Introduction

For this exercise I'd like to introduce you to my friend Jack. Jack is a talented chef, his mom is french, his dad is italian, he grew up in asia and lives in Manhatten. Jack can cook any cuisine you can think of to perfection. Here is the problem: Jack wants to open a restaurant/foodvenue on Manhattan, and he wants to use data to help him make the hard descition of the kind of restaurant he should open and where he should open it. Ideally he would like a recommendation for each neighborhood on Manhattan, based on the existing food places in the neighborhood.

Dataset

To complete the task I will need to isolate the Manhattan data from the New York dataset and the dataset must fulfill these requirements:

1) The data set must be food venues only.
2) The data set must include food category.
3) The data set must include latitude and logitude coordinates of each neighborhood.

Methodology

1) First I'll download the data.
2) Draw a map of the Manhattan Neiborhoods.
3) Find out the top10 food venue types for all of Manhattan.
4) Isolate top10 food-venue type and make a dataframe for these and the Manhattan neiborhoods.
5) Based on the average amount of food-venue type, rate each food-venue type for each neiborhood.

1) Download and prep the data

```
!wget -q -O 'newyork_data.json' https://cocl.us/new_york_dataset
print('Data downloaded!')
```

```
Data downloaded!
```

```
with open('newyork_data.json') as json_data:
    newyork_data = json.load(json_data)
```

```
neighborhoods_data = newyork_data['features']
```

The above just returns a json object for all of NY, next step is to tranform the data into a dataframe

```
# define the dataframe columns
column_names = ['Borough', 'Neighborhood', 'Latitude', 'Longitude']

# instantiate the dataframe
neighborhoods = pd.DataFrame(columns=column_names)
neighborhoods
```

Borough	Neighborhood	Latitude	Longitude

```
#Fill in Manhattan data
for data in neighborhoods_data:
    borough = neighborhood_name = data['properties']['borough']
    neighborhood_name = data['properties']['name']
    neighborhood_latlon = data['geometry']['coordinates']
    neighborhood_lat = neighborhood_latlon[1]
    neighborhood_lon = neighborhood_latlon[0]
    neighborhoods = neighborhoods.append({'Borough': borough,
                                          'Neighborhood': neighborhood_name,
                                          'Latitude': neighborhood_lat,
                                          'Longitude': neighborhood_lon}, ignore_index=True)
neighborhoods.head()
```

	Borough	Neighborhood	Latitude	Longitude
0	Bronx	Wakefield	40.894705	-73.847201
1	Bronx	Co-op City	40.874294	-73.829939
2	Bronx	Eastchester	40.887556	-73.827806
3	Bronx	Fieldston	40.895437	-73.905643
4	Bronx	Riverdale	40.890834	-73.912585

```
print('The dataframe has {} boroughs and {} neighborhoods.'.format(
        len(neighborhoods['Borough'].unique()),
        neighborhoods.shape[0]
    )
)
```

```
The dataframe has 5 boroughs and 306 neighborhoods.
```

Time to isolate Mahattan data

```
#Isolate Mahattan Data
manhattan_data = neighborhoods[neighborhoods['Borough'] == 'Manhattan'].reset_index(drop=True)
manhattan_data.head()
```

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

2) Draw a map of the Manhattan Neiborhoods

Get Manhattan coordinates for the map viewport

```
address = 'Manhattan, NY'

geolocator = Nominatim(user_agent="ny_explorer")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print('The geograpical coordinate of Manhattan are {}, {}.'.format(latitude, longitude))
```

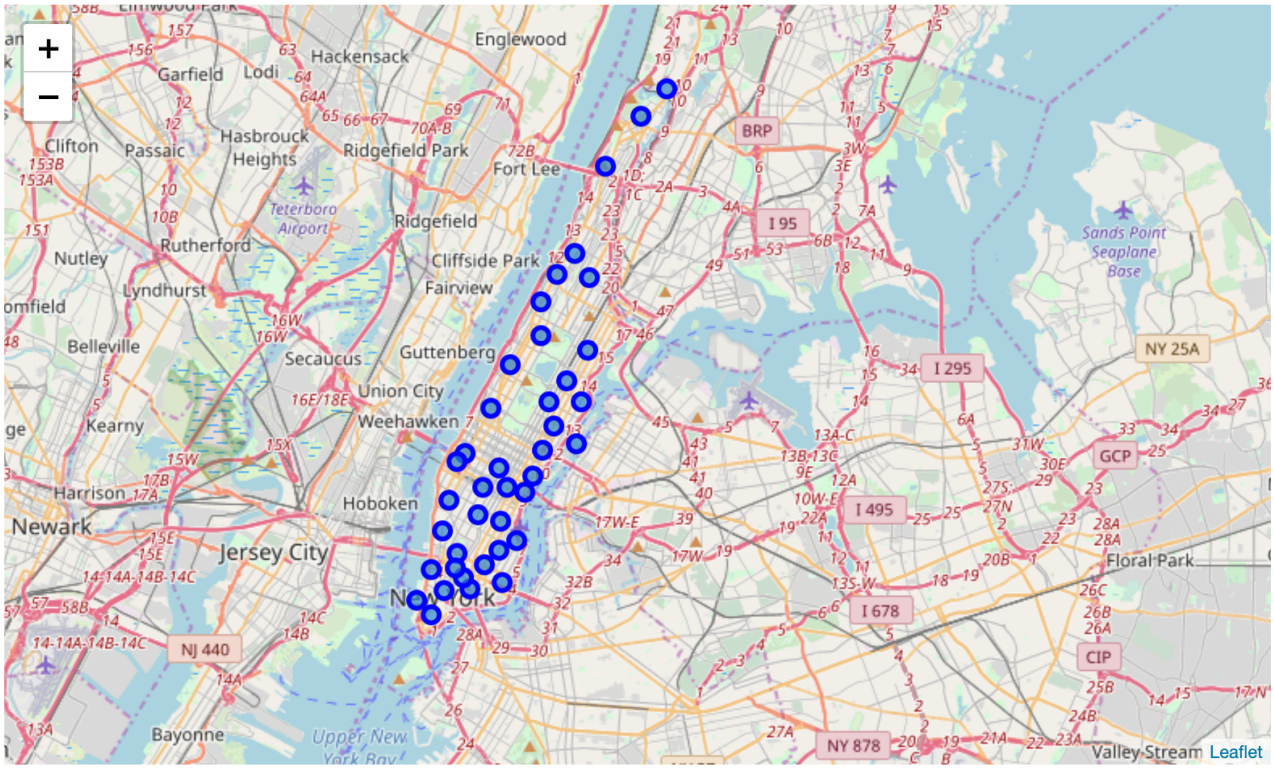
```
The geograpical coordinate of Manhattan are 40.7900869, -73.9598295.
```

Create map of Manhattan using latitude and longitude values

```
# Create map
map_manhattan = folium.Map(location=[latitude, longitude], zoom_start=11)

# add markers 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_manhattan)

map_manhattan
```



3) Find out the top10 food venue types for all of Manhattan.

Next step is to create a function that gets the food-venue data. Notice the section is set to food, so it only returns food-venues.

```
def getNearbyVenues(names, latitudes, longitudes, radius=500):
    section = 'food'
    venues_list=[]
    LIMIT = 100
    for name, lat, lng in zip(names, latitudes, longitudes):
        print(name)

        # create the API request URL
        url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={}&v={}&ll={},{}&radius={}&sec
tion={}&limit={}'.format(
            CLIENT_ID,
            CLIENT_SECRET,
            VERSION,
            lat,
            lng,
            radius,
            section,
            LIMIT)

        # make the GET request
        results = requests.get(url).json()["response"]['groups'][0]['items']

        # 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 = ['Neighborhood',
                  'Neighborhood Latitude',
                  'Neighborhood Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']

    return(nearby_venues)
```

Set up dataframe for all neighborhoods

```
manhattan_venues = getNearbyVenues(names=manhattan_data['Neighborhood'],
                                   latitudes=manhattan_data['Latitude'],
                                   longitudes=manhattan_data['Longitude']
                                  )
```

4) Isolate top10 food-venue type and make a dataframe for these and the Manhattan neiborhoods.

Neighborhood	Italian Restaurant	Pizza Place	Café	American Restaurant	Deli / Bodega	Sandwich Place	Mexican Restaurant	Chinese Restaurant	Bakery	French Restaurant
Sutton Place	7.000	9.000	1.000	4.000	1.000	1.000	3.00	3.00	1.00	4.00
Tribeca	7.000	1.000	5.000	7.000	4.000	3.000	1.00	2.00	3.00	2.00
Tudor City	2.000	5.000	7.000	3.000	9.000	3.000	5.00	2.00	0.00	1.00
Turtle Bay	10.000	3.000	6.000	3.000	9.000	3.000	1.00	0.00	0.00	3.00
Upper East Side	15.000	5.000	2.000	6.000	4.000	1.000	2.00	2.00	3.00	4.00
Upper West Side	7.000	3.000	2.000	2.000	0.000	0.000	2.00	1.00	3.00	2.00
Washington Heights	2.000	9.000	3.000	1.000	7.000	3.000	4.00	6.00	4.00	0.00
West Village	17.000	4.000	2.000	9.000	0.000	2.000	4.00	2.00	1.00	4.00
Yorkville	10.000	10.000	2.000	2.000	9.000	4.000	3.00	3.00	3.00	1.00
AVG	6.175	4.225	3.575	3.525	3.275	2.975	2.75	2.75	2.65	2.25

```
# I want to know the 10 most popular food venues
df = manhattan_venues['Venue Category'].value_counts()
df.head(10)
```

```
Italian Restaurant     247
Pizza Place            169
Café                   143
American Restaurant    141
Deli / Bodega          131
Sandwich Place         119
Chinese Restaurant     110
Mexican Restaurant     110
Bakery                 106
French Restaurant       90
Name: Venue Category, dtype: int64
```

Now we set up a dataframe that shows the neighborhood-location of each type of restaurant.

```
# one hot encoding
manhattan_onehot = pd.get_dummies(manhattan_venues[['Venue Category']], prefix="", prefix_sep="")

# add neighborhood column back to dataframe
manhattan_onehot['Neighborhood'] = manhattan_venues['Neighborhood']

# move neighborhood column to the first column
fixed_columns = [manhattan_onehot.columns[-1]] + list(manhattan_onehot.columns[:-1])
manhattan_onehot = manhattan_onehot[fixed_columns]
manhattan_onehot.shape
manhattan_onehot.head()
```

	Neighborhood	Afghan Restaurant	African Restaurant	American Restaurant	Arepa Restaurant	Argentinian Restaurant	Asian Restaurant	Australian Restaurant	Austrian Restaurant	BBQ Joint	Bagel Shop	Bakery	Belgian Restaurant	Bistro	I
0	Marble Hill	0	0	0	0	0	0	0	0	0	0	0	0	0	
1	Marble Hill	0	0	0	0	0	0	0	0	0	0	0	0	0	
2	Marble Hill	0	0	0	0	0	0	0	0	0	0	0	0	0	
3	Marble Hill	0	0	0	0	0	0	0	0	0	0	0	0	0	
4	Marble Hill	0	0	0	0	0	0	0	0	0	0	0	0	0	

Next, I group rows by neighborhood and summarize for each category. The top 10 venue categories are specified in the code below, to clear out all other categories.

```
manhattan_grouped = manhattan_onehot.groupby('Neighborhood').sum()
manhattan_grouped = manhattan_grouped[['Italian Restaurant','Pizza Place','Café','American Restaurant','Deli / Bodega'
,'Sandwich Place','Mexican Restaurant','Chinese Restaurant','Bakery','French Restaurant']]
manhattan_grouped.head()
```

Neighborhood	Italian Restaurant	Pizza Place	Café	American Restaurant	Deli / Bodega	Sandwich Place	Mexican Restaurant	Chinese Restaurant	Bakery	French Restaurant
Battery Park City	4	4	0	2	0	2	1	3	1	0
Carnegie Hill	4	8	5	2	1	0	2	1	6	3
Central Harlem	0	3	1	2	3	2	0	4	1	2
Chelsea	5	4	5	4	2	3	4	2	8	6
Chinatown	1	2	3	3	0	3	4	19	7	0

Done, its starting to look prety useful. Next step is to find the average number of the top 10 restaurants per Neighborhood, and add that as the last row (called AVG).

```
# Lets find the average number of the top 10 restaurants per Neighborhood, and add that as the last row.
manhattan_grouped.loc['AVG'] = manhattan_grouped.mean()
manhattan_grouped.tail(10)
```

Neighborhood	Italian Restaurant	Pizza Place	Café	American Restaurant	Deli / Bodega	Sandwich Place	Mexican Restaurant	Chinese Restaurant	Bakery	French Restaurant
Sutton Place	7.000	9.000	1.000	4.000	1.000	1.000	3.00	3.00	1.00	4.00
Tribeca	7.000	1.000	5.000	7.000	4.000	3.000	1.00	2.00	3.00	2.00
Tudor City	2.000	5.000	7.000	3.000	9.000	3.000	5.00	2.00	0.00	1.00
Turtle Bay	10.000	3.000	6.000	3.000	9.000	3.000	1.00	0.00	0.00	3.00
Upper East Side	15.000	5.000	2.000	6.000	4.000	1.000	2.00	2.00	3.00	4.00
Upper West Side	7.000	3.000	2.000	2.000	0.000	0.000	2.00	1.00	3.00	2.00
Washington Heights	2.000	9.000	3.000	1.000	7.000	3.000	4.00	6.00	4.00	0.00
West Village	17.000	4.000	2.000	9.000	0.000	2.000	4.00	2.00	1.00	4.00
Yorkville	10.000	10.000	2.000	2.000	9.000	4.000	3.00	3.00	3.00	1.00
AVG	6.175	4.225	3.575	3.525	3.275	2.975	2.75	2.75	2.65	2.25

5) Based on the average amount of food-venue type, rate each food-venue type for each neighbourhood.

Now we are getting to the recommendation part. I'll deduct the average from the actual number of food places for a certain category, and thereby give each food-category a score for a neighborhood. **For example** if the average number of pizzaplaces is 4 and a neighborhood only has two, the Pizzaplace score for this Neighborhood will be +2. If a neighborhood already has 8 pizzaplaces, the score is -4, meaning 'Dont open a pizzaplace here, there are plenty'.

```
# Subtract the average [AVG] from all rows, and flip the sign (+/-) to make negative potins negative and positive poin
ts positive
df = manhattan_grouped.iloc[0:40].subtract(manhattan_grouped.iloc[40])
df = df*-1
df.head(10)
```

Neighborhood	Italian Restaurant	Pizza Place	Café	American Restaurant	Deli / Bodega	Sandwich Place	Mexican Restaurant	Chinese Restaurant	Bakery	French Restaurant
Battery Park City	2.175	0.225	3.575	1.525	3.275	0.975	1.75	-0.25	1.65	2.25
Carnegie Hill	2.175	-3.775	-1.425	1.525	2.275	2.975	0.75	1.75	-3.35	-0.75
Central Harlem	6.175	1.225	2.575	1.525	0.275	0.975	2.75	-1.25	1.65	0.25
Chelsea	1.175	0.225	-1.425	-0.475	1.275	-0.025	-1.25	0.75	-5.35	-3.75
Chinatown	5.175	2.225	0.575	0.525	3.275	-0.025	-1.25	-16.25	-4.35	2.25
Civic Center	-5.825	1.225	-1.425	-2.475	0.275	-5.025	-2.25	1.75	-1.35	-3.75
Clinton	-3.825	1.225	-0.425	-4.475	-4.725	-2.025	-0.25	-2.25	1.65	0.25
East Harlem	6.175	-1.775	1.575	3.525	-2.725	0.975	-4.25	1.75	-2.35	1.25
East Village	1.175	-4.775	0.575	1.525	0.275	2.975	-2.25	-3.25	1.65	-1.75
Financial District	0.175	-1.775	-2.425	-3.475	-0.725	-5.025	-2.25	1.75	0.65	1.25

It works! Looks like it would be a good idea to look into opening an Italian restaurant in Central Harlem, and a less good ideas to open a Chinese restaurant in Chinatown, or a bakery in Chelsea. Would be nice if it was easier to read the data though..

Highlight the results with pandas style function:

```
def highlight_max(s):
    '''
    highlight the maximum in a Series yellow.
    '''
    is_max = s == s.max()
    return ['background-color: yellow' if v else '' for v in is_max]
```

```
df.style.apply(highlight_max)
```

Neighborhood	Italian Restaurant	Pizza Place	Café	American Restaurant	Deli / Bodega	Sandwich Place	Mexican Restaurant	Chinese Restaurant	Bakery	French Restaurant
Battery Park City	2.175	0.225	3.575	1.525	3.275	0.975	1.75	-0.25	1.65	2.25
Carnegie Hill	2.175	-3.775	-1.425	1.525	2.275	2.975	0.75	1.75	-3.35	-0.75
Central Harlem	6.175	1.225	2.575	1.525	0.275	0.975	2.75	-1.25	1.65	0.25
Chelsea	1.175	0.225	-1.425	-0.475	1.275	-0.025	-1.25	0.75	-5.35	-3.75
Chinatown	5.175	2.225	0.575	0.525	3.275	-0.025	-1.25	-16.25	-4.35	2.25
Civic Center	-5.825	1.225	-1.425	-2.475	0.275	-5.025	-2.25	1.75	-1.35	-3.75
Clinton	-3.825	1.225	-0.425	-4.475	-4.725	-2.025	-0.25	-2.25	1.65	0.25
East Harlem	6.175	-1.775	1.575	3.525	-2.725	0.975	-4.25	1.75	-2.35	1.25
East Village	1.175	-4.775	0.575	1.525	0.275	2.975	-2.25	-3.25	1.65	-1.75
Financial District	0.175	-1.775	-2.425	-3.475	-0.725	-5.025	-2.25	1.75	0.65	1.25
Flatiron	-8.825	2.225	-1.425	-2.475	3.275	-3.025	-1.25	1.75	-0.35	-0.75
Gramercy	1.175	0.225	1.575	-0.475	0.275	-0.025	-1.25	1.75	1.65	2.25
Greenwich Village	-12.825	0.225	-2.425	-1.475	3.275	-0.025	1.75	-1.25	0.65	-1.75
Hamilton Heights	5.175	-0.775	-0.425	3.525	-6.725	-0.025	-4.25	-1.25	0.65	2.25
Hudson Yards	2.175	4.225	-2.425	-1.475	1.275	0.975	2.75	1.75	2.65	2.25
Inwood	6.175	-1.775	-0.425	1.525	-1.725	1.975	-4.25	0.75	-0.35	2.25
Lenox Hill	-8.825	-3.775	-2.425	1.525	-3.725	0.975	-0.25	-1.25	-0.35	-0.75
Lincoln Square	0.175	2.225	-6.425	-1.475	1.275	2.975	1.75	-0.25	1.65	-0.75

Little Italy	-7.825	1.225	-3.425	1.525	2.275	-1.025	1.75	-4.25	-7.35	-0.75
Lower East Side	5.175	-0.775	0.575	2.525	0.275	0.975	0.75	-0.25	-0.35	1.25
Manhattan Valley	5.175	-0.775	1.575	2.525	0.275	2.975	-0.25	0.75	1.65	0.25
Manhattanville	4.175	3.225	2.575	2.525	-1.725	0.975	-0.25	-1.25	1.65	2.25
Marble Hill	6.175	3.225	3.575	1.525	1.275	-1.025	2.75	2.75	1.65	2.25
Midtown	3.175	0.225	1.575	-4.475	0.275	-4.025	-0.25	1.75	-3.35	-1.75
Midtown South	2.175	2.225	1.575	-2.475	2.275	-2.025	2.75	0.75	-1.35	1.25
Morningside Heights	5.175	0.225	0.575	0.525	-1.725	0.975	1.75	0.75	2.65	2.25
Murray Hill	2.175	1.225	0.575	-2.475	2.275	-6.025	0.75	-0.25	0.65	-1.75
Noho	-5.825	-5.775	-0.425	0.525	2.275	-1.025	-2.25	2.75	-0.35	-2.75
Roosevelt Island	6.175	3.225	2.575	3.525	1.275	0.975	2.75	1.75	2.65	2.25
Soho	-9.825	2.225	-4.425	-2.475	2.275	-1.025	0.75	1.75	-2.35	-8.75
Stuyvesant Town	6.175	4.225	3.575	3.525	2.275	1.975	2.75	2.75	1.65	2.25
Sutton Place	-0.825	-4.775	2.575	-0.475	2.275	1.975	-0.25	-0.25	1.65	-1.75
Tribeca	-0.825	3.225	-1.425	-3.475	-0.725	-0.025	1.75	0.75	-0.35	0.25
Tudor City	4.175	-0.775	-3.425	0.525	-5.725	-0.025	-2.25	0.75	2.65	1.25
Turtle Bay	-3.825	1.225	-2.425	0.525	-5.725	-0.025	1.75	2.75	2.65	-0.75
Upper East Side	-8.825	-0.775	1.575	-2.475	-0.725	1.975	0.75	0.75	-0.35	-1.75
Upper West Side	-0.825	1.225	1.575	1.525	3.275	2.975	0.75	1.75	-0.35	0.25
Washington Heights	4.175	-4.775	0.575	2.525	-3.725	-0.025	-1.25	-3.25	-1.35	2.25
West Village	-10.825	0.225	1.575	-5.475	3.275	0.975	-1.25	0.75	1.65	-1.75
Yorkville	-3.825	-5.775	1.575	1.525	-5.725	-1.025	-0.25	-0.25	-0.35	1.25

Result

The dataframe above is the result of the exercise. Jack's wish was a recommendation for each neighbourhood on Manhattan, based on the existing food places in the neighbourhood. The resulting dataframe provides that, as it highlights the restaurants for each neighbourhood. The category that scores the highest is "Italian restaurant" with a value of 6.175 in 6 neighbourhoods, so I would recommend Jack to start looking at those first. We also get some strong advise for not opening an Italian restaurant in West village (negative 10.8 points).

Discussion

Although the results does satisfy the business problem, there are ways to improve it. We could for example add population for each neighbourhood, to make sure there are enough people living there, compared to the amount of restaurants. Also, the model is based on existing competition, and therefore it would not recommend opening a chinese restaurant in chinatown, but instead recommend opening an italian restaurant in chinatown. This might or might not be an issue with the model, it's hard to say at this point and would require more research.

Conclusion

Jack’s wish was a recommendation for each neighbourhood on Manhattan, based on the existing food places in the neighbourhood. The resulting dataframe provides that, as it highlights the restaurants for each neighbourhood, that scores the highest, is in the topten most liked food categories, and not have too much competition in the neighbourhood.