Coursera Data Science Capstone Project :

Priyank Shah

1 Introduction

This project is to analyze the districts in Houston and find one for a young couple in their 20's. They enjoy going to wine bars and yoga studios and would prefer a location where these are in plenty. The objective of this project is to analyze the districts in Houston and find one that fits this couples needs. This is a great project to use for the Foursquare API

2 Data

There are multiple datasets that are needed to analyze this data. These datasets will be used to analyze the different districts and then select one based on the couples needs.

2.1 District Data Set:

This dataset is extracted from the Wikipedia page and we use Beautiful Soup to scrape the webpage.

2.2 Geocoding Data:

This dataset is obtained using the geocoding API that are then used in the dataframe.

2.3 Venue:

This dataset is obtained from the Foursquare API. We get information about venues around the Houston districts.

3 Methodology

3.1 Website scraping:

This involves using Beautiful soup to get data of the Wikipedia page for Houston: 'https://en.wikipedia.org/wiki/List_of_Houston_neighborhoods'

In [4]:
```
table = soup.find('table', class_ = 'wikitable sortable') # Gets the table from the webpage
rows = table.find_all('tr') # Gets the table rows

postcodes = [] # Initializes the raw postcodes list
boroughs = [] # Initializes the raw boroughs list
neighbourhoods = [] # Initializes the raw neighbourhoods list

for row in rows:
    columns = row.find_all('td')
    try :
        if columns[1].text != 'Not assigned':  # To skip if the borough name is 'Not Assigned'

            postcode = columns[0].text
            postcodes.append(postcode)

            borough = columns[1].text
            boroughs.append(borough)

            neighbourhood = columns[2].text.split('\n')[0] # Removing the newline character at the end

            if neighbourhood == 'Not assigned': # Assigning the same name to neighbourhood if it is 'Not Assigned'
                neighbourhood = borough

            neighbourhoods.append(neighbourhood)

    except Exception as e : # To skip the first row which contains column names
        pass

postcode_explored = [] # Initializing the list of explored postcodes
for index_i, postcode_i in enumerate(postcodes) :
    if postcode_i not in postcode_explored :
        nbds = neighbourhoods[index_i]
        for index_f, postcode_f in enumerate(postcodes) :
            if postcode_i == postcode_f and index_i != index_f:
                nbds = nbds + ', ' + neighbourhoods[index_f] # Concatenating the neighbourhood names
        csv_writer.writerow([postcode_i, boroughs[index_i], nbds]) # Writing the rows in the csv file
        postcode_explored.append(postcode_i)
```

3.2 Geocoding:

This involves using the Geocoding API to get the location coordinates for each district

```
API_KEY='65183f4caad7471ba05770b07bb594a1'
import json

latitudes = [] # Initializing the latitude array
longitudes = [] # Initializing the longitude array

for postal_code in postal_codes :
    place_name = postal_code + " Houston, TX" # Formats the place name
    url = 'https://api.opencagedata.com/geocode/v1/json?q={}&key={}'.format(place_name, API_KEY) # Gets the proper url to make t
    obj = json.loads(requests.get(url).text) # Loads the JSON file in the form of a python dictionary

    results = obj['results'] # Extracts the results information out of the JSON file
    lat = results[0]['geometry']['lat'] # Extracts the latitude value
    lng = results[0]['geometry']['lng'] # Extracts the longitude value

    latitudes.append(lat) # Appending to the list of latitudes
    longitudes.append(lng) # Appending to the list of longitudes
```

3.3 Foursquare API:

This API was used to get all the venues in each district

```
explore_df_list = []

for i, nbd_name in enumerate(df['Postcode']):

    try :
        ### Getting the data of neighbourhood
        nbd_name = df.loc[i, 'Postcode']
        nbd_lat = df.loc[i, 'Latitude']
        nbd_lng = df.loc[i, 'Longitude']

        radius = 500 # Setting the radius as 500 metres
        LIMIT = 100 # Getting the top 100 venues

        url = 'https://api.foursquare.com/v2/venues/explore?client_id={} \
        &client_secret={}&ll={},{}&v={}&radius={}&limit={}'\
        .format(CLIENT_ID, CLIENT_SECRET, nbd_lat, nbd_lng, VERSION, radius, LIMIT)

        results = json.loads(requests.get(url).text)
        results = results['response']['groups'][0]['items']

        nearby = json_normalize(results) # Flattens JSON

        # Filtering the columns
        filtered_columns = ['venue.name', 'venue.categories', 'venue.location.lat', 'venue.location.lng']
        nearby = nearby.loc[:, filtered_columns]

        # Renaming the columns
        columns = ['Name', 'Category', 'Latitude', 'Longitude']
        nearby.columns = columns

        # Gets the categories
        nearby['Category'] = nearby.apply(get_category_type, axis=1)

        # Gets the data required
        for i, name in enumerate(nearby['Name']):
            explore_df_list.append([nbd_name, nbd_lat, nbd_lng] + nearby.loc[i, :].values.tolist())

    except Exception as e:
        pass
```

3.4 Clustering:

I have used the K-means clustering algorithm to create clusters of neighborhoods. Since there are so many venues, K-means was the best option

K means clustering

```
In [49]: kclusters = 7
         toronto_grouped_clustering = toronto_grouped.drop('Neighbourhood', 1)
         kmeans = KMeans(n_clusters = kclusters, random_state = 0).fit(toronto_grouped_clustering)
         kmeans.labels_[0:10]
         neighbourhoods_venues_sorted.insert(0, 'Cluster Labels', kmeans.labels_)
```

3.5 Folium:

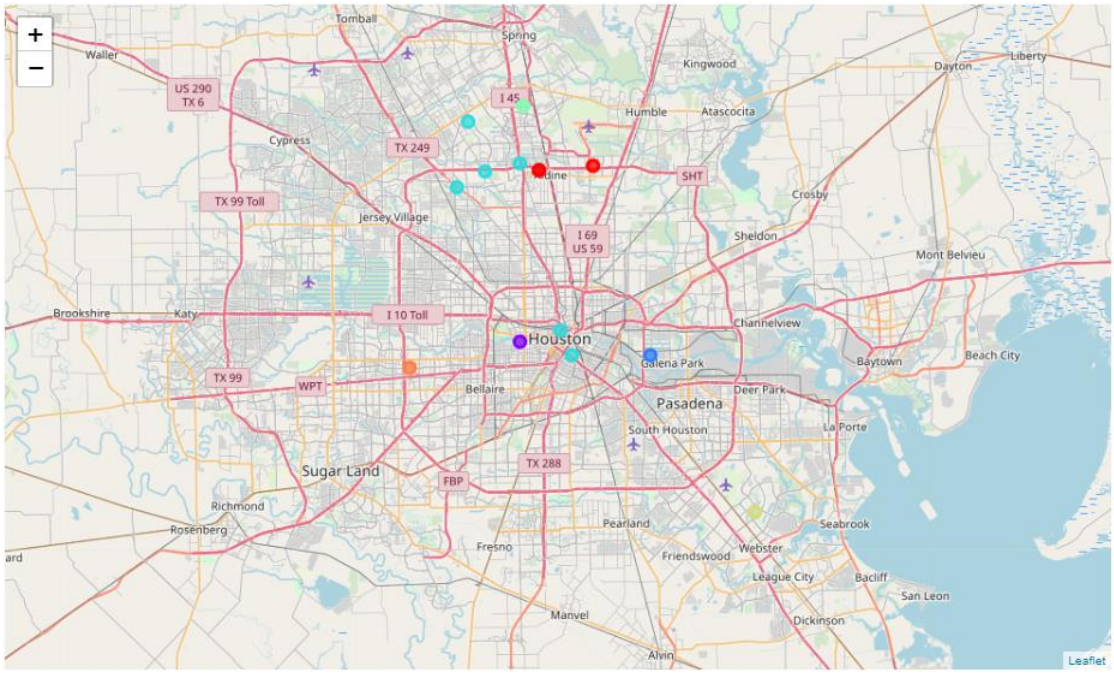
The folium library was used to display data on maps.

In [51]:
```
# Create map
map_clusters = folium.Map(location=[tor_lat, tor_lng], zoom_start=11)

# 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(toronto_merged['Latitude'], toronto_merged['Longitude'], toronto_merged['Postcode'], toronto_m
    label = folium.Popup(str(poi) + ' (Cluster ' + str(cluster) + ')', parse_html=True)
    map_clusters.add_child(
        folium.features.CircleMarker(
        [lat, lon],
        radius=5,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7))

map_clusters
```

4 Results

The districts are divided into clusters using the K means clustering algorithm. They are grouped into clusters based on the venues that they have.

Out[51]:



Each cluster is also examined to see the top venues. E.g. for Cluster ==0 below.

Checking clusters

In [54]:
```
toronto_merged[toronto_merged['Cluster Labels']==0]
```

Out[54]:

	Postcode	Borough	Neighbourhood	Latitude	Longitude	Cluster Labels	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	
8	International Management District	Alief and Little Saigon\r\n	Westpark Tollway to the north, Beltway 8 to the east, Bissonnet Street and Bellfort Street to the south, Texas State Highway 6 to the west	29.941288	-95.327246	0	Hotel	American Restaurant	Breakfast Spot	Gym / Fitness Center	Diner	Intersection	BBQ Joint	
13	Southwest Management District	Sharpstown, Mahatma Gandhi District, portions of Chinatown	Westpark Tollway to the north, Hillcroft Road to the east, Bissonnet Street to the south, Beltway 8 to the west	29.937680	-95.392751	0	Hotel	Hotel Pool	New American Restaurant	Fried Chicken Joint	Gastropub	Hotel Bar	American Restaurant	S S Re
14	Spring Branch Management District	Spring Branch	Tanner Road to the north, Hempstead Highway to the east, Interstate 10 to the south, and Beltway 8 to the west	29.937680	-95.392751	0	Hotel	Hotel Pool	New American Restaurant	Fried Chicken Joint	Gastropub	Hotel Bar	American Restaurant	S S Re

5 Discussion

From the results it is evident that the best cluster for Wine Bars and Yoga studios is cluster 2. This is the Greater East End Management District. The couple should look to move into this area based on their love for wine and yoga.

Out[65]:

	Postcode	Borough	Neighbourhood	Latitude	Longitude	Cluster Labels	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
5	Greater East End Management District	East End, Magnolia Park, and Harrisburg	East of the East Downtown Management District, north of Interstate 45, south of Clinton Drive, and east of Interstate 610	29.741647	-95.256811	2	Wine Bar	Yoga Studio	Fried Chicken Joint	Discount Store	Donut Shop	Fast Food Restaurant	Flower Shop	Food

6 Conclusion

The couple should look to move to the Greater East End Management district based on their love for wine and yoga. This area would fit their needs the best.