Introduction

Istanbul is quite a popular tourist and vacation destination for people all around the world. They are diverse and multicultural and offer a wide variety of experiences that are widely sought after. We try to group the neighborhoods of Istanbul respectively and draw insights into what they look like now.

Problem

The aim is to help tourists choose their destinations depending on the experiences that the neighbourhoods have to offer and what they would want to have. This also helps people make decisions if they are thinking about migrating to Istanbul or even if they want to relocate neighbourhoods within the city. Our findings will help stakeholders make informed decisions and address any concerns they have including the different kinds of cuisines, provision stores and what the city has to offer.

Data Description

Istanbul neighborhoods geolocation data is required. I used arranged data set including City, Town, neighborhood. Information of Lat Long is retrieved from neighborhood name using geopy arcgis geolocator.

Data Sources : http://www.birkenarayazdiklarim.com/index.php/2020/04/30/turkiyenin-il-ilce-semt-mahalle-koy-veritabani/

GeoPy ArcGIS API

Geopy is a Python client for several popular geocoding web services. Geopy makes it easy for Python developers to locate the coordinates of addresses, cities, countries, and landmarks across the globe using third-party geocoders and other data sources.

More specifically, we use Geopy to get the geo locations of the neighbourhoods of Istanbul. The following columns are added to our initial dataset which prepares our data.

1. *latitude* : Latitude for Neighbourhood
2. *longitude* : Longitude for Neighbourhood

Foursquare API Data

We need venues in different neighborhoods of that specific borough. In order to gain that information "Foursquare" is used. Foursquare provides flourish information about venues.Such information includes venue names, locations, menus and even photos. As such, the foursquare location platform will be used as the sole data source since all the stated required information can be obtained through the API.

After finding the list of neighbourhoods, we then connect to the Foursquare API to gather information about venues inside each and every neighbourhood. For each neighbourhood, we have chosen the radius to be 500 meters.

The data retrieved from Foursquare contained information of venues within a specified distance of the longitude and latitude of the postcodes. The information obtained per venue as follows:

1. *Neighbourhood* : Name of the Neighbourhood
2. *Neighbourhood Latitude* : Latitude of the Neighbourhood
3. *Neighbourhood Longitude* : Longitude of the Neighbourhood
4. *Venue* : Name of the Venue
5. *Venue Latitude* : Latitude of Venue
6. *Venue Longitude* : Longitude of Venue
7. *Venue Category* : Category of Venue

Methodology

I used below libraries.

```
import pandas as pd
import requests
import numpy as np
import matplotlib.cm as cm
import matplotlib.colors as colors
import folium

# import k-means for the clustering stage
from sklearn.cluster import KMeans
```

Package breakdown:

- *Pandas* : To collect and manipulate data in JSON and HTMl and then data analysis
- *matplotlib* : Detailing the generated maps
- *folium* : Generating maps of London and Paris
- *sklearn* : To import Kmeans which is the machine learning model that we are using.

The approach taken here is to explore each of the cities individually, plot the map to show the neighbourhoods being considered and then build our 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.

Data Collection

I used following information. I manipulated and cleaned data using Excel.

http://www.birkenarayazdiklarim.com/index.php/2020/04/30/turkiyenin-il-ilce-semt-mahalle-koy-veritabani/

Modified Data which download from URL link.

	A	B	C	D	E
1	IL_ID	ILCE_ID	SEMT_ADI	SEMT_ID	POSTA_KODU
2	Istanbul	Bakırköy	Zeytinlik	1811	34140
3	Istanbul	Bakırköy	Cevizlik	1812	34142
4	Istanbul	Bakırköy	Kartaltepe	1813	34144
5	Istanbul	Bakırköy	Zuhuratbaba	1814	34147
6	Istanbul	Bakırköy	Yeşilköy	1815	34149
7	Istanbul	Bakırköy	Florya	1816	34153
8	Istanbul	Bakırköy	Ataköy	1817	34158
9	Istanbul	Bayrampaşa	Numunebağ	1818	34030
10	Istanbul	Bayrampaşa	Altıntepsi	1819	34035
11	Istanbul	Bayrampaşa	Muratpaşa	1820	34040
12	Istanbul	Bayrampaşa	Yıldırım	1821	34045
13	Istanbul	Beşiktaş	Levent	1822	34330
14	Istanbul	Beşiktaş	Akatlar	1823	34335
15	Istanbul	Beşiktaş	Etiler	1824	34337
16	Istanbul	Beşiktaş	Levazım	1825	34340
17	Istanbul	Beşiktaş	Bebek	1826	34342
18	Istanbul	Beşiktaş	Arnavutköy	1827	34345
19	Istanbul	Beşiktaş	Ortaköy	1828	34347
20	Istanbul	Beşiktaş	Gayrettepe	1829	34349
21	Istanbul	Beşiktaş	Abbasağa	1830	34353

DATA IMPORTING

Data is cleaned and arranged for processing.

In [3]:

```
df_besiktas = pd.read_csv('Istanbul.txt')
df_besiktas
```

Out[3]:

	IL_ID	ILCE_ID	SEMT_ADI	SEMT_ID	POSTA_KODU
0	Istanbul	Bakırköy	Zeytinlik	1811	34140.0
1	Istanbul	Bakırköy	Cevizlik	1812	34142.0
2	Istanbul	Bakırköy	Kartaltepe	1813	34144.0
3	Istanbul	Bakırköy	Zuhuratbaba	1814	34147.0
4	Istanbul	Bakırköy	Yeşilköy	1815	34149.0
5	Istanbul	Bakırköy	Florya	1816	34153.0
6	Istanbul	Bakırköy	Ataköy	1817	34158.0
7	Istanbul	Bayrampaşa	Numunebağ	1818	34030.0
8	Istanbul	Bayrampaşa	Altıntepsi	1819	34035.0
9	Istanbul	Bayrampaşa	Muratpaşa	1820	34040.0

Finding lat long information of neighborhood

I used geopu geocode to find lat long.

In [4]:
```python
from geopy.geocoders import ArcGIS
geolocator = ArcGIS(scheme="https")
```

Finding information Lat, Long for neighborhood

In [5]:
```python
def get_x_y_uk(address1):
    from geopy.geocoders import ArcGIS
    geolocator = ArcGIS(scheme="https")
    lat_coords = 0
    lng_coords = 0
    g=geolocator.geocode(address1)
    lng_coords = g[1][1]
    lat_coords = g[1][0]
    return str(lat_coords) +","+ str(lng_coords)
```

In [6]:
```python
get_x_y_uk(34330)
```

Out[6]: '43.58668000000006,2.74869050500007'

In [13]:
```python
geolocator.geocode('Beşiktaş,Akat Mahallesi')
```

Out[13]: Location(Akat, Beşiktaş, İstanbul, (41.08554000000004, 29.025630000000035, 0.0))

Feature Engineering

Both of our Datasets actually contain information related to all the cities in the country. We can narrow down and further process the data by selecting only the neighbourhoods pertaining to 'ISTANBUL'.

Looking over our Istanbul dataset, we can see that we don't have the geolocation data. We need to extrapolate the missing data for our neighbourhoods. We perform this by leveraging the Geopy Arcgis API. With the Help of ArcGIS API we can get the latitude and longitude of our Istanbul neighbourhood data.

In [24]:
```python
besiktas_merged = pd.concat([df_besiktas,lat_besiktas.astype(float), lng_besiktas.astype(float)], axis=1)
besiktas_merged.columns= ['city','town','borough','borough_ID','post_code','latitude','longitude']
besiktas_merged

```

Out[24]:

	city	town	borough	borough_ID	post_code	latitude	longitude
0	Istanbul	Bakırköy	Zeytinlik	1811	34140.0	40.975380	28.872350
1	Istanbul	Bakırköy	Cevizlik	1812	34142.0	40.976560	28.876940
2	Istanbul	Bakırköy	Kartaltepe	1813	34144.0	40.985710	28.875990
3	Istanbul	Bakırköy	Zuhuratbaba	1814	34147.0	40.987380	28.870390
4	Istanbul	Bakırköy	Yeşilköy	1815	34149.0	40.960130	28.824780
5	Istanbul	Bakırköy	Florya	1816	34153.0	40.972990	28.787930
6	Istanbul	Bakırköy	Ataköy	1817	34158.0	40.991947	28.852414
7	Istanbul	Bayrampaşa	Numunebağ	1818	34030.0	41.032685	28.914072
8	Istanbul	Bayrampaşa	Altıntepsi	1819	34035.0	41.039260	28.902950
9	Istanbul	Bayrampaşa	Muratpaşa	1820	34040.0	41.050470	28.906860
10	Istanbul	Bayrampaşa	Yıldırım	1821	34045.0	41.066380	28.892660
11	Istanbul	Beşiktaş	Levent	1822	34330.0	41.075860	29.017200
12	Istanbul	Beşiktaş	Akatlar	1823	34335.0	41.081300	29.026040
13	Istanbul	Beşiktaş	Etiler	1824	34337.0	41.082470	29.036510
14	Istanbul	Beşiktaş	Levazım	1825	34340.0	41.063990	29.018900
15	Istanbul	Beşiktaş	Bebek	1826	34342.0	41.081110	29.044160
16	Istanbul	Beşiktaş	Arnavutköy	1827	34345.0	41.066140	29.041540
17	Istanbul	Beşiktaş	Ortaköy	1828	34347.0	41.051500	29.027710
18	Istanbul	Beşiktaş	Gayrettepe	1829	34349.0	41.061090	29.007110
19	Istanbul	Beşiktaş	Abbasağa	1830	34353.0	41.047590	29.006210
20	Istanbul	Beşiktaş	Türkali	1831	34357.0	[illegible]	[illegible]

Visualize the Map of Istanbul

In [26]:

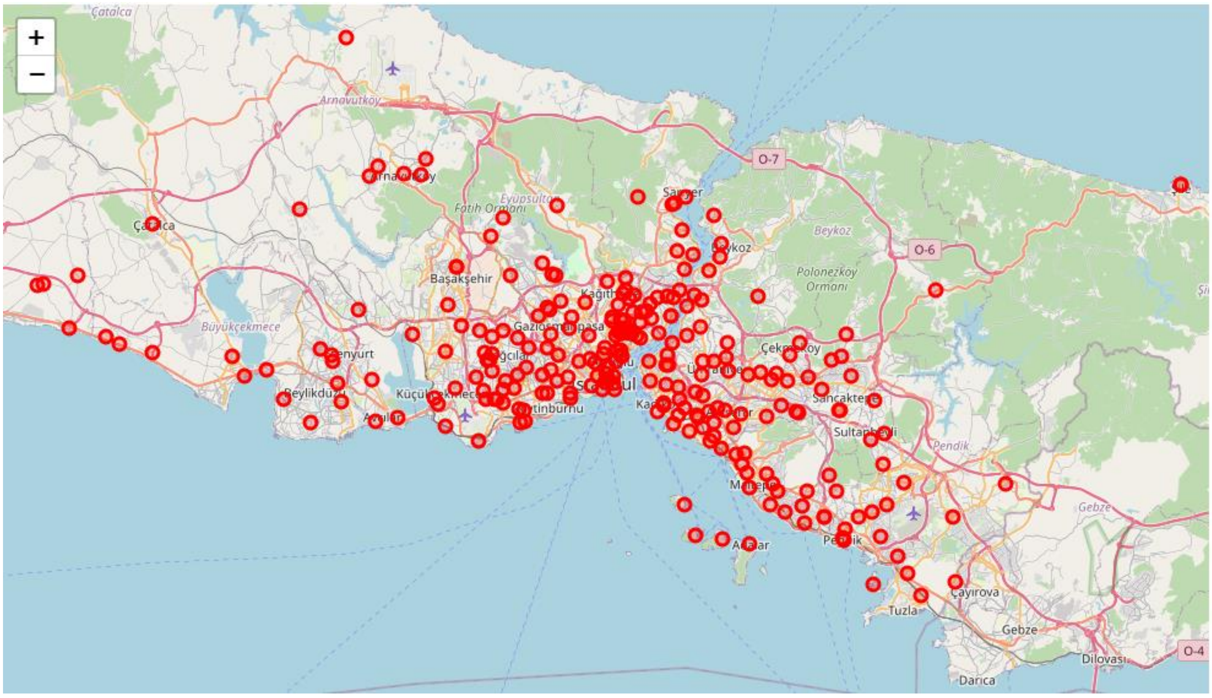
```
Besiktas = geolocator.geocode('Beşiktaş')
Besiktas_lng_coords = Besiktas[1][1]
Besiktas_lat_coords = Besiktas[1][0]

```

In [27]:

```
# Creating the map of London
map_Besiktas = folium.Map(location=[Besiktas_lat_coords, Besiktas_lng_coords], zoom_start=12)
# adding markers to map
for latitude, longitude, borough, town in zip(besiktas_merged['latitude'], besiktas_merged['longitude'], besiktas_merged['bo
    label = '{}, {}'.format(town, borough)
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [latitude, longitude],
        radius=5,
        popup=label,
        color='red',
        fill=True
        ).add_to(map_Besiktas)
map_Besiktas
```

Neighborhood of Istanbul



Now that we have visualized the neighbourhoods, we need to find out what each neighbourhood is like and what are the common venue and venue categories within a 500m radius.

This is where Foursquare comes into play. With the help of Foursquare we define a function which collects information pertaining to each neighbourhood including that of the name of the neighbourhood, geo-coordinates, venue and venue categories.

```
CLIENT_ID = 'NC5E1OAAGOLRE2TRI4MLINGQJVJPFYDQXAOIZGYFJ1EVZNIO'
CLIENT_SECRET = 'MBZ05TETAG31KX4AOCJIKGHMSWDFP0PYP4FQIYCZOXRFQI3Y'
VERSION = '20180605' # Foursquare API version
```

Venues in Istanbul

Venues in Istanbul

To proceed with the next part, we need to define Foursquare API credentials.

Using Foursquare API, we are able to get the venue and venue categories around each neighbourhood in Istanbul.

In [29]:
```python
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={}&limit={}'
            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)
```

Grouping by Venue Categories

Since we are trying to find out what are the different kinds of venue categories present in each neighbourhood and then calculate the top 10 common venues to base our similarity on, we use the One Hot Encoding to work with our categorical datatype of the venue categories. This helps to convert the categorical data into numeric data.

We won't be using label encoding in this situation since label encoding might cause our machine learning model to have a bias or a sort of ranking which we are trying to avoid by using One Hot Encoding.

We perform one hot encoding and then calculate the mean of the grouped venue categories for each of the neighbourhoods.

One Hot Encoding

We need to Encode our venue categories to get a better result for our clustering

In [33]:
```
Besiktas_venue_cat  = pd.get_dummies(venues_in_Besiktas[['Venue Category']], prefix="", prefix_sep="")
Besiktas_venue_cat
```

Out[33]:

	ATM	Accessories Store	Adult Boutique	Advertising Agency	Afghan Restaurant	African Restaurant	Airport	Airport Terminal	American Restaurant	Amphitheater	...	Waterfront	Wedding Hall	Whisky Bar	W
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	
2	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	
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
5	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
6	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
7	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
8	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
9	0	0	0	0	0	0	0	0	0	0	...	0	0	0	

Top Venues in the Neighbourhoods

In our next step, We need to rank and label the top venue categories in our neighbourhood. Let's define a function to get the top venue categories in the neighbourhood. There are many categories, we will consider top 10 categories to avoid data skew. Defining a function to label them accurately

Getting the top venue categories in the neighbourhoods of Istanbul

In [36]:
```
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]
```

There are way too many venue categories, we can take the top 10 to cluster the neighbourhoods.

Creating a function to label the columns of the venue correctly

In [37]:
```
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))
```

Top venue categories

Getting the top venue categories in Istanbul

In [38]:
```
# create a new dataframe for Istanbul
neighborhoods_venues_sorted_besiktas = pd.DataFrame(columns=columns)
neighborhoods_venues_sorted_besiktas['Neighbourhood'] = Besiktas_grouped['Neighbourhood']

for ind in np.arange(Besiktas_grouped.shape[0]):
    neighborhoods_venues_sorted_besiktas.iloc[ind, 1:] = return_most_common_venues(Besiktas_grouped.iloc[ind, :], num_top_ve

neighborhoods_venues_sorted_besiktas.head()
```

Out[38]:

	Neighbourhood	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
0	39587	Café	Coffee Shop	Restaurant	Clothing Store	Gym	Hotel	Boutique	Breakfast Spot	Bar	Yoga Studio
1	Abbasağa	Coffee Shop	Café	Performing Arts Venue	Burger Joint	Park	Bar	Hookah Bar	Lounge	Art Gallery	Historic Site
2	Abdurrahmannafizgürman	Café	Turkish Restaurant	Restaurant	Clothing Store	Gym	Department Store	Furniture / Home Store	Bakery	Soccer Field	Toy / Game Store
3	Acıbadem	Gym	Turkish Restaurant	Café	Coffee Shop	Convenience Store	Restaurant	Bakery	Park	Pharmacy	Manti Place
4	Akatlar	Café	Performing Arts Venue	Grocery Store	Kebab Restaurant	Park	Sporting Goods Shop	Stationery Store	Steakhouse	Fondue Restaurant	Music Venue

Model Building - KMeans

K Means Let's cluster the city of istanbul to roughly 5 to make it easier to analyze.

We use the K Means clustering technique to do so.

In [39]:
```
# set number of clusters
k_num_clusters = 5

Besiktas_grouped_clustering = Besiktas_grouped.drop('Neighbourhood', 1)

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

Out[39]:
```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
    n_clusters=5, n_init=10, n_jobs=None, precompute_distances='auto',
    random_state=0, tol=0.0001, verbose=0)
```

Labelling Clustered Data

Labelling Clustered Data

In [90]:
```
kmeans_Besiktas.labels_
```

Out[90]:
```
array([0, 1, 4, 0, 3, 1, 1, 2, 1, 0])
```

In [40]:
```
neighborhoods_venues_sorted_besiktas.insert(0, 'Cluster Labels', kmeans_Besiktas.labels_ +1)
```

In [41]:
```
Besiktas_data = besiktas_merged

Besiktas_data = Besiktas_data.join(neighborhoods_venues_sorted_besiktas.set_index('Neighbourhood'), on='borough')

Besiktas_data
```

Out[41]:

	city	town	borough	borough_ID	post_code	latitude	longitude	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue
0	Istanbul	Bakırköy	Zeytinlik	1811	34140.0	40.975380	28.872350	3.0	Café	Coffee Shop	Restaurant	Turkish Restaurant
1	Istanbul	Bakırköy	Cevizlik	1812	34142.0	40.976560	28.876940	1.0	Café	Coffee Shop	Seafood Restaurant	Turkish Restaurant
2	Istanbul	Bakırköy	Kartaltepe	1813	34144.0	40.985710	28.875990	3.0	Bakery	Park	Turkish Restaurant	Gym
3	Istanbul	Bakırköy	Zuhuratbaba	1814	34147.0	40.987380	28.870390	3.0	Café	Bakery	Turkish Restaurant	Pizza Place
4	Istanbul	Bakırköy	Yeşilköy	1815	34149.0	40.960130	28.824780	3.0	Seafood Restaurant	Café	Restaurant	Coffee Shop
5	Istanbul	Bakırköy	Florya	1816	34153.0	40.972990	28.787930	1.0	Café	Steakhouse	Hotel	Pool

Visualizing the clustered neighbourhood

Visualizing the clustered neighbourhood

Let's plot the clusters

In [43]:

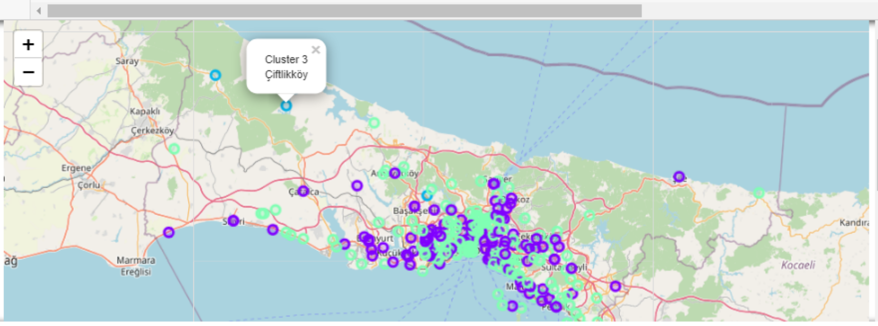
```
map_clusters_Besiktas = folium.Map(location=[Besiktas_lat_coords, Besiktas_lng_coords], zoom_start=12)

# set color scheme for the clusters
x = np.arange(k_num_clusters)
ys = [i + x + (i*x)**2 for i in range(k_num_clusters)]
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(Besiktas_data_nonan['latitude'], Besiktas_data_nonan['longitude'], Besiktas_data_nonan['bo
    label = folium.Popup('Cluster ' + str(int(cluster) +1) + '\n' + str(poi) , parse_html=True)
    folium.CircleMarker(
        [lat, lon],
        radius=5,
        popup=label,
        color=rainbow[int(cluster-1)],
        fill=True,
        fill_color=rainbow[int(cluster-1)]
        ).add_to(map_clusters_Besiktas)

map_clusters_Besiktas
```

Out[43]:



Results and Discussion

The neighbourhoods of Istanbul are very mulitcultural. There are a lot of different cusines including Syrian, Italian, Turkish and Chinese. Istanbul seems to take a step further in this direction by having a lot of Restaurants, bars, juice bars, coffee shops, Fish and Chips shop and Breakfast spots. It has a lot of shopping options too with that of the Flea markets, flower shops, fish markets, Fishing stores, clothing stores. The main modes of transport seem to be Buses and trains. For leisure, the neighbourhoods are set up to have lots of parks, gyms and Historic sites.

Overall, the city of Istanbul offers a multicultural, diverse and certainly an entertaining experience.

Conclusion

The purpose of this project was to explore the cities of Istanbul and see how attractive it is to potential tourists and migrants. We explored Istanbul 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.