Final Report, Toronto Neighborhood Analysis

1. Introduction
    a. Schmeck Industries has commissioned us to find a solution to a relocation request for its top officers in its company. The company headquarters is relocating to Toronto from Charlotte. The President of the company has asked for help in finding a neighborhood in Toronto that satisfies certain criteria, and will use that analysis to help officers find new residences.
    b. Requirements:
        i. All officers will be asked to live within the same neighborhood to promote camaraderie and fellowship amongst the team.
        ii. All officers will work remotely within the neighborhood and will be highly mobile travelling to each other's homes for work purposes as there will not be an official corporate headquarters office being built for at least 10 years.
        iii. Neighborhood Must Haves
        iv. Bars, taverns, and restaurants available within walking distance of the center of the neighborhood, as the officers enjoy getting together after workdays to discuss business plans, market conditions, and football while having beers. "Within walking distance" is defined as within 700 meters of the center of the neighborhood.
        v. Cafes or coffee shops within walking distance of the center of the neighborhood, as the officers often require a caffeine boost in the morning after events from the night before.
        vi. Parks or greenway trails within walking distance of the center of the neighborhood for the officer's families to take advantage of
        vii. Available city transit service within walking distance of the center of the neighborhood, as most officers do not have automobiles and depend on transit services to get around, even though most work will be done remotely.
        viii. The neighborhood cannot be subject to any airport noise higher than 45dB
        ix. A total of 3 neighborhoods will need be presented, with basic details listed for each so the team can make their decisions on where to relocate as a group.
        x. Local real estate market, pricing, and availability are not of concern for this purpose as the company is funding all home purchases for the officers to relocate, and will handle the finances of those purchases outside of this analysis
2. Data
    a. Neighborhood Location Data
        i. Neighborhood Location Data was scraped from the Toronto Neighborhood Wikipedia page at https://en.wikipedia.org/w/index.php?title=List_of_postal_codes_of_Canada:_M&oldid=890001695
        ii. Neighborhood data was sorted by latitude and longitude, and plotted on a map for visual reference.
        iii. Data was then run through the Foursquare Developer API to identify local venues close to the center of each neighborhood, within 700 meters
        iv. Venues were organized by frequency in each neighborhood
        v. Venues were then filtered for key words based on the requirements given, and the top 3 results were set as the final neighborhood targets
    b. Airport Noise levels
        i. Noise levels were based on inbound/outbound flights from Pearson Intl Airport
        ii. Map was extracted from https://cdn.torontopearson.com/-/media/project/pearson/content/community/get-involved/community-conversations/quieter-operations/six-ideas-stakeholder-roundtable.pdf?modified=20190221005347&la=en
        iii. Flight paths on the map were examined and a longitude was determined to help eliminate neighborhoods from the flight paths of aircraft

c. Neighborhood Amenities
    i. Neighborhood amenities were extracted from https://www.neighbourhoodguide.com/toronto/scarborough/...
    ii. Amenities were concolidated and summarized into 4 comparable sections
        1. Basic Overview
        2. Nightlife/Entertainment
        3. City Transit
        4. Parks and Family activities

3. Methodology
    a. Web scraped the listing of neighborhoods sorted by postal code from the following website: https://en.wikipedia.org/w/index.php?title=List_of_postal_codes_of_Canada:_M&oldid=890001695 and read the contents into a Pandas df

Webscrape for neighborhoods from Wiki entry

Using pandas to scrape website with details of postal codes in the toronto area, and create a dataframe of the table contents

In [4]:
```
# define url of Toronto Wiki entry
url = 'https://en.wikipedia.org/w/index.php?title=List_of_postal_codes_of_Canada:_M&oldid=890001695'

# Read the table from the url into a pandas dataframe
table = pd.read_html(url)
df = table[0]
```

    b. Removed all the “Not Assigned” neighborhoods, and eliminated one specific row that was titled “Business Reply mail…”

freq 8 11 18

In [7]:
```
# remove "Not Assigned" values, remove on PO Box for business related mail processing
final_df = df[df.Borough != "Not assigned"]
final_df = final_df.set_index('Neighbourhood')
final_df = final_df.drop('Business Reply Mail Processing Centre 969 Eastern')
final_df = final_df.reset_index()
final_df.head(10)
```

Out[7]:

	Neighbourhood	Postcode	Borough
0	Parkwoods	M3A	North York
1	Victoria Village	M4A	North York
2	Harbourfront	M5A	Downtown Toronto
3	Regent Park	M5A	Downtown Toronto
4	Lawrence Heights	M6A	North York
5	Lawrence Manor	M6A	North York
6	Not assigned	M7A	Queen's Park
7	Islington Avenue	M9A	Etobicoke
8	Rouge	M1B	Scarborough
9	Malvern	M1B	Scarborough

    c. Checked shape to determine number of unique neighborhoods in df

In [8]:
```
# review final dataframe
final_df.describe()
```

Out[8]:

	Neighbourhood	Postcode	Borough
count	210	210	210
unique	208	102	11
top	Runnymede	M8Y	Etobicoke
freq	2	8	45

In [9]:
```
final_df.shape
```

Out[9]: (210, 3)

d. Using the geocoder library, we then listed each postal codes lat/long coordinates in order to map and show geospatial location. We dropped the postal code column and listed the neighborhoods in the first column.

```
In [54]: toronto_df = result
toronto_df=toronto_df.drop('Postcode', axis=1)
toronto_df=toronto_df.set_index('Neighbourhood')
toronto_df=toronto_df.reset_index()
toronto_df
```

Out[54]:

	Neighbourhood	Lat	Lng	Borough
0	Rouge	43.806686	-79.194353	Scarborough
1	Malvern	43.806686	-79.194353	Scarborough
2	Highland Creek	43.784535	-79.160497	Scarborough
3	Rouge Hill	43.784535	-79.160497	Scarborough
4	Port Union	43.784535	-79.160497	Scarborough
5	Guildwood	43.763573	-79.188711	Scarborough
6	Morningside	43.763573	-79.188711	Scarborough
7	West Hill	43.763573	-79.188711	Scarborough
8	Woburn	43.770992	-79.216917	Scarborough
9	Cedarbrae	43.773136	-79.239476	Scarborough
10	Scarborough Village	43.744734	-79.239476	Scarborough
11	East Birchmount Park	43.727929	-79.262029	Scarborough
12	Ionview	43.727929	-79.262029	Scarborough
13	Kennedy Park	43.727929	-79.262029	Scarborough
14	Clairlea	43.711112	-79.284577	Scarborough
15	Golden Mile	43.711112	-79.284577	Scarborough

e. We then used the folium library to render a map of Toronto with all neighborhoods listed using circle markers

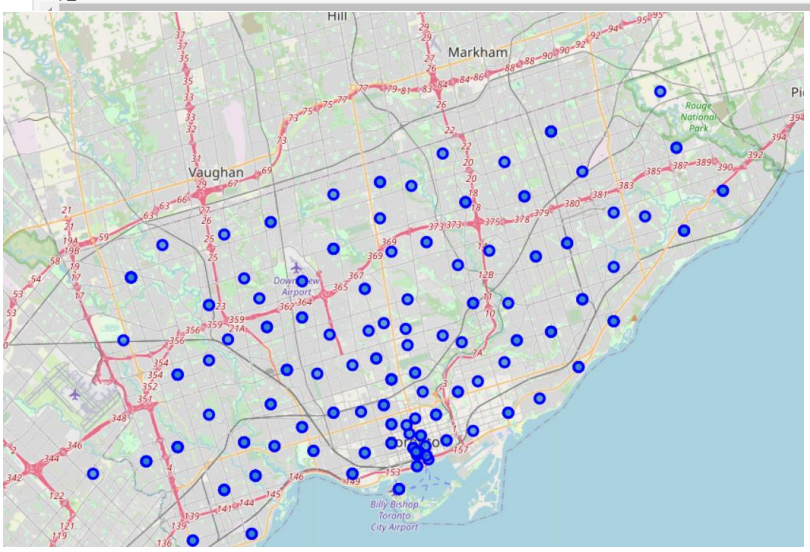
```
5]: #center map around lat/long of Toronto
address = 'Toronto, CA'

# initiate a geolocator
geolocator = Nominatim(user_agent="tor_explorer")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude

# create map of Toronto using latitude and longitude values
map_tor = folium.Map(location=[latitude, longitude])

# add markers to map based on neighborhood data
for lat, lng, borough, neighborhood in zip(toronto_df['Lat'], toronto_df['Lng'], toronto_df['Borough'], toronto_df['Neighbourhood']
    label = '{}, {}'.format(neighborhood, borough)
    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_tor)

map_tor
```



f. We reviewed the airport noise levels and determined that the no neighborhoods wst of Woodbine Heights would satisfy the requirements given, and we dropped any neighborhoods with a longitude to the West of -79.194353, and discovered 44 neighborhoods in the data set

Based on airport noise survey, we decided to eliminate all neighborhoods west of longitude -79.318389

7]:
```
# dropping any rows with longitude < -79.318389
totonto_df = toronto_df.drop(toronto_df[toronto_df['Lng'] < -79.318390].index, inplace = True)

toronto_df.shape
```

7]: (44, 4)

g. We then mapped the new data set

Show map of new neighborhood set

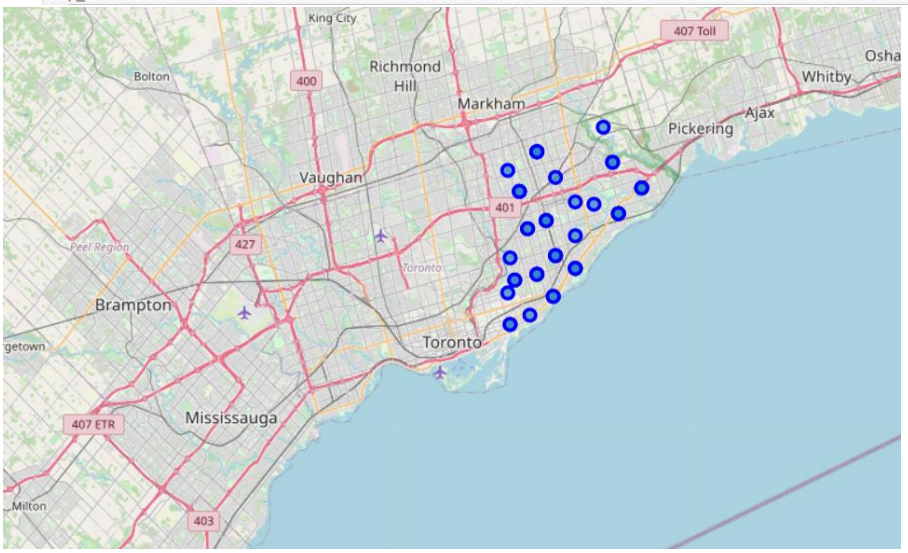
59]:
```
#center map around lat/long of Toronto
address = 'Toronto, CA'

# initiate a geolocator
geolocator = Nominatim(user_agent="tor_explorer")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude

# create map of Toronto using latitude and longitude values
map_tor = folium.Map(location=[latitude, longitude])

# add markers to map based on neighborhood data
for lat, lng, borough, neighborhood in zip(toronto_df['Lat'], toronto_df['Lng'], toronto_df['Borough'], toronto_df['Neighbourhood']):
    label = '{}, {}'.format(neighborhood, borough)
    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_tor)

map_tor
```



h. We then used the FourSquare Developer API to list all venues in these 44 neighborhoods to begin the comparison to the requirements

```
#define a function to create a "venues_list" dataframe for all neighborhoods within 100 meters of the center of the neighborhood

def getNearbyVenues(names, latitudes, longitudes, radius=700):

    venues_list=[]
    for name, lat, lng in zip(names, latitudes, longitudes):

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

```
2]: # run the function on the toronto neighborhood locations, storing the venue information in the "toronto_venues" dataframe

toronto_venues = getNearbyVenues(names=toronto_df['Neighbourhood'],
                                 latitudes=toronto_df['Lat'],
                                 longitudes=toronto_df['Lng']
                                )
```

i. We then grouped the venues by neighborhood

```
]: # see how many venues were returned in each neighborhood
toronto_venues.groupby('Neighborhood').count()
```

]:

Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
Agincourt	9	9	9	9	9	9
Agincourt North	17	17	17	17	17	17
Birch Cliff	8	8	8	8	8	8
Cedarbrae	15	15	15	15	15	15
Clairlea	14	14	14	14	14	14
Clarks Corners	22	22	22	22	22	22
Cliffcrest	4	4	4	4	4	4
Cliffside	4	4	4	4	4	4
Cliffside West	8	8	8	8	8	8
Dorset Park	13	13	13	13	13	13

j. We then summarized each neighborhood by its venues, and created a new dataframe with each neighborhood and its frequency of each venue type

Summarize each neighborhood based on venue frequency, category

[67]:
```
# one hot encoding
toronto_onehot = pd.get_dummies(toronto_venues[['Venue Category']], prefix="", prefix_sep="")

# add neighborhood column back to dataframe
toronto_onehot['Neighborhood'] = toronto_venues['Neighborhood']

toronto_onehot.head()
```

t[67]:

	Asian Restaurant	Athletics & Sports	Auto Garage	BBQ Joint	Badminton Court	Bagel Shop	Bakery	Bank	Bar	Baseball Field	Beer Store	Board Shop	Breakfast Spot	Brewery	Burger Joint	Burrito Place	Bus Line	Bus Station
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Group each category by neighborhood, and the frequency of each category in each neighborhood

[68]:
```
toronto_grouped = toronto_onehot.groupby('Neighborhood').mean().reset_index()
toronto_grouped.head()
```

[68]:

	Neighborhood	Asian Restaurant	Athletics & Sports	Auto Garage	BBQ Joint	Badminton Court	Bagel Shop	Bakery	Bank	Bar	Baseball Field	Beer Store	Board Shop	Breakfast Spot	Brewery	Burger Joint
0	Agincourt	0.0	0.000000	0.0	0.000000	0.111111	0.0	0.000000	0.000000	0.0	0.0	0.0	0.0	0.111111	0.0	0.0
1	Agincourt North	0.0	0.000000	0.0	0.058824	0.000000	0.0	0.000000	0.000000	0.0	0.0	0.0	0.0	0.000000	0.0	0.0
2	Birch Cliff	0.0	0.000000	0.0	0.000000	0.000000	0.0	0.000000	0.000000	0.0	0.0	0.0	0.0	0.000000	0.0	0.0
3	Cedarbrae	0.0	0.066667	0.0	0.000000	0.000000	0.0	0.066667	0.066667	0.0	0.0	0.0	0.0	0.000000	0.0	0.0
4	Clairlea	0.0	0.000000	0.0	0.000000	0.000000	0.0	0.142857	0.000000	0.0	0.0	0.0	0.0	0.000000	0.0	0.0

k. We then created a new dataframe with each neighborhood showing its top 7 most common venues

Create a new dataframe with each neighborhood listed with their top 7 venues

]:
```
# Define a new function to find the top venues
def return_most_common_venues(row, num_top_venues):
    row_categories = row.iloc[1:]
    row_categories_sorted = row_categories.sort_values(ascending=False)

    return row_categories_sorted.index.values[0:num_top_venues]
```

]:
```
# execute the function and create a new dataframe with the neighborhoods listed with their top 5 venue categories
num_top_venues = 7

indicators = ['st', 'nd', 'rd']

# create columns according to number of top venues
columns = ['Neighborhood']
for ind in np.arange(num_top_venues):
    try:
        columns.append('{}{} Most Common Venue'.format(ind+1, indicators[ind]))
    except:
        columns.append('{}th Most Common Venue'.format(ind+1))

# create a new dataframe
tor_venues = pd.DataFrame(columns=columns)
tor_venues['Neighborhood'] = toronto_grouped['Neighborhood']

for ind in np.arange(toronto_grouped.shape[0]):
    tor_venues.iloc[ind, 1:] = return_most_common_venues(toronto_grouped.iloc[ind, :], num_top_venues)

tor_venues
```

]:

	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
0	Agincourt	Sandwich Place	Lounge	Skating Rink	Pool Hall	Badminton Court	Newsagent	Latin American Restaurant
1	Agincourt North	Intersection	Shopping Plaza	Playground	Pharmacy	Park	Food Court	Noodle House
2	Birch Cliff	Park	Diner	College Stadium	Thai Restaurant	Café	General Entertainment	Skating Rink
3	Cedarbrae	Indian Restaurant	Playground	Athletics & Sports	Caribbean Restaurant	Chinese Restaurant	Coffee Shop	Gym / Fitness Center

l. We then did manual sorts within the dataframe to find the neighborhoods with the most common venues that matched the listed requirements, and found that the neighborhoods at Highland Creek, Port Union, and Rouge Hill had the best fit. We then created a final dataframe for visualization purposes to

map the 3 neighborhoods in question.

```
list =  ['Highland Creek', 'Port Union', 'Rouge Hill']
filter_venues = tor_venues[tor_venues['Neighborhood'].isin(list)]
filter_venues
```

	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
13	Highland Creek	Breakfast Spot	Burger Joint	Bar	Mexican Restaurant	Playground	Pizza Place	Pharmacy
25	Port Union	Breakfast Spot	Burger Joint	Bar	Mexican Restaurant	Playground	Pizza Place	Pharmacy
27	Rouge Hill	Breakfast Spot	Burger Joint	Bar	Mexican Restaurant	Playground	Pizza Place	Pharmacy

```
new_map = toronto_df[toronto_df['Neighbourhood'].isin(list)]
new_map
```

	Neighbourhood	Lat	Lng	Borough
2	Highland Creek	43.784535	-79.160497	Scarborough
3	Rouge Hill	43.784535	-79.160497	Scarborough
4	Port Union	43.784535	-79.160497	Scarborough

m. Created a map of the neighborhoods

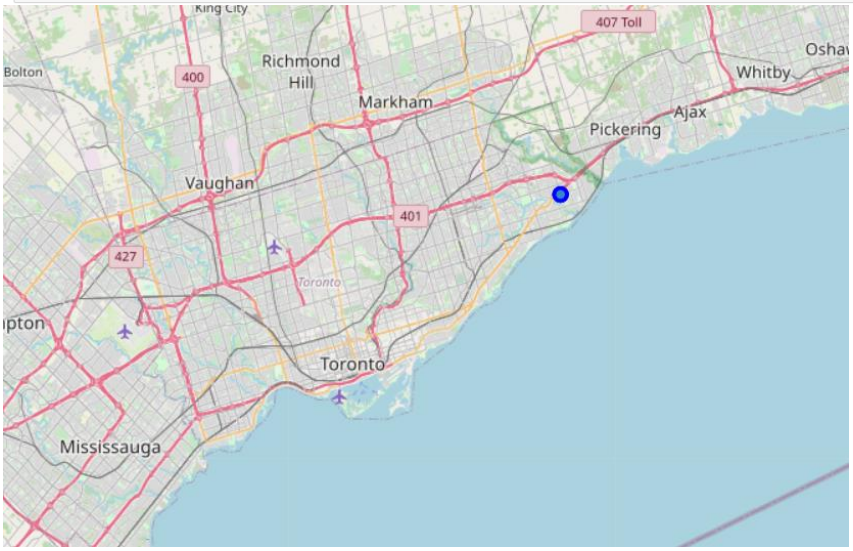
```
#center map around lat/long of Toronto
address = 'Toronto, CA'

# initiate a geolocator
geolocator = Nominatim(user_agent="tor_explorer")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude

# create map of Toronto using latitude and longitude values
final_map = folium.Map(location=[latitude, longitude])

# add markers to map based on neighborhood data
for lat, lng, borough, neighborhood in zip(new_map['Lat'], new_map['Lng'], new_map['Borough'], new_map['Neighbourhood']):
    label = '{}, {}'.format(neighborhood, borough)
    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(final_map)

final_map
```



n. We then used the website https://www.neighbourhoodguide.com/toronto/scarborough/... To read about each neighborhood and its amenities, history, geography, real estate, etc.

4. Results:
   a. Each neighborhood was broken down into 4 categories to help inform the stakeholders of the results
   b. Each neighborhood was then ranked based on each requirement, with Highland Creek being the most well rounded neighborhood, Port Union holding the most entertainment value, and Rouge hill holding the highest Family value

5. Discussion will be based on the results factors and how we can further investigate the locations for the stakeholders to make a well informed decision.
6. Conclusion draws one last breath for the stakeholders to ask questions and reset the analysis for further research if needed.