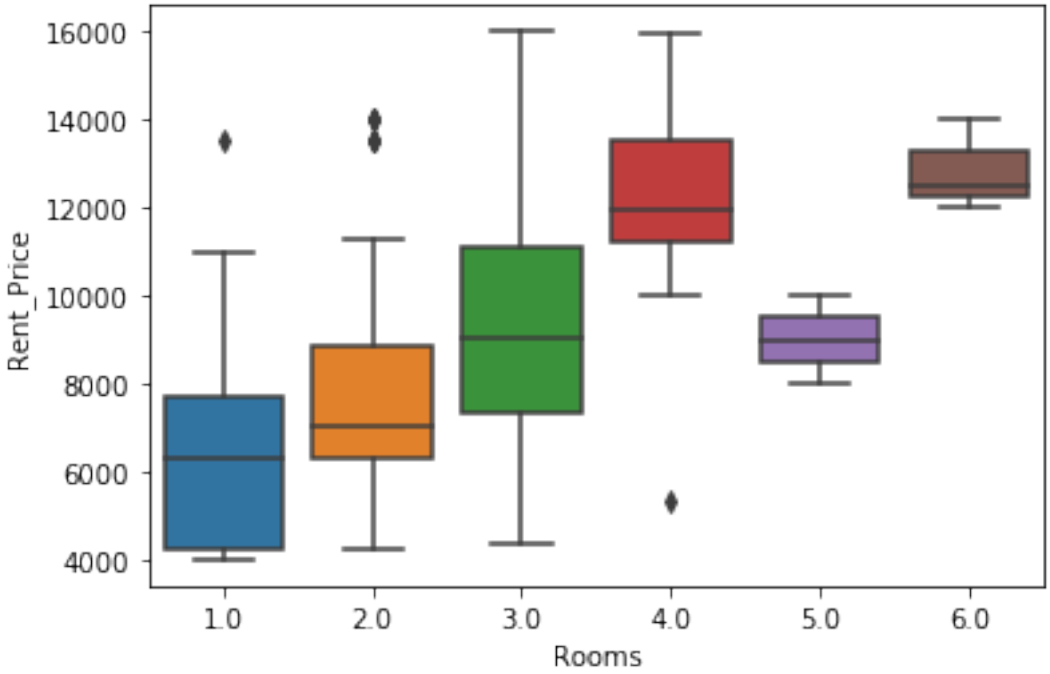
[26]:
```
import seaborn as sns
sns.distplot(mh_rent['Price_per_ft2'],bins=15)
```

[26]: <matplotlib.axes._subplots.AxesSubplot at 0x7f3e6fd8ce80>



[27]:
```
sns.boxplot(x='Rooms', y= 'Rent_Price', data=mh_rent)
```

[27]: <matplotlib.axes._subplots.AxesSubplot at 0x7f3e6fd00908>



11 Map of Manhattan apartments for rent

11.0.1 The popups will indicate the address and the monthly price for rent thus making it convenient to select the target appartment with the price condition estipulated (max US7000)

[28]:
```
# create map of Manhattan using latitude and longitude values from Nominatim
latitude= 40.7308619
longitude= -73.9871558

map_manhattan_rent = folium.Map(location=[latitude, longitude], zoom_start=12.5)

# add markers to map
for lat, lng, label in zip(mh_rent['Lat'], mh_rent['Long'],'$ ' +␣
 ↪mh_rent['Rent_Price'].astype(str)+ ',  '+ mh_rent['Address']):
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=6,
        popup=label,
        color='blue',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.7,
        parse_html=False).add_to(map_manhattan_rent)


map_manhattan_rent
```

[28]:
```
<folium.folium.Map at 0x7f3e6fbfacc0>
```

12 Map of Manhattan showing the places for rent and the cluster of venues

12.0.1 Now, one can point to a rental place for price and address location information while knowing the cluster venues around it.

12.0.2 This is an insightful way to explore rental possibilites

[29]:
```
# create map of Manhattan using latitude and longitude values from Nominatim
latitude= 40.7308619
longitude= -73.9871558

# create map with clusters
kclusters=5
```

```
map_clusters2 = folium.Map(location=[latitude, longitude], zoom_start=13)

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

# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(manhattan_merged['Latitude'],␣
 ↪manhattan_merged['Longitude'], manhattan_merged['Neighborhood'],␣
 ↪manhattan_merged['Cluster Labels']):
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    folium.CircleMarker(
        [lat, lon],
        radius=20,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7).add_to(map_clusters2)

# add markers to map for rental places
for lat, lng, label in zip(mh_rent['Lat'], mh_rent['Long'],'$ ' +␣
 ↪mh_rent['Rent_Price'].astype(str)+ mh_rent['Address']):
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=6,
        popup=label,
        color='blue',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.7,
        parse_html=False).add_to(map_clusters2)

    # Adds tool to the top right
from folium.plugins import MeasureControl
map_manhattan_rent.add_child(MeasureControl())

# FMeasurement ruler icon to establish distnces on map
from folium.plugins import FloatImage
url = ('https://media.licdn.com/mpr/mpr/shrinknp_100_100/
 ↪AAEAAQAAAAAAAAlgAAAAJGE3OTA4YTdlLTkzZjUtNDFjYy1iZThlLWQ5OTNkYzlhNzM4OQ.jpg')
FloatImage(url, bottom=5, left=85).add_to(map_manhattan_rent)
```

```
map_clusters2
```

[29]: <folium.folium.Map at 0x7f3e6f99eeb8>

13 Now one can explore a particular rental place and its venues in detail

13.1 In the map above, examination of appartments with rental place below 7000/month is straightforwad while knowing the venues around it.

13.2 We could find an appartment with at the right price and in a location with desirable venues. The next step is to see if it is located near a subway metro station, in next cells work.

[30]:
```
## kk is the cluster number to explore
kk = 3
manhattan_merged.loc[manhattan_merged['Cluster Labels'] == kk, manhattan_merged.
 ↪columns[[1] + list(range(5, manhattan_merged.shape[1]))]]
```

[30]:

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue
3	Inwood	Mexican Restaurant	Lounge	Pizza Place	Café
5	Manhattanville	Deli / Bodega	Italian Restaurant	Seafood Restaurant	Mexican Restaurant
10	Lenox Hill	Sushi Restaurant	Italian Restaurant	Coffee Shop	Gym / Fitness Center
12	Upper West Side	Italian Restaurant	Bar	Bakery	Vegetarian / Vegan Restaurant
16	Murray Hill	Sandwich Place	Hotel	Japanese Restaurant	Gym / Fitness Center
17	Chelsea	Coffee Shop	Italian Restaurant	Ice Cream Shop	Bakery
18	Greenwich Village	Italian Restaurant	Sushi Restaurant	French Restaurant	Clothing Store
27	Gramercy	Italian Restaurant	Restaurant	Thrift / Vintage Store	Cocktail Bar
29	Financial District	Coffee Shop	Hotel	Gym	Wine Shop
31	Noho	Italian Restaurant	French Restaurant	Cocktail Bar	Gift Shop
32	Civic Center	Gym / Fitness Center	Bakery		
35	Turtle Bay	Italian Restaurant	Coffee Shop		
36	Tudor City	Café	Park		
38	Flatiron	Italian Restaurant	American Restaurant		

32	Italian Restaurant	Cocktail Bar	
35	Steakhouse	Wine Bar	
36	Pizza Place	Mexican Restaurant	
38	Gym	Gym / Fitness Center	

	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue
3	Wine Bar	Bakery	American Restaurant
5	Sushi Restaurant	Beer Garden	Coffee Shop
10	Pizza Place	Burger Joint	Deli / Bodega
12	Indian Restaurant	Coffee Shop	Cosmetics Shop
16	Coffee Shop	Salon / Barbershop	Burger Joint
17	Nightclub	Theater	Art Gallery
18	Chinese Restaurant	Café	Indian Restaurant
27	Bagel Shop	Coffee Shop	Pizza Place
29	Steakhouse	Bar	Italian Restaurant
31	Bookstore	Grocery Store	Mexican Restaurant
32	French Restaurant	Sandwich Place	Coffee Shop
35	Sushi Restaurant	Hotel	Noodle House
36	Greek Restaurant	Sushi Restaurant	Hotel
38	Yoga Studio	Vegetarian / Vegan Restaurant	Bakery

	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
3	Park	Frozen Yogurt Shop	Spanish Restaurant
5	Falafel Restaurant	Bike Trail	Other Nightlife
10	Gym	Sporting Goods Shop	Thai Restaurant
12	Wine Bar	Mexican Restaurant	Sushi Restaurant
16	French Restaurant	Bar	Italian Restaurant
17	Seafood Restaurant	American Restaurant	Hotel
18	Bakery	Seafood Restaurant	Electronics Store
27	Mexican Restaurant	Grocery Store	Wine Shop
29	Pizza Place	Park	Gym / Fitness Center
31	Hotel	Sushi Restaurant	Coffee Shop
32	Gym	Yoga Studio	Park
35	Indian Restaurant	Japanese Restaurant	French Restaurant
36	Deli / Bodega	Diner	Dog Run
38	Clothing Store	Cosmetics Shop	Cycle Studio

14 Mapping Manhattan Subway locations

14.0.1 Manhattan subway metro locations (address) was obtained from webscrapping sites such as Wikipedia, Google and NY Metro Transit. For simplification, a csv file was produced from the 'numbers' (Apple excel) so that the reading of this file is the starting point here.

14.0.2 The geodata will be obtain via Nominatim using the algorythm below.

[31]:
```
# A csv file summarized the subway station and the addresses for next step to␣
 ↪determine geodata
mh=pd.read_csv('NYC_subway_list.csv')
mh.head()
```

[31]:

	sub_station	sub_address
0	Dyckman Street Subway Station	170 Nagle Ave, New York, NY 10034, USA
1	57 Street Subway Station	New York, NY 10106, USA
2	Broad St	New York, NY 10005, USA
3	175 Street Station	807 W 177th St, New York, NY 10033, USA
4	5 Av and 53 St	New York, NY 10022, USA

14.0.3 Add colums labeled 'lat' and 'long' to be filled with geodata

[32]:
```
#  Add columns 'lat'  and 'long' to mh dataframe - with random temporary␣
 ↪numbers to get started
sLength = len(mh['sub_station'])
lat = pd.Series(np.random.randn(sLength))
long=pd.Series(np.random.randn(sLength))
mh = mh.assign(lat=lat.values)
mh = mh.assign(long=long.values)
```

[34]:
```
## Algorythm to find latitude and longitud for each subway metro station and␣
 ↪add them to dataframe

for n in range(len(mh)):
    address= mh['sub_address'][n]
    geolocator = Nominatim()
    location = geolocator.geocode(address)
    latitude = location.latitude
    longitude = location.longitude
    mh['lat'][n]=latitude
    mh['long'][n]=longitude
    #print(n,latitude,longitude)
    time.sleep(2)

print('Geodata completed')
# save dataframe to csv file
```

```
mh.to_csv('MH_subway.csv',index=False)
mh.shape
```

```
/home/jupyterlab/conda/envs/python/lib/python3.6/site-
packages/ipykernel_launcher.py:5: DeprecationWarning: Using Nominatim with the
default "geopy/1.22.0" `user_agent` is strongly discouraged, as it violates
Nominatim's ToS https://operations.osmfoundation.org/policies/nominatim/ and may
possibly cause 403 and 429 HTTP errors. Please specify a custom `user_agent`
with `Nominatim(user_agent="my-application")` or by overriding the default
`user_agent`: `geopy.geocoders.options.default_user_agent = "my-application"`.
In geopy 2.0 this will become an exception.
  """
/home/jupyterlab/conda/envs/python/lib/python3.6/site-
packages/ipykernel_launcher.py:9: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  if __name__ == '__main__':
/home/jupyterlab/conda/envs/python/lib/python3.6/site-
packages/ipykernel_launcher.py:10: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  # Remove the CWD from sys.path while we load stuff.

Geodata completed
```

[34]:
```
(76, 4)
```

[35]:
```
mh=pd.read_csv('MH_subway.csv')
print(mh.shape)
mh.head()
```

```
(76, 4)
```

[35]:

	sub_station	sub_address	lat	long
0	Dyckman Street Subway Station	170 Nagle Ave, New York, NY 10034, USA	40.861857	-73.924509
1	57 Street Subway Station	New York, NY 10106, USA	40.758798	-73.962343
2	Broad St	New York, NY 10005, USA	40.712728	-74.006015
3	175 Street Station	807 W 177th St, New York, NY 10033, USA		
4	5 Av and 53 St	New York, NY 10022, USA		

```
3  40.847991 -73.939785
4  40.758798 -73.962343
```

[36]:
```
# removing duplicate rows and creating new set mhsub1
mhsub1=mh.drop_duplicates(subset=['lat','long'], keep="last").
 ↪reset_index(drop=True)
mhsub1.shape
```

[36]:
```
(21, 4)
```

[37]:
```
mhsub1.tail()
```

[37]:
```
                  sub_station                              sub_address  \
16   190 Street Subway Station   Bennett Ave, New York, NY 10040, USA
17  59 St-Lexington Av Station     E 60th St, New York, NY 10065, USA
18           23 Street Station      New York, NY 10010, United States
19            14 Street / 8 Av      New York, NY 10014, United States
20          MTA New York City   525 11th Ave, New York, NY 10018, USA

          lat       long
16  40.858113 -73.932983
17  40.762794 -73.967564
18  40.758798 -73.962343
19  40.712728 -74.006015
20  40.759809 -73.999282
```

14.1 MAP of Manhattan showing the location of subway stations

[38]:
```
# map subway stations
# create map of Manhattan using latitude and longitude values obtain previoulsy␣
 ↪via Moninatim geolocator
latitude=40.7308619
longitude=-73.9871558

map_mhsub1 = folium.Map(location=[latitude, longitude], zoom_start=12)

# add markers of subway locations to map
for lat, lng, label in zip(mhsub1['lat'], mhsub1['long'], ␣
 ↪mhsub1['sub_station'].astype(str) ):
    label = folium.Popup(label, parse_html=True)
    folium.RegularPolygonMarker(
        [lat, lng],
        number_of_sides=6,
        radius=6,
        popup=label,
        color='red',
```

```
        fill_color='red',
        fill_opacity=2.5,
    ).add_to(map_mhsub1)
map_mhsub1
```

[38]:
```
<folium.folium.Map at 0x7f3e6f5b17b8>
```

15 Map of Manhattan showing places for rent and the subway locations nearby

15.1 Now, we can visualize the desirable rental places and their nearest subway station. Popups display rental address and monthly rental price and the subway station name.

15.1.1 Notice that the icon in the top-right corner is a "ruler" that allows to measure the distance from a rental place to an specific subway station

[39]:
```
mh_rent.head()
```

[39]:

	Address	Area	Price_per_ft2	Rooms	Area-ft2	Rent_Price	Lat	Long
0	West 105th Street	Upper West Side	2.94	5.0	3400	10000	40.799771	-73.966213
1	East 97th Street	Upper East Side	3.57	3.0	2100	7500	40.788517	-73.955118
2	West 105th Street	Upper West Side	1.89	4.0	2800	5300	40.799771	-73.966213
3	CARMINE ST.	West Village	3.03	2.0	1650	5000	40.730337	-74.002476
4	171 W 23RD ST.	Chelsea	3.45	2.0	1450	5000	40.744118	-73.995299

[40]:
```
# create map of Manhattan using latitude and longitude values from Nominatim
latitude= 40.7308619
longitude= -73.9871558

map_manhattan_rent = folium.Map(location=[latitude, longitude], zoom_start=13.3)

# add markers to map
for lat, lng, label in zip(mh_rent['Lat'], mh_rent['Long'],'$ ' +␣
 ↪mh_rent['Rent_Price'].astype(str)+ mh_rent['Address']):
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=6,
```

```
        popup=label,
        color='blue',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.7,
        parse_html=False).add_to(map_manhattan_rent)

    # add markers of subway locations to map
for lat, lng, label in zip(mhsub1['lat'], mhsub1['long'], ␣
 ↪mhsub1['sub_station'].astype(str) ):
    label = folium.Popup(label, parse_html=True)
    folium.RegularPolygonMarker(
        [lat, lng],
        number_of_sides=6,
        radius=6,
        popup=label,
        color='red',
        fill_color='red',
        fill_opacity=2.5,
    ).add_to(map_manhattan_rent)

    # Adds tool to the top right
from folium.plugins import MeasureControl
map_manhattan_rent.add_child(MeasureControl())

# Measurement ruler icon tool to measure distances in map
from folium.plugins import FloatImage
url = ('https://media.licdn.com/mpr/mpr/shrinknp_100_100/
 ↪AAEAAQAAAAAAAAlgAAAAJGE3OTA4YTdlLTkzZjUtNDFjYy1iZThlLWQ5OTNkYzlhNzM4OQ.jpg')
FloatImage(url, bottom=5, left=85).add_to(map_manhattan_rent)

map_manhattan_rent
```

[40]: <folium.folium.Map at 0x7f3e6f4dfb38>

16 4.0 Results

16.1 ONE CONSOLIDATE MAP

16.1.1 Let's consolidate all the required inforamtion to make the apartment selection in one map

17 Map of Manhattan with rental places, subway locations and cluster of venues

17.0.1 Red dots are Subway stations, Blue dots are apartments available for rent, Bubbles are the clusters of venues

[41]:
```
# create map of Manhattan using latitude and longitude values from Nominatim
latitude= 40.7308619
longitude= -73.9871558

map_mh_one = folium.Map(location=[latitude, longitude], zoom_start=13.3)

# add markers to map
for lat, lng, label in zip(mh_rent['Lat'], mh_rent['Long'],'$ ' +␣
 ↪mh_rent['Rent_Price'].astype(str)+ ', '+mh_rent['Address']):
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=6,
        popup=label,
        color='blue',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.7,
        parse_html=False).add_to(map_mh_one)

    # add markers of subway locations to map
for lat, lng, label in zip(mhsub1['lat'], mhsub1['long'], ␣
 ↪mhsub1['sub_station'].astype(str) ):
    label = folium.Popup(label, parse_html=True)
    folium.RegularPolygonMarker(
        [lat, lng],
        number_of_sides=6,
        radius=6,
        popup=label,
        color='red',
        fill_color='red',
        fill_opacity=2.5,
    ).add_to(map_mh_one)
```

```
# set color scheme for the clusters
kclusters=5
x = np.arange(kclusters)
ys = [i+x+(i*x)**2 for i in range(kclusters)]
colors_array = cm.rainbow(np.linspace(0, 1, len(ys)))
rainbow = [colors.rgb2hex(i) for i in colors_array]

# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(manhattan_merged['Latitude'],␣
 ↪manhattan_merged['Longitude'], manhattan_merged['Neighborhood'],␣
 ↪manhattan_merged['Cluster Labels']):
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    folium.CircleMarker(
        [lat, lon],
        radius=15,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7).add_to(map_mh_one)

    # Adds tool to the top right
from folium.plugins import MeasureControl
map_mh_one.add_child(MeasureControl())

# Measurement ruler icon tool to measure distances in map
from folium.plugins import FloatImage
url = ('https://media.licdn.com/mpr/mpr/shrinknp_100_100/
 ↪AAEAAQAAAAAAAAlgAAAAJGE3OTA4YTdlLTkzZjUtNDFjYy1iZThlLWQ5OTNkYzlhNzM4OQ.jpg')
FloatImage(url, bottom=5, left=85).add_to(map_mh_one)

map_mh_one
```

[41]: <folium.folium.Map at 0x7f3e6f1d57f0>

18 Problem Resolution

18.1 The above consolidate map was used to explore options.

18.1.1 After examining, I have chosen two locations that meet the requirements which will assess to make a choice.

- Apartment 1: 305 East 63rd Street in the Sutton Place Neighborhood and near 'subway 59th Street' station, Cluster # 2 Monthly rent : 7500 Dollars
- Apartment 2: 19 Dutch Street in the Financial District Neighborhood and near 'Fulton Street Subway' station, Cluster # 3 Monthly rent : 6935 Dollars

18.2 Venues for Apartment 1 - Cluster 2

[42]:
```
## kk is the cluster number to explore
kk = 2
manhattan_merged.loc[manhattan_merged['Cluster Labels'] == kk, manhattan_merged.
 ↪columns[[1] + list(range(5, manhattan_merged.shape[1]))]]
```

[42]:

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0	Marble Hill	Coffee Shop	Discount Store	Yoga Studio	Steakhouse	Supplement Shop	Tennis Stadium	Shoe Store	Gym	Bank	Seafood Restaurant
1	Chinatown	Chinese Restaurant	Cocktail Bar	Dim Sum Restaurant	American Restaurant	Vietnamese Restaurant	Salon / Barbershop	Noodle House	Bakery	Bubble Tea Shop	Ice Cream Shop
6	Central Harlem	African Restaurant	Seafood Restaurant	French Restaurant	American Restaurant	Cosmetics Shop	Chinese Restaurant	Event Space	Liquor Store	Beer Bar	Gym / Fitness Center
9	Yorkville	Coffee Shop	Gym	Bar	Italian Restaurant	Sushi Restaurant	Pizza Place	Mexican Restaurant	Deli / Bodega	Japanese Restaurant	Pub
14	Clinton	Theater	Italian Restaurant	Coffee Shop	American Restaurant	Gym / Fitness Center	Hotel	Wine Shop	Spa	Gym	Indie Theater
23	Soho	Clothing Store	Boutique	Women's Store	Shoe Store	Men's Store	Furniture / Home Store	Italian Restaurant	Mediterranean Restaurant	Art Gallery	Design Studio
26	Morningside Heights	Coffee Shop	American Restaurant	Park	Bookstore	Pizza Place	Sandwich Place	Burger Joint	Café		
34	Sutton Place	Gym / Fitness Center	Italian Restaurant	Furniture / Home Store	Indian Restaurant	Dessert Shop	American Restaurant	Bakery	Juice Bar		
39	Hudson Yards	Coffee Shop	Italian Restaurant	Hotel	Theater	American Restaurant	Café	Gym / Fitness Center	Thai Restaurant		

```
26         Deli / Bodega           Tennis Court
34              Boutique       Sushi Restaurant
39            Restaurant                   Gym
```

[ ]:
```
## Venues for Apartment 2 - Cluster 3
```

[43]:
```
## kk is the cluster number to explore
kk = 3
manhattan_merged.loc[manhattan_merged['Cluster Labels'] == kk, manhattan_merged.
↪columns[[1] + list(range(5, manhattan_merged.shape[1]))]]
```

[43]:

```
          Neighborhood 1st Most Common Venue 2nd Most Common Venue  \
3               Inwood    Mexican Restaurant                Lounge
5        Manhattanville         Deli / Bodega    Italian Restaurant
10          Lenox Hill      Sushi Restaurant    Italian Restaurant
12     Upper West Side    Italian Restaurant                   Bar
16         Murray Hill        Sandwich Place                 Hotel
17             Chelsea           Coffee Shop    Italian Restaurant
18    Greenwich Village    Italian Restaurant      Sushi Restaurant
27            Gramercy    Italian Restaurant            Restaurant
29   Financial District           Coffee Shop                 Hotel
31                 Noho    Italian Restaurant     French Restaurant
32         Civic Center  Gym / Fitness Center                Bakery
35           Turtle Bay    Italian Restaurant           Coffee Shop
36           Tudor City                 Café                  Park
38             Flatiron    Italian Restaurant   American Restaurant

     3rd Most Common Venue          4th Most Common Venue  \
3              Pizza Place                          Café
5       Seafood Restaurant             Mexican Restaurant
10              Coffee Shop          Gym / Fitness Center
12                   Bakery  Vegetarian / Vegan Restaurant
16     Japanese Restaurant          Gym / Fitness Center
17          Ice Cream Shop                        Bakery
18       French Restaurant               Clothing Store
27  Thrift / Vintage Store                 Cocktail Bar
29                     Gym                    Wine Shop
31            Cocktail Bar                    Gift Shop
32      Italian Restaurant                 Cocktail Bar
35              Steakhouse                     Wine Bar
36              Pizza Place             Mexican Restaurant
38                     Gym          Gym / Fitness Center

   5th Most Common Venue          6th Most Common Venue 7th Most Common Venue  \
3               Wine Bar                         Bakery   American Restaurant
5       Sushi Restaurant                   Beer Garden           Coffee Shop
10           Pizza Place                   Burger Joint         Deli / Bodega
```

12	Indian Restaurant	Coffee Shop	Cosmetics Shop	
16	Coffee Shop	Salon / Barbershop	Burger Joint	
17	Nightclub	Theater	Art Gallery	
18	Chinese Restaurant	Café	Indian Restaurant	
27	Bagel Shop	Coffee Shop	Pizza Place	
29	Steakhouse	Bar	Italian Restaurant	
31	Bookstore	Grocery Store	Mexican Restaurant	
32	French Restaurant	Sandwich Place	Coffee Shop	
35	Sushi Restaurant	Hotel	Noodle House	
36	Greek Restaurant	Sushi Restaurant	Hotel	
38	Yoga Studio	Vegetarian / Vegan Restaurant	Bakery	

	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
3	Park	Frozen Yogurt Shop	Spanish Restaurant
5	Falafel Restaurant	Bike Trail	Other Nightlife
10	Gym	Sporting Goods Shop	Thai Restaurant
12	Wine Bar	Mexican Restaurant	Sushi Restaurant
16	French Restaurant	Bar	Italian Restaurant
17	Seafood Restaurant	American Restaurant	Hotel
18	Bakery	Seafood Restaurant	Electronics Store
27	Mexican Restaurant	Grocery Store	Wine Shop
29	Pizza Place	Park	Gym / Fitness Center
31	Hotel	Sushi Restaurant	Coffee Shop
32	Gym	Yoga Studio	Park
35	Indian Restaurant	Japanese Restaurant	French Restaurant
36	Deli / Bodega	Diner	Dog Run
38	Clothing Store	Cosmetics Shop	Cycle Studio

19 Apartment Selection

19.1 Using the "one map" above, I was able to explore all possibilities since the popups provide the information needed for a good decision.

19.1.1 Apartment 1 rent cost is US7500 slightly above the US7000 budget. Apt 1 is located 400 meters from subway station at 59th Street and work place (Park Ave and 53rd) is another 600 meters way. I can walk to work place and use subway for other places aroung. Venues for this apt are as of Cluster 2 and it is located in a fine district in the East side of Manhattan.

19.1.2 Apartment 2 rent cost is US6935, just under the US7000 budget. Apt 2 is located 60 meters from subway station at Fulton Street, but I will have to ride the subway daily to work , possibly 40-60 min ride. Venues for this apt are as of Cluster 3.¶

19.2 Based on current Singapore venues, I feel that Cluster 2 type of venues is a closer resemblance to my current place. That means that APARTMENT 1 is a better choice since the extra monthly rent is worth the conveniences it provides.