The Battle of Neighborhoods

by

Iaroslav Mokroguz

March 16, 2020

1. Introduction

1.1 Description of the Problem

Open new business in London is quite challenge. To do so, yours enterprise should have at least one strong market advantage. Capital of Great Britain consist of multicultural environments according to it long imperial history. That fact could be a very strong advantage, as this could make new market slot based on cultural specialization. Another way to increase chance to survive in competition is to choose high profit way of business. One of it is restaurant venture.

But according to everything said before, other competitors will think the same. To fight with them we need something great as data science. We can use Foursquare API and ethnic distribution data to find our market share in London neighborhoods.

1.2 Discussion of the Background

"Ukudla meals" restaurant chain is my client. This company mostly specialized on African cuisine in United States. They are looking to expand operation into Europe through London. They want to create a high-end restaurant that can survive hard competition. Their target is not only Africans, but they are pro-organic and healthy eating. Their slogan: "every meal counts and counts as a royal when you eat".

Since the London demography is so big, my client needs deeper insight from available data in other to decide where to establish the first Europe “palace” restaurant. This company spends a lot on research and provides customers with data insight into the ingredients used at restaurants.

1.3 Interest

Considering the diversity of London, there is a high multicultural sense. London is a place where different shades live. As such, in the search for a high-end African-inclined restaurant, there is a high shortage. The target audience is broad: it ranges from Londoners, tourists and those who are passionate about organic food.

2. Data acquisition and cleaning

2.1 Description of Data

This project will rely on public data from Wikipedia and Foursquare.

2.1.1 Dataset 1

In this project, London will be used as synonymous to the "Greater London Area" in this project. Within the Greater London Area, there are areas that are within the London Area Postcode. The focus of this project will be the neighborhoods are that are within the London Post Code area.

The London Area consists of 32 Boroughs and the "City of London". Our data will be from the link: https://en.wikipedia.org/wiki/List_of_areas_of_London

The web scrapped of the Wikipedia page for the Greater London Area data is provided below:

Out[8]:

	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

London is big and due to the limitations in the number of calls for the Foursquare API, the following assumptions are made to confine this project to only South East London.

Assumption 1: Where the Postcode are more than one, (for example, in Acton, there are 2 postcodes - *W3* and *W4*), the postcodes are spread to multi-rows and assigned the same values from the other columns.

Out[15]:

	Location	Borough	Post-town	Dial-code	OSGridRef	Postcode
0	Abbey Wood	Bexley, Greenwich	LONDON	020	TQ465785	SE2
1	Acton	Ealing, Hammersmith and Fulham	LONDON	020	TQ205805	W3
1	Acton	Ealing, Hammersmith and Fulham	LONDON	020	TQ205805	W4
10	Angel	Islington	LONDON	020	TQ345665	EC1
10	Angel	Islington	LONDON	020	TQ345665	N1

As seen above, there are separate rows for Postcodes - *W3*and *W4*; same goes for the others too.

Assumption 2: From the data, only the *'Location'*, *'Borough'*, *'Postcode'*, *'Post-town'* will be used for this project. So they are extracted into a new data frame.

Out[18]:

	Location	Borough	Postcode	Post-town
0	Abbey Wood	Bexley, Greenwich	SE2	LONDON
1	Acton	Ealing, Hammersmith and Fulham	W3	LONDON
2	Acton	Ealing, Hammersmith and Fulham	W4	LONDON
3	Angel	Islington	EC1	LONDON
4	Angel	Islington	N1	LONDON

Assumption 3: Now, only the *Boroughs* with London *Post-town* will be used for our search of location. Therefore, all the non-post-town are dropped.

Out[24]:

	Location	Borough	Postcode
0	Abbey Wood	Bexley, Greenwich	SE2
1	Acton	Ealing, Hammersmith and Fulham	W3
2	Acton	Ealing, Hammersmith and Fulham	W4
3	Angel	Islington	EC1
4	Angel	Islington	N1
5	Church End	Brent	NW10
6	Church End	Barnet	N3
7	Clapham	Lambeth, Wandsworth	SW4
8	Clerkenwell	Islington	EC1
9	Colindale	Barnet	NW9

From assumption 3, there are now 380 instances, which is a drop from 638 because of the drop of non-London post-towns.

Assumption 4: Due to its more diverse outlook, proximity to Afro-Caribbean markets and accessible facilities, only the South East areas of London will be considered for our analysis. The South East area has postcodes starting with SE.

So, first, we remove the whitespaces at the start of some of the postcodes and then drop the other non-SE postcodes.

Out[30]:

	Location	Borough	Postcode
0	Abbey Wood	Bexley, Greenwich	SE2
1	Crofton Park	Lewisham	SE4
2	Crossness	Bexley	SE2
3	Crystal Palace	Bromley	SE19
4	Crystal Palace	Bromley	SE20
5	Crystal Palace	Bromley	SE26
6	Denmark Hill	Southwark	SE5
7	Deptford	Lewisham	SE8
8	Dulwich	Southwark	SE21
9	East Dulwich	Southwark	SE22

Assumption 5: This assumption will focus on the demography of London where there are predominantly more multicultural groups. According to the proportion of races by London borough as seen in Demography of London, the top 5 Black Africans or Caribbean's are shown below:

Out[38]:

	Local authority	White	Mixed	Asian	Black	Other
22	Lewisham	53.5	7.4	9.3	27.2	2.6
27	Southwark	54.3	6.2	9.4	26.9	3.3
21	Lambeth	57.1	7.6	6.9	25.9	2.4
11	Hackney	54.7	6.4	10.5	23.1	5.3
7	Croydon	55.1	6.6	16.4	20.2	1.8

Assumption 6: Our next assumption will be based on the top 5 areas will significantly high "Black", "Mixed" and other races. These leave us with Lewisham, Southwark, Lambeth, Hackney and Croydon.

Just to be sure with syntax, Hackney is in North London, so it will not be returned.

Out[41]:

	Location	Borough	Postcode
0	Crofton Park	Lewisham	SE4
1	Denmark Hill	Southwark	SE5
2	Deptford	Lewisham	SE8
3	Dulwich	Southwark	SE21
4	East Dulwich	Southwark	SE22

2.1.2 Dataset 2

In obtaining the location data of the locations, the *Geocoder* package is used with the *arcgis_geocoder* to obtain the latitude and longitude of the needed locations.

These will help to create a new data-frame that will be used subsequently for the South East London areas. We will proceed to store the location data - latitude and longitude:

Out[49]:

	Location	Borough	Postcode	Latitude	Longitude
0	Crofton Park	Lewisham	SE4	51.46268	-0.03558
1	Denmark Hill	Southwark	SE5	51.47480	-0.09313
2	Deptford	Lewisham	SE8	51.48114	-0.02467
3	Dulwich	Southwark	SE21	51.44100	-0.08897
4	East Dulwich	Southwark	SE22	51.45256	-0.07076

2.1.3 Dataset 3

The Foursquare API will be used to obtain the South East London Area venues for the geographical location data. These will be used to explore the neighborhoods of London accordingly.

The venues within the neighborhoods of South East London like the areas restaurants and proximity to amenities would be correlated. Also, accessibility and ease of supplies would be considered as it relates to venues.

3. Methodology

3.1 Data Exploration

3.1.1 Single Neighborhood

An initial exploration of a single Neighborhood within the London area was done to examine the Foursquare workability. The Lewisham Borough postcode *SE13* and Location - *Lewisham* is used for this.

Out[55]:

	Location	Borough	Postcode	Latitude	Longitude
0	Crofton Park	Lewisham	SE4	51.46268	-0.03558
1	Denmark Hill	Southwark	SE5	51.47480	-0.09313
2	Deptford	Lewisham	SE8	51.48114	-0.02467
3	Dulwich	Southwark	SE21	51.44100	-0.08897
4	East Dulwich	Southwark	SE22	51.45256	-0.07076
5	Elephant and Castle	Southwark	SE1	51.49960	-0.09613
6	Elephant and Castle	Southwark	SE11	51.49084	-0.11108
7	Elephant and Castle	Southwark	SE17	51.48764	-0.09542
8	Bankside	Southwark	SE1	51.49960	-0.09613
9	Forest Hill	Lewisham	SE23	51.44122	-0.04764
10	Gipsy Hill	Lambeth	SE19	51.41990	-0.08808
11	Gipsy Hill	Lambeth	SE27	51.43407	-0.10375
12	Grove Park	Lewisham	SE12	51.44759	0.01350
13	Herne Hill	Lambeth	SE24	51.45529	-0.09928
14	Hither Green	Lewisham	SE13	51.46196	-0.00754
15	Honor Oak	Lewisham	SE23	51.44122	-0.04764
16	Ladywell	Lewisham	SE4	51.46268	-0.03558
17	Ladywell	Lewisham	SE13	51.46196	-0.00754
18	Lambeth	Lambeth	SE1	51.49960	-0.09613
19	Lee	Lewisham	SE12	51.44759	0.01350
20	Lewisham	Lewisham	SE13	51.46196	-0.00754

Now, let's use the Lewisham with the index location *20*. The latitude and longitude values of Lewisham with postcode SE13, are 51.46196000000003, -0.007539999999949032. Let's explore the top 100 venues that are within a 2000

meters radius of Lewisham. And then, let's create the *GET* request *URL*, and then the *url* is named:

'https://api.foursquare.com/v2/venues/explore?&client_id=0HH2B0MRFB2FALD3CL3SQAGF5KPCVO53DS5OEOKOP4MWUCJO&client_secret=D5KMPZK1RAFC0RSUS3VCUOIAIIA2KVCOWHIP1RJX3D1L0UQS&v=20190528&ll=51.46196000000003,-0.007539999999949032&radius=2000&limit=100'

Then, send the GET request and examine the results:

```
Out[60]: {'meta': {'code': 200, 'requestId': '5e6ed2921d67cb001ba94095'},
          'response': {'suggestedFilters': {'header': 'Tap to show:',
            'filters': [{'name': 'Open now', 'key': 'openNow'}]},
           'headerLocation': 'Lewisham Central',
           'headerFullLocation': 'Lewisham Central, London',
           'headerLocationGranularity': 'neighborhood',
           'totalResults': 185,
           'suggestedBounds': {'ne': {'lat': 51.47996001800005,
             'lng': 0.021296961190459426},
            'sw': {'lat': 51.44395998200002, 'lng': -0.03637696119035749}},
           'groups': [{'type': 'Recommended Places',
             'name': 'recommended',
             'items': [{'reasons': {'count': 0,
                'items': [{'summary': 'This spot is popular',
                  'type': 'general',
                  'reasonName': 'globalInteractionReason'}]},
               'venue': {'id': '535823bc498ec8d8da9aad5f',
                'name': 'Street Feast Model Market',
                'location': {'address': '196 Lewisham High St',
```

From the *results*, the necessary information needs to be obtained from *items* key. To do this, the *get_category_type* function is used from the Foursquare lab. The result is then cleaned up from json to a structured *pandas* data-frame as shown below:

Out[63]:

	name	categories	lat	lng
0	Street Feast Model Market	Street Food Gathering	51.460209	-0.012199
1	Maggie's Kitchen	Café	51.465380	-0.011213
2	Levante restaurant	Restaurant	51.462072	-0.009491
3	Levante Pide Restaurant	Turkish Restaurant	51.459848	-0.011476
4	Corte	Coffee Shop	51.459776	-0.011554
5	Manor House Gardens	Park	51.456686	0.004684
6	Dirty South	Pub	51.458846	-0.002666
7	Côte Brasserie	French Restaurant	51.467378	0.007176
8	Blackheath Farmers' Market	Farmers Market	51.465913	0.007945
9	Gennaro Delicatessan	Deli / Bodega	51.461765	-0.009726
10	The Sausage Man	Food Truck	51.462507	-0.010248

Out[65]:

	Count
Pub	13
Café	8
Gastropub	6
Park	5
Garden	4

Interestingly, even though there are restaurants are the Lewisham area, they are not even in the top 5 venues. It should be noted that since we are limited by data availability, our perspectives will be on what we have.

“100 venues were returned by Foursquare”. So in this case, 100 venues were returned for Lewisham.

3.1.2 Multiple Neighborhoods

Now let's explore (Multiple) Neighborhoods in the South East London area. To do this, the function *getNearbyVenues* is used and it's created to repeat the same process for all neighborhoods.

In [67]:
```
def getNearbyVenues(names, latitudes, longitudes, radius=2000):

    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={}'.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 = ['Neighbourhood',
                  'Neighbourhood Latitude',
                  'Neighbourhood Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']

    return(nearby_venues)
```

The created function - *getNearbyVenues* is then used on each neighborhoods and creates a new data-frame called *london_venues*.

Out[72]:

	Neighbourhood	Neighbourhood Latitude	Neighbourhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Crofton Park	51.46268	-0.03558	The Orchard	51.463678	-0.035699	Gastropub
1	Crofton Park	51.46268	-0.03558	Brockley's Rock	51.459457	-0.033868	Fish & Chips Shop
2	Crofton Park	51.46268	-0.03558	Browns Of Brockley	51.464513	-0.037346	Coffee Shop
3	Crofton Park	51.46268	-0.03558	Waterintobeer	51.463712	-0.038826	Beer Store
4	Crofton Park	51.46268	-0.03558	Saka Maka	51.464826	-0.036437	Indian Restaurant

The number of venues returned for each neighborhoods is then explored as follows:

Out[73]:

Neighbourhood	Neighbourhood Latitude	Neighbourhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
Bankside	100	100	100	100	100	100
Bellingham	70	70	70	70	70	70
Bermondsey	100	100	100	100	100	100
Blackheath	87	87	87	87	87	87
Brixton	100	100	100	100	100	100
Brockley	100	100	100	100	100	100
Camberwell	100	100	100	100	100	100
Catford	70	70	70	70	70	70
Chinbrook	57	57	57	57	57	57
Crofton Park	100	100	100	100	100	100
Denmark Hill	100	100	100	100	100	100
Deptford	100	100	100	100	100	100
Dulwich	100	100	100	100	100	100
East Dulwich	82	82	82	82	82	82
Elephant and Castle	300	300	300	300	300	300
Forest Hill	100	100	100	100	100	100
Gipsy Hill	200	200	200	200	200	200
Grove Park	57	57	57	57	57	57
Herne Hill	100	100	100	100	100	100
Hither Green	100	100	100	100	100	100

The next step is to check how many unique categories can be returned for the venues: “There are 198 unique categories”

Out[76]:

	Count
Pub	437
Coffee Shop	294
Café	267
Park	202
Grocery Store	149

Out[77]:

	Count
count	198.000000
mean	21.474747
std	47.512829
min	1.000000
25%	4.000000
50%	8.000000
75%	18.750000
max	437.000000

3.2 Clustering

For this section, the neighborhoods in South East London will be clustered based on the processed data obtained above.

3.2.1 Libraries

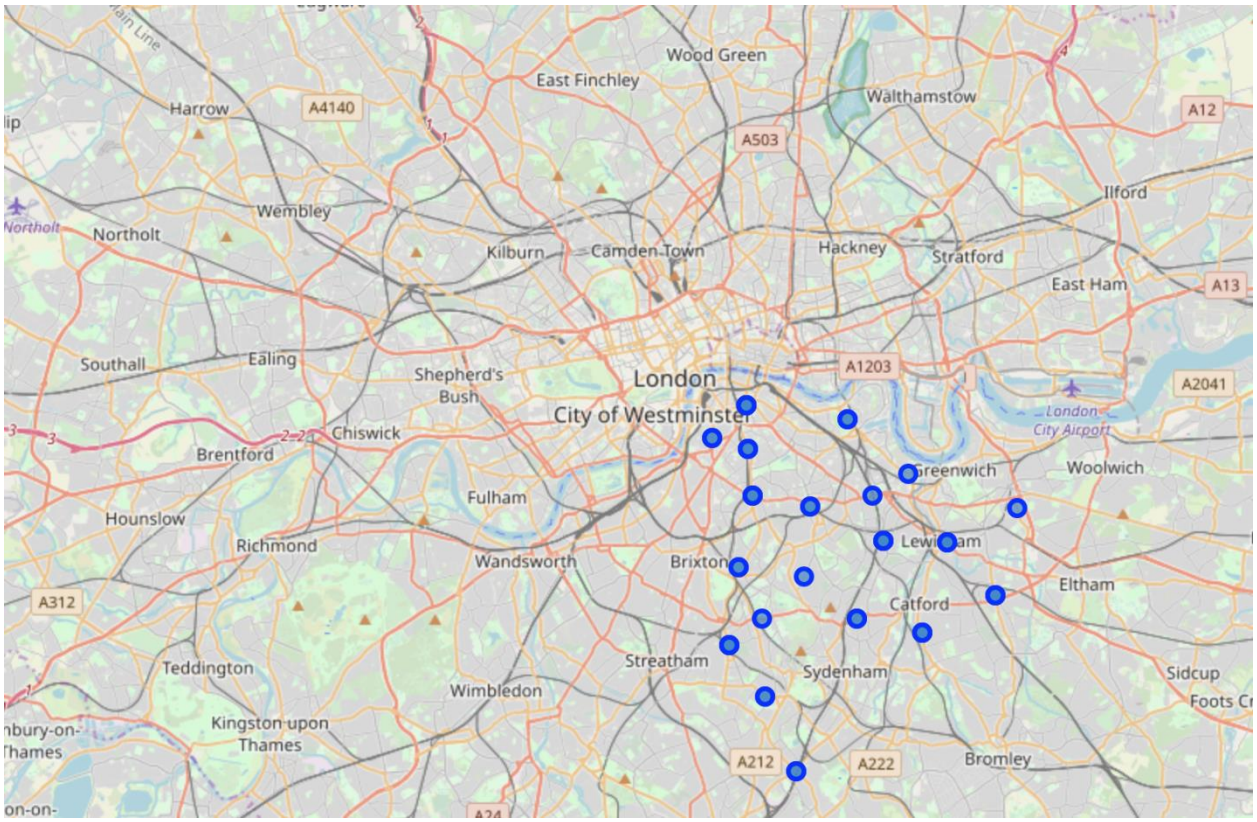
To get started, all the necessary libraries have been called in the libraries section:

In [1]:
```
# library for BeautifulSoup
from bs4 import BeautifulSoup

# library to handle data in a vectorized manner
import numpy as np

# library for data analsysis
import pandas as pd
pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', None)

# library to handle JSON files
import json
print('numpy, pandas, ..., imported...')

!pip -q install geopy
# conda install -c conda-forge geopy --yes # uncomment this line if you haven't completed the Foursquare API lab
print('geopy installed...')
# convert an address into latitude and longitude values
from geopy.geocoders import Nominatim
print('Nominatim imported...')

# library to handle requests
import requests
print('requests imported...')

# tranform JSON file into a pandas dataframe
from pandas.io.json import json_normalize
print('json_normalize imported...')

# Matplotlib and associated plotting modules
import matplotlib.cm as cm
import matplotlib.colors as colors
print('matplotlib imported...')

# import k-means from clustering stage
from sklearn.cluster import KMeans
print('Kmeans imported...')

# install the Geocoder
!pip -q install geocoder
import geocoder
```

3.2.2 Map Visualization

Using the *geopy* library, the latitude and longitude values of London is obtained: “The geographical coordinate of London are 51.5073219, -0.1276474”. The *folium* library is then used to obtain the coordinates:

The South East London neighborhoods are then superimposed on top as shown below, still using the *folium* library:



3.2.3 Analyzing Each Neighborhood

In this section, the objective is to check and explore the venues in each neighborhood.

First the *Neighbourhood* column is added back to the data-frame. There is some re-arrangement - move the new *Neighbourhood* column to the first column. Therefore, the new one hot encoded data-frame is:

Out[87]:

	Neighbourhood	African Restaurant	American Restaurant	Antique Shop	Aquarium	Argentinian Restaurant	Art Gallery	Art Museum	Arts & Crafts Store	Asian Restaurant	Athletics & Sports	Australian Restaurant	BBQ Joint	Bakery	Bar	Beach	Beer Bar	Beer Garden	Beer Store	Bike Shop	Bistro	Bookstore	Brazilian Restaurant	Breakfast Spot	Brewery
1896	Lewisham	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1897	Lewisham	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1898	Lewisham	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1899	Lewisham	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1900	Lewisham	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1901	Lewisham	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1902	Lewisham	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1903	Lewisham	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1904	Lewisham	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

As can be seen from above, Lewisham with its demography has no African restaurants within the top spots. The new data-frame size is (4252, 199).

Then we started “Regrouping and Category Statistics”:

Out[90]:

	Neighbourhood	African Restaurant	American Restaurant	Antique Shop	Aquarium	Argentinian Restaurant	Art Gallery	Art Museum	Arts & Crafts Store	Asian Restaurant	Athletics & Sports	Australian Restaurant	BBQ Joint	Bakery	Bar	Beach	Beer Bar	Beer Garden	Beer Store	Bike Shop	Bistro	Bookstore
0	Bankside	0.00	0.000000	0.0	0.0	0.010000	0.01	0.03	0.0	0.02	0.0	0.0	0.0	0.020000	0.020000	0.01	0.00	0.0	0.01	0.01	0.01	0.01
1	Bellingham	0.00	0.000000	0.0	0.0	0.000000	0.00	0.00	0.0	0.00	0.0	0.0	0.0	0.014286	0.014286	0.00	0.00	0.0	0.00	0.00	0.00	0.00
2	Bermondsey	0.00	0.000000	0.0	0.0	0.010000	0.01	0.03	0.0	0.02	0.0	0.0	0.0	0.020000	0.020000	0.01	0.00	0.0	0.01	0.01	0.01	0.01
3	Blackheath	0.00	0.011494	0.0	0.0	0.011494	0.00	0.00	0.0	0.00	0.0	0.0	0.0	0.034483	0.011494	0.00	0.00	0.0	0.00	0.00	0.00	0.00
4	Brixton	0.01	0.000000	0.0	0.0	0.000000	0.01	0.00	0.0	0.00	0.0	0.0	0.0	0.020000	0.030000	0.00	0.03	0.0	0.01	0.01	0.00	0.00

Before “One-hot encoding”: (46, 5)

After “One-hot encoding”: (40, 199)

Grouping of each Neighborhood with 10 common venues:

```
----Bankside----
                  venue  freq
0           Coffee Shop  0.08
1                 Hotel  0.06
2                   Pub  0.06
3    Italian Restaurant  0.05
4               Theater  0.04
5      Tapas Restaurant  0.03
6    Seafood Restaurant  0.03
7            Art Museum  0.03
8         Cocktail Bar  0.02
9  Street Food Gathering  0.02
```

Putting the common venues into *pandas* dataframe, the following *return_most_common_venuesis* used to sort the venues in descending order. Then we create a new panda data-frame with 10 most common venues as shown below:

Out[96]:

	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
0	Bankside	Coffee Shop	Hotel	Pub	Italian Restaurant	Theater	Seafood Restaurant	Art Museum	Tapas Restaurant	Pizza Place	
1	Bellingham	Grocery Store	Park	Supermarket	Pub	Fast Food Restaurant	Coffee Shop	Italian Restaurant	Café	Fried Chicken Joint	
2	Bermondsey	Coffee Shop	Hotel	Pub	Italian Restaurant	Theater	Seafood Restaurant	Art Museum	Tapas Restaurant	Pizza Place	
3	Blackheath	Pub	Grocery Store	Park	Coffee Shop	Garden	Bakery	Italian Restaurant	Clothing Store	Supermarket	
4	Brixton	Café	Park	Coffee Shop	Pub	Cocktail Bar	Italian Restaurant	Middle Eastern Restaurant	Indian Restaurant	Beer Bar	

3.2.4 Clustering of Neighborhoods

The next thing to do now is to create clusters of the neighborhood using the *k-means* to cluster the neighborhood into 5 clusters and creating a new data-frame that includes the clusters as well as the top 10 venues for each neighborhoods.

Out[105]:

	Location	Borough	Postcode	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	9th Most Common Venue	10th Most Common Venue
0	Crofton Park	Lewisham	SE4	51.46268	-0.03558	3	Coffee Shop	Pub	Café	Park	Bar	Gastropub	Cocktail Bar	Indian Restaurant	Italian Restaurant	Bakery
1	Denmark Hill	Southwark	SE5	51.47480	-0.09313	1	Café	Park	Coffee Shop	Pub	Cocktail Bar	Italian Restaurant	Middle Eastern Restaurant	Indian Restaurant	Beer Bar	Pizza Place
2	Deptford	Lewisham	SE8	51.48114	-0.02467	3	Pub	Coffee Shop	Café	Bar	Park	Trail	Cocktail Bar	Brewery	Deli / Bodega	Market
3	Dulwich	Southwark	SE21	51.44100	-0.08897	4	Pub	Café	Grocery Store	Bakery	Park	Coffee Shop	Gym / Fitness Center	Italian Restaurant	Brewery	Pizza Place
4	East Dulwich	Southwark	SE22	51.45256	-0.07076	1	Pub	Café	Pizza Place	Park	Italian Restaurant	Coffee Shop	Gastropub	Burger Joint	Cocktail Bar	Restaurant

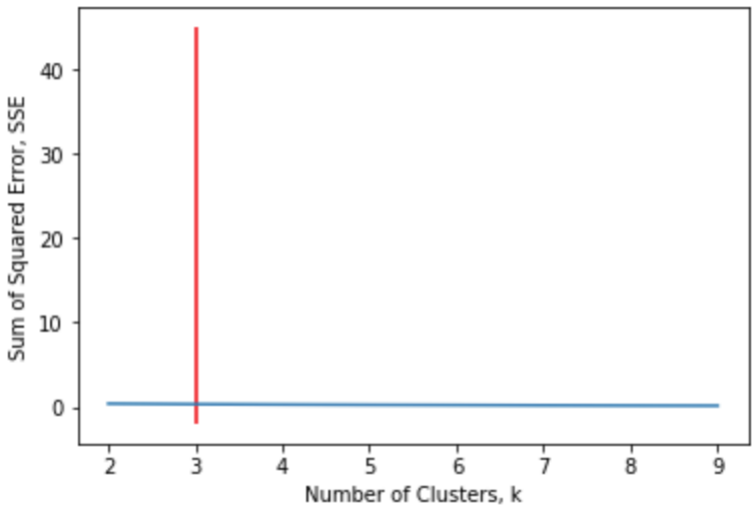
3.2.5 Optimal Number of Clusters for K-mean

To get the optimal number of clusters to be used for the *K-mean*, there are a number ways possible for the evaluation. Therefore, in this task, the following are used:

3.2.5.1 Elbow Method

The *elbow method* is used to solve the problem of selecting *k*. Interestingly, the elbow method is not perfect either but it gives significant insight that is perhaps not top optimal but sub-optimal to choosing the optimal number of clusters by fitting the model with a range of values for *k*.

The approach for this is to run the k-means clustering for a range of value *k* and for each value of *k*, the *Sum of the Squared Errors (SSE)* is calculated. When this is done, a plot of *k* and the corresponding *SSEs* are then made. At the elbow (just like arm), that is where the optimal value of *k* is. And that will be the number of clusters to be used. The whole idea is to have minimum *SSE*.



Depending on the number of iteration (in this case, 500 iterations were used), the number of cluster, *k* is 3.

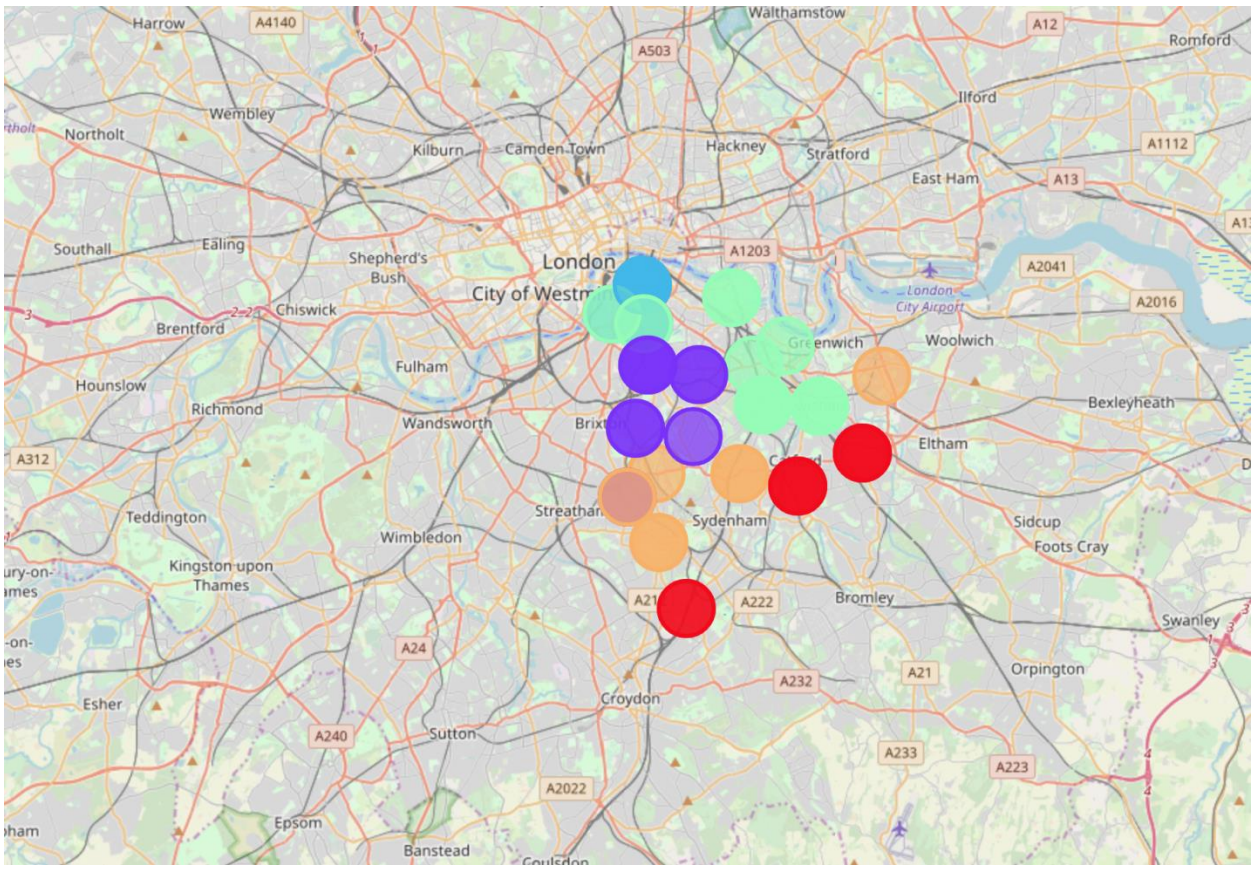
3.2.5.2 Silhouette Coefficient

To find the optimal value of the number of clusters, *k*, the number of clusters is iterated corresponding *Silhouette Coefficients* is calculated for each of the k-values used. The highest Silhouette Coefficient gives the best match to its own cluster. Please see below:

```
Where n_clusters = 2, the Silhouette Coefficient is 0.6081085930798573
Where n_clusters = 3, the Silhouette Coefficient is 0.6157992881218252
Where n_clusters = 4, the Silhouette Coefficient is 0.6365055758470731
Where n_clusters = 5, the Silhouette Coefficient is 0.674324874789089
Where n_clusters = 6, the Silhouette Coefficient is 0.769792007465332
Where n_clusters = 7, the Silhouette Coefficient is 0.8095362649872676
Where n_clusters = 8, the Silhouette Coefficient is 0.8666861116629072
Where n_clusters = 9, the Silhouette Coefficient is 0.9337555386129051
```

From this result: the high the *n_clusters* the better is the silhouette coefficient. For this project, a cluster value of 5 will be used.

3.2.6 Visualizing the Resulting Clusters


Harrow
A4140
Wembley
Northolt
Kilburn
Camden Town
A503
Walthamstow
A12
Romford
Ilford
Hackney
Stratford
A1112
East Ham
A13
Southall
Ealing
Shepherd's Bush
London
City of Westm
A1203
A2041
London City Airport
A2016
Chiswick
Brentford
Greenwich
Woolwich
Fulham
Hounslow
Richmond
Wandsworth
Bexleyheath
Eltham
A312
Teddington
Streatham
Sydenham
Sidcup
Foots Cray
Wimbledon
Kingston upon Thames
A222
Bromley
Swanley
A24
A232
A21
Orpington
Esher
Croydon
A240
Sutton
A233
A223
A2022
Epsom
Banstead

4. Results

The following are the highlights of the 5 clusters above:

- Pubs, Cafe, Coffee Shops are popular in the South East London.
- As for restaurants, the Italian Restaurants are very popular in the South East London area (especially in Southwark and Lambeth areas).
- With the Lewisham area being the most condensed area of Africans in the South East Area, it is surprising to see how in the top 10 venues, you can barely see restaurants in the top 5 venues.
- Although, the Clusters have variations, a very visible presence is the predominance of pubs.

5. Discussion

It is very important to note that Clusters 2 and 3 are the most viable clusters to create a brand African Restaurant. Their proximity to other amenities and accessibility to station are paramount. These 2 clusters do not have top restaurants that could rival their standards if they are created.

6. Conclusion

In conclusion, this project would have had better results if there were more data in terms of crime data within the area, traffic access and allowance of more venues exploration with the Foursquare (limited venues for free calls).

Also, getting the ratings and feedbacks of the current restaurants within the clusters would have helped in providing more insight into the best location