# The Battle of Neighborhoods in Paddington, London, UK

## Applied Data Science Capstone Week 5 Peer-Graded Assignment

## Introduction to the opportunity

Paddington is an area within the City of Westminster, in central London, located in the West End of London. First a medieval parish then a metropolitan borough, it was integrated with Westminster and Greater London in 1965. Three important landmarks of the district are Paddington station, designed by the celebrated engineer Isambard Kingdom Brunel and opened in 1847; St Mary's Hospital; and Paddington Green Police Station (the most important high-security police station in the United Kingdom).

The Paddington district is centered around Paddington railway station. The conventional recognized boundary of the district is much smaller than the longstanding pre-mid-19th century parish. That parish was virtually equal to the borough abolished in 1965. It is divided from a northern offshoot Maida Vale by the Regent's Canal; its overlap is the artisan and touristic neighborhood of Little Venice. In the east of the district around Paddington Green, it remains divided from Marylebone by Edgware Road (as commonly heard in spoken form, the Edgware Road). In the southwest, it is bounded by its south and western offshoot Bayswater. A final offshoot, Westbourne, rises to the northwest.

Commercial traffic on the Grand Junction Canal (which became the Grand Union Canal in 1929) dwindled because of railway competition in the late-19th and early-20th centuries, and freight then moved from rail to road after World War II, leading to the abandonment of the goods yards in the early 1980s. The land lay derelict until the Paddington Waterside Partnership was established in 1998 to co-ordinate the regeneration of the area between the Westway, Praed Street, and Westbourne Terrace. This includes major developments on the goods yard site (now branded Paddington Central) and around the canal (Paddington Basin). As of October 2017 much of these developments have been completed and are in use.

A major project called Paddington Waterside aims to regenerate former railway and canal land between 1998 and 2018, and the area is seeing many new developments. Offshoot districts (historically within Paddington) are Maida Vale, Westbourne, and Bayswater including Lancaster Gate.

As the city grows and develops, it becomes increasingly important to examine and understand it quantitatively to develop services for the benefit of its citizens.

Consultants, developers, policy makers and/or city planners have an interest in answering the following questions:

1. If someone is looking to open a restaurant, where would they open it?
2. If a contractor is trying to start their own business, where would they setup their office?
3. Using Foursquare data, what venues are most common in different locations within the city?

## Data

**To understand and explore we will need the following data:**

1. About the Paddington district in London: https://en.wikipedia.org/wiki/Paddington (https://en.wikipedia.org/wiki/Paddington)
2. The postcode area of Paddington: https://en.wikipedia.org/wiki/W_postcode_area (https://en.wikipedia.org/wiki/W_postcode_area)
3. UK Postcodes with Latitude and Longitude: https://www.freemaptools.com/download/outcode-postcodes/postcode-outcodes.csv (https://www.freemaptools.com/download/outcode-postcodes/postcode-outcodes.csv)
4. Foursquare Developers Access to venue data: https://foursquare.com/ (https://foursquare.com/)

In this assignment, we will explore, segment, and cluster the neighborhoods in Paddington. However, the neighborhood data is not readily available on the internet. For the Paddington neighborhood data, a Wikipedia page exists that has all the information we need to explore and cluster the neighborhoods in Toronto. We will scrape the Wikipedia page and wrangle the data, clean it, and then read it into a pandas data frame so that it is in a structured format like a dataset. Once the data is in a structured format, we can analyze the dataset to explore and cluster the neighborhoods in Paddington.

# Methodology

All steps are referenced below in the Appendix.

The methodology will include:

1. Loading the data
2. Exploring the data
3. Exploring The Neighbourhoods of Western and Paddington, London
4. Clustering Paddington Location by using k-means

## Loading the data

Start by creating a new Notebook for this assignment. Use the Notebook to build the code to scrape the following Wikipedia page, https://en.wikipedia.org/wiki/W_postcode_area (https://en.wikipedia.org/wiki/W_postcode_area), in order to obtain the data that is in the table of postal codes and to transform the data into a pandas data frame. Another way, which would help to learn for more complicated cases of web scraping is using the BeautifulSoup package. Here is the package's main documentation page: http://beautiful-soup-4.readthedocs.io/en/latest/ (http://beautiful-soup-4.readthedocs.io/en/latest/). The package is so popular that there is a plethora of tutorials and examples on how to use it. Here is a very good Youtube video on how to use the BeautifulSoup package: https://www.youtube.com/watch?v=ng2o98k983k (https://www.youtube.com/watch?v=ng2o98k983k). Use pandas, or the BeautifulSoup package, or any other way you are comfortable with to transform the data in the table on the Wikipedia page into the above pandas dataframe.

## Exploring the data

Now we will have been equipped with the skills and the tools to use location data to explore a geographical location. We will have the opportunity to be as creative as we want and come up with an idea to leverage the Foursquare location data to explore or compare neighborhoods or cities of our choice or to come up with a problem that we can use the Foursquare location data to solve.

## Exploring The Neighbourhoods of Western and Paddington, London

Exploring and clustering the neighborhoods in Paddington can help to explain what you decided to do and to report any observations you make and to generate maps to visualize the neighborhoods and how they cluster together. Using this data will allow exploration and examination to answer the questions. Paddington locations of interest will then allow us to cluster and quantitatively understand the venues most common to that location.

### Clustering Paddington Location by using k-means

K-means clustering is a type of unsupervised learning, which is used when you have unlabeled data (i.e., data without defined categories or groups). The goal of this algorithm is to find groups in the data, with the number of groups represented by the variable K. The algorithm works iteratively to assign each data point to one of K groups based on the features that are provided. Data points are clustered based on feature similarity.

Loading the Paddington Locations data enables us to perform statistical analysis on the most common venues by location. Plotting the latitude and longitude coordinates of the locations of interest that enables us to now study the most common venues by using the Foursquare data. Analyzing each location by grouping rows by location and the mean of the frequency of occurrence of each category we venue categories we study the top five most common venues. Putting this data into a pandas dataframe we can then determine the most common venues by location and plot onto a map.

## Results

The analysis enabled us to discover and describe visually and quantitatively:

1. Identify the Neighborhoods in Paddington
2. Determine the top 10 most common venues by location of interest within a 1 km radius of the center by using k-means.
3. Evaluate the number of restaurants/business offices within each cluster of locations by statistics.

## Discussion

With the findings of the top 10 most frequent venues by locations of interest as determined from the Foursquare dataset, consultants, developers, policymakers, and/or city planners have an interest in answering the following questions:

1. If someone is looking to open a restaurant, where would they open it?
2. If a contractor is trying to start their own business, where would they setup their office?
3. What venues are most common in different locations within the city?

## Conclusion

Using a combination of datasets of Paddington and Foursquare venue data we were able to analyze, discover, and describe neighborhoods, and statistically describe quantitatively venues by locations of interest. In brief, the datasets of Paddington are interesting, they miss the details required for true valued quantitative analysis and predictive analytics which would be most valued by investors and developers to make appropriate investments and to minimize risk.

## Appendix

## Import Libraries

Entrée [ ]:

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

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

# import plotting modules
import matplotlib.cm as cm
import matplotlib.colors as colors

# import k-means
from sklearn.cluster import KMeans

# import Beautiful Soup
from bs4 import BeautifulSoup

import json # import json to handle JSON files

!conda install -c conda-forge geopy --yes
from geopy.geocoders import Nominatim # from an address into latitude and longitude values

import requests # import library to handle requests
from pandas.io.json import json_normalize # from JSON file into a pandas dataframe

import xml

!conda install -c conda-forge folium=0.5.0 --yes
import folium # import folium for rendering

print('The libraries loaded successfully.')
```

## Loading the data: Scrapping Wikipedia page

Entrée [ ]:

```
url = requests.get('https://en.wikipedia.org/wiki/W_postcode_area').text
bs = BeautifulSoup(url,'lxml')
```

## Creating the dataframe with four columns: 'Postcode district', 'Post town', 'Coverage', 'Local authority area'

Entrée [4]:

```
tbl_post = bs.find('table', { "class" : "wikitable sortable" })
rows= tbl_post.find_all('th')
cols= tbl_post.find_all('td')

postcode = []
posttown = []
coverage = []
local = []

for i in range(4, len(rows)):
    postcode.append(rows[i].text.strip())

for i in range(0, len(cols), 3):
    posttown.append(cols[i+0].text.strip())
    coverage.append(cols[i+1].text.strip())
    local.append(cols[i+2].text.strip())

df_post = pd.DataFrame(data=[postcode, posttown, coverage, local]).transpose()
df_post.columns = ['Postcode', 'Post town', 'Coverage', 'Local authority area']
df_post
```

Out[4]:

| | Postcode | Post town | Coverage | Local authority area |
|---|---|---|---|---|
| 0 | W1A | LONDON | PO boxes & Admail codes in W1[7] | non-geographic |
| 1 | W1B | LONDON | Portland Place, Regent Street | Westminster |
| 2 | W1C | LONDON | Oxford Street (west) | Westminster |
| 3 | W1D | LONDON | Soho (south east); Chinatown, Soho Square | Westminster |
| 4 | W1F | LONDON | Soho (north west) | Westminster |
| 5 | W1G | LONDON | Harley Street | Westminster |
| 6 | W1H | LONDON | Marylebone | Westminster |
| 7 | W1J | LONDON | Mayfair (south), Piccadilly | Westminster |
| 8 | W1K | LONDON | Mayfair (north), Grosvenor Square | Westminster |

## Exploring data

## Only process the cells that have an assigned locak authority area. Ignore cells with a borough that is non-geographic.

Entrée [5]:

```
df_post['Local authority area'].replace('non-geographic', np.nan, inplace=True)
df_post.dropna(subset=['Local authority area'], inplace=True)

df_post
```

Out[5]:

| | Postcode | Post town | Coverage | Local authority area |
|---|---|---|---|---|
| **1** | W1B | LONDON | Portland Place, Regent Street | Westminster |
| **2** | W1C | LONDON | Oxford Street (west) | Westminster |
| **3** | W1D | LONDON | Soho (south east); Chinatown, Soho Square | Westminster |
| **4** | W1F | LONDON | Soho (north west) | Westminster |
| **5** | W1G | LONDON | Harley Street | Westminster |
| **6** | W1H | LONDON | Marylebone | Westminster |
| **7** | W1J | LONDON | Mayfair (south), Piccadilly | Westminster |
| **8** | W1K | LONDON | Mayfair (north), Grosvenor Square | Westminster |
| **9** | W1S | LONDON | Mayfair (east), Hanover Square, Savile Row, Ro... | Westminster |
| **10** | W1T | LONDON | Fitzrovia, Tottenham Court Road | Camden |
| **11** | W1U | LONDON | Marylebone | Westminster |
| **12** | W1W | LONDON | Great Portland Street, Fitzrovia | Westminster |
| **13** | W2 | LONDON | Paddington head district: Paddington, Bayswate... | Kensington and Chelsea, Westminster |
| **14** | W3 | LONDON | Acton district: Acton, West Acton, North Acton... | Ealing, Hammersmith and Fulham, Hounslow |
| **15** | W4 | LONDON | Chiswick district: Chiswick, Gunnersbury, Turn... | Ealing, Hammersmith and Fulham, Hounslow |
| **16** | W5 | LONDON | Ealing district: Ealing, South Ealing, Ealing ... | Ealing, Hounslow |
| **17** | W6 | LONDON | Hammersmith district: Fulham, Hammersmith, Rav... | Hammersmith and Fulham, Hounslow |
| **18** | W7 | LONDON | Hanwell district: Hanwell, Boston Manor (part) | Ealing, Hounslow |
| **19** | W8 | LONDON | Kensington district: Kensington, Holland Park ... | Kensington and Chelsea |
| **20** | W9 | LONDON | Maida Hill district: Maida Hill, Maida Vale, L... | Brent, Camden, Westminster |
| **21** | W10 | LONDON | North Kensington district: North Kensington, K... | Brent, Hammersmith and Fulham, Kensington and ... |
| **22** | W11 | LONDON | Notting Hill district: Notting Hill, Ladbroke ... | Hammersmith and Fulham, Kensington and Chelsea... |
| **23** | W12 | LONDON | Shepherds Bush district: Shepherds Bush, White... | Hammersmith and Fulham |
| **24** | W13 | LONDON | West Ealing district: West Ealing, Northfields... | Ealing |

| Postcode | Post town | Coverage | Local authority area |
| --- | --- | --- | --- |

Entrée [6]:

```
df_postna = df_post.groupby(['Postcode', 'Post town', 'Coverage'])['Local authority area'].
df_postna.columns = ['Postcode', 'Post town', 'Coverage', 'Local authority area']
df_postna
```

Out[6]:

| | Postcode | Post town | Coverage | Local authority area |
|---|---|---|---|---|
| **0** | W10 | LONDON | North Kensington district: North Kensington, K... | Brent, Hammersmith and Fulham, Kensington and ... |
| **1** | W11 | LONDON | Notting Hill district: Notting Hill, Ladbroke ... | Hammersmith and Fulham, Kensington and Chelsea... |
| **2** | W12 | LONDON | Shepherds Bush district: Shepherds Bush, White... | Hammersmith and Fulham |
| **3** | W13 | LONDON | West Ealing district: West Ealing, Northfields... | Ealing |
| **4** | W14 | LONDON | West Kensington district: West Kensington, Ken... | Hammersmith and Fulham, Kensington and Chelsea |
| **5** | W1B | LONDON | Portland Place, Regent Street | Westminster |
| **6** | W1C | LONDON | Oxford Street (west) | Westminster |
| **7** | W1D | LONDON | Soho (south east); Chinatown, Soho Square | Westminster |
| **8** | W1F | LONDON | Soho (north west) | Westminster |
| **9** | W1G | LONDON | Harley Street | Westminster |
| **10** | W1H | LONDON | Marylebone | Westminster |
| **11** | W1J | LONDON | Mayfair (south), Piccadilly | Westminster |
| **12** | W1K | LONDON | Mayfair (north), Grosvenor Square | Westminster |
| **13** | W1S | LONDON | Mayfair (east), Hanover Square, Savile Row, Ro... | Westminster |
| **14** | W1T | LONDON | Fitzrovia, Tottenham Court Road | Camden |
| **15** | W1U | LONDON | Marylebone | Westminster |
| **16** | W1W | LONDON | Great Portland Street, Fitzrovia | Westminster |
| **17** | W2 | LONDON | Paddington head district: Paddington, Bayswate... | Kensington and Chelsea, Westminster |
| **18** | W3 | LONDON | Acton district: Acton, West Acton, North Acton... | Ealing, Hammersmith and Fulham, Hounslow |
| **19** | W4 | LONDON | Chiswick district: Chiswick, Gunnersbury, Turn... | Ealing, Hammersmith and Fulham, Hounslow |
| **20** | W5 | LONDON | Ealing district: Ealing, South Ealing, Ealing ... | Ealing, Hounslow |
| **21** | W6 | LONDON | Hammersmith district: Fulham, Hammersmith, Rav... | Hammersmith and Fulham, Hounslow |
| **22** | W7 | LONDON | Hanwell district: Hanwell, Boston Manor (part) | Ealing, Hounslow |
| **23** | W8 | LONDON | Kensington district: Kensington, Holland Park ... | Kensington and Chelsea |

| | Postcode | Post town | Coverage | Local authority area |
|---|---|---|---|---|
| **24** | W9 | LONDON | Maida Hill district: Maida Hill, Maida Vale, L... | Brent, Camden, Westminster |

Entrée [7]:

```
df_postna['Local authority area'].replace('non-geographic', "Westminster", inplace=True)

df_postna
```

Out[7]:

| | Postcode | Post town | Coverage | Local authority area |
|---|---|---|---|---|
| **0** | W10 | LONDON | North Kensington district: North Kensington, K... | Brent, Hammersmith and Fulham, Kensington and ... |
| **1** | W11 | LONDON | Notting Hill district: Notting Hill, Ladbroke ... | Hammersmith and Fulham, Kensington and Chelsea... |
| **2** | W12 | LONDON | Shepherds Bush district: Shepherds Bush, White... | Hammersmith and Fulham |
| **3** | W13 | LONDON | West Ealing district: West Ealing, Northfields... | Ealing |
| **4** | W14 | LONDON | West Kensington district: West Kensington, Ken... | Hammersmith and Fulham, Kensington and Chelsea |
| **5** | W1B | LONDON | Portland Place, Regent Street | Westminster |
| **6** | W1C | LONDON | Oxford Street (west) | Westminster |
| **7** | W1D | LONDON | Soho (south east); Chinatown, Soho Square | Westminster |
| **8** | W1F | LONDON | Soho (north west) | Westminster |
| **9** | W1G | LONDON | Harley Street | Westminster |
| **10** | W1H | LONDON | Marylebone | Westminster |
| **11** | W1J | LONDON | Mayfair (south), Piccadilly | Westminster |
| **12** | W1K | LONDON | Mayfair (north), Grosvenor Square | Westminster |
| **13** | W1S | LONDON | Mayfair (east), Hanover Square, Savile Row, Ro... | Westminster |
| **14** | W1T | LONDON | Fitzrovia, Tottenham Court Road | Camden |
| **15** | W1U | LONDON | Marylebone | Westminster |
| **16** | W1W | LONDON | Great Portland Street, Fitzrovia | Westminster |
| **17** | W2 | LONDON | Paddington head district: Paddington, Bayswate... | Kensington and Chelsea, Westminster |
| **18** | W3 | LONDON | Acton district: Acton, West Acton, North Acton... | Ealing, Hammersmith and Fulham, Hounslow |
| **19** | W4 | LONDON | Chiswick district: Chiswick, Gunnersbury, Turn... | Ealing, Hammersmith and Fulham, Hounslow |
| **20** | W5 | LONDON | Ealing district: Ealing, South Ealing, Ealing ... | Ealing, Hounslow |
| **21** | W6 | LONDON | Hammersmith district: Fulham, Hammersmith, Rav... | Hammersmith and Fulham, Hounslow |
| **22** | W7 | LONDON | Hanwell district: Hanwell, Boston Manor (part) | Ealing, Hounslow |
| **23** | W8 | LONDON | Kensington district: Kensington, Holland Park ... | Kensington and Chelsea |

| | Postcode | Post town | Coverage | Local authority area |
|---|---|---|---|---|
| **24** | W9 | LONDON | Maida Hill district: Maida Hill, Maida Vale, L... | Brent, Camden, Westminster |

Entrée [8]:

```
df_postna.shape
```

Out[8]:

```
(25, 4)
```

Entrée [9]:

```
df_gp = pd.read_csv('https://www.freemaptools.com/download/outcode-postcodes/postcode-outco
df_gp.columns = ['id','Postcode', 'Latitude', 'Longitude']
```

Entrée [10]:

```
df_merge = pd.merge(df_postna, df_gp, on=['Postcode'], how='inner')

df_result = df_merge[['Post town', 'Coverage', 'Local authority area', 'id', 'Postcode', 'L

df_result
```

Out[10]:

| | Post town | Coverage | Local authority area | id | Postcode | Latitude | Longitude |
|---|---|---|---|---|---|---|---|
| **0** | LONDON | North Kensington district: North Kensington, K... | Brent, Hammersmith and Fulham, Kensington and ... | 2682 | W10 | 51.52103 | -0.21397 |
| **1** | LONDON | Notting Hill district: Notting Hill, Ladbroke ... | Hammersmith and Fulham, Kensington and Chelsea... | 2683 | W11 | 51.51189 | -0.20424 |
| **2** | LONDON | Shepherds Bush district: Shepherds Bush, White... | Hammersmith and Fulham | 2684 | W12 | 51.50777 | -0.22890 |
| **3** | LONDON | West Ealing district: West Ealing, Northfields... | Ealing | 2685 | W13 | 51.51270 | -0.31951 |
| **4** | LONDON | West Kensington district: West Kensington, Ken... | Hammersmith and Fulham, Kensington and Chelsea | 2686 | W14 | 51.49488 | -0.20923 |
| **5** | LONDON | Portland Place, Regent Street | Westminster | 2687 | W1B | 51.51357 | -0.13931 |
| **6** | LONDON | Oxford Street (west) | Westminster | 2688 | W1C | 51.51371 | -0.14795 |
| **7** | LONDON | Soho (south east); Chinatown, Soho Square | Westminster | 2689 | W1D | 51.51344 | -0.13066 |
| **8** | LONDON | Soho (north west) | Westminster | 2690 | W1F | 51.51261 | -0.13502 |
| **9** | LONDON | Harley Street | Westminster | 2691 | W1G | 51.51818 | -0.14633 |
| **10** | LONDON | Marylebone | Westminster | 2692 | W1H | 51.51659 | -0.15936 |
| **11** | LONDON | Mayfair (south), Piccadilly | Westminster | 2693 | W1J | 51.50735 | -0.14388 |
| **12** | LONDON | Mayfair (north), Grosvenor Square | Westminster | 2694 | W1K | 51.51104 | -0.14950 |
| **13** | LONDON | Mayfair (east), Hanover Square, Savile Row, Ro... | Westminster | 2696 | W1S | 51.51090 | -0.14086 |
| **14** | LONDON | Fitzrovia, Tottenham Court Road | Camden | 2697 | W1T | 51.51980 | -0.13473 |
| **15** | LONDON | Marylebone | Westminster | 2698 | W1U | 51.51827 | -0.15209 |
| **16** | LONDON | Great Portland Street, Fitzrovia | Westminster | 2699 | W1W | 51.51897 | -0.13909 |
| **17** | LONDON | Paddington head district: Paddington, Bayswate... | Kensington and Chelsea, Westminster | 2700 | W2 | 51.51508 | -0.17816 |

| | Post town | Coverage | Local authority area | id | Postcode | Latitude | Longitude |
|---|---|---|---|---|---|---|---|
| **18** | LONDON | Acton district: Acton, West Acton, North Acton... | Ealing, Hammersmith and Fulham, Hounslow | 2701 | W3 | 51.50925 | -0.26775 |
| **19** | LONDON | Chiswick district: Chiswick, Gunnersbury, Turn... | Ealing, Hammersmith and Fulham, Hounslow | 2702 | W4 | 51.49118 | -0.26268 |
| **20** | LONDON | Ealing district: Ealing, South Ealing, Ealing ... | Ealing, Hounslow | 2703 | W5 | 51.51243 | -0.30078 |
| **21** | LONDON | Hammersmith district: Fulham, Hammersmith, Rav... | Hammersmith and Fulham, Hounslow | 2704 | W6 | 51.49246 | -0.22805 |
| **22** | LONDON | Hanwell district: Hanwell, Boston Manor (part) | Ealing, Hounslow | 2705 | W7 | 51.51110 | -0.33398 |
| **23** | LONDON | Kensington district: Kensington, Holland Park ... | Kensington and Chelsea | 2706 | W8 | 51.50003 | -0.19317 |
| **24** | LONDON | Maida Hill district: Maida Hill, Maida Vale, L... | Brent, Camden, Westminster | 2707 | W9 | 51.52607 | -0.19070 |

Entrée [11]:

```
address = 'London, UK'

geo_locator = Nominatim()
location = geo_locator.geocode(address)
latitude_loc = location.latitude
longitude_loc = location.longitude
print('The geograpical coordinate of London City: {}, {}.'.format(latitude_loc, longitude_l
```

```
C:\Users\VuNHA\anaconda3\lib\site-packages\ipykernel_launcher.py:3: Deprecat
ionWarning: Using Nominatim with the default "geopy/1.21.0" `user_agent` is
strongly discouraged, as it violates Nominatim's ToS https://operations.osmf
oundation.org/policies/nominatim/ (https://operations.osmfoundation.org/poli
cies/nominatim/) and may possibly cause 403 and 429 HTTP errors. Please spec
ify a custom `user_agent` with `Nominatim(user_agent="my-application")` or b
y overriding the default `user_agent`: `geopy.geocoders.options.default_user
_agent = "my-application"`. In geopy 2.0 this will become an exception.
  This is separate from the ipykernel package so we can avoid doing imports
 until

The geograpical coordinate of London City: 51.5073219, -0.1276474.
```

## Visualizing data

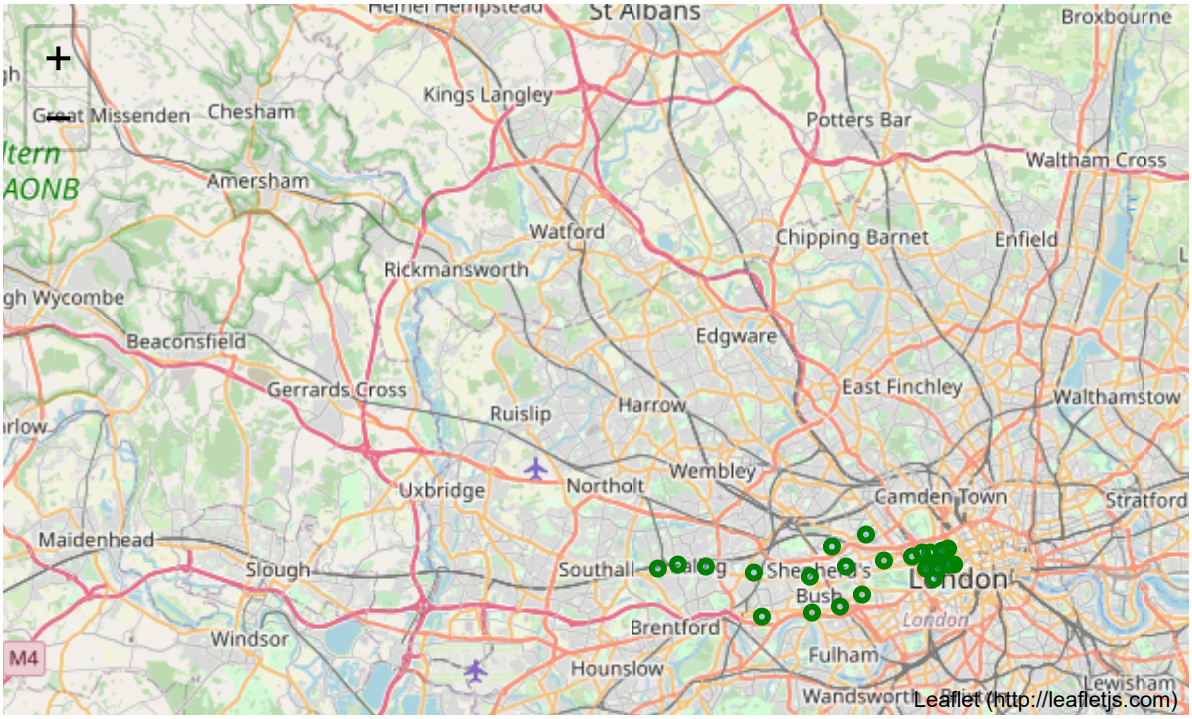
Entrée [12]:

```
# Create the map of London City by using latitude and longitude values
london_map = folium.Map(location=[latitude_loc, longitude_loc], zoom_start=10)

# add markers to map
for lat, lng, cov, local in zip(df_result['Latitude'], df_result['Longitude'], df_result['C
    label = '{}, {}'.format(cov, local)
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=3,
        popup=label,
        color='green',
        fill=True,
        fill_color='#3199cc',
        fill_opacity=0.3,
        parse_html=False).add_to(london_map)

london_map
```

Out[12]:




## Exploring The Neighbourhoods of Western and Paddington, London

### Defining Foursquare Credentials and Version

Entrée [13]:

```
CLIENT_ID = 'JGGBRN5XODTLZGJOMCSWIQMRH1JLGJKPSFR10XNB2R5U25GR' # Foursquare ID
CLIENT_SECRET = 'KWRAMLK2HOJBQ2XLICLKXRU3M4HOCC1U2VG4Y4OPP5JF03QX' # Foursquare Secret
VERSION = '20180605' # Foursquare API version

print('Your credentails:')
print('CLIENT_ID: ' + CLIENT_ID)
print('CLIENT_SECRET:' + CLIENT_SECRET)
```

```
Your credentails:
CLIENT_ID: JGGBRN5XODTLZGJOMCSWIQMRH1JLGJKPSFR10XNB2R5U25GR
CLIENT_SECRET:KWRAMLK2HOJBQ2XLICLKXRU3M4HOCC1U2VG4Y4OPP5JF03QX
```

## Just Selecting the Neighbourhoods of Western and Paddington, London

Entrée [14]:

```
df_t4 = df_result[df_result['Local authority area'].str.contains('Westminster')]

to_data = df_t4.reset_index(drop=True)
to_data
```

Out[14]:

| | Post town | Coverage | Local authority area | id | Postcode | Latitude | Longitude |
|---|---|---|---|---|---|---|---|
| **0** | LONDON | North Kensington district: North Kensington, K... | Brent, Hammersmith and Fulham, Kensington and ... | 2682 | W10 | 51.52103 | -0.21397 |
| **1** | LONDON | Notting Hill district: Notting Hill, Ladbroke ... | Hammersmith and Fulham, Kensington and Chelsea... | 2683 | W11 | 51.51189 | -0.20424 |
| **2** | LONDON | Portland Place, Regent Street | Westminster | 2687 | W1B | 51.51357 | -0.13931 |
| **3** | LONDON | Oxford Street (west) | Westminster | 2688 | W1C | 51.51371 | -0.14795 |
| **4** | LONDON | Soho (south east); Chinatown, Soho Square | Westminster | 2689 | W1D | 51.51344 | -0.13066 |
| **5** | LONDON | Soho (north west) | Westminster | 2690 | W1F | 51.51261 | -0.13502 |
| **6** | LONDON | Harley Street | Westminster | 2691 | W1G | 51.51818 | -0.14633 |
| **7** | LONDON | Marylebone | Westminster | 2692 | W1H | 51.51659 | -0.15936 |
| **8** | LONDON | Mayfair (south), Piccadilly | Westminster | 2693 | W1J | 51.50735 | -0.14388 |
| **9** | LONDON | Mayfair (north), Grosvenor Square | Westminster | 2694 | W1K | 51.51104 | -0.14950 |
| **10** | LONDON | Mayfair (east), Hanover Square, Savile Row, Ro... | Westminster | 2696 | W1S | 51.51090 | -0.14086 |
| **11** | LONDON | Marylebone | Westminster | 2698 | W1U | 51.51827 | -0.15209 |
| **12** | LONDON | Great Portland Street, Fitzrovia | Westminster | 2699 | W1W | 51.51897 | -0.13909 |
| **13** | LONDON | Paddington head district: Paddington, Bayswate... | Kensington and Chelsea, Westminster | 2700 | W2 | 51.51508 | -0.17816 |
| **14** | LONDON | Maida Hill district: Maida Hill, Maida Vale, L... | Brent, Camden, Westminster | 2707 | W9 | 51.52607 | -0.19070 |

## Just for the map of Western and Paddington, London Neighbourhoods

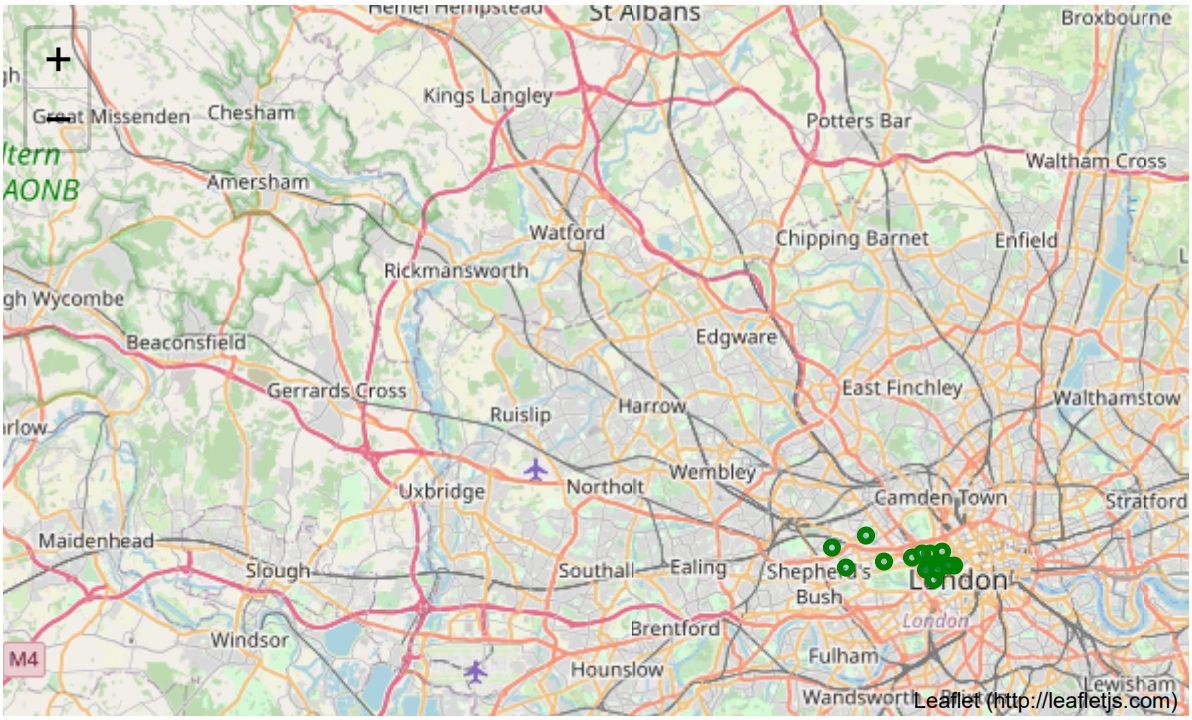
Entrée [15]:

```
# Creating the map of Western and Paddington, London by using latitude and longitude values
nhood_map = folium.Map(location=[latitude_loc, longitude_loc], zoom_start=10)

# adding markers to mAP
for lat, lng, cov, local in zip(to_data['Latitude'], to_data['Longitude'], to_data['Coverag
    label = '{}, {}'.format(cov, local)
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=3,
        popup=label,
        color='green',
        fill=True,
        fill_color='#3199cc',
        fill_opacity=0.3,
        parse_html=False).add_to(nhood_map)

nhood_map
```

Out[15]:




## For the first neighbourhood

Entrée [16]:

```
to_data.loc[0, 'Local authority area']
```

Out[16]:

```
'Brent, Hammersmith and Fulham, Kensington and Chelsea, Westminster'
```

## For the neighbourhood long and lat values

Entrée [17]:

```
neighbourhood_latitude = to_data.loc[0, 'Latitude'] # neighbourhood latitude value
neighbourhood_longitude = to_data.loc[0, 'Longitude'] # neighbourhood longitude value

neighbourhood_name = to_data.loc[0, 'Local authority area'] # neighbourhood name

print('Latitude and longitude values of {} are {}, {}.'.format(neighbourhood_name,
                                                               neighbourhood_latitude,
                                                               neighbourhood_longitude))
```

```
Latitude and longitude values of Brent, Hammersmith and Fulham, Kensington a
nd Chelsea, Westminster are 51.521029999999996, -0.21397.
```

## For the top 100 venus within a radius from the centroid of 500 meters

Entrée [18]:

```
LIMIT = 100
radius = 500

url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={}&v={}&ll=
    CLIENT_ID,
    CLIENT_SECRET,
    VERSION,
    neighbourhood_latitude,
    neighbourhood_longitude,
    radius,
    LIMIT)
url
```

Out[18]:

```
'https://api.foursquare.com/v2/venues/explore?&client_id=JGGBRN5XODTLZGJOMCS
WIQMRH1JLGJKPSFR10XNB2R5U25GR&client_secret=KWRAMLK2HOJBQ2XLICLKXRU3M4HOCC1U
2VG4Y4OPP5JF03QX&v=20180605&ll=51.521029999999996,-0.21397&radius=500&limit=
100'
```

Entrée [19]:

```
results = requests.get(url).json()
results
```

Out[19]:


```
{'meta': {'code': 200, 'requestId': '5ea12526b57e88001bd99e69'},
 'response': {'headerLocation': 'St. Charles',
  'headerFullLocation': 'St. Charles, London',
  'headerLocationGranularity': 'neighborhood',
  'totalResults': 33,
  'suggestedBounds': {'ne': {'lat': 51.5255300045,
    'lng': -0.20675141258254473},
   'sw': {'lat': 51.516529995499994, 'lng': -0.22118858741745526}},
  'groups': [{'type': 'Recommended Places',
    'name': 'recommended',
    'items': [{'reasons': {'count': 0,
       'items': [{'summary': 'This spot is popular',
         'type': 'general',
         'reasonName': 'globalInteractionReason'}]},
      'venue': {'id': '4c188695834e2d7f63ff2880',
       'name': 'Lowry & Baker',
       'location': {'address': '339 Portobello Rd.',
```


Entrée [20]:

```
# function that extracts the category of the venue
def get_category_type(row):
    try:
        categories_list = row['categories']
    except:
        categories_list = row['venue.categories']

    if len(categories_list) == 0:
        return None
    else:
        return categories_list[0]['name']
```

## Cleaning json and structuring into a pandas dataframe

Entrée [21]:

```
venues = results['response']['groups'][0]['items']

nearby_venues = json_normalize(venues) # flatten JSON

# filtering columns
filtered_columns = ['venue.name', 'venue.categories', 'venue.location.lat', 'venue.location
nearby_venues =nearby_venues.loc[:, filtered_columns]

# filtering the category for each row
nearby_venues['venue.categories'] = nearby_venues.apply(get_category_type, axis=1)

# cleaning columns
nearby_venues.columns = [col.split(".")[-1] for col in nearby_venues.columns]

nearby_venues.head()
```

```
C:\Users\VuNHA\anaconda3\lib\site-packages\ipykernel_launcher.py:3: FutureWa
rning: pandas.io.json.json_normalize is deprecated, use pandas.json_normaliz
e instead
  This is separate from the ipykernel package so we can avoid doing imports
until
```

Out[21]:

| | name | categories | lat | lng |
|---|---|---|---|---|
| **0** | Lowry & Baker | Café | 51.521356 | -0.210165 |
| **1** | The Eagle | Pub | 51.522233 | -0.212528 |
| **2** | Georges Portobello Fish Bar | Fish & Chips Shop | 51.521189 | -0.209813 |
| **3** | Pizza East | Pizza Place | 51.521041 | -0.209452 |
| **4** | 333 HotShoe | Coffee Shop | 51.521203 | -0.209837 |

## How many venus were returned from Foursquare?

Entrée [22]:

```
print('{} venues were returned by Foursquare.'.format(nearby_venues.shape[0]))
```

```
33 venues were returned by Foursquare.
```

### Look at nearby venues

Entrée [23]:

```
def getNearbyVenues(names, latitudes, longitudes, radius=1000, LIMIT=100):

    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={}&
            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']['id'],
            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 = ['Location',
                  'Location Latitude',
                  'Location Longitude',
                  'Venue',
                  'Venue id',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category'
                   ]

    return(nearby_venues)
```

Entrée [24]:

```
paddington_data_venues = getNearbyVenues(names=df_result['Coverage'],
                                         latitudes=df_result['Latitude'],
                                         longitudes=df_result['Longitude']
                                        )
```

```
North Kensington district: North Kensington, Kensal Town, Ladbroke Grove (no
rth), Queen's Park (part)
Notting Hill district: Notting Hill, Ladbroke Grove (south), Holland Park (p
art)
Shepherds Bush district: Shepherds Bush, White City, Wormwood Scrubs, East A
cton (east)
West Ealing district: West Ealing, Northfields (north and west)
West Kensington district: West Kensington, Kensington Olympia, Holland Park
Portland Place, Regent Street
Oxford Street (west)
Soho (south east); Chinatown, Soho Square
Soho (north west)
Harley Street
Marylebone
Mayfair (south), Piccadilly
Mayfair (north), Grosvenor Square
Mayfair (east), Hanover Square, Savile Row, Royal Academy
Fitzrovia, Tottenham Court Road
Marylebone
Great Portland Street, Fitzrovia
Paddington head district: Paddington, Bayswater, Hyde Park, Westbourne Gree
n, Little Venice (part), Notting Hill (part)
Acton district: Acton, West Acton, North Acton (part), South Acton, East Act
on (west), Park Royal (south), Hanger Hill Garden Estate, Gunnersbury Park
Chiswick district: Chiswick, Gunnersbury, Turnham Green, Acton Green, South
Acton (part), Bedford Park
Ealing district: Ealing, South Ealing, Ealing Common, North Ealing, Northfie
lds, (south and east), Pitshanger, Hanger Lane
Hammersmith district: Fulham, Hammersmith, Ravenscourt Park, Stamford Brook
(part)
Hanwell district: Hanwell, Boston Manor (part)
Kensington district: Kensington, Holland Park (part)
Maida Hill district: Maida Hill, Maida Vale, Little Venice (part)
```

Entrée [26]:

```
print(paddington_data_venues.shape)
paddington_data_venues
```

```
(2322, 8)
```

Entrée [27]:

```
print('There are {} unique venue categories.'.format(len(paddington_data_venues['Venue Cate
```

```
There are 217 unique venue categories.
```

Entrée [30]:

```
print('There are {} unique venues.'.format(len(paddington_data_venues['Venue id'].unique())
```

```
There are 1380 unique venues.
```

Entrée [32]:

```
univen = paddington_data_venues.groupby('Location').nunique('Venue Category')
univen
```

Out[32]:

| Location | Location | Location Latitude | Location Longitude | Venue | Venue id | Venue Latitude | Venue Longitude | Venue Category |
|---|---|---|---|---|---|---|---|---|
| **Acton district: Acton, West Acton, North Acton (part), South Acton, East Acton (west), Park Royal (south), Hanger Hill Garden Estate, Gunnersbury Park** | 1 | 1 | 1 | 42 | 43 | 43 | 43 | 27 |
| **Chiswick district: Chiswick, Gunnersbury, Turnham Green, Acton Green, South Acton (part), Bedford Park** | 1 | 1 | 1 | 100 | 100 | 100 | 100 | 55 |
| **Ealing district: Ealing, South Ealing, Ealing Common, North Ealing, Northfields, (south and east), Pitshanger, Hanger Lane** | 1 | 1 | 1 | 96 | 100 | 100 | 100 | 51 |
| **Fitzrovia, Tottenham Court Road** | 1 | 1 | 1 | 98 | 100 | 100 | 100 | 55 |
| **Great Portland Street, Fitzrovia** | 1 | 1 | 1 | 97 | 100 | 100 | 100 | 61 |
| **Hammersmith district: Fulham, Hammersmith, Ravenscourt Park, Stamford Brook (part)** | 1 | 1 | 1 | 97 | 100 | 100 | 100 | 51 |
| **Hanwell district: Hanwell, Boston Manor (part)** | 1 | 1 | 1 | 38 | 40 | 40 | 40 | 25 |
| **Harley Street** | 1 | 1 | 1 | 94 | 100 | 100 | 100 | 59 |
| **Kensington district: Kensington, Holland Park (part)** | 1 | 1 | 1 | 97 | 100 | 100 | 100 | 51 |

| Location | Location | Location Latitude | Location Longitude | Venue | Venue id | Venue Latitude | Venue Longitude | Venue Category |
|---|---|---|---|---|---|---|---|---|
| **Maida Hill district: Maida Hill, Maida Vale, Little Venice (part)** | 1 | 1 | 1 | 86 | 86 | 86 | 86 | 47 |
| **Marylebone** | 1 | 2 | 2 | 127 | 136 | 136 | 136 | 73 |
| **Mayfair (east), Hanover Square, Savile Row, Royal Academy** | 1 | 1 | 1 | 99 | 100 | 100 | 100 | 56 |
| **Mayfair (north), Grosvenor Square** | 1 | 1 | 1 | 95 | 100 | 100 | 100 | 48 |
| **Mayfair (south), Piccadilly** | 1 | 1 | 1 | 100 | 100 | 100 | 100 | 50 |
| **North Kensington district: North Kensington, Kensal Town, Ladbroke Grove (north), Queen's Park (part)** | 1 | 1 | 1 | 84 | 84 | 84 | 84 | 51 |
| **Notting Hill district: Notting Hill, Ladbroke Grove (south), Holland Park (part)** | 1 | 1 | 1 | 100 | 100 | 100 | 100 | 57 |
| **Oxford Street (west)** | 1 | 1 | 1 | 94 | 100 | 100 | 100 | 48 |
| **Paddington head district: Paddington, Bayswater, Hyde Park, Westbourne Green, Little Venice (part), Notting Hill (part)** | 1 | 1 | 1 | 98 | 100 | 100 | 100 | 58 |
| **Portland Place, Regent Street** | 1 | 1 | 1 | 98 | 100 | 100 | 100 | 55 |
| **Shepherds Bush district: Shepherds Bush, White City, Wormwood Scrubs, East Acton (east)** | 1 | 1 | 1 | 100 | 100 | 100 | 100 | 64 |
| **Soho (north west)** | 1 | 1 | 1 | 99 | 100 | 100 | 100 | 58 |
| **Soho (south east); Chinatown, Soho Square** | 1 | 1 | 1 | 96 | 100 | 100 | 100 | 60 |

| Location | Location Latitude | Location Longitude | Venue | Venue id | Venue Latitude | Venue Longitude | Venue Category |
|---|---|---|---|---|---|---|---|
| **Location** | | | | | | | |
| **West Ealing district: West Ealing, Northfields (north and west)** | 1 | 1 | 1 | 69 | 69 | 69 | 69 | 43 |
| **West Kensington district: West Kensington, Kensington Olympia, Holland Park** | 1 | 1 | 1 | 99 | 100 | 100 | 100 | 51 |

Entrée [33]:

```
paddington_data_venues.groupby('Venue Category').nunique()
```

Out[33]:

| Venue Category | Location | Location Latitude | Location Longitude | Venue | Venue id | Venue Latitude | Venue Longitude | Venue Category |
|---|---|---|---|---|---|---|---|---|
| **African Restaurant** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **American Restaurant** | 5 | 6 | 6 | 4 | 4 | 4 | 4 | 1 |
| **Antique Shop** | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 |
| **Argentinian Restaurant** | 6 | 7 | 7 | 4 | 4 | 4 | 4 | 1 |
| **Art Gallery** | 15 | 16 | 16 | 12 | 12 | 12 | 12 | 1 |
| **Art Museum** | 7 | 7 | 7 | 3 | 3 | 3 | 3 | 1 |

### Analyze each Location

Entrée [34]:

```
# one hot encoding
paddington_onehot = pd.get_dummies(paddington_data_venues[['Venue Category']], prefix="", p

# add neighbourhood column back to dataframe
paddington_onehot['Location'] = paddington_data_venues['Location']

# move neighbourhood column to the first column
fixed_columns = [paddington_onehot.columns[-1]] + list(paddington_onehot.columns[:-1])
paddington_onehot = paddington_onehot[fixed_columns]

paddington_onehot.head()
```

Out[34]:

| | Location | African Restaurant | American Restaurant | Antique Shop | Argentinian Restaurant | Art Gallery | Art Museum | Arts & Crafts Store | Asi Restaura |
|---|---|---|---|---|---|---|---|---|---|
| **0** | North Kensington district: North Kensington, K... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| **1** | North Kensington district: North Kensington, K... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| **2** | North Kensington district: North Kensington, K... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| **3** | North Kensington district: North Kensington, K... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| **4** | North Kensington district: North Kensington, K... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

Entrée [35]:

```
paddington_onehot.shape
```

Out[35]:

```
(2322, 218)
```

**Group rows by location and by the mean of the frequency of occurrence of each category**

Entrée [36]:

```
paddington_grouped = paddington_onehot.groupby('Location').mean().reset_index()
paddington_grouped
```

Out[36]:

| | Location | African Restaurant | American Restaurant | Antique Shop | Argentinian Restaurant | Art Gallery | Art Museum | Arts & Crafts Store | Res |
|---|---|---|---|---|---|---|---|---|---|
| 0 | Acton district: Acton, West Acton, North Acton... | 0.00 | 0.000 | 0.00 | 0.00 | 0.000000 | 0.00 | 0.00 | 0 |
| 1 | Chiswick district: Chiswick, Gunnersbury, Turn... | 0.00 | 0.000 | 0.01 | 0.01 | 0.000000 | 0.01 | 0.00 | 0 |
| 2 | Ealing district: Ealing, South Ealing, Ealing ... | 0.00 | 0.000 | 0.00 | 0.00 | 0.010000 | 0.00 | 0.00 | 0 |
| 3 | Fitzrovia, Tottenham Court Road | 0.00 | 0.010 | 0.00 | 0.00 | 0.010000 | 0.00 | 0.02 | 0 |
| 4 | Great Portland Street, Fitzrovia | 0.00 | 0.010 | 0.00 | 0.00 | 0.020000 | 0.00 | 0.01 | 0 |
| 5 | Hammersmith district: Fulham, Hammersmith, Rav... | 0.00 | 0.000 | 0.00 | 0.00 | 0.000000 | 0.00 | 0.00 | 0 |
| 6 | Hanwell district: Hanwell, Boston Manor (part) | 0.00 | 0.000 | 0.00 | 0.00 | 0.000000 | 0.00 | 0.00 | 0 |
| 7 | Harley Street | 0.00 | 0.010 | 0.00 | 0.01 | 0.030000 | 0.00 | 0.00 | 0 |
| 8 | Kensington district: Kensington, Holland Park ... | 0.00 | 0.000 | 0.00 | 0.00 | 0.010000 | 0.00 | 0.00 | 0 |
| 9 | Maida Hill district: Maida Hill, Maida Vale, L... | 0.00 | 0.000 | 0.00 | 0.00 | 0.000000 | 0.00 | 0.00 | 0 |
| 10 | Marylebone | 0.00 | 0.015 | 0.00 | 0.02 | 0.015000 | 0.00 | 0.00 | 0 |
| 11 | Mayfair (east), Hanover Square, Savile Row, Ro... | 0.00 | 0.000 | 0.00 | 0.00 | 0.050000 | 0.01 | 0.01 | 0 |
| 12 | Mayfair (north), Grosvenor Square | 0.00 | 0.000 | 0.00 | 0.01 | 0.050000 | 0.01 | 0.00 | 0 |

| | Location | African Restaurant | American Restaurant | Antique Shop | Argentinian Restaurant | Art Gallery | Art Museum | Arts & Crafts Store | Res |
|---|---|---|---|---|---|---|---|---|---|
| **13** | Mayfair (south), Piccadilly | 0.00 | 0.010 | 0.00 | 0.00 | 0.040000 | 0.01 | 0.00 | 0 |
| **14** | North Kensington district: North Kensington, K... | 0.00 | 0.000 | 0.00 | 0.00 | 0.000000 | 0.00 | 0.00 | 0 |
| **15** | Notting Hill district: Notting Hill, Ladbroke ... | 0.00 | 0.000 | 0.01 | 0.00 | 0.020000 | 0.00 | 0.00 | 0 |
| **16** | Oxford Street (west) | 0.00 | 0.000 | 0.00 | 0.01 | 0.060000 | 0.00 | 0.00 | 0 |
| **17** | Paddington head district: Paddington, Bayswate... | 0.00 | 0.000 | 0.00 | 0.02 | 0.000000 | 0.00 | 0.00 | 0 |
| **18** | Portland Place, Regent Street | 0.00 | 0.000 | 0.00 | 0.00 | 0.050000 | 0.01 | 0.01 | 0 |
| **19** | Shepherds Bush district: Shepherds Bush, White... | 0.01 | 0.000 | 0.00 | 0.00 | 0.000000 | 0.00 | 0.00 | 0 |
| **20** | Soho (north west) | 0.00 | 0.000 | 0.00 | 0.00 | 0.040000 | 0.02 | 0.01 | 0 |
| **21** | Soho (south east); Chinatown, Soho Square | 0.00 | 0.000 | 0.00 | 0.00 | 0.010000 | 0.01 | 0.01 | 0 |
| **22** | West Ealing district: West Ealing, Northfields... | 0.00 | 0.000 | 0.00 | 0.00 | 0.014493 | 0.00 | 0.00 | 0 |
| **23** | West Kensington district: West Kensington, Ken... | 0.00 | 0.000 | 0.00 | 0.00 | 0.000000 | 0.00 | 0.00 | 0 |

Entrée [37]:

```
paddington_grouped.shape
```

Out[37]:

```
(24, 218)
```

**Top 5 most common venues of each Location**

Entrée [38]:

```
num_top_venues = 5

for hood in paddington_grouped['Location']:
    print("----"+hood+"----")
    temp = paddington_grouped[paddington_grouped['Location'] == 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_ven
    print('\n')
```

```
----Soho (south east); Chinatown, Soho Square----
            venue  freq
0         Theater  0.09
1    Cocktail Bar  0.08
2           Hotel  0.03
3       Bookstore  0.03
4  Ice Cream Shop  0.03


----West Ealing district: West Ealing, Northfields (north and west)----
           venue  freq
0    Coffee Shop  0.07
1          Hotel  0.06
2            Pub  0.04
3  Grocery Store  0.04
4           Park  0.04


----West Kensington district: West Kensington, Kensington Olympia, Holland
```

**Put into a pandas dataframe**

Entrée [39]:

```
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]
```

Entrée [41]:

```
num_top_venues = 10

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

# create columns according to number of top venues
columns = ['Location']
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
location_venues_sorted = pd.DataFrame(columns=columns)
location_venues_sorted['Location'] = paddington_grouped['Location']

for ind in np.arange(paddington_grouped.shape[0]):
    location_venues_sorted.iloc[ind, 1:] = return_most_common_venues(paddington_grouped.ilo

location_venues_sorted
```

Out[41]:

| | Location | 1st Most Common Venue | 2nd Most Common Venue | 3rd Most Common Venue | 4th Most Common Venue | 5th Most Common Venue | 6th Most Common Venue | 7 C |
|---|---|---|---|---|---|---|---|---|
| 0 | Acton district: Acton, West Acton, North Acton... | Pub | Gym / Fitness Center | Grocery Store | Café | Park | Train Station | Fa Re |
| 1 | Chiswick district: Chiswick, Gunnersbury, Turn... | Pub | Café | Coffee Shop | Bakery | Park | Burger Joint | Re |
| 2 | Ealing district: Ealing, South Ealing, Ealing ... | Coffee Shop | Pub | Hotel | Italian Restaurant | Park | Burger Joint | |
| 3 | Fitzrovia, Tottenham Court Road | Coffee Shop | Cocktail Bar | Hotel | Pizza Place | French Restaurant | Theater | |
| 4 | Great Portland Street, Fitzrovia | Coffee Shop | Pizza Place | Cocktail Bar | Hotel | Wine Bar | French Restaurant | |
| 5 | Hammersmith district: Fulham, Hammersmith, Rav... | Pub | Coffee Shop | Italian Restaurant | Café | Park | Sandwich Place | |
| 6 | Hanwell district: Hanwell, Boston Manor (part) | Coffee Shop | Pub | Café | Supermarket | Hotel | Fast Food Restaurant | |
| 7 | Harley Street | Coffee Shop | Hotel | Burger Joint | Cocktail Bar | French Restaurant | Café | Art |

| | Location | 1st Most Common Venue | 2nd Most Common Venue | 3rd Most Common Venue | 4th Most Common Venue | 5th Most Common Venue | 6th Most Common Venue | 7 C |
|---|---|---|---|---|---|---|---|---|
| **8** | Kensington district: Kensington, Holland Park ... | Hotel | Pub | Italian Restaurant | Café | Thai Restaurant | Bakery | Re |
| **9** | Maida Hill district: Maida Hill, Maida Vale, L... | Pub | Café | Coffee Shop | Grocery Store | Pizza Place | Garden | |
| **10** | Marylebone | Hotel | Juice Bar | Café | Sandwich Place | Bakery | Burger Joint | H |
| **11** | Mayfair (east), Hanover Square, Savile Row, Ro... | Hotel | Art Gallery | Cocktail Bar | Lounge | Indian Restaurant | Clothing Store | E |
| **12** | Mayfair (north), Grosvenor Square | Hotel | French Restaurant | Clothing Store | Hotel Bar | Art Gallery | Boutique | Re |
| **13** | Mayfair (south), Piccadilly | Hotel | Cocktail Bar | Hotel Bar | French Restaurant | Lounge | Art Gallery | |
| **14** | North Kensington district: North Kensington, K... | Pub | Gym / Fitness Center | Café | Italian Restaurant | Bakery | Park | |
| **15** | Notting Hill district: Notting Hill, Ladbroke ... | Italian Restaurant | Pub | Bakery | Gym / Fitness Center | Pizza Place | Bookstore | |
| **16** | Oxford Street (west) | Hotel | French Restaurant | Hotel Bar | Art Gallery | Clothing Store | Juice Bar | |
| **17** | Paddington head district: Paddington, Bayswate... | Garden | Hotel | Pub | Café | Coffee Shop | Greek Restaurant | Re |
| **18** | Portland Place, Regent Street | Cocktail Bar | Hotel | Art Gallery | Coffee Shop | Theater | Clothing Store | Bo |
| **19** | Shepherds Bush district: Shepherds Bush, White... | Clothing Store | Middle Eastern Restaurant | Chinese Restaurant | Cosmetics Shop | Café | Coffee Shop | Re |
| **20** | Soho (north west) | Theater | Cocktail Bar | Art Gallery | Bookstore | Hotel | Coffee Shop | |
| **21** | Soho (south east); Chinatown, Soho Square | Theater | Cocktail Bar | Steakhouse | Bookstore | Hotel | Ice Cream Shop | |
| **22** | West Ealing district: West Ealing, Northfields... | Coffee Shop | Hotel | Bakery | Park | Grocery Store | Pub | Re |

| | Location | 1st Most Common Venue | 2nd Most Common Venue | 3rd Most Common Venue | 4th Most Common Venue | 5th Most Common Venue | 6th Most Common Venue | 7 C |
|---|---|---|---|---|---|---|---|---|
| **23** | West Kensington district: West Kensington, Ken... | Pub | Café | Italian Restaurant | Hotel | Grocery Store | Persian Restaurant | Re |

## Clustering Paddington Locations

### Run k-means to cluster Locations into 5 clusters

Entrée [45]:

```
# set number of clusters
kclusters = 5

paddington_grouped_clustering = paddington_grouped.drop('Location', 1)

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

# check cluster labels generated for each row in the dataframe
kmeans.labels_[0:24]
```

Out[45]:

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

### Creating a new dataframe including the cluster as well as the top 10 venues for each Location

Entrée [ ]:

```
paddington_merged = df_result

# add clustering labels
paddington_merged['Cluster Labels'] = kmeans.labels_

# merge paddington_grouped with df_result to add latitude/longitude for each location
paddington_merged = paddington_merged.join(location_venues_sorted.set_index('Location'), on

paddington_merged # check the last columns!
```

Entrée [ ]:

```
# 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(paddington_merged['Latitude'], paddington_merged['Longitu
    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_opacity=0.7).add_to(map_clusters)
map_clusters
```