REPORT

INTRODUCTION

1. A description of the problem and a discussion of the background

We are a digital start up in the process of launching a new revolutionary app.
Our product = **“Pub Quiz Champions”**

An App that will help you to organize any Pub Quiz like a professional.
Thanks to the apps you can play with your friend in face to face at home or in a pub or you can play virtually from any distance which is quite convenient during this difficult time of COVID lockdown.

But first think first : What is a Pub Quiz ?

Pub quiz
From Wikipedia,

A pub quiz is a quiz held in a pub or bar. These events are also called quiz nights[1], trivia nights[2], or bar trivia[3] and may be held in other settings. Pub quizzes may attract customers to a pub who are not found there on other days. The pub quiz is a modern example of a pub game. Although different pub quizzes can cover a range of formats and topics, they have many features in common. The pub quiz was established in the UK in the 1970s by Burns and Porter and became part of British culture.[4] The Great British Pub Quiz challenge is an annual event.[4] In continental Europe, pub quizzes are a staple event at Irish pubs, where they are usually held in English.

We are now in the last phases of the projects and are preparing our communication plan for the launch of the product.

The customer audience we want to target is :

- Fan of quiz , pub aficionados , pub owners
- Our test market is London UK

Our communication plan will be composed of :

- Organization of live events in different Pub where we will demonstrate the added values of our products
- Digital advertising campaigns (social media + display ads) that will be geo targeted on the areas where we find the most frequented pub.

The problem :

We don't have a huge communication budget so it's not possible for us to target the 3500 London pubs.
To make our communication plan a success:
We need to identify and create a selection of London areas where we will start our communication plan.
We want to identify which areas of London are distinguished by the frequency of pub visits.

DATA

2. **Data where you describe the data that will be used to solve the problem and the source of the data**

We will use the following data :

1. List of London Borough & their GPS coordinates : Longitude - Latitude that i have available in an excel format from a previous project. :
    a. https://github.com/Philreb/coursera_capstone_project/blob/main/london_coordinates2.xlsx

2. Venue data that will be extracted from Foursquare API and will be used for the clustering of the neighbourhood

METHODOLOGY:

1. Visualize the Borough of London
   a. Download and read coordinates data
   b. Use **Folium package** to create a map of London with Borough superimposed on top
2. Explore neighborhood
   a. **Foursquare API**
   b. Define Foursquare credentials and version
   c. Explore the neighbourhood in our dataframe and extract the top 50 venues in a 5000 radius.
   d. Convert the venues list in a new dataframe
   e. Add the venues to the london map
3. Analyse neighborhood
   a. Proceed a one hot encoding
   b. Group by rows
   c. Define top venues
4. Cluster the neighborhood
   a. Create the clusters with with **k-means**
   b. Visualize the clusters by adding them to the map

METHODOLOGY:

1. Visualize the Borough of London
   a. Download and read coordinates data

We use pandas read function to extract data from file : “ london_coordinates2.xlsx”

```
]: # 1 - Visualize the Borough of London
       # Download and read coordinates data
       # Use Folium package to create a map of London with Borough superimposed on top
```

```
2]: import pandas as pd
```

```
]: # Extract the coordinates from excel file
```

```
3]: london = pd.read_excel(r'london_coordinates2.xlsx')
    london.head(30)
```

Here is the *List of London Borough & their GPS coordinates : Longitude - Latitude*

]:

	Borough	Population	latitude	longitude
0	Barking	194352	51.536563	0.075766
1	Barnet	369088	51.656923	-0.194925
2	Bexley	236687	51.439933	0.154327
3	Brent	317264	51.567281	-0.271057
4	Bromley	317899	51.406025	0.013156
5	Camden	229719	51.551706	-0.158826
6	Croydon	372752	51.376165	-0.098234
7	Ealing	342494	51.525026	-0.341500
8	Enfield	320524	51.652299	-0.080712
9	Greenwich	264008	51.482577	-0.007659
10	Hackney	257379	53.156314	-1.575022
11	Hammersmith	178685	51.491188	-0.223731
12	Haringey	263386	51.590611	-0.110971
13	Harrow	243372	51.580559	-0.341995
14	Havering	242080	51.577924	0.212083
15	Hillingdon	286806	51.535183	-0.448138
16	Hounslow	262407	51.460922	-0.373149
17	Islington	215667	51.546506	-0.105806
18	Kensington and Chelsea	155594	51.499080	-0.193825
19	Kingston upon Thames	166793	51.412330	-0.300689
20	Lambeth	314242	51.457148	-0.123068
21	Lewisham	286180	51.441458	-0.011701
22	Merton	203223	50.891854	-4.095316
23	Newham	318227	51.525516	0.035216
24	Redbridge	288272	51.588610	0.082398
25	Richmond upon Thames	191365	51.461311	-0.303742
26	Southwark	298464	51.502781	-0.087738
27	Sutton	195914	51.361428	-0.193961
28	Tower Hamlets	272890	51.520261	-0.029340
29	Waltham Forest	265797	51.588638	-0.011763

b. Use **Folium package** to create a map of London with Borough superimposed on top

We import Folium

```
[ ]: # Use Folium package to create a map of London with Borough superimposed on top
```

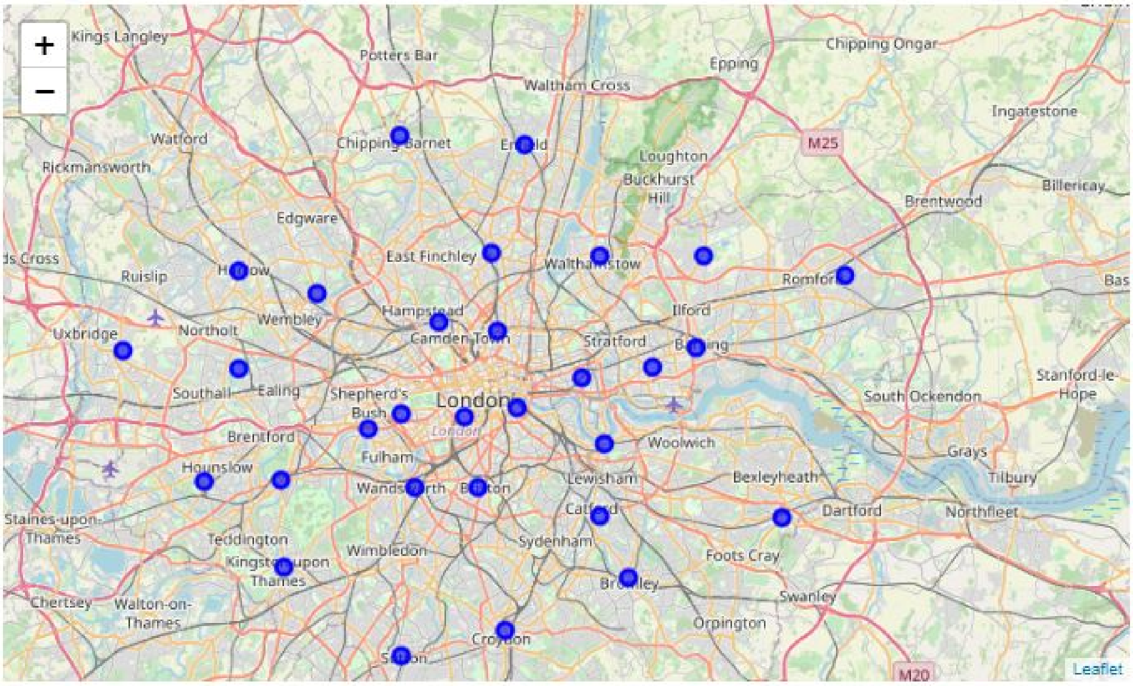
```
[4]: ! pip install folium==0.5.0
     import folium
```

Based on Barking coordinates, we create a map with all Borough

```
5]: latitude = london.loc[0].latitude
    longitude =london.loc[0].longitude
    print('Barking coordinates are' , latitude,longitude)

    Barking coordinates are 51.536563 0.075766
```

```
6]: london_map = folium.Map(location=[latitude, longitude], zoom_start=13)

    # add Borough as blue circle markers

    for lat, lng, label in zip(london.latitude, london.longitude, london.Borough):
        folium.CircleMarker(
            [lat, lng],
            radius=5,
            color='blue',
            popup=label,
            fill = True,
            fill_color='blue',
            fill_opacity=0.6
        ).add_to(london_map)
    london_map
```

6]:



2. Explore neighborhood
    a. **Foursquare API**
    b. Define Foursquare credentials and version

```
# FOURSQUARE API
```

```
# Define Foursquare credentials and version
```

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

```
Your credentails:
CLIENT_ID: [redacted]
CLIENT_SECRET:[redacted]
```

I just removed visibility on client ID & SECRET before sharing document

    c. Explore the neighbourhood in our dataframe and extract the top 50 venues in a 5000 radius.

```
radius = 5000
LIMIT = 50

venues = []

for lat, long, neighborhood in zip(london['latitude'], london['longitude'], london['Borough']):

    # create the API request URL
    url = "https://api.foursquare.com/v2/venues/explore?client_id={}&client_secret={}&v={}&ll={},{}&radius={}&limi
        CLIENT_ID,
        CLIENT_SECRET,
        VERSION,
        lat,
        long,
        radius,
        LIMIT)

    # make the GET request
    results = requests.get(url).json()["response"]['groups'][0]['items']

    # return only relevant information for each nearby venue
    for venue in results:
        venues.append((
            neighborhood,
            lat,
            long,
            venue['venue']['name'],
            venue['venue']['location']['lat'],
            venue['venue']['location']['lng'],
            venue['venue']['categories'][0]['name']))
```

d. Convert the venues list in a new dataframe

```python
# convert the venues list into a new DataFrame

venues_df = pd.DataFrame(venues)

# define the column names
venues_df.columns = ['Neighborhood', 'Latitude', 'Longitude', 'VenueName', 'VenueLatitude', 'VenueLongitude', 'Ver

print(venues_df.shape)
venues_df.head(10)
```

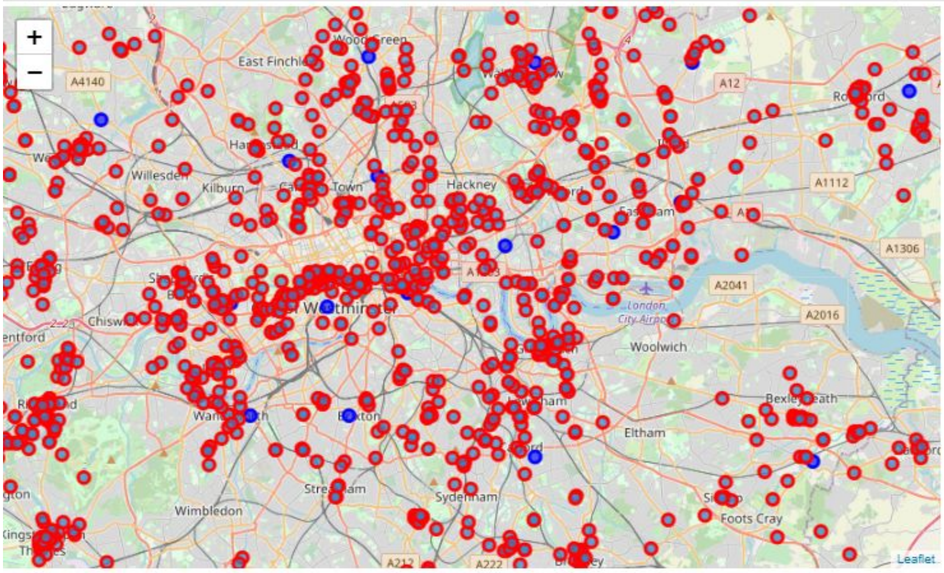
(1533, 7)

	Neighborhood	Latitude	Longitude	VenueName	VenueLatitude	VenueLongitude	VenueCategory
0	Barking	51.536563	0.075766	Barking Abbey	51.535352	0.076054	Park
1	Barking	51.536563	0.075766	Barking Park	51.545217	0.086134	Park
2	Barking	51.536563	0.075766	McDonald's	51.534031	0.053797	Fast Food Restaurant
3	Barking	51.536563	0.075766	The Miller's Well (Wetherspoon)	51.533406	0.056379	Pub
4	Barking	51.536563	0.075766	Capital Karts	51.531792	0.118739	Go Kart Track
5	Barking	51.536563	0.075766	Eastbury Manor House	51.532973	0.099741	History Museum
6	Barking	51.536563	0.075766	Pets at Home	51.520473	0.070494	Pet Store
7	Barking	51.536563	0.075766	Taste Of India	51.542572	0.050107	Indian Restaurant
8	Barking	51.536563	0.075766	Cristina's	51.536523	0.076672	Steakhouse
9	Barking	51.536563	0.075766	The Reach Bar + Kitchen	51.506730	0.073015	Gastropub

e. Add the venues to the london map

```python
# Add the venues to London map
```

```python
# add markers to map

for lat, lng, neighborhood in zip(venues_df['VenueLatitude'], venues_df['VenueLongitude'], venues_df['VenueCategor
    label = '{}'.format(neighborhood)
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=5,
        popup=label,
        color='red',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.7).add_to(london_map)

london_map
```



Map seems heavy, we want to check the number of unique venues:

```
venues_df.VenueCategory.nunique()
```

```
212
```

3. Analyse neighborhood
    a. Proceed a one hot encoding

To prepare data for clustering we will proceed one hot encoding

```
# 3 - Analyze Each Neighborhood
        # one hot encoding
        # grouping
```

```
# one hot encoding
london_onehot = pd.get_dummies(venues_df[['VenueCategory']], prefix="", prefix_sep="")

# add neighborhood column back to dataframe

london_onehot['Neighborhood'] = venues_df['Neighborhood']

# move neighborhood column to the first column

fixed_columns = [london_onehot.columns[-1]] + list(london_onehot.columns[:-1])
london_onehot = london_onehot[fixed_columns]

london_onehot.head()
```

	Neighborhood	American Restaurant	Aquarium	Argentinian Restaurant	Art Gallery	Art Museum	Arts & Crafts Store	Asian Restaurant	Athletics & Sports	Australian Restaurant	...	Waterfront	W
0	Barking	0	0	0	0	0	0	0	0	0	...	0	
1	Barking	0	0	0	0	0	0	0	0	0	...	0	
2	Barking	0	0	0	0	0	0	0	0	0	...	0	
3	Barking	0	0	0	0	0	0	0	0	0	...	0	
4	Barking	0	0	0	0	0	0	0	0	0	...	0	

5 rows × 213 columns

```
london_onehot.shape
```

```
(1533, 213)
```

b. Group by rows

```python
# Next, let's group rows by neighborhood and
# by taking the mean of the frequency of occurrence of each category
```

```python
london_grouped = london_onehot.groupby('Neighborhood').mean().reset_index()
london_grouped
```

	Neighborhood	American Restaurant	Aquarium	Argentinian Restaurant	Art Gallery	Art Museum	Arts & Crafts Store	Asian Restaurant	Athletics & Sports	Australian Restaurant	...	Waterfront
0	Barking	0.00	0.000000	0.00	0.000000	0.00	0.00	0.00	0.00	0.00	...	0.00
1	Barnet	0.02	0.000000	0.00	0.000000	0.00	0.00	0.00	0.00	0.00	...	0.00
2	Bexley	0.02	0.000000	0.00	0.000000	0.00	0.02	0.00	0.00	0.00	...	0.00
3	Brent	0.00	0.000000	0.00	0.000000	0.00	0.00	0.02	0.00	0.00	...	0.00
4	Bromley	0.02	0.000000	0.00	0.000000	0.00	0.00	0.02	0.00	0.00	...	0.00
5	Camden	0.00	0.000000	0.00	0.000000	0.00	0.00	0.00	0.02	0.00	...	0.00
6	Croydon	0.00	0.000000	0.00	0.000000	0.00	0.02	0.00	0.00	0.00	...	0.00
7	Ealing	0.00	0.000000	0.00	0.000000	0.00	0.00	0.00	0.00	0.00	...	0.00
8	Enfield	0.00	0.000000	0.00	0.000000	0.00	0.00	0.00	0.02	0.00	...	0.00
9	Greenwich	0.00	0.000000	0.02	0.000000	0.00	0.00	0.00	0.00	0.02	...	0.00
10	Hackney	0.00	0.035714	0.00	0.035714	0.00	0.00	0.00	0.00	0.00	...	0.00
11	Hammersmith	0.00	0.000000	0.00	0.000000	0.00	0.00	0.00	0.00	0.00	...	0.00
12	Haringey	0.00	0.000000	0.00	0.000000	0.00	0.00	0.00	0.00	0.00	...	0.00
13	Harrow	0.00	0.000000	0.00	0.000000	0.00	0.00	0.00	0.00	0.00	...	0.00
14	Havering	0.00	0.000000	0.00	0.000000	0.00	0.00	0.00	0.00	0.00	...	0.00
15	Hillingdon	0.00	0.000000	0.00	0.000000	0.00	0.00	0.00	0.02	0.00	...	0.00
16	Hounslow	0.00	0.000000	0.00	0.020000	0.00	0.00	0.00	0.00	0.00	...	0.02
17	Islington	0.00	0.000000	0.00	0.000000	0.00	0.00	0.00	0.00	0.00	...	0.00
18	Kensington and Chelsea	0.00	0.000000	0.00	0.020000	0.02	0.00	0.00	0.00	0.00	...	0.00
19	Kingston upon Thames	0.00	0.000000	0.00	0.000000	0.00	0.00	0.02	0.00	0.00	...	0.00
20	Lambeth	0.00	0.000000	0.00	0.020000	0.02	0.00	0.00	0.00	0.00	...	0.00
21	Lewisham	0.00	0.000000	0.02	0.000000	0.00	0.00	0.00	0.00	0.00	...	0.00
22	Merton	0.00	0.000000	0.00	0.000000	0.00	0.00	0.00	0.00	0.00	...	0.00
23	Newham	0.00	0.000000	0.02	0.000000	0.00	0.00	0.02	0.00	0.00	...	0.00
24	Redbridge	0.00	0.000000	0.00	0.000000	0.00	0.00	0.00	0.00	0.00	...	0.00
25	Richmond upon Thames	0.00	0.000000	0.02	0.020000	0.00	0.00	0.00	0.00	0.00	...	0.02
26	Southwark	0.00	0.000000	0.00	0.040000	0.02	0.00	0.00	0.00	0.00	...	0.00
27	Sutton	0.02	0.000000	0.00	0.000000	0.00	0.00	0.00	0.00	0.00	...	0.00
28	Tower Hamlets	0.00	0.000000	0.00	0.020000	0.00	0.00	0.00	0.00	0.00	...	0.00
29	Waltham Forest	0.00	0.000000	0.00	0.040000	0.00	0.00	0.00	0.00	0.00	...	0.00
30	Wandsworth	0.00	0.000000	0.00	0.000000	0.00	0.00	0.02	0.00	0.00	...	0.00
31	Westminster	0.00	0.000000	0.00	0.020000	0.06	0.00	0.02	0.00	0.00	...	0.00

32 rows × 213 columns

```python
# confirm size
```

```python
london_grouped.shape
```

```
(32, 213)
```

c. Define top venues

We now define the top venues expecting to see some “pubs”

```
num_top_venues = 5

for hood in london_grouped['Neighborhood']:
    print("----"+hood+"----")
    temp = london_grouped[london_grouped['Neighborhood'] == hood].T.reset_index()
    temp.columns = ['venue','freq']
    temp = temp.iloc[1:]
    temp['freq'] = temp['freq'].astype(float)
    temp = temp.round({'freq': 2})
    print(temp.sort_values('freq', ascending=False).reset_index(drop=True).head(num_top_venues))
    print('\n')
```

```
----Barking----
              venue  freq
0              Park  0.18
1  Indian Restaurant  0.12
2               Pub  0.08
3              Café  0.08
4             Hotel  0.06


----Barnet----
               venue  freq
0                Pub  0.14
1               Café  0.12
2  Turkish Restaurant  0.08
3         Coffee Shop  0.06
4         Supermarket  0.04


----Bexley----
               venue  freq
0                Pub  0.14
1               Park  0.10
2      Grocery Store  0.08
3  Italian Restaurant  0.06
4           Pharmacy  0.04
```

We are now going to put them in a dataframe and sort by most common venues

6]:
```
#Let's put that into a pandas dataframe
#First, let's write a function to sort the venues in descending order.
```

0]:
```
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]
```

8]:
```
#Now let's create the new dataframe and display the top 10 venues for each neighborhood.
```

1]:
```
import numpy as np
```

2]:
```
num_top_venues = 10

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
neighborhoods_venues_sorted = pd.DataFrame(columns=columns)
neighborhoods_venues_sorted['Neighborhood'] = london_grouped['Neighborhood']

for ind in np.arange(london_grouped.shape[0]):
    neighborhoods_venues_sorted.iloc[ind, 1:] = return_most_common_venues(london_grouped.iloc[ind, :], num_top_ver

neighborhoods_venues_sorted.head(35)
```

2]:

	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	8th Most Common Venue	9th Most Common Venue
0	Barking	Park	Indian Restaurant	Pub	Café	Hotel	Gym / Fitness Center	Restaurant	Steakhouse	Movie Theater
1	Barnet	Pub	Café	Turkish Restaurant	Coffee Shop	Fish & Chips Shop	Supermarket	Park	Breakfast Spot	Sushi Restaurant
2	Bexley	Pub	Park	Grocery Store	Italian Restaurant	Coffee Shop	Pharmacy	Furniture / Home Store	Clothing Store	Café
3	Brent	Indian Restaurant	Pizza Place	Park	Middle Eastern Restaurant	Clothing Store	Supermarket	Grocery Store	Hookah Bar	Food Court
4	Bromley	Pub	Coffee Shop	Gym / Fitness Center	Pizza Place	Park	Grocery Store	Indie Movie Theater	Gastropub	Sandwich Place
5	Camden	Park	Garden	Pub	Coffee Shop	Breakfast Spot	Movie Theater	Market	Zoo Exhibit	Gastropub
6	Croydon	Pub	Park	Café	Coffee Shop	Golf Course	Bookstore	Caribbean Restaurant	Clothing Store	Movie Theater

4. Cluster the neighborhood
    a. Create the clusters with with **k-means**

We will use **k mean** to create the cluster

k-means clustering

From Wikipedia, the free encyclopedia

Not to be confused with K-nearest neighbors algorithm.

***k*-means clustering** is a method of vector quantization, originally from signal processing, that aims to partition *n* observations into *k* clusters in which each observation belongs to the cluster with the nearest mean (cluster centers or cluster centroid), serving as a prototype of the cluster. This results in a partitioning of the data space into Voronoi cells. It is popular for cluster analysis in data mining. *k*-means clustering minimizes within-cluster variances (squared Euclidean distances), but not regular Euclidean distances, which would be the more difficult Weber problem: the mean optimizes squared errors, whereas only the geometric median minimizes Euclidean distances. For instance, better Euclidean solutions can be found using k-medians and k-medoids.

```
# 4. Cluster Neighborhoods
    # Create the clusters with with k-means
    # Visualize the clusters by adding them to the map
```

```
# Create the clusters with with k-means
```

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

```
# set number of clusters
kclusters = 5

london_grouped_clustering = london_grouped.drop('Neighborhood', 1)

# run k-means clustering
kmeans = KMeans(n_clusters=kclusters, random_state=0).fit(london_grouped_clustering)

# check cluster labels generated for each row in the dataframe
kmeans.labels_[0:10]
```

```
array([2, 1, 1, 0, 1, 0, 1, 2, 2, 1], dtype=int32)
```

And we put results in a dataframe

```
# Let's create a new dataframe that includes the cluster
# as well as the top 10 venues for each neighborhood.
```

```
# add clustering labels

neighborhoods_venues_sorted.insert(0,'Cluster Labels',kmeans.labels_)
neighborhoods_venues_sorted.head(35)
```

	Cluster Labels	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	8th Most Common Venue
0	2	Barking	Park	Indian Restaurant	Pub	Café	Hotel	Gym / Fitness Center	Restaurant	Steakhouse
1	1	Barnet	Pub	Café	Turkish Restaurant	Coffee Shop	Fish & Chips Shop	Supermarket	Park	Breakfast Spot
2	1	Bexley	Pub	Park	Grocery Store	Italian Restaurant	Coffee Shop	Pharmacy	Furniture / Home Store	Clothing Store
3	0	Brent	Indian Restaurant	Pizza Place	Park	Middle Eastern Restaurant	Clothing Store	Supermarket	Grocery Store	Hookah Bar
4	1	Bromley	Pub	Coffee Shop	Gym / Fitness Center	Pizza Place	Park	Grocery Store	Indie Movie Theater	Gastropub
5	0	Camden	Park	Garden	Pub	Coffee Shop	Breakfast Spot	Movie Theater	Market	Zoo Exhibit
6	1	Croydon	Pub	Park	Café	Coffee Shop	Golf Course	Bookstore	Caribbean Restaurant	Clothing Store
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	Fish & Chips [illegible]	Scenic [illegible]	[illegible]

b. Visualize the clusters by adding them to the map

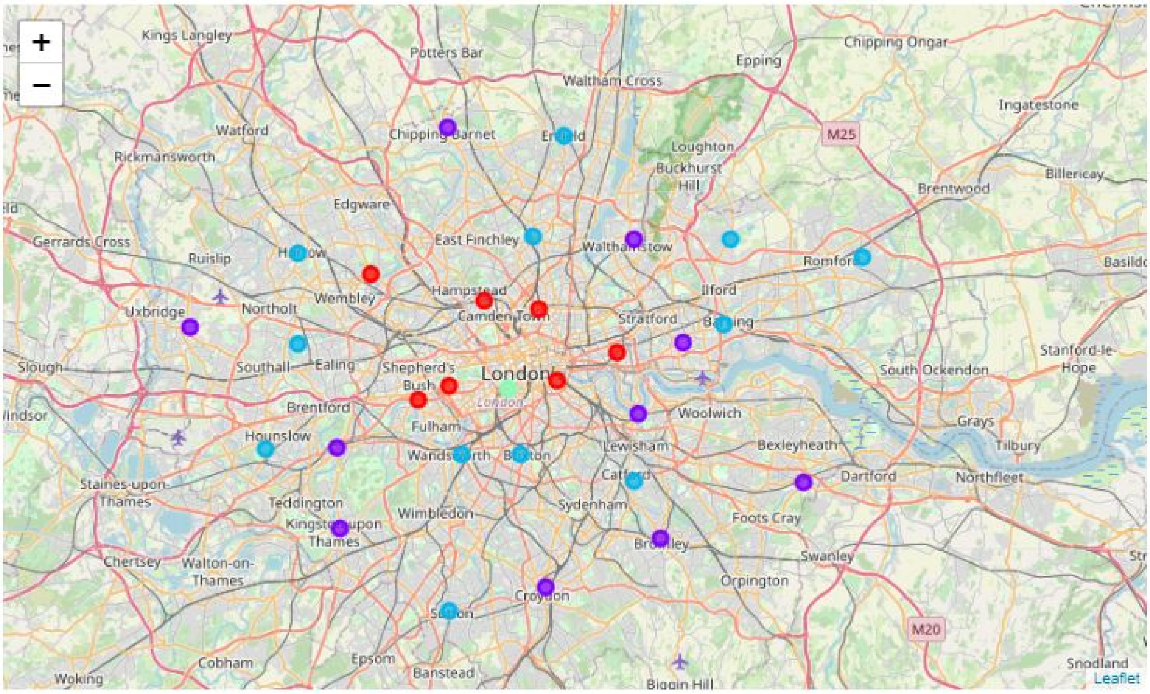
Using folium , we had the cluster to the map :

```
]: # Visualize the clusters by adding them to the map
```

```
]: # Matplotlib and associated plotting modules
import matplotlib.cm as cm
import matplotlib.colors as colors
```

```
]: # create map
map_clusters = folium.Map(location=[latitude, longitude], 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(neighborhoods_venues_sorted['latitude'], neighborhoods_venues_sorted['longitude'
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    folium.CircleMarker(
        [lat, lon],
        radius=5,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7).add_to(map_clusters)

map_clusters
```

]:



RESULTS:

5. Examine cluster & conclusion
    a. Identify the cluster with “pub” as most common venues

⇒ We are now going to examine each cluster and check the venues in the top 3 positions:

```
]: ## 5. Examine Clusters
]: # cluster 0
]: neighborhoods_venues_sorted.loc[neighborhoods_venues_sorted['Cluster Labels'] == 0, neighborhoods_venues_sorted.cc
```

	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	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
3	Brent	Indian Restaurant	Pizza Place	Park	Middle Eastern Restaurant	Clothing Store	Supermarket	Grocery Store	Hookah Bar	Food Court	Movie Theater
5	Camden	Park	Garden	Pub	Coffee Shop	Breakfast Spot	Movie Theater	Market	Zoo Exhibit	Gastropub	Deli / Bodega
11	Hammersmith	Café	Pizza Place	Park	Middle Eastern Restaurant	Coffee Shop	Persian Restaurant	Bubble Tea Shop	Pub	Museum	Gastropub
17	Islington	Cocktail Bar	Theater	Pizza Place	Market	Hotel	Ice Cream Shop	Coffee Shop	Bakery	Canal	Pub
18	Kensington and Chelsea	Department Store	Ice Cream Shop	Park	Hotel	Pizza Place	Dessert Shop	Gym / Fitness Center	Science Museum	Garden	Japanese Restaurant
26	Southwark	Hotel	Scenic Lookout	Grocery Store	Street Food Gathering	Performing Arts Venue	Cocktail Bar	Art Gallery	Gym / Fitness Center	Coffee Shop	Theater
28	Tower Hamlets	Coffee Shop	Pub	Hotel	Beer Bar	Market	Brewery	Cocktail Bar	Pizza Place	Park	Stadium

```python
# cluster 1
```

```python
neighborhoods_venues_sorted.loc[neighborhoods_venues_sorted['Cluster Labels'] == 1, neighborhoods_venues_sorted.c
```

	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	8th Most Common Venue	9th Most Common Venue	10th Mc Comm Ven
1	Barnet	Pub	Café	Turkish Restaurant	Coffee Shop	Fish & Chips Shop	Supermarket	Park	Breakfast Spot	Sushi Restaurant	Restaura
2	Bexley	Pub	Park	Grocery Store	Italian Restaurant	Coffee Shop	Pharmacy	Furniture / Home Store	Clothing Store	Café	Gard
4	Bromley	Pub	Coffee Shop	Gym / Fitness Center	Pizza Place	Park	Grocery Store	Indie Movie Theater	Gastropub	Sandwich Place	Portugue Restaura
6	Croydon	Pub	Park	Café	Coffee Shop	Golf Course	Bookstore	Caribbean Restaurant	Clothing Store	Movie Theater	Food Cou
9	Greenwich	Pub	Coffee Shop	Turkish Restaurant	Garden	Park	Café	Street Food Gathering	Brewery	Historic Site	Playgrou
10	Hackney	Pub	Bar	Hotel	Fast Food Restaurant	Indian Restaurant	Thrift / Vintage Store	Department Store	Resort	Coffee Shop	Fish Chips Sh
15	Hillingdon	Coffee Shop	Pub	Indian Restaurant	Multiplex	Gym / Fitness Center	Restaurant	Park	Supermarket	Middle Eastern Restaurant	Gard Cent
19	Kingston upon Thames	Pub	Coffee Shop	Burger Joint	Café	Japanese Restaurant	Garden	Supermarket	Sushi Restaurant	Gastropub	Pa
23	Newham	Pub	Park	Café	Hotel	Bar	Gym / Fitness Center	Indian Restaurant	Stadium	Fish & Chips Shop	Burg Jo
25	Richmond upon Thames	Pub	Garden	Coffee Shop	Botanical Garden	Café	Hotel	Italian Restaurant	Park	Bakery	Gastrop
29	Waltham Forest	Pub	Coffee Shop	Restaurant	Brewery	Café	Pizza Place	Art Gallery	Park	Farm	Bake

```
#cluster 2
```

```
neighborhoods_venues_sorted.loc[neighborhoods_venues_sorted['Cluster Labels'] == 2, neighborhoods_venues_sorted.co
```

	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	8th Most Common Venue	9th Most Common Venue
0	Barking	Park	Indian Restaurant	Pub	Café	Hotel	Gym / Fitness Center	Restaurant	Steakhouse	Movie Theater
7	Ealing	Pub	Park	Café	Coffee Shop	Pizza Place	Fish & Chips Shop	Scenic Lookout	Hobby Shop	Noodle House
8	Enfield	Coffee Shop	Park	Pub	Turkish Restaurant	Supermarket	Gym / Fitness Center	Garden Center	Grocery Store	Fish & Chips Shop
12	Haringey	Turkish Restaurant	Park	Café	Pub	Coffee Shop	Trail	Bakery	Pizza Place	Organic Grocery
13	Harrow	Indian Restaurant	Park	Coffee Shop	Supermarket	Pub	Gastropub	Portuguese Restaurant	Chocolate Shop	Ice Cream Shop
14	Havering	Coffee Shop	Pub	Park	Café	Italian Restaurant	Supermarket	Department Store	Clothing Store	Grocery Store
16	Hounslow	Park	Pub	Coffee Shop	Clothing Store	Indian Restaurant	Supermarket	Hotel	Gastropub	Rugby Stadium
20	Lambeth	Café	Park	Coffee Shop	Market	Pizza Place	Beer Store	Bakery	Italian Restaurant	Pub
21	Lewisham	Park	Pub	Turkish Restaurant	Coffee Shop	Café	Italian Restaurant	Farmers Market	Pizza Place	Food Truck
24	Redbridge	Park	Pub	Coffee Shop	Gym / Fitness Center	Pizza Place	English Restaurant	Restaurant	Supermarket	Indian Restaurant
27	Sutton	Park	Pub	Supermarket	Italian Restaurant	Coffee Shop	Café	Hotel	Garden Center	Soccer Field
30	Wandsworth	Park	Café	Pub	Pizza Place	Coffee Shop	Mini Golf	Beer Store	Museum	Movie Theater

```
#cluster 3
```

```
neighborhoods_venues_sorted.loc[neighborhoods_venues_sorted['Cluster Labels'] == 3, neighborhoods_venues_sorted.co
```

	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	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue	latitude
31	Westminster	Hotel	Plaza	Boutique	Art Museum	Clothing Store	Indian Restaurant	Park	Lounge	Spa	Fountain	51.497495

```
[ ]: #cluster 5
```

```
36]: neighborhoods_venues_sorted.loc[neighborhoods_venues_sorted['Cluster Labels'] == 5, neighborhoods_venues_sorted.cc
```

36]:

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	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue	latitude	lon

CONCLUSION & DISCUSSION

Our problem was :

We don't have a huge communication budget so it's not possible for us to target the 3500 London pubs.
To make our communication plan a success:
We need to identify and create a selection of London areas where we will start our communication plan.
We want to identify which areas of London are distinguished by the frequency of pub visits.

The **CLUSTER 1** show a clear predominant "pub" in the **1st Most common venues**

```
]: # cluster 1
```

```
]: neighborhoods_venues_sorted.loc[neighborhoods_
```

]:

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue
1	Barnet	Pub	Café	Turkish Restaurant
2	Bexley	Pub	Park	Grocery Store
4	Bromley	Pub	Coffee Shop	Gym / Fitness Center
6	Croydon	Pub	Park	Café
9	Greenwich	Pub	Coffee Shop	Turkish Restaurant
10	Hackney	Pub	Bar	Hotel
15	Hillingdon	Coffee Shop	Pub	Indian Restaurant
19	Kingston upon Thames	Pub	Coffee Shop	Burger Joint
23	Newham	Pub	Park	Café
25	Richmond upon Thames	Pub	Garden	Coffee Shop
29	Waltham Forest	Pub	Coffee Shop	Restaurant

As a conclusion : All the Borough identified in cluster 1 will be the ones where we will start our communication plans.