THE BATTLE OF NEIGHBOURHOODS

In this project, I have investigated where would be the best location in West London to open a new Korean restaurant. This research is aimed at clients looking to open new Korean restaurants and could be easily changed for other types of restaurants and other types of venues.

For this case I have created a scenario where I have a client who wants to open a new premium Korean restaurant in West London. I have assumed that a successful new restaurant would be in a busy area with many other restaurants where the number of potential clients walking past the restaurant would be high. However, an ideal location would also be in an area with a low number of other Korean restaurants to there being too much competition.

LIBRARIES

- Numpy - Library to handle data in a vectorized manner
- Pandas - Library for data analysis
- Requests - Library to handle requests
- SKLearn – I used DBSCAN to perform density based clustering and StandardScalar to normalise data
- Pyplot – Library to create plots
- Nominatim - Module to convert an address into latitude and longitude values
- Pandas.io.json – Library to transform json file into a pandas data frame
- Folium - Plotting library for creating maps

DATA

Foursquare Venue Data — To collect our data I used the Foursquare API to find basic information on all the Asian restaurants in our search area. Each request returns a maximum of 50 results meaning that for some areas not all of the restaurants are returned by a single search. To remedy this, I search for Chinese, Japanese and

Korean restaurants as well as searching for the more general ‘Asian restaurants’ category. This ensured that I retrieves the relevant data for all the restaurants in our area.

I specified the category ID’s which can be found on https://developer.foursquare.com/docs/build-with-foursquare/categories/ and specify a radius for each search in metres.

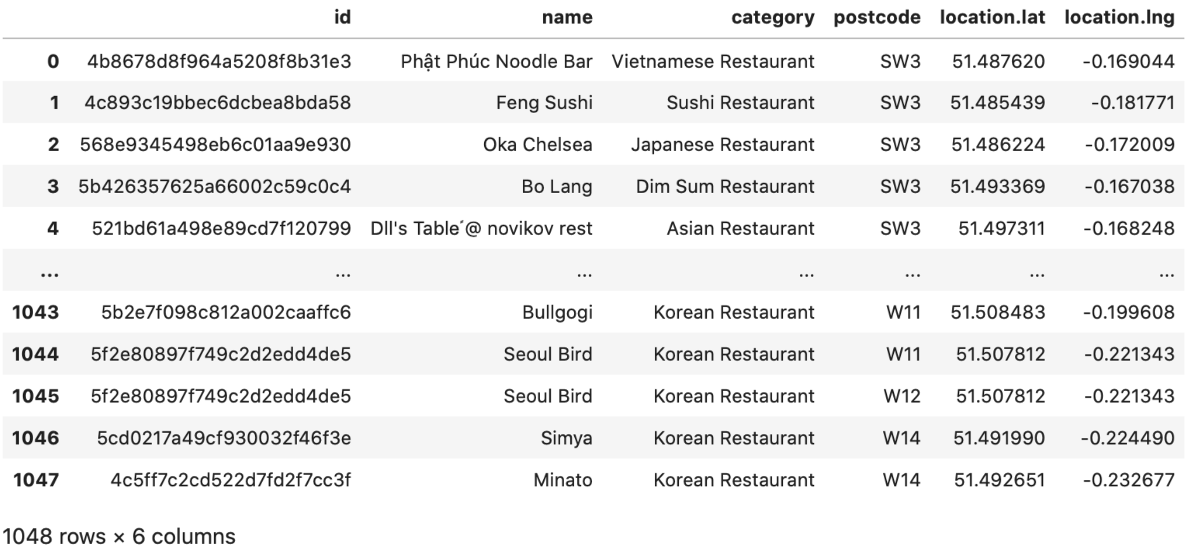
- 4bf58dd8d48988d142941735 – Asian Restaurants
- 4bf58dd8d48988d145941735 – Chinese Restaurants
- 4bf58dd8d48988d113941735 – Japanese Restaurants
- 4bf58dd8d48988d113941735 – Korean Restaurants

I then specified the postcodes for our search.

1. SW3
2. SW5
3. SW6
4. SW7
5. SW10
6. W2
7. W6
8. W8
9. W9
10. W10
11. W11
12. W1
13. W14

These restrict my search to West-central London, ignoring SW1, allowing us to keep the number of restaurants I request information on under 500 which is the limit for a free Foursquare account.

I then performed the search for each postcode and category using a for loop creating a pandas data frame as shown below.

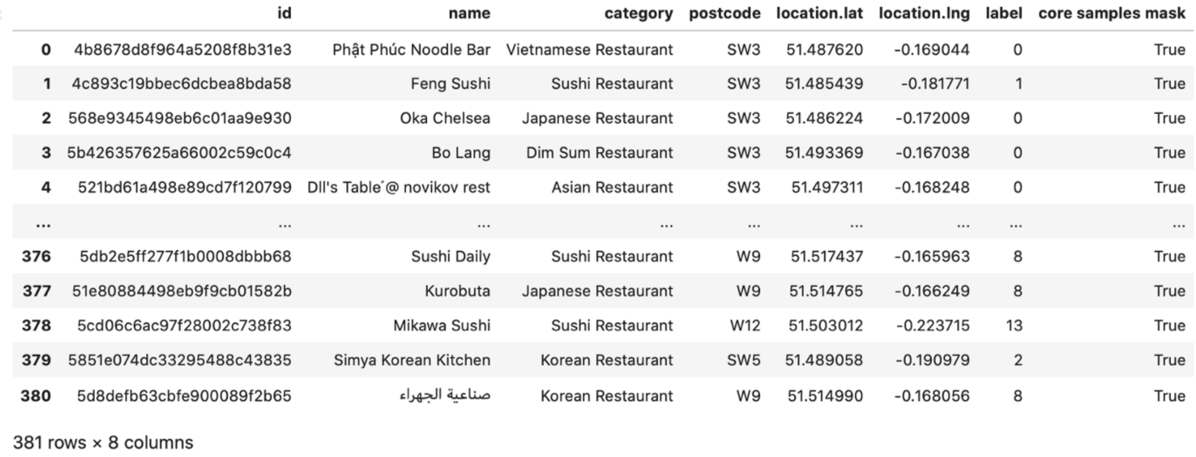


	id	name	category	postcode	location.lat	location.lng
0	4b8678d8f964a5208f8b31e3	Phật Phúc Noodle Bar	Vietnamese Restaurant	SW3	51.487620	-0.169044
1	4c893c19bbec6dcbea8bda58	Feng Sushi	Sushi Restaurant	SW3	51.485439	-0.181771
2	568e9345498eb6c01aa9e930	Oka Chelsea	Japanese Restaurant	SW3	51.486224	-0.172009
3	5b426357625a66002c59c0c4	Bo Lang	Dim Sum Restaurant	SW3	51.493369	-0.167038
4	521bd61a498e89cd7f120799	Dll's Table´@ novikov rest	Asian Restaurant	SW3	51.497311	-0.168248
...	...	...	...	...	...	...
1043	5b2e7f098c812a002caaffc6	Bullgogi	Korean Restaurant	W11	51.508483	-0.199608
1044	5f2e80897f749c2d2edd4de5	Seoul Bird	Korean Restaurant	W11	51.507812	-0.221343
1045	5f2e80897f749c2d2edd4de5	Seoul Bird	Korean Restaurant	W12	51.507812	-0.221343
1046	5cd0217a49cf930032f46f3e	Simya	Korean Restaurant	W14	51.491990	-0.224490
1047	4c5ff7c2cd522d7fd2f7cc3f	Minato	Korean Restaurant	W14	51.492651	-0.232677

1048 rows × 6 columns

For each restaurant I have retrieved the venue ID, the name, the category, the postcode and the latitude and longitude coordinates.

Because we have our search has used overlapping categories (‘Chinese Restaurant’ is a subcategory of ‘Asian Restaurant’) and overlapping areas (The area covered by each postcode is usually less than the circle created by the 1km radius meaning that the circles overlap) means that this data frame contains many of duplicates. We remove duplicates by creating a list of venue ID’s and dropping each entry that whose venue ID is already an element of the list. This creates a new data frame of unique entries.



	id	name	category	postcode	location.lat	location.lng	label	core samples mask
0	4b8678d8f964a5208f8b31e3	Phật Phúc Noodle Bar	Vietnamese Restaurant	SW3	51.487620	-0.169044	0	True
1	4c893c19bbec6dcbea8bda58	Feng Sushi	Sushi Restaurant	SW3	51.485439	-0.181771	1	True
2	568e9345498eb6c01aa9e930	Oka Chelsea	Japanese Restaurant	SW3	51.486224	-0.172009	0	True
3	5b426357625a66002c59c0c4	Bo Lang	Dim Sum Restaurant	SW3	51.493369	-0.167038	0	True
4	521bd61a498e89cd7f120799	Dll's Table´@ novikov rest	Asian Restaurant	SW3	51.497311	-0.168248	0	True
...	...	...	...	...	...	...	...	...
376	5db2e5ff277f1b0008dbbb68	Sushi Daily	Sushi Restaurant	W9	51.517437	-0.165963	8	True
377	51e80884498eb9f9cb01582b	Kurobuta	Japanese Restaurant	W9	51.514765	-0.166249	8	True
378	5cd06c6ac97f28002c738f83	Mikawa Sushi	Sushi Restaurant	W12	51.503012	-0.223715	13	True
379	5851e074dc33295488c43835	Simya Korean Kitchen	Korean Restaurant	SW5	51.489058	-0.190979	2	True
380	5d8defb63cbfe900089f2b65	صناعية الجهراء	Korean Restaurant	W9	51.514990	-0.168056	8	True

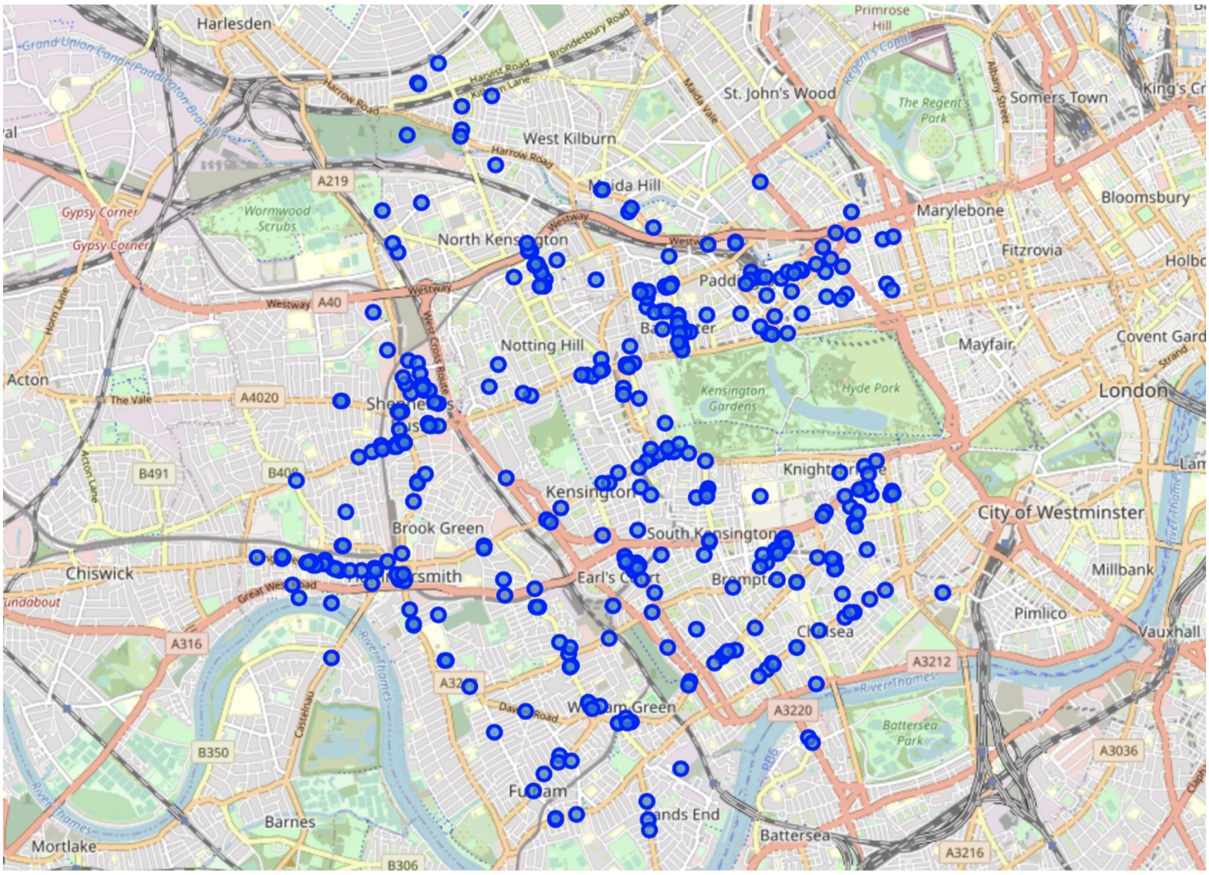
381 rows × 8 columns

We will also use the foursquare API to request more details on a subset of the restaurants by using each restaurants venue id. This lets us retrieve the average rating and price bracket of each restaurant.

We are only able to make 500 of these calls each day so make sure that our code keeps all necessary data.

METHODOLOGY

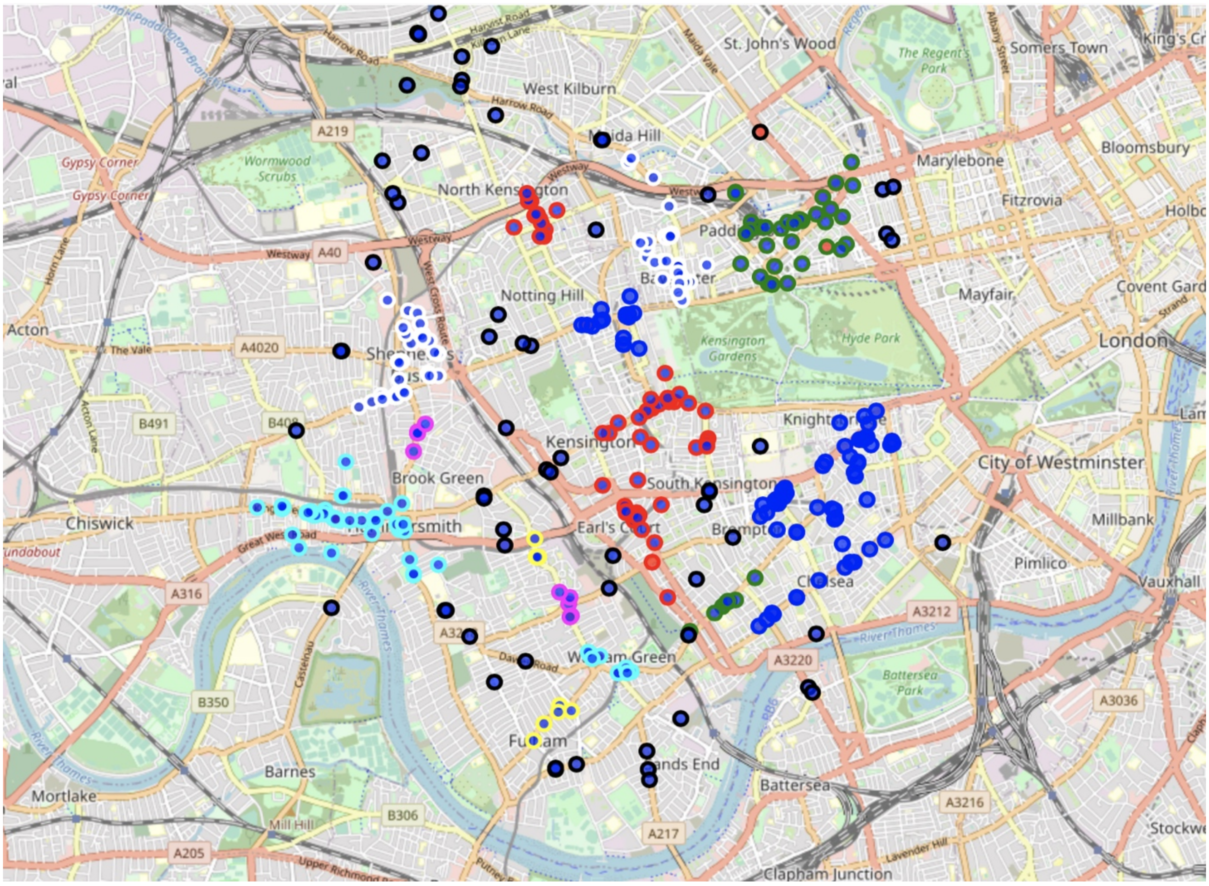
To start, we create a map of London using Folium so that we can visualise Londons restaurants. We use the latitude and longitude coordinates in our data frame to place markers on the map and label each marker with the name of the restaurant and its category.



From this map we can see that the density of restaurants varies across the city. We want to identify areas with a high density of Asian restaurants and to do this we use density-based clustering to group the restaurants together as well as to identify outliers.

We used the DBSCAN algorithm on the latitude and longitude coordinates to label each restaurant with it's cluster's number or with -1 if it is an outlier. We then used these labels to choose colours for each cluster so that they can be easily identified on the map.

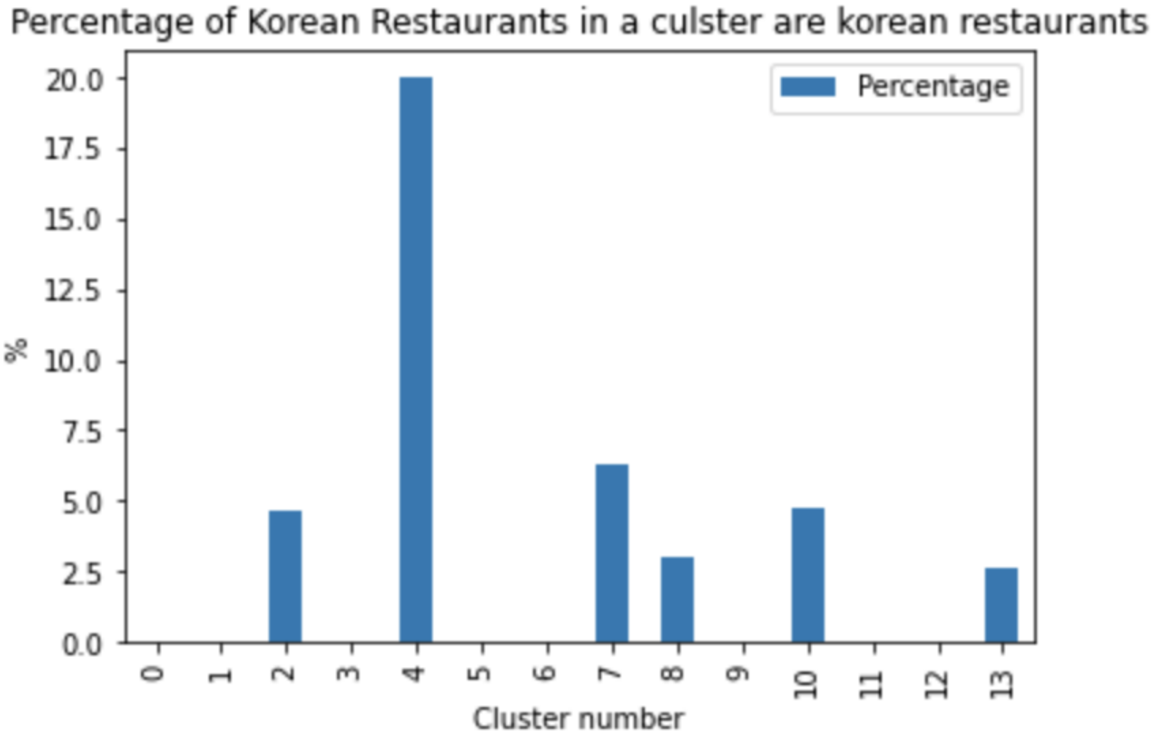
I then created a new map using the new coloured markers. I also changed the markers fill colour so that Korean restaurants had a red fill and other restaurants havd a blue fill.



The clusters are

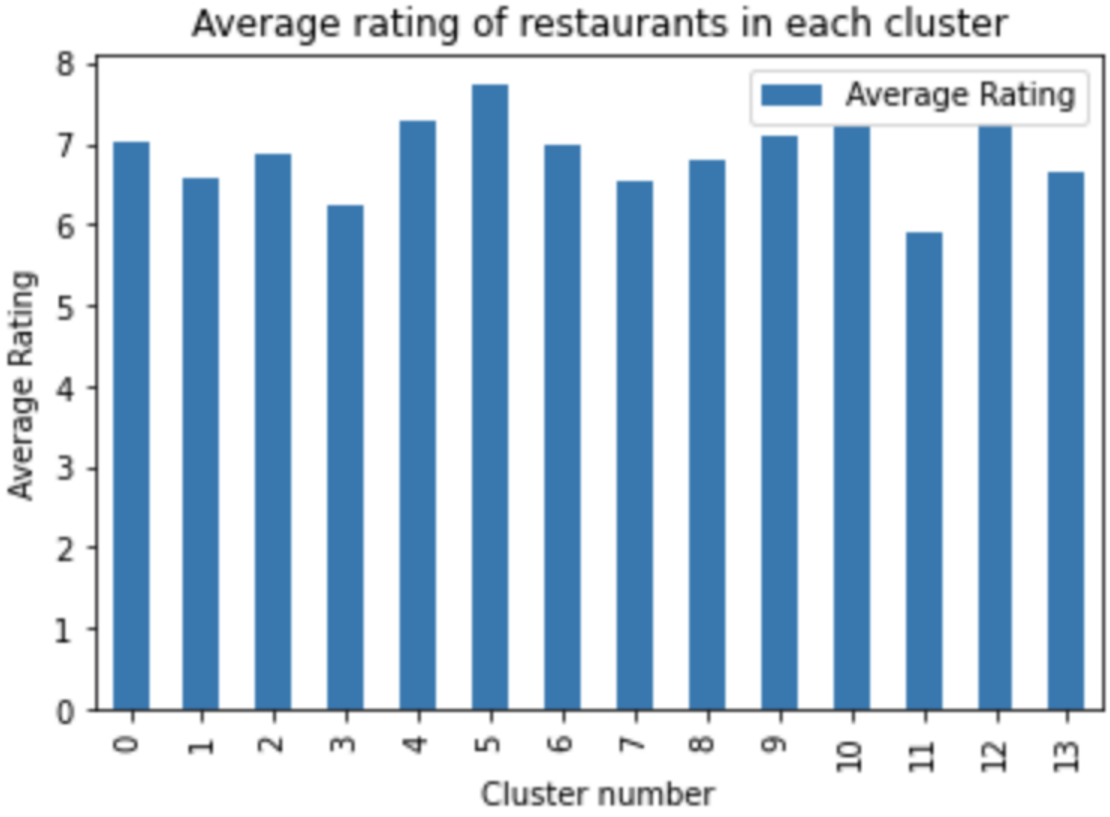
0. Chelsea, South Kensington and Knightsbridge — Blue
1. Fulham Road, Chelsea — Green
2. Earls Court and High Street Kensington — Red
3. Fulham Broadway — Cyan
4. North End Road, Fulham — Magenta
5. Fulham Road, Parsons Green — Yellow
6. Bayswater — White
7. Notting Hill — Blue
8. Paddington — Green
9. North Kensington — Red
10. Hammersmith — Cyan
11. Brook Green — Magenta
12. North End Road, West Kensington — Yellow
13. Shepherd's Bush — White

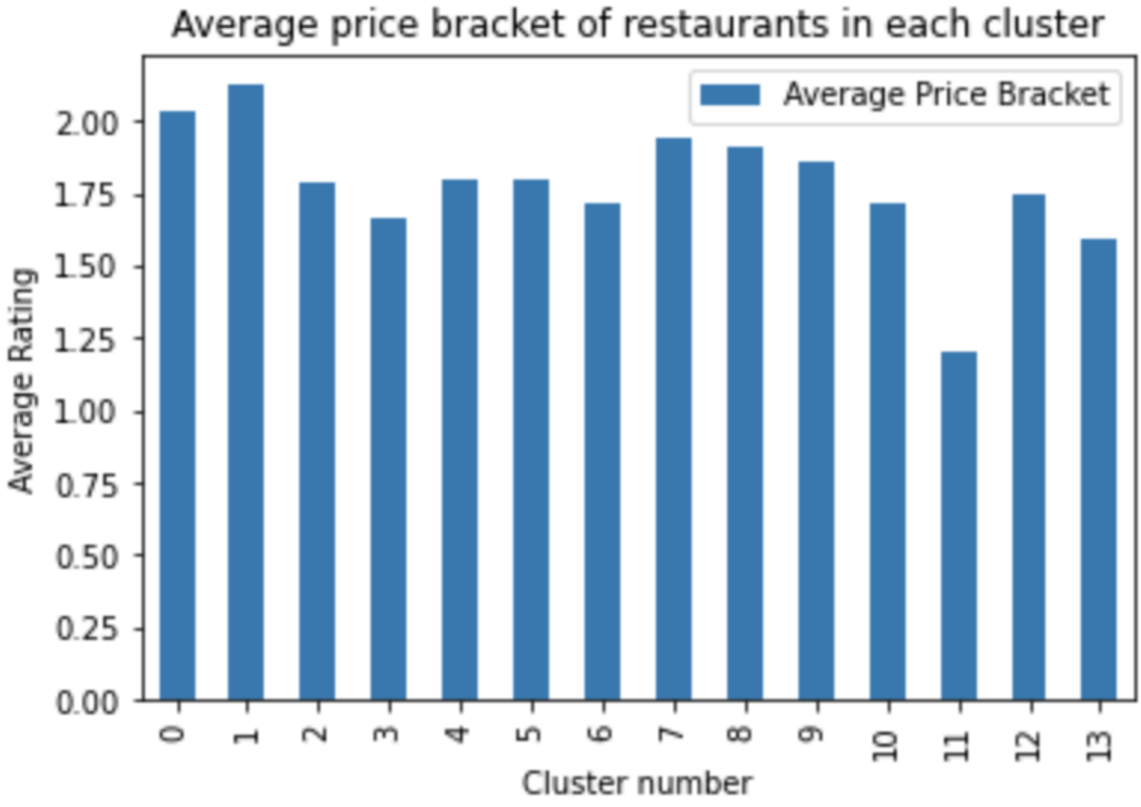
We now want to perform some analysis on each cluster: For each cluster, we calculate what percentage of Asian restaurants are Korean restaurants and then plot this as a bar chart using matplotlib.



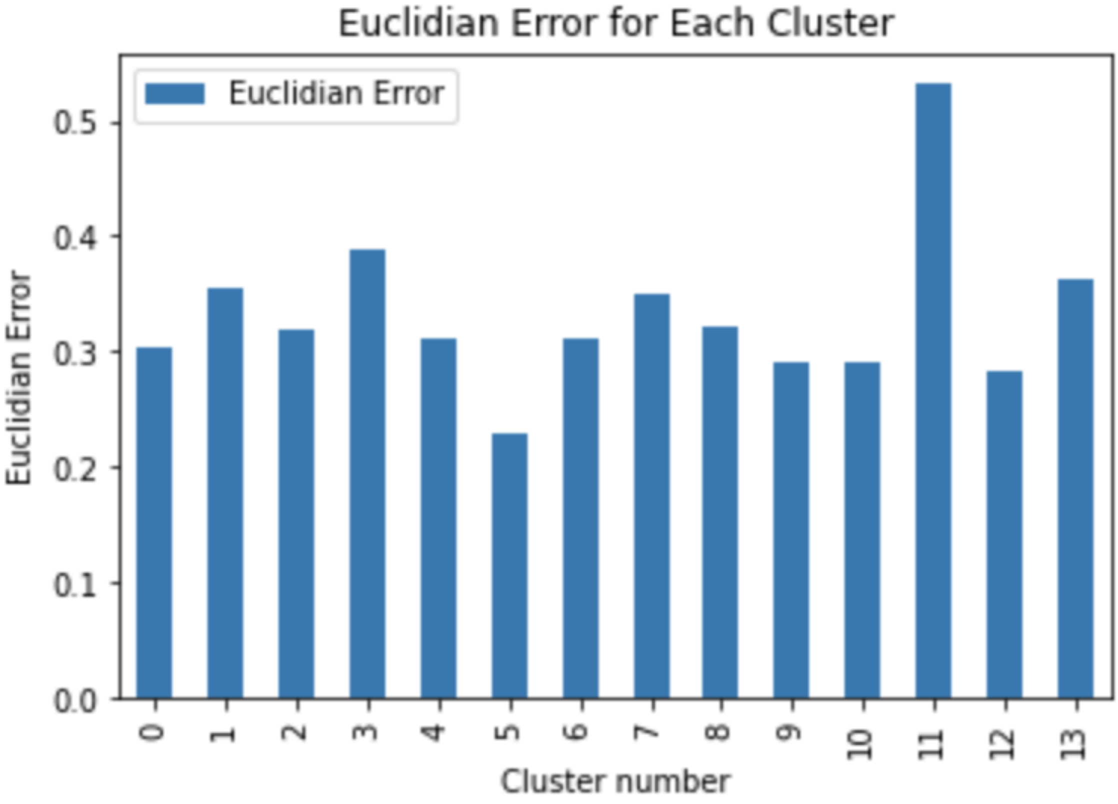
This shows us that cluster 4 already has a high number of Korean restaurants and should be avoided as it would have too much competition.

We similarly find the average rating (out of 10) and the average price bracket (from 1 to 4) for each cluster and plot them as bar charts. To do this requires making a premium call on the Foursquare API for each restaurant in the clusters.





To evaluate which cluster is most suitable for our client's new restaurant I first define a set of values for average rating, average price bracket and percentage of restaurants in a cluster that are Korean restaurants, set at 10, 1.9 and 0 respectively. I calculated the difference between the real value and the ideal value for each cluster and normalised these values by dividing by what is roughly their theoretical maximum. (100% for the difference in percentage, 2 for the difference in average price bracket and 10 for the difference in average rating). I then created a metric called the Euclidian error which is the square root of the sum of the squares of the three normalised differences and plotted it as a bar chart.



Results

From our chart of the Euclidian error for each cluster we find that cluster 5 is the best candidate as it is the closest to a perfect cluster. This cluster is located on Fulham Road, near to Parsons Green tube station.

Discussion and Conclusion

Our method makes a lot of assumptions but is a good simplified model. In a future version I could include a wider selection of categories to create the clusters, but I think the difference would be marginal as the clusters are already quite well defined. In addition, I could use other clustering methods to perform experimental data analysis.

Another area of interest would be to use a property API such as the Rightmove API to estimate the price of opening the new restaurant by examining commercial rents in various postcodes. This would be a major factor in any choice of location.

Using historical data by changing the Foursquare version in requests would allow me to examine which restaurants are more successful and which ones are likely to close down. This would allow me to eliminate my assumptions regarding which locations are likely to lead to successful restaurants. K-Means on a number of factors would help me to work what types of restaurants are likely to stay open for long periods.

Our method used above has clearly separated London Asian restaurants into discrete clusters and provided interesting insights into these various areas.