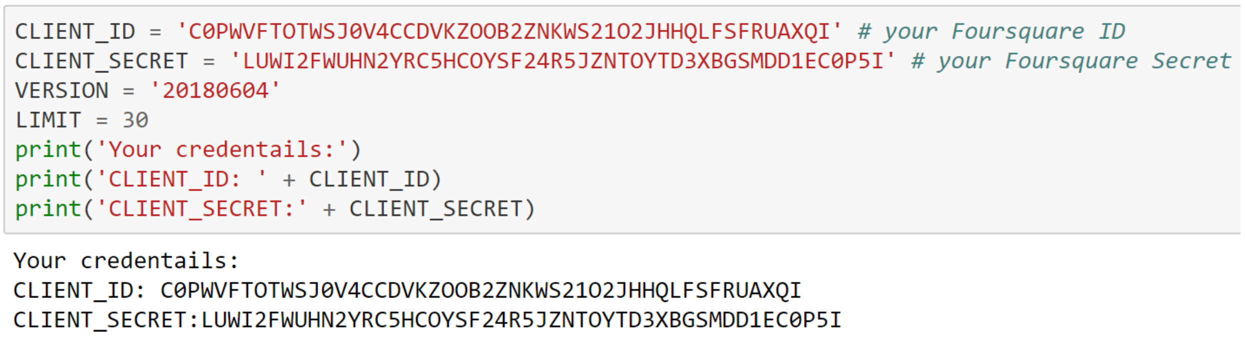
Introduction.

New York is the most populous city in the United States. Now imagine, that you have to work half a day in one location, and second half of a day in another location. You are moving from one office to another by taxi. And you want to have lunch somewhere nearby your offices. You like, for example, Greek cuisine. You are looking for a good, trusted venue with the highest rate. To solve this problem, you want to leverage the Foursquare location data to find all existing Greek venues nearby both offices and compare those venues inside one neighbourhood and then between two neighbourhoods, to find one, which suits all your requests.

Data Description.

To solve this problem, you need Forsquare API to get the most common venues of given Borough of New York nearby your offices.

To do so, firstly you need to define Foursquare credentials and version.

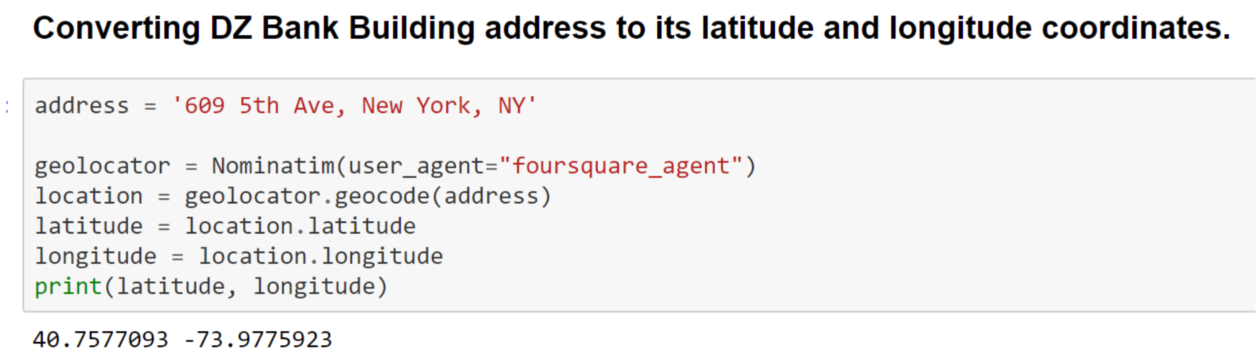


```
CLIENT_ID = 'C0PWVFTOTWSJ0V4CCDVKZOOB2ZNKWS21O2JHHQLFSFRUAXQI' # your Foursquare ID
CLIENT_SECRET = 'LUWI2FWUHN2YRC5HCOYSF24R5JZNTOYTD3XBGSMDD1EC0P5I' # your Foursquare Secret
VERSION = '20180604'
LIMIT = 30
print('Your credentails:')
print('CLIENT_ID: ' + CLIENT_ID)
print('CLIENT_SECRET:' + CLIENT_SECRET)
```

```
Your credentails:
CLIENT_ID: C0PWVFTOTWSJ0V4CCDVKZOOB2ZNKWS21O2JHHQLFSFRUAXQI
CLIENT_SECRET:LUWI2FWUHN2YRC5HCOYSF24R5JZNTOYTD3XBGSMDD1EC0P5I
```

Then you need to convert both offices locations to its latitude and longitude coordinates.

First office, DZ Bank Bulding.



Converting DZ Bank Building address to its latitude and longitude coordinates.

```
address = '609 5th Ave, New York, NY'

geolocator = Nominatim(user_agent="foursquare_agent")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print(latitude, longitude)
```

```
40.7577093 -73.9775923
```

Second office, Heffner Agency.

Converting Heffner Agency address to its latitude and longitude coordinates.

```
address = '40 Wall St, New York, NY'

geolocator = Nominatim(user_agent="foursquare_agent")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print(latitude, longitude)
```

```
40.707021850000004 -74.00966692266792
```

Now you can search both locations.

Methodology.

Let's assume that it is lunch time, and you are craving Greek food. So, let's define a query to search for Greek food that is within 500 metres from the DZ Bank Bulding and Heffner Agency. Define the corresponding URL:

```
search_query = 'Greek'
radius = 500
print(search_query + ' .... OK!')
```

```
Greek .... OK!
```

```
url = 'https://api.foursquare.com/v2/venues/search?client_id={}&client_secret={}&ll={},{}&v={}&query={}&radius={}&limit={}'.form
at(CLIENT_ID, CLIENT_SECRET, latitude, longitude, VERSION, search_query, radius, LIMIT)
url
```

```
'https://api.foursquare.com/v2/venues/search?client_id=C0PWVFTOTWSJ0V4CCDVKZOOB2ZNKWS21O2JHHQLFSFRUAXQI&client_secret=LUWI2FWUH
N2YRC5HCOYSF24R5JZNTOYTD3XBGSMDD1EC0P5I&ll=40.7577093,-73.9775923&v=20180604&query=Greek&radius=500&limit=30'
```

Get relevant part of JSON and transform it into a pandas dataframe:

```
# assign relevant part of JSON to venues
venues = results['response']['venues']

# tranform venues into a dataframe
dataframe = json_normalize(venues)
dataframe.head()
```

```
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages/ipykernel_launcher.py:5: FutureWarning: pandas.io.json.json_norm
alize is deprecated, use pandas.json_normalize instead
  """
```

	id	name	categories	referralId	hasPerk	location.address	location.crossStreet	location.lat	location.lng	loca
0	5414bab4498e79ad23ecce9b	GRK Fresh Greek	[{'id': '4bf58dd8d48988d10e941735', 'name': 'G...	v-1593680805	False	451 Lexington Ave	44th Street	40.753128	-73.974666	
1	5ae8484cd7627e002cc4e055	Greek On 6 Ave	[{'id': '4bf58dd8d48988d1cb941735', 'name': 'F...	v-1593680805	False	NaN	NaN	40.759335	-73.981336	4
2	4c0fa77198102d7fa18ae506	Joannne's Amazing Greek Cart	[{'id': '4bf58dd8d48988d1cb941735', 'name': 'F...	v-1593680805	False	NaN	47th St & Park Avenue	40.756834	-73.972084	4

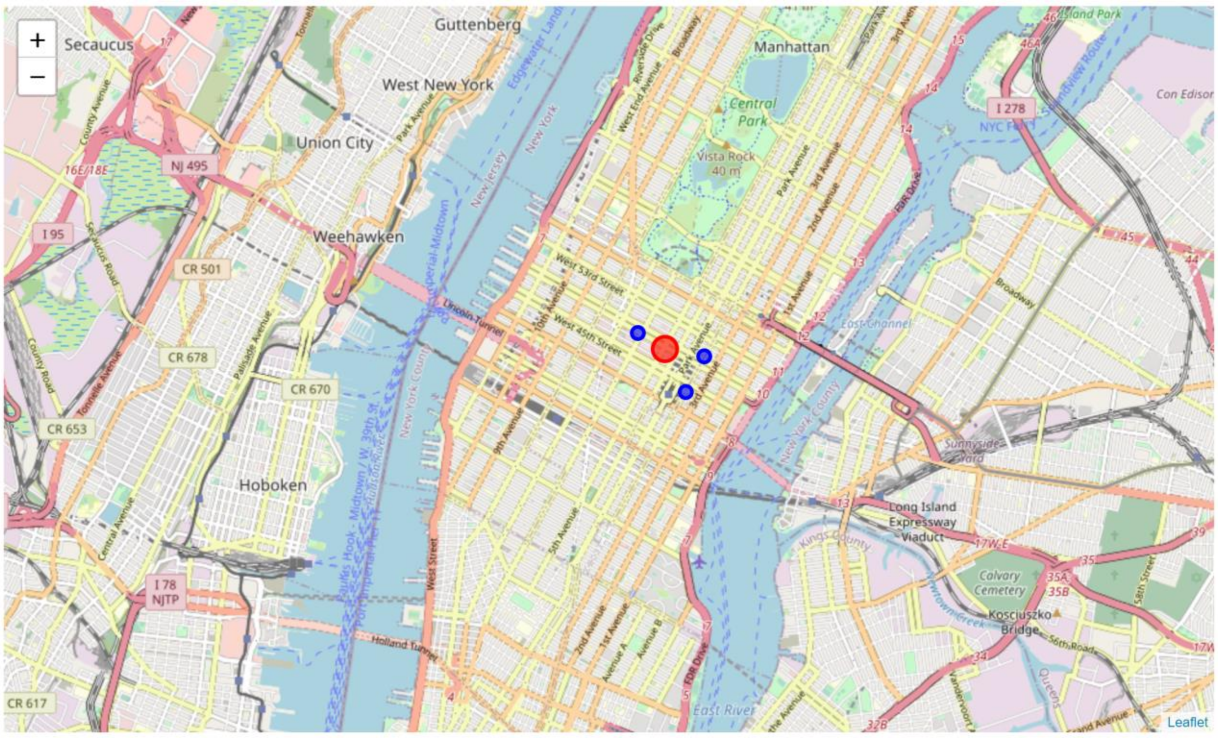
3 rows × 24 columns

We can see, that there are three restaurants nearby the first office. Define information of interest and filter dataframe:

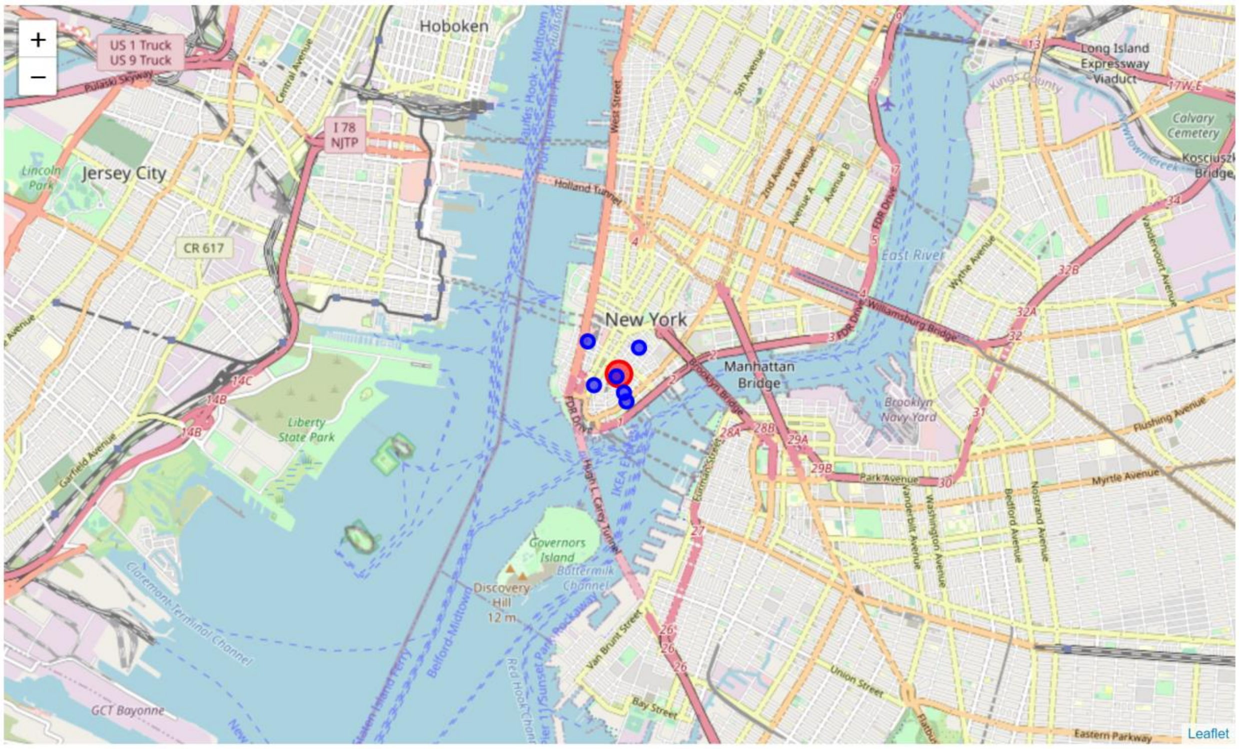
```python
# keep only columns that include venue name, and anything that is associated with location
filtered_columns = ['name', 'categories'] + [col for col in dataframe.columns if col.startswith('location.')] + ['id']
dataframe_filtered = dataframe.loc[:, filtered_columns]

# 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']

# filter the category for each row
dataframe_filtered['categories'] = dataframe_filtered.apply(get_category_type, axis=1)

# clean column names by keeping only last term
dataframe_filtered.columns = [column.split('.')[-1] for column in dataframe_filtered.columns]

dataframe_filtered
```

	name	categories	address	crossStreet	lat	lng	labeledLatLngs	distance	postalCode	cc	city	state	country	formattedAddress
0	GRK Fresh Greek	Greek Restaurant	451 Lexington Ave	44th Street	40.753128	-73.974666	[{'label': 'display', 'lat': 40.753128, 'lng':...	566	10017	US	New York	NY	United States	[451 Lexington Ave (44th Street), New York, NY...
1	Greek On 6 Ave	Food Truck	NaN	NaN	40.759335	-73.981336	[{'label': 'display', 'lat': 40.75933511750253...	363	10019	US	New York	NY	United States	[New York, NY 10019, United States]
2	Joanne's Amazing Greek Cart	Food Truck	NaN	47th St & Park Avenue	40.756834	-73.972084	[{'label': 'display', 'lat': 40.75683449877979...	474	NaN	US	New York	NY	United States	[New York, NY, United States]

Now we can see names, categories, address and so on. Let's visualize the Greek restaurants that are nearby:



We can repeat the same search for the second location and visualize the result as well:



Results.

Now you want to know restaurants ratings to choose the best option. Get the venue's overall rating for each place:

```
venue_id = '5047c785e4b0bcc0f416cdb3' # ID of GRK Fresh Greek Restaurant
url = 'https://api.foursquare.com/v2/venues/{}?client_id={}&client_secret={}&v={}'.format(venue_id, CLIENT_ID, CLIENT_SECRET, VE
RSION)

result = requests.get(url).json()
try:
    print(result['response']['venue']['rating'])
except:
    print('This venue has not been rated yet.')
```

```
7.8
```

The resaults in the table below:

Restraunts	Rating
GRK Fresh Greek	7.3
Greek On 6 Ave	This venue has not been rated yet.
Joannne's Amazing Greek Cart	This venue has not been rated yet.
GRK Fresh Greek - Financial District	7.8
Ehhnviko Greek Cart	This venue has not been rated yet.
Greek From Greece (GFG)	6.6
Saint Nicholas Greek Orthodox Church	This venue has not been rated yet.
Hanover Greek	5.6
Absolute Greek Food Truck	This venue has not been rated yet.

Discussion.

We can see that the best place is in the district number two nearby Heffner Agency.