The Battle of Neighborhoods (Week 1)

Part 1 : Introduction and Data Sections

In [ ]:
```
import numpy as np # library to handle data in a vectorized manner
import time
import pandas as pd # library for data analysyis
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
```

In [ ]:
```
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

print('Libraries imported.')
```

1. Introduction Section :

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

Description of the Problem and Background

Scenario:

I am a student living Singapore. I want to use this opportunity to practice my learnings in Coursera in order to answer relevant questions arisen. The key question is : How can I find a convenient and enjoyable place similar to mine now in Singapore? Certainly, I can use available real estate apps and Google but the idea is to use and apply myself the learned tools during the course. In order to make a comparison and evaluation of the rental options in Manhattan NY, I must set some basis, therefore the apartment in Manhattan must meet the following demands:

```
apartment must be 2 or 3 bedrooms
desired location is near a metro station in the Manhattan area and within 1.0 mile (1.6 km) r
adius
price of rent not exceed $7,000 per month
top ammenities in the selected neighborhood shall be similar to current residence
desirable to have venues such as coffee shops, restaurants Asian Thai, wine stores, gym and f
ood shops
as a reference, I have included a map of venues near current residence in Singapore.
```

Business Problem:

The challenge is to find a suitable apartment for rent in Manhattan NY that complies with the demands on location, price and venues. The data required to resolve this challenge is described in the following section 2, below. Interested Audience

I believe this is a relevant challenge with valid questions for anyone moving to other large city in US, EU or Asia. The same methodology can be applied in accordance to demands as applicable. This case is also applicable for anyone interested in exploring starting or locating a new business in any city. Lastly, it can also serve as a good practical exercise to develop Data Science skills.

2. Data Section:¶

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

Description of the Data:¶

The following data is required to answer the issues of the problem:

```
List of Boroughs and neighborhoods of Manhattan with their geodata (latitud and longitud)
List of Subway metro stations in Manhattan with their address location
List of apartments for rent in Manhattan area with their addresses and price
Preferably, a list of apartment for rent with additional information, such as price, address,
area, # of beds, etc
Venues for each Manhattan neighborhood ( than can be clustered)
Venues for subway metro stations, as needed
```

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 cluste
red in groups ( as per Course LAB)
Use foursquare and geopy data to map the location of subway metro stations , separately and o
n top of the above clustered map in order to be able to identify the venues and ammenities ne
ar 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 t
he subway locations.
create a map that depicts, for instance, the average rental price per square ft, around a rad
ius of 1.0 mile (1.6 km) around each subway station - or a similar metrics. I will be able t
o 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 re
ading, libraries or other government agencies such as Metro New York MTA, etc.
```

The procesing of these DATA will allow to answer the key questions to make a decision:

```
what is the cost of rent (per square ft) 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 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.
```

Reference of venues around current residence in Singapore for comparison to Manhattan place

In [ ]:
```
# 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))
```

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

In [16]:
```
CLIENT_ID = 'DVCTZDPDYXTS0BRJFPLM0M323APGXNWZ15PLRQlVC0CFLF1T' # your Foursquare ID
CLIENT_SECRET = '5NWAGXRLXIXAV0L3DNYY1EPIHNMAAAISFDFELY5YXL5LFWL1' # your Foursquare Secret
VERSION = '20180605' # Foursquare API version

print('Your credentails:')
print('CLIENT_ID: ' + CLIENT_ID)
print('CLIENT_SECRET:' + CLIENT_SECRET)
```

```
Your credentails:
CLIENT_ID: DVCTZDPDYXTS0BRJFPLM0M323APGXNWZ15PLRQlVC0CFLF1T
CLIENT_SECRET:5NWAGXRLXIXAV0L3DNYY1EPIHNMAAAISFDFELY5YXL5LFWL1
```

In [17]:
```
LIMIT = 100 # limit of number of venues returned by Foursquare API
radius = 500 # define radius

# create URL
url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={}&v={}&ll={},{}&rad
ius={}&limit={}'.format(
    CLIENT_ID,
    CLIENT_SECRET,
    VERSION,
    neighborhood_latitude,
    neighborhood_longitude,
    radius,
    LIMIT)
url # display URL
```

Out[17]:
```
'https://api.foursquare.com/v2/venues/explore?&client_id=DVCTZDPDYXTS0BRJFPLM0M323APGXNWZ15PLRQl
VC0CFLF1T&client_secret=5NWAGXRLXIXAV0L3DNYY1EPIHNMAAAISFDFELY5YXL5LFWL1&v=20180605&ll=1.279265
5,103.8480938&radius=500&limit=100'
```

In [18]:
```
results = requests.get(url).json()
#results
```

In [19]:
```
# function that extracts the category of the venue
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']
```

In [20]:
```
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.head(10)
```

Out[20]:

	name	categories	lat	lng
0	Napoleon Food & Wine Bar	Wine Bar	1.279925	103.847333
1	Native	Cocktail Bar	1.280135	103.846844
2	Pepper Bowl	Asian Restaurant	1.279371	103.846710
3	Park Bench Deli	Deli / Bodega	1.279872	103.847287
4	Muchachos	Burrito Place	1.279072	103.847026
5	Mellower Coffee	Café	1.277814	103.848188
6	Freehouse	Beer Garden	1.281254	103.848513
7	Sofitel So Singapore	Hotel	1.280017	103.849813
8	Dumpling Darlings	Dumpling Restaurant	1.280483	103.846942
9	PS.Cafe	Café	1.280468	103.846264

In [21]:
```
# create map of Singapore place  using latitude and longitude values
map_sg = folium.Map(location=[latitude, longitude], zoom_start=20)

# 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=4,
        radius=10,
        popup=label,
        color='blue',
        fill_color='#0f0f0f',
        fill_opacity=0.7,
    ).add_to(map_sg)

map_sg
```

Out[21]: