The Battle of Neighborhoods
London vs Paris

1. Introduction

Which is better, London or Paris! Well, both cities have their unique aspects, making it hard to point out which city is better than the other in general.

Therefore, when it comes to London vs Paris, it will solely depend on what you are looking for, as well as your values. While some may prefer London, others will opt for Paris.

London and Paris are both diverse and multicultural cities and offer variety of experiences that you could ever imagine. I have tried to group the neighborhoods of London and Paris to draw insights to what you would expect and experience if you happen to be there.

And, with the below provided python program, I hope you can identify the one that best matches your preferences.

2. Business Problem

With the help of this Notebook visitor choose their destination depending on the leisure and entertainment neighborhoods have to offer. This will help tourists make decisions if they are planning to visit London or Paris. I’m sure these findings will help tourist and visitorsa make well informed decisions.

3. Data Description

To have this notebook package compile, We collected geographical location data for both cities. We collected data mainly as below:

- Postal Codes
- Categories
- Neighborhoods
- Boroughs
- Venues and Categories.

London

We scraped our data from **https://en.wikipedia.org/wiki/List_of_areas_of_London.** This wikipedia page has information about all the neighborhoods. Data we selected is Neighborhood (borough), Town (name of borough), and postal code. This wikipedia does not have geographical information so I have used Arcgis API to get geo locations of the neighborhoods i.e. latitude and longitude for London's neighborhoods.

Paris

I have used JSON data that was available at https://www.data.gouv.fr/fr/datasets/r/e88c6fda-1d09-42a0-a069-606d3259114e to get the data for our solution. We only selected data for Paris and selected below noted data columns

- postal_code : Postal codes
- nom_comm : Name of Neighborhood
- nom_dept : Name of the Town
- geo_point_2d : latitude and longitude of each Neighborhood.

Foursquare API

We used Foursquare to get data for different venues in different neighborhoods. Foursquare proides locations data related to venues and events within an area of interest. You can get venue names, locations, menus and location photos. We used foursquare location as the sole data source since all the stated information can be obtained through Foursquare API.

After compiling the list of neighborhoods, we connected through Foursquare API to get information related to venues inside each neighborhood. We limited the radius to be within 500 meters.

We retrieved below data for each venue as follows:

- Neighborhood : Name of the Neighborhood
- Latitude : Latitude of the Neighborhood
- Longitude : Longitude of the Neighborhood
- Venue : Name of the Venue
- Venue Latitude : Latitude of Venue
- Venue Longitude : Longitude of Venue
- Venue Category : Category of Venue

We cluster the neighborhoods based on similar venue categories and then presented observations and findings. Stakeholders should be able to take necessary decision after using this data.

4. Methodology

I have imported below noted Python Packages to create our Notebook model

```
Pandas – To collect and manipulate JSON and HTML data
folium – To Generate Maps
matplotlib – Detailing the Maps
sklearn – To import KMeans
requests – Http Request
html5lib – to parse HTML files
lambda - functions that take other functions as their arguments
arcgis - mapping, geocoding, routing, and spatial analysis
numpy - multi-dimensional array and matrix data structures and mathematical
operations
```

This model provides exploration of each cities individually. Plot the map to show the neighborhoods and build the model by clustering all of the similar neighbourhoods together and finally plot the new map with the clustered neighbourhoods. We draw insights and then compare and discuss our findings.

4.1 Data Collection

In the data collection stage, we begin with collecting the required data for the cities of London and Paris. We need data that has the postal codes, neighborhoods and boroughs specific to each of the cities.

We scraped the List of areas of London wikipedia page and created a second table using the following code:

url_london_uk = "https://en.wikipedia.org/wiki/List_of_areas_of_London"

wiki_london_url = requests.get(url_london_uk)

wiki_london_data = pd.read_html(wiki_london_url.text)

wiki_london_data = wiki_london_data[1]

wiki_london_data

The data looks like this:

	Location	London borough	Post town	Postcode district	Dial code	OS grid ref
0	Abbey Wood	Bexley, Greenwich [7]	LONDON	SE2	020	TQ465785
1	Acton	Ealing, Hammersmith and Fulham[8]	LONDON	W3, W4	020	TQ205805
2	Addington	Croydon[8]	CROYDON	CR0	020	TQ375645
3	Addiscombe	Croydon[8]	CROYDON	CR0	020	TQ345665
4	Albany Park	Bexley	BEXLEY, SIDCUP	DA5, DA14	020	TQ478728
...	...	...	...	...	...	...
526	Woolwich	Greenwich	LONDON	SE18	020	TQ435795

Using JSON I downloaded Paris data from https://www.data.gouv.fr/fr/datasets/r/e88c6fda-1d09-42a0-a069-606d3259114e and loaded data using Pandas after reading the JSON file:

```
!wget -q -O 'france-data.json' https://www.data.gouv.fr/fr/datasets/r/e88c6fda-1d09-42a0-a069-606d3259114e
```

	datasetid	recordid	fields	geometry	record_timestamp
0	correspondances-code-insee-code-postal	2bf36b38314b6c39dfbcd09225f97fa532b1fc45	{'code_comm': '645', 'nom_dept': 'ESSONNE', 's...	{'type': 'Point', 'coordinates': [2.2517129721...	2016-09-21T00:29:06.175+02:00
1	correspondances-code-insee-code-postal	7ee82e74e059b443df18bb79fc5a19b1f05e5a88	{'code_comm': '133', 'nom_dept': 'SEINE-ET-MAR...	{'type': 'Point', 'coordinates': [3.0529405055...	2016-09-21T00:29:06.175+02:00
2	correspondances-code-insee-code-postal	e2cd3186f07286705ed482a10b6aebd9de633c81	{'code_comm': '378', 'nom_dept': 'ESSONNE', 's...	{'type': 'Point', 'coordinates': [2.1971816504...	2016-09-21T00:29:06.175+02:00

4.2 Data Reprocessing and preparation

London: Formatted London data to replace the spaces and underscores in the title. The borough column has numbers within square brackets that we remove using below code:

```
Wiki_london_uk_data.rename(columns=lambda x: x.strip().replace(" ", "_"), inplace=True)
Wiki_london_uk_data
```

Paris: Created a dataframe for each nested fields as below:

```
paris_field_data = pd.DataFrame()
for f in paris_france_raw.fields:
    dict_new = f
    paris_field_data = paris_field_data.append(dict_new, ignore_index=True)
paris_field_data.head()
```

4.3 Feature Selection

We only needed borough, neighborhood, postal codes and geolocations (latitude and longitude) for each datasets.

London:

```
df1_uk = Wiki_london_uk_data.drop( [ Wiki_london_uk_data.columns[0], Wiki_london_uk_data.columns[4], Wiki_london_uk
```

Paris:

```
df_paris_2 = paris_field_data[['postal_code','nom_comm','nom_dept','geo_point_2d']]
df_paris_2
```

4.4 Feature Engineering

Both datasets contains information for all cities in the country. We narrow down data selection and \only selected neighbourhoods that belongs to London and Paris.

London:

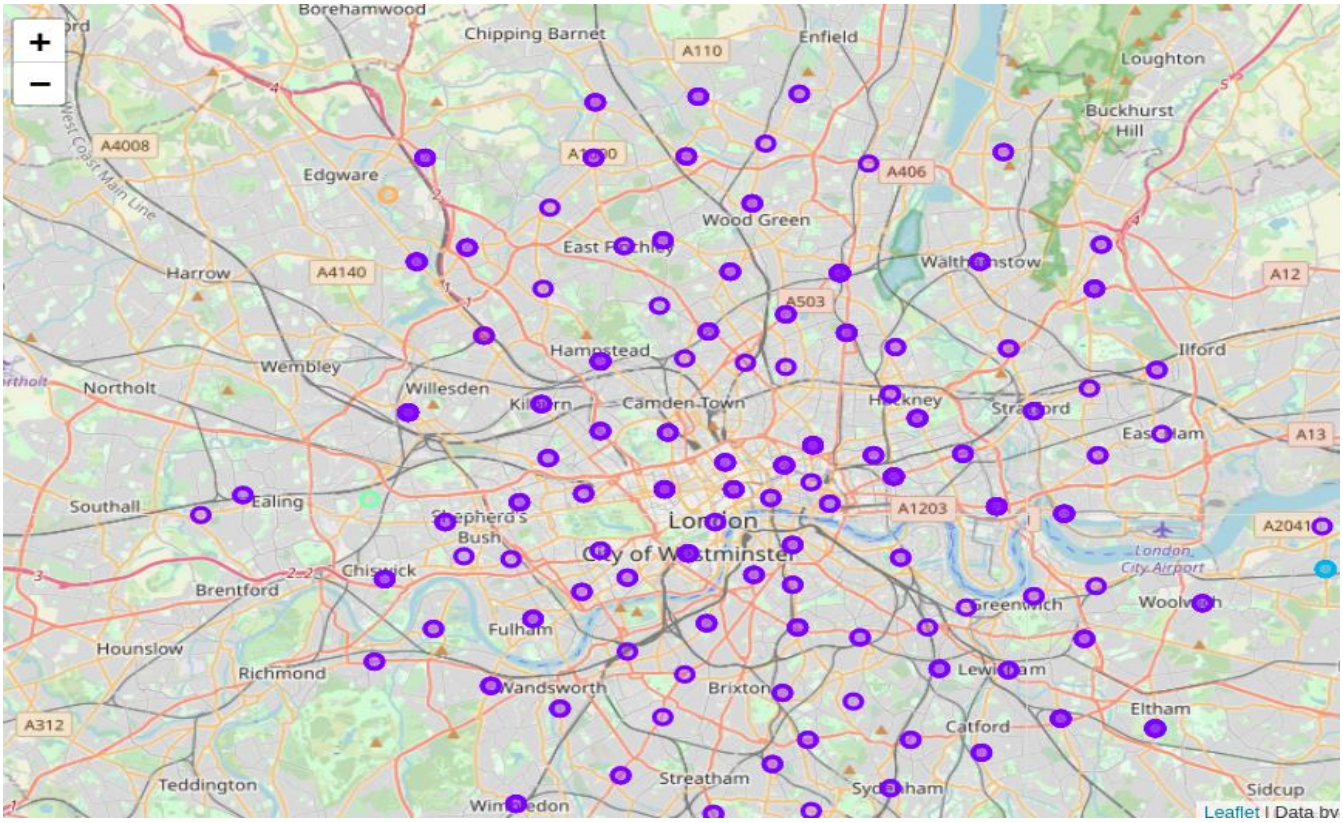
```
df1_uk = df1_uk[df1_uk['town'].str.contains('LONDON')]
df1_uk
```

Paris:

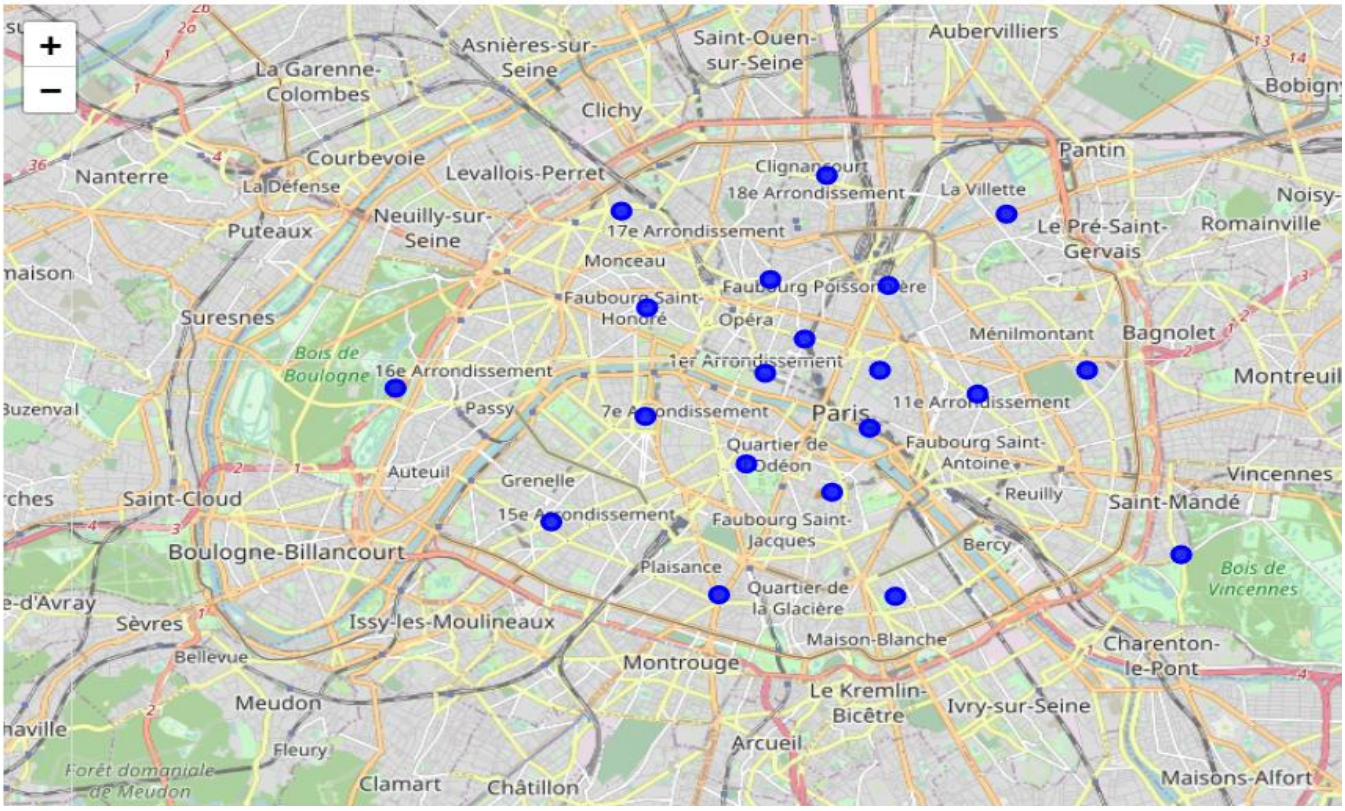
```
df_paris = df_paris_2[df_paris_2['nom_dept'].str.contains('PARIS')].reset_index(drop=True)
df_paris
```

I used `ArcGIS API to` get the latitude and longitude for London neighborhood data and used Passing postal codes to map the geographical co-ordinates. We merged source data with the geographical coordinates for further processing of data.

Map of London:



Neighbourhood map of Paris:



After we visualized each neighborhood, we needed to figure out what are the common venue and venue categories within a 500m radius. We used Foursquare to perform this task, we defined a function which collects information related to each neighborhood including venue and venue categories and limited to 100.

```
LIMIT=100
def getNearbyVenues(names, latitudes, longitudes, radius=500):

    venues_list=[]
    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={}&
            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']['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 = ['Neighbourhood',
                  'Neighbourhood Latitude',
                  'Neighbourhood Longitude',
                  'Venue',
                  'Venue Category']
    return(nearby_venues)
```

Resulting data looks like:

	Neighbourhood	Neighbourhood Latitude	Neighbourhood Longitude	Venue	Venue Category
0	Bexley, Greenwich	51.49245	0.12127	Sainsbury's	Supermarket
1	Bexley, Greenwich	51.49245	0.12127	Lesnes Abbey	Historic Site
2	Bexley, Greenwich	51.49245	0.12127	Lidl	Supermarket
3	Bexley, Greenwich	51.49245	0.12127	Abbey Wood Railway Station (ABW)	Train Station
4	Bexley, Greenwich	51.49245	0.12127	Bean @ Work	Coffee Shop

4.5 One Hot Encoding

I used One Hot Encoding to work with categorical datatype of each venue categories. This helps to convert the categorical data into numeric data. I performed one hot encoding and then calculated the mean of the grouped venue categories for each of the neighborhood.

In [ ]:
```
London_venue_cat = pd.get_dummies(venues_in_London[['Venue Category']], prefix="", prefix_sep="")
London_venue_cat
```

In [ ]:
```
London_venue_cat['Neighbourhood'] = venues_in_London['Neighbourhood']
# moving neighborhood column to the first column
fixed_columns = [London_venue_cat.columns[-1]] + list(London_venue_cat.columns[:-1])
London_venue_cat = London_venue_cat[fixed_columns]

London_venue_cat.head()
```

Venue categories mean value We will group the Neighbourhoods and calculate the mean venue categories value in each Neighbourhood

In [ ]:
```
London_grouped = London_venue_cat.groupby('Neighbourhood').mean().reset_index()
London_grouped.head()
```

Let's make a function to get the top most common venue categories

In [ ]:
```
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]
```

	Neighbourhood	Accessories Store	Adult Boutique	African Restaurant	American Restaurant	Antique Shop	Arcade	Arepa Restaurant	Argentinian Restaurant	Art Gallery	Art Museum	Arts & Crafts Store	Asian Restaurant	Athletics & Sports
0	Barnet	0.0	0.0	0.0	0.001887	0.0	0.0	0.0	0.007547	0.0	0.0	0.0	0.020755	0.0
1	Barnet, Brent, Camden	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.000000	0.0
2	Bexley	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.000000	0.0
3	Bexley, Greenwich	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.000000	0.0
4	Bexley, Greenwich	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.000000	0.0

4.6 Top Venues in the Neighbourhoods

We needed to rank and label the top venue categories in our neighborhood.

Define a function to get the top venue categories in the neighborhood.

There are many categories, we will consider top 10 categories to avoid data skew.

```
num_top_venues = 10

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

# create columns according to number of top venues
columns = ['Neighbourhood']
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 for London
```

```
# create a new dataframe for London
neighborhoods_venues_sorted_london = pd.DataFrame(columns=columns)
neighborhoods_venues_sorted_london['Neighbourhood'] = London_grouped['Neighbourhood']

for ind in np.arange(London_grouped.shape[0]):
    neighborhoods_venues_sorted_london.iloc[ind, 1:] = return_most_common_venues(London_grouped.iloc[ind, :], num_t

neighborhoods_venues_sorted_london.head()
```

4.7 Model Building with KMeans

I used KMeans algorithm to cluster similar neighborhoods together and we have five number of clusterss.

```
# set number of clusters
k_num_clusters = 5

London_grouped_clustering = London_grouped.drop('Neighbourhood', 1)

# run k-means clustering
kmeans_london_uk = KMeans(n_clusters=k_num_clusters, random_state=0).fit(London_grouped_clustering)
kmeans_london_uk
```

We then join London_merged with our neighborhood venues sorted to add latitude & longitude for each of the neighborhood to prepare it for visualization.

```
neighborhoods_venues_sorted_london.insert(0, 'Cluster Labels', kmeans_london_uk.labels_ +1)
```

Join London_merged with our neighbourhood venues sorted to add latitude & longitude for each of the neighborhood to prepare it for plotting

```
london_data = london_uk_merged

london_data = london_data.join(neighborhoods_venues_sorted_london.set_index('Neighbourhood'), on='borough')

london_data.head()
```

Drop all the NaN values to prevent data skew

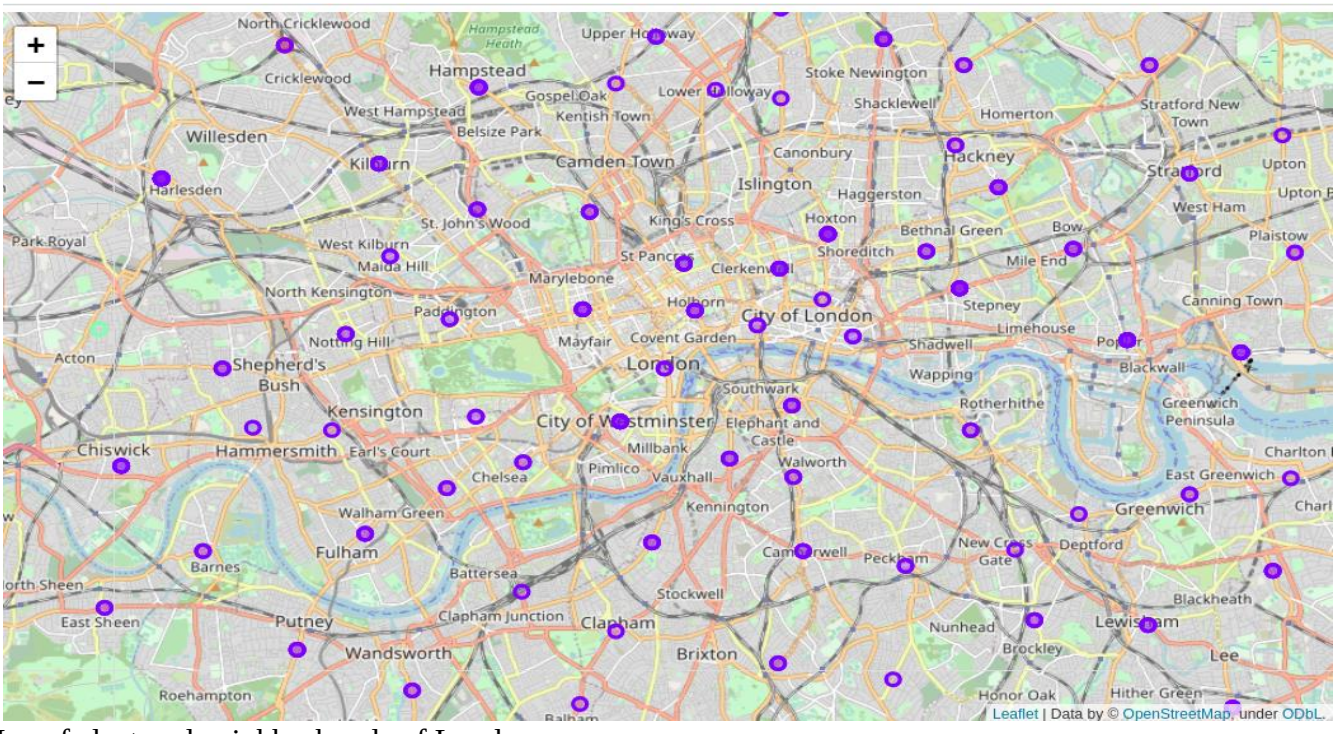
```
london_data_formated = london_data.dropna(subset=['Cluster Labels'])
```

4.8 Visualizing the clustered Neighbourhoods

I used Folium to drop all NaN values to prevent data skew

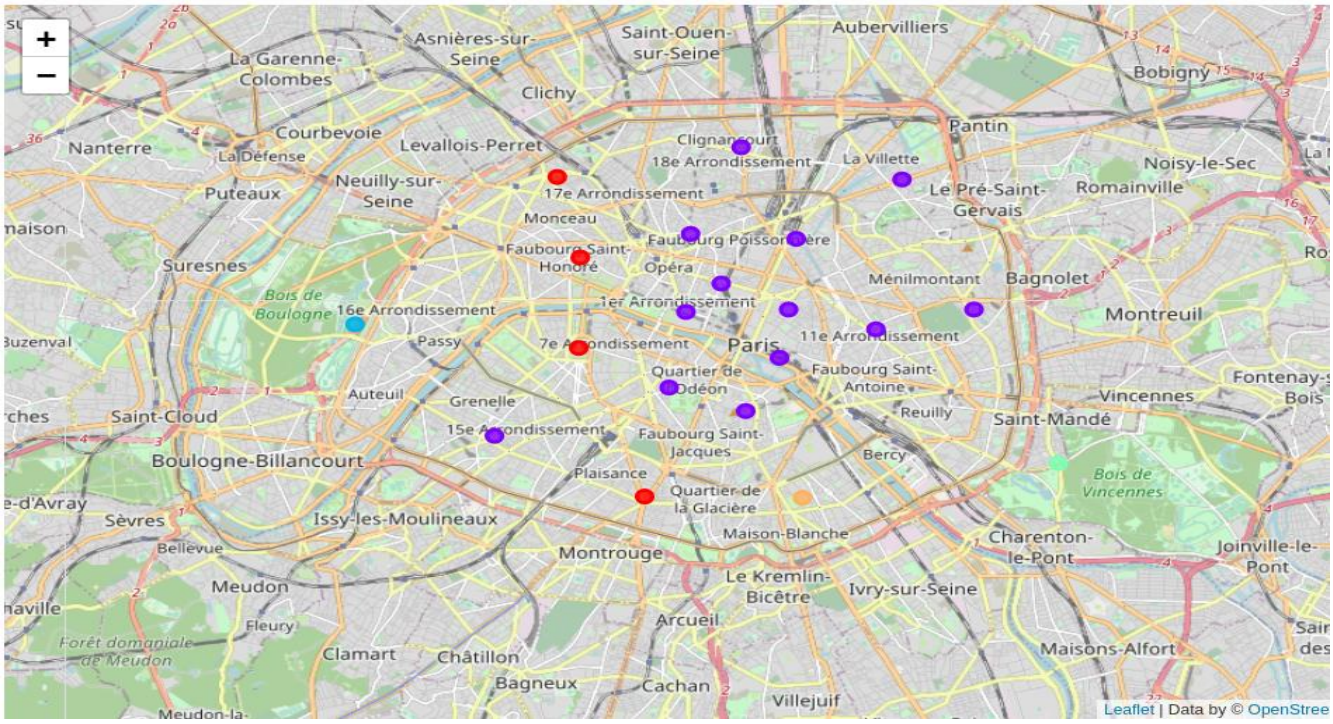
london_data_nonan = london_data.dropna(subset=['Cluster Labels'])

Map of clustered neighbourhoods of London:



Map of clustered neighborhoods of London

Map of Clustered Neighborhoods of Paris



4.9 Examining our Clusters

Cluster 1

```
london_data_formated.loc[london_data_formated['Cluster Labels'] == 1, london_data_formated.columns[[1] + list(range
```

	town	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	9th Most Common Venue	10th Most Common Venue
6	LONDON	1	Italian Restaurant	Coffee Shop	Hotel	Gym / Fitness Center	Pub	Restaurant	Wine Bar	Cocktail Bar	Sandwich Place	Salad Place
7	LONDON	1	Hotel	Coffee Shop	Sandwich Place	Café	Pub	Italian Restaurant	Hotel Bar	Theater	Restaurant	Sushi Restaurant
9	LONDON	1	Supermarket	Convenience Store	Hotel	Grocery Store	Fast Food Restaurant	Park	Bus Stop	Gym / Fitness Center	Golf Course	Gastropub
10	LONDON	1	Coffee Shop	Pub	Food Truck	Café	Park	Vietnamese Restaurant	Italian Restaurant	Hotel	Cocktail Bar	Gym / Fitness Center
12	LONDON	1	Coffee Shop	Pub	Food Truck	Café	Park	Vietnamese Restaurant	Italian Restaurant	Hotel	Cocktail Bar	Gym / Fitness Center
...	...	...	...	...	...	...	...	...	...	...	...	...

I can examine clusters by expanding the code using the `cluster Label` column:

5. Results and Discussion

The neighborhoods of London and Paris are very multicultural. There are lots eat-in places including Chinese, Italian, Indian and Turkish. In London public transport is pretty good and there are lots of bars, coffee shops, Fish and Chips shop and Flea Markets. Paris is little small but there are lots variety of cuisines including French, Thai, Asian, Chinese and Pakistanis. Public transport in Paris is different IE. buses, boats and ferries. There are lots of shopping, Parks, Art galleries and Museums and Historic sites. Overall, Paris seems like a better vacation spot with a mix of possibilities.

6. Conclusion

The purpose of this project was to explore the cities of London and Paris and see how attractive it is to potential tourists and migrants. We explored both the cities based on their postal codes and then extrapolated the common venues present in each of the neighbourhoods finally concluding with clustering similar neighbourhoods together.

We could see that each of the neighbourhoods in both the cities have a wide variety of experiences to offer which is unique in it's own way. The cultural diversity is quite evident which also gives the feeling of a sense of inclusion.

Both Paris and London seem to offer a vacation stay or a romantic getaway with a lot of places to explore, beautiful landscapes, amazing food and a wide variety of culture. Overall, it's upto the stakeholders to decide which experience they would prefer more and which would more to their liking.

Link to my Notebook: https://github.com/shahidmian1/IBM-Data-Science/blob/main/week5-capstone-final.ipynb