In [ ]:

```
try:
    print("Importing libraries...\n")
    from progressbar import ProgressBar
    from bs4 import BeautifulSoup as bts # library for web scraping
    import numpy as np # library to handle data in a vectorized manner
    import pandas as pd # library for data analysis
    from pandas.io.json import json_normalize
    import matplotlib.cm as cm
    import matplotlib.colors as colors
    import requests # library to handle requests
    from geopy.geocoders import Nominatim # convert an address into latitude and longitude values
    import matplotlib as mp # library for visualization
    from sklearn.cluster import KMeans # import k-means from clustering stage
    from geopy.geocoders import Nominatim # convert an address into latitude and longitude values
    import folium # map rendering library
    import lxml
    import re
    from time import sleep

    from matplotlib import pyplot as plt
    from matplotlib.pyplot import figure

    import datetime
    import dateutil
    print("All libraries imported successfully!\n")
except:
    print("ERROR: Could not import all libraries!\n")

%matplotlib inline
```

Set Up

Get London coordinates for map visualisations

In [249…

```
address = 'London'

geolocator = Nominatim(user_agent="ldn_explorer")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print('The geograpical coordinate of London are {}, {}.'.format(latitude, longitude))
```

```
The geograpical coordinate of London are 51.5073219, -0.1276474.
```

Retrieve transformed crime data. Due to limitations with the ward level data noted in data exploration, it is unlikely to be of muvh use, but it will be used for visualisations just in case there is sufficient to show a pattern.

In [250…

```
adjCrimeByWardMinor = pd.read_csv('PopAdjCrimeByWardGranularClass.csv')
adjCrimeByWardMajor = pd.read_csv('PopAdjCrimeByWard.csv')

adjCrimeByWardMajor.set_index('Ward', inplace=True)
adjCrimeByWardMinor.set_index('Ward', inplace=True)
```

In [251…

```
adjCrimeByBoroughMinor = pd.read_csv('PopAdjCrimeByBoroughGranularClass.csv')
adjCrimeByBoroughMajor = pd.read_csv('PopAdjCrimeByBorough.csv')

adjCrimeByBoroughMajor.set_index('Borough', inplace=True)
adjCrimeByBoroughMinor.set_index('Borough', inplace=True)
```

In [252…

```
adjCrimeByBoroughMajor.head()
```

Out[252…

Borough	Arson and Criminal Damage	Burglary	Drug Offences	Miscellaneous Crimes Against Society	Possession of Weapons	Public Order Offences	Robbery	Sexual Offences	Theft	Vehicle Offences	Violence Against the Person	Total Crime
Barking and Dagenham	129.282297	123.062201	142.296651	31.674641	15.885167	115.215311	75.071770	61.291866	312.870813	233.014354	616.650718	1856.315713
Barnet	99.050308	151.360370	55.261807	19.070842	7.802875	96.406571	50.462012	29.774127	275.205339	290.143737	387.089322	1461.627269
Bexley	121.080639	92.181744	67.785510	20.343840	9.496521	101.473598	24.928367	31.354892	190.912812	217.355710	430.904625	1307.818256
Brent	122.794339	132.339657	127.311051	23.456790	15.688046	123.366456	62.451069	36.916591	298.705209	252.363746	533.514002	1728.906908
Bromley	112.991766	119.518146	74.290942	20.646539	9.911558	100.365965	28.209820	31.381519	265.233303	236.992986	394.510522	1394.053114

In [253…

```
adjCrimeByBoroughMinor.head()
```

Out[253…

Borough	Absconding from Lawful Custody	Aggravated Vehicle Taking	Arson	Bail Offences	Bicycle Theft	Burglary - Business and Community	Burglary - Residential	Criminal Damage	Dangerous Driving	Disclosure, Obstruction, False or Misleading State	...	Robbery of Personal Property	Shoplifting	Theft from Person	Theft from a Motor Vehicle	Theft or Taking of a Motor Vehicle	Threat or Possession With Intent to Commit Crimina	Violence with Injury	Violenc withou Injur
Barking and Dagenham	0.000000	2.966507	5.454545	0.047847	16.746411	28.373206	94.688995	123.827751	1.483254	0.047847	...	69.090909	77.894737	48.516746	102.631579	102.392344	6.746411	190.813397	425.69378
Barnet	0.000000	1.899384	2.926078	0.000000	14.091376	29.645791	121.714579	96.124230	0.590349	0.051335	...	45.251540	79.081109	31.801848	183.957906	67.864476	4.055441	114.784394	272.12525
Bexley	0.000000	2.496930	5.034793	0.000000	7.572657	21.203438	70.978305	116.045845	1.023332	0.081867	...	21.162505	63.487515	11.420385	122.513303	60.049120	6.999591	139.255014	291.44494
Brent	0.210780	2.439024	4.034929	0.030111	21.860885	26.136706	106.202951	118.759410	1.114122	0.060223	...	57.091238	62.210178	44.173442	148.479374	68.684131	5.088829	169.015357	363.89641
Bromley	0.030497	1.890820	6.312900	0.000000	9.210125	27.203416	92.314730	106.678866	1.006404	0.000000	...	23.360781	115.004575	21.164989	138.487344	58.706923	6.251906	125.282098	269.13693

5 rows × 47 columns

Analysis

In [276…

```
london_venues_with_data=pd.read_csv('cleaned_neighborhood_venue.csv')
london_venues_with_data.head()
```

Out[276…

	Unnamed: 0	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category	Neighborhood Geocodable
0	0	Abbey Wood	51.487621	0.11405	Lesnes Abbey	51.489526	0.125839	Historic Site	True
1	1	Abbey Wood	51.487621	0.11405	Dagenham Sunday Market	51.517026	0.111949	Flea Market	True
2	2	Abbey Wood	51.487621	0.11405	Morrisons	51.507276	0.105392	Supermarket	True
3	3	Abbey Wood	51.487621	0.11405	wilko	51.505596	0.103845	Furniture / Home Store	True

	Unnamed: 0	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category	Neighborhood Geocodable
4	4	Abbey Wood	51.487621	0.11405	Lidl	51.496152	0.118417	Supermarket	True

For this analysis let's focus on neigbourhood data retrieved by neighbourhood name

In [277…
```
london_venues_with_data=london_venues_with_data[london_venues_with_data['Neighborhood Geocodable']==True]
```

Observe there is a venue category "Neighborhood" as well as a column neighborhood

In [278…
```
london_venues_with_data[london_venues_with_data['Venue Category']=='Neighborhood']
```

Out[278…

	Unnamed: 0	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category	Neighborhood Geocodable
5938	6302	Covent Garden	51.512874	-0.122544	Seven Dials	51.513779	-0.126948	Neighborhood	True
12720	13355	Holborn	51.519598	-0.113727	Seven Dials	51.513779	-0.126948	Neighborhood	True

Pivot the data, to obtain columns for each venue category

In [279…
```
# one hot encoding
london_onehot = pd.get_dummies(london_venues_with_data[['Venue Category']], prefix="", prefix_sep="")

# add neighborhood column back to dataframe. Use UK spelling to distringuish from venue categoru
london_onehot['Neighbourhood'] = london_venues_with_data['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()
```

Out[279…

	Neighbourhood	ATM	Accessories Store	Afghan Restaurant	African Restaurant	Airport	Airport Lounge	Airport Service	Airport Terminal	American Restaurant	...	Windmill	Wine Bar	Wine Shop	Winery	Wings Joint	Women's Store	Xinjiang Restaurant	Yoga Studio	Zoo	Zoo Exhibit
0	Abbey Wood	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
1	Abbey Wood	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
2	Abbey Wood	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
3	Abbey Wood	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
4	Abbey Wood	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0

5 rows × 421 columns

In [280…
```
london_onehot.shape
```

Out[280… (24501, 421)

In [281…
```
london_onehot =london_onehot.drop_duplicates()
```

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

In [282…

```
london_grouped = london_onehot.groupby('Neighbourhood').mean().reset_index()
london_grouped.head(10)
```

Out[282…

	Neighbourhood	ATM	Accessories Store	Afghan Restaurant	African Restaurant	Airport	Airport Lounge	Airport Service	Airport Terminal	American Restaurant	...	Windmill	Wine Bar	Wine Shop	Winery	Wings Joint	Women's Store	Xinjiang Restaurant	Yoga Studio	Zoo	Zoo Exhibit
0	Abbey Wood	0.0	0.0	0.000000	0.000000	0.000000	0.0	0.0	0.0	0.000000	...	0.0	0.000000	0.000000	0.0	0.0	0.0	0.0	0.000000	0.0	0.0
1	Acton	0.0	0.0	0.000000	0.000000	0.000000	0.0	0.0	0.0	0.000000	...	0.0	0.000000	0.000000	0.0	0.0	0.0	0.0	0.000000	0.0	0.0
2	Addington	0.0	0.0	0.000000	0.027778	0.027778	0.0	0.0	0.0	0.000000	...	0.0	0.000000	0.000000	0.0	0.0	0.0	0.0	0.000000	0.0	0.0
3	Aldgate	0.0	0.0	0.000000	0.000000	0.000000	0.0	0.0	0.0	0.000000	...	0.0	0.020000	0.020000	0.0	0.0	0.0	0.0	0.000000	0.0	0.0
4	Aldwych	0.0	0.0	0.000000	0.000000	0.000000	0.0	0.0	0.0	0.016667	...	0.0	0.016667	0.016667	0.0	0.0	0.0	0.0	0.000000	0.0	0.0
5	Anerley	0.0	0.0	0.000000	0.000000	0.000000	0.0	0.0	0.0	0.000000	...	0.0	0.000000	0.000000	0.0	0.0	0.0	0.0	0.000000	0.0	0.0
6	Angel	0.0	0.0	0.015385	0.000000	0.000000	0.0	0.0	0.0	0.000000	...	0.0	0.000000	0.015385	0.0	0.0	0.0	0.0	0.015385	0.0	0.0
7	Arkley	0.0	0.0	0.025641	0.000000	0.000000	0.0	0.0	0.0	0.000000	...	0.0	0.000000	0.000000	0.0	0.0	0.0	0.0	0.000000	0.0	0.0
8	Balham	0.0	0.0	0.000000	0.000000	0.000000	0.0	0.0	0.0	0.000000	...	0.0	0.000000	0.000000	0.0	0.0	0.0	0.0	0.024390	0.0	0.0
9	Barbican	0.0	0.0	0.000000	0.000000	0.000000	0.0	0.0	0.0	0.000000	...	0.0	0.000000	0.000000	0.0	0.0	0.0	0.0	0.000000	0.0	0.0

10 rows × 421 columns

Define a function to work out the top venues categories for a neighbourhood.

In [283…

```
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]
```

Determine the top 10 venue categories for a neighbourhood

In [284…

```
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['Neighbourhood']

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_venues)

neighborhoods_venues_sorted
```

Out[284…

	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

	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
0	Abbey Wood	Supermarket	Warehouse Store	Gym / Fitness Center	Train Station	Grocery Store	Historic Site	Clothing Store	Mobile Phone Shop	Fast Food Restaurant	Flea Market
1	Acton	Fast Food Restaurant	Brewery	Train Station	Hookah Bar	Food & Drink Shop	Bakery	Middle Eastern Restaurant	Bar	Chinese Restaurant	Pastry Shop
2	Addington	Gaming Cafe	Forest	Gym / Fitness Center	Rugby Pitch	Mediterranean Restaurant	Tram Station	Train Station	Golf Course	Pharmacy	Sandwich Place
3	Aldgate	Garden	Speakeasy	Bakery	Mediterranean Restaurant	Hotel Bar	Hotel	Office	French Restaurant	Beer Bar	Middle Eastern Restaurant
4	Aldwych	Restaurant	Burger Joint	Gelato Shop	Opera House	Noodle House	Gym	Spanish Restaurant	Gym / Fitness Center	Steakhouse	Museum
...	...	...	...	...	...	...	...	...	...	...	...
320	Woodside Park	Grocery Store	Gym / Fitness Center	Gift Shop	Newsagent	Golf Course	Café	Gourmet Shop	Greek Restaurant	Burger Joint	Beer Bar
321	Woolwich	Burger Joint	Platform	Boat or Ferry	Hotel	Pier	Tunnel	Steakhouse	Sandwich Place	Bakery	Pharmacy
322	Worcester Park	Pizza Place	Steakhouse	English Restaurant	Pharmacy	Grocery Store	Supermarket	Coffee Shop	BBQ Joint	Train Station	Bus Stop
323	Yeading	Hotel	Café	Sandwich Place	Park	Coffee Shop	Pharmacy	Supermarket	Clothing Store	Mexican Restaurant	Military Base
324	Yiewsley	Burger Joint	Gym	Coffee Shop	Bar	Fast Food Restaurant	Theater	Fish & Chips Shop	Chinese Restaurant	Gym / Fitness Center	Bookstore

325 rows × 11 columns

Conduct k means clustering, to cluster neighbourhoods with similar venue profiles

In [285…

```
# set number of clusters
kclusters = 8

london_grouped_clustering = london_grouped.drop('Neighbourhood', 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]
```

Out[285… array([7, 5, 3, 2, 2, 4, 2, 1, 1, 2])

Retrieve list of London neighbourhoods and their coordinates, to enable the clusters to be plotted

In [286…

```
wiki_df=pd.read_csv('london_neighbourhoods.csv')
```

In [287…

```
wiki_df['Neighborhood']= wiki_df['Location']
wiki_df.head()
```

Out[287…

	Unnamed: 0	Unnamed: 0.1	Location	London borough	Post town	Postcode district	Dial code	OS grid ref	latitude	longitude	Neighborhood
0	0	0	Abbey Wood	Bexley, Greenwich [7]	LONDON	SE2	020	TQ465785	51.487621	0.114050	Abbey Wood
1	1	1	Acton	Ealing, Hammersmith and Fulham[8]	LONDON	W3, W4	020	TQ205805	51.508140	-0.273261	Acton
2	2	2	Addington	Croydon[8]	CROYDON	CR0	020	TQ375645	51.358637	-0.031635	Addington
3	3	3	Addiscombe	Croydon[8]	CROYDON	CR0	020	TQ345665	51.379692	-0.074282	Addiscombe

	Unnamed: 0	Unnamed: 0.1	Location	London borough	Post town	Postcode district	Dial code	OS grid ref	latitude	longitude	Neighborhood
4	4	4	Albany Park	Bexley	BEXLEY, SIDCUP	DA5, DA14	020	TQ478728	51.435384	0.125965	Albany Park

In [288…

```
# add clustering labels
neighborhoods_venues_sorted.insert(0, 'Cluster Labels', kmeans.labels_)

london_merged = wiki_df

# merge toronto_grouped with toronto_data to add latitude/longitude for each neighborhood
london_merged = london_merged.join(neighborhoods_venues_sorted.set_index('Neighborhood'), on='Neighborhood')

# view the merged data
london_merged.head()
```

Out[288…

	Unnamed: 0	Unnamed: 0.1	Location	London borough	Post town	Postcode district	Dial code	OS grid ref	latitude	longitude	...	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	0	0	Abbey Wood	Bexley, Greenwich [7]	LONDON	SE2	020	TQ465785	51.487621	0.114050	...	Supermarket	Warehouse Store	Gym / Fitness Center	Train Station	Grocery Store	Historic Site	Clothing Store	Mobile Phone Shop	R
1	1	1	Acton	Ealing, Hammersmith and Fulham[8]	LONDON	W3, W4	020	TQ205805	51.508140	-0.273261	...	Fast Food Restaurant	Brewery	Train Station	Hookah Bar	Food & Drink Shop	Bakery	Middle Eastern Restaurant	Bar	R
2	2	2	Addington	Croydon[8]	CROYDON	CR0	020	TQ375645	51.358637	-0.031635	...	Gaming Cafe	Forest	Gym / Fitness Center	Rugby Pitch	Mediterranean Restaurant	Tram Station	Train Station	Golf Course	
3	3	3	Addiscombe	Croydon[8]	CROYDON	CR0	020	TQ345665	51.379692	-0.074282	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
4	4	4	Albany Park	Bexley	BEXLEY, SIDCUP	DA5, DA14	020	TQ478728	51.435384	0.125965	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

5 rows × 22 columns

In [289…

```
#london_merged.to_csv('london_merged2306.csv')
```

In [290…

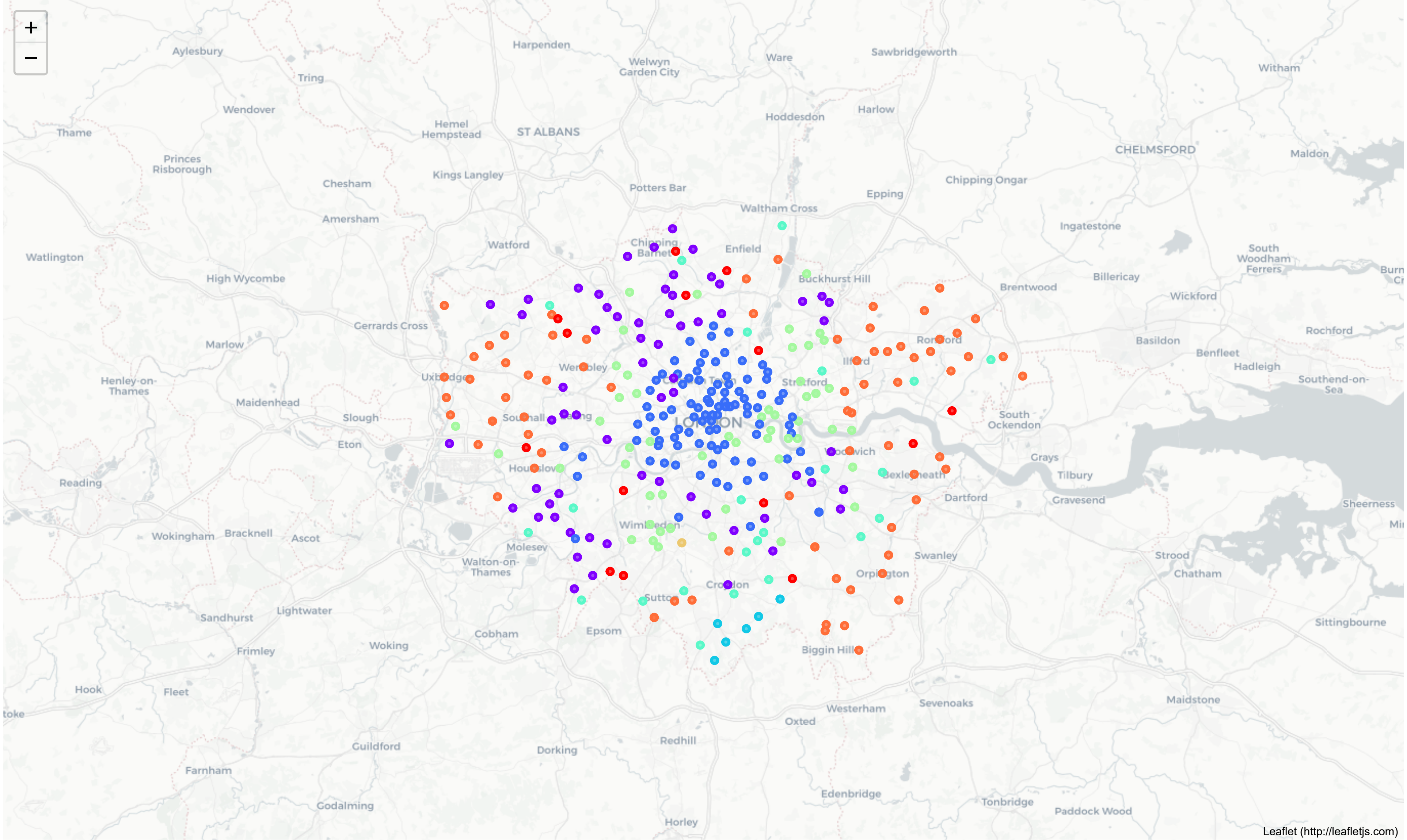
```
london_merged=london_merged.dropna(axis=0)
```

In [291…

```
# create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=10, tiles ='cartodbpositron')

# 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(london_merged['latitude'], london_merged['longitude'], london_merged['Neighborhood'], london_merged['Cluster Labels']):
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
```

```
    cluster= int(cluster)
    folium.CircleMarker(
        [lat, lon],
        radius=3,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7).add_to(map_clusters)

map_clusters
```

Out[291…



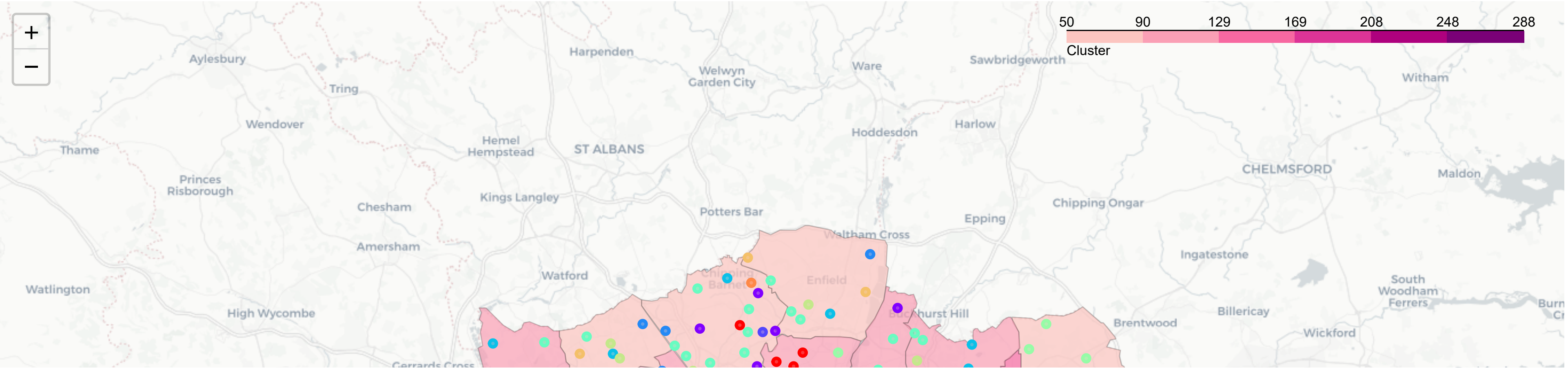
In [270…

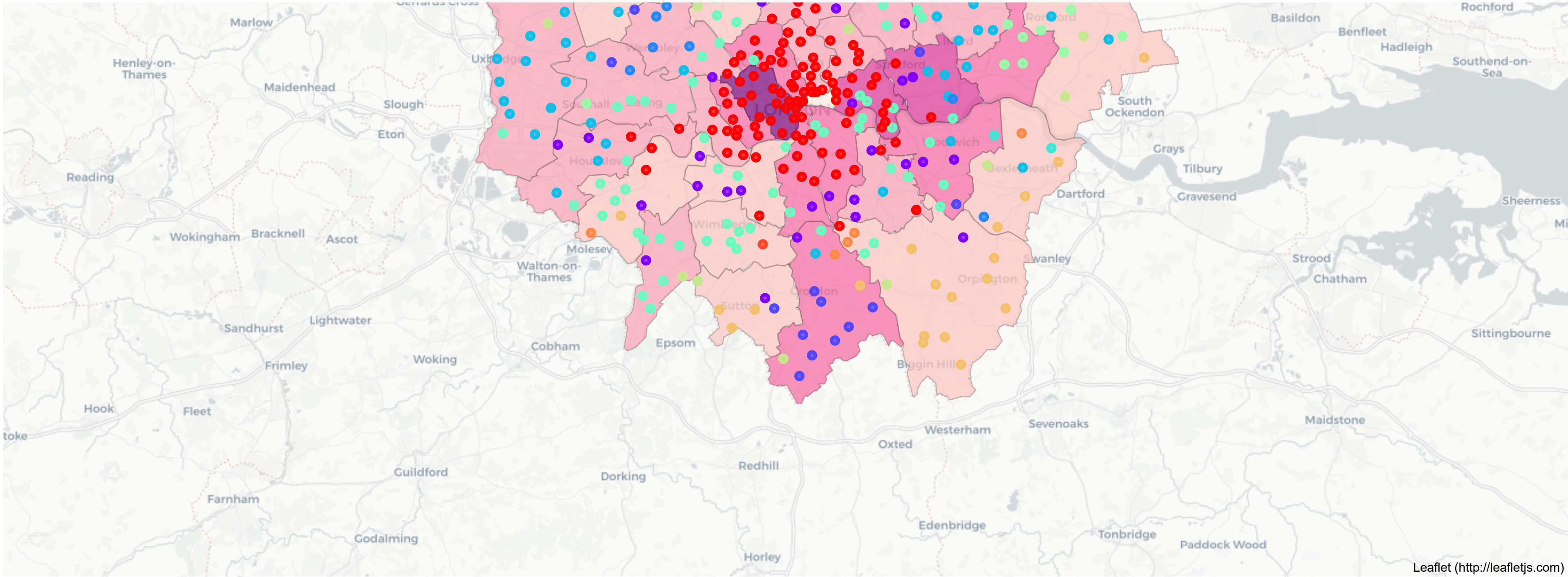
```
adjCrimeByWardMajor['WardCode'] = adjCrimeByWardMajor.index
adjCrimeByWardMinor['WardCode'] = adjCrimeByWardMinor.index
```

In [271…

```
# create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=10,    tiles ='cartodbpositron')
lnd_geo = r'london_boroughs_proper.geojson'

adjCrimeByBoroughMajor['Borough'] = adjCrimeByBoroughMajor.index
# 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]

map_clusters.choropleth(
    geo_data=lnd_geo,
    data=adjCrimeByBoroughMajor,
    columns=['Borough','Drug Offences'],
    key_on='feature.properties.name',
    fill_color='RdPu',
    fill_opacity=0.7,
    line_opacity=0.2,
    legend_name='Cluster'
    )

# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(london_merged['latitude'], london_merged['longitude'], london_merged['Neighborhood'], london_merged['Cluster Labels']):
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    cluster= int(cluster)
    folium.CircleMarker(
        [lat, lon],
        radius=3,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7).add_to(map_clusters)

map_clusters
```

Out[271…



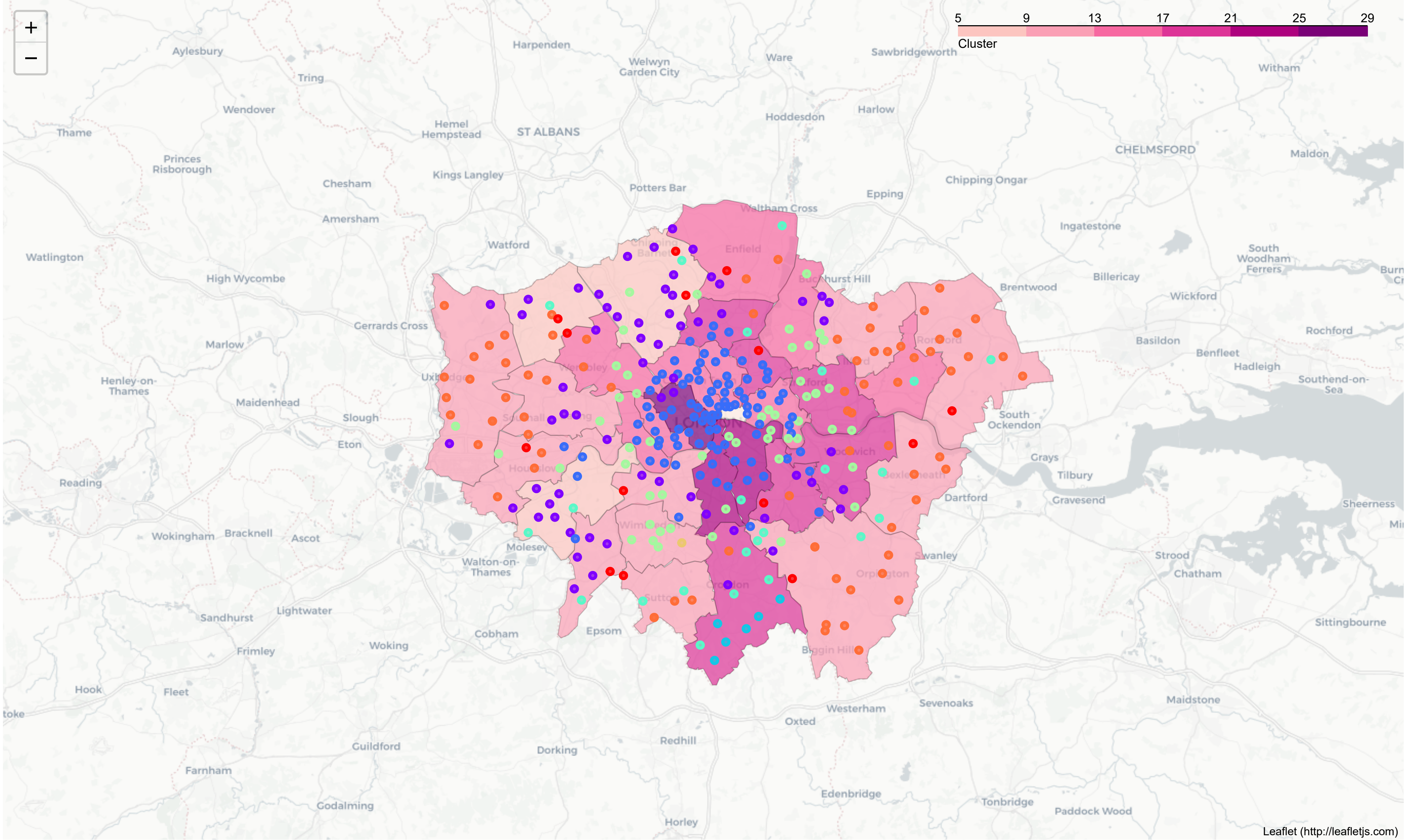


In [292…

```
# create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=10,    tiles ='cartodbpositron')

# 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]

map_clusters.choropleth(
    geo_data=lnd_geo,
    data=adjCrimeByBoroughMajor,
    columns=['Borough','Possession of Weapons'],
    key_on='feature.properties.name',
    fill_color='RdPu',
    fill_opacity=0.7,
    line_opacity=0.2,
    legend_name='Cluster'
)

# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(london_merged['latitude'], london_merged['longitude'],  london_merged['Neighborhood'], london_merged['Cluster Labels']):
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    cluster= int(cluster)
    folium.CircleMarker(
        [lat, lon],
        radius=3,
        popup=label,
        color=rainbow[cluster-1],
```

```
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7).add_to(map_clusters)

map_clusters
```

Out[292…



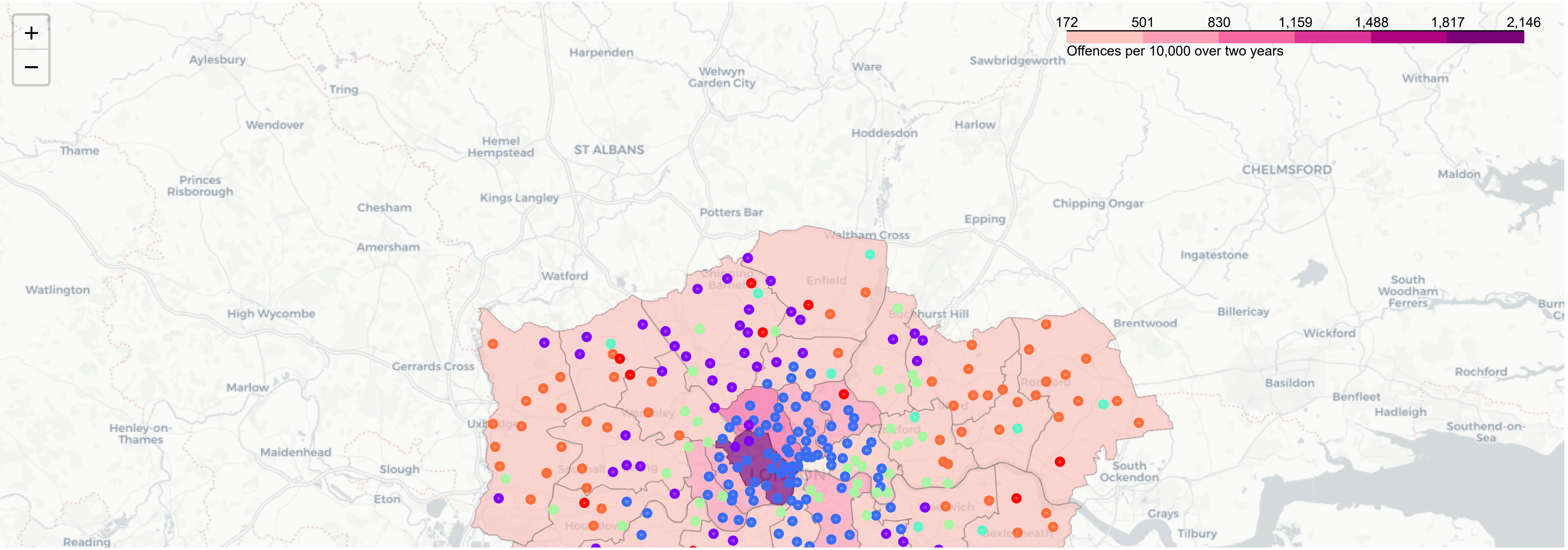
Focussing on the crimes most correlated with facilities

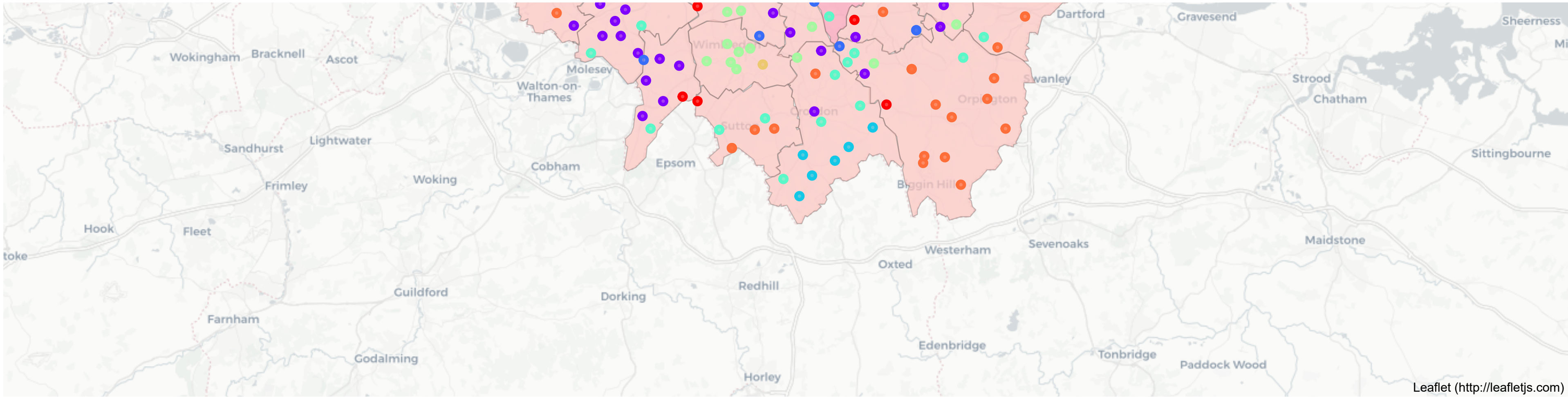
In [300…

```
# create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=10,    tiles ='cartodbpositron')
```

```python
# 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]

map_clusters.choropleth(
    geo_data=lnd_geo,
    data=adjCrimeByBoroughMajor,
    columns=['Borough','Theft'],
    key_on='feature.properties.name',
    fill_color='RdPu',
    fill_opacity=0.7,
    line_opacity=0.2,
    legend_name='Offences per 10,000 over two years'
)

# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(london_merged['latitude'], london_merged['longitude'],  london_merged['Neighborhood'], london_merged['Cluster Labels']):
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    cluster= int(cluster)
    folium.CircleMarker(
        [lat, lon],
        radius=3,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7).add_to(map_clusters)

map_clusters
```

Out[300…

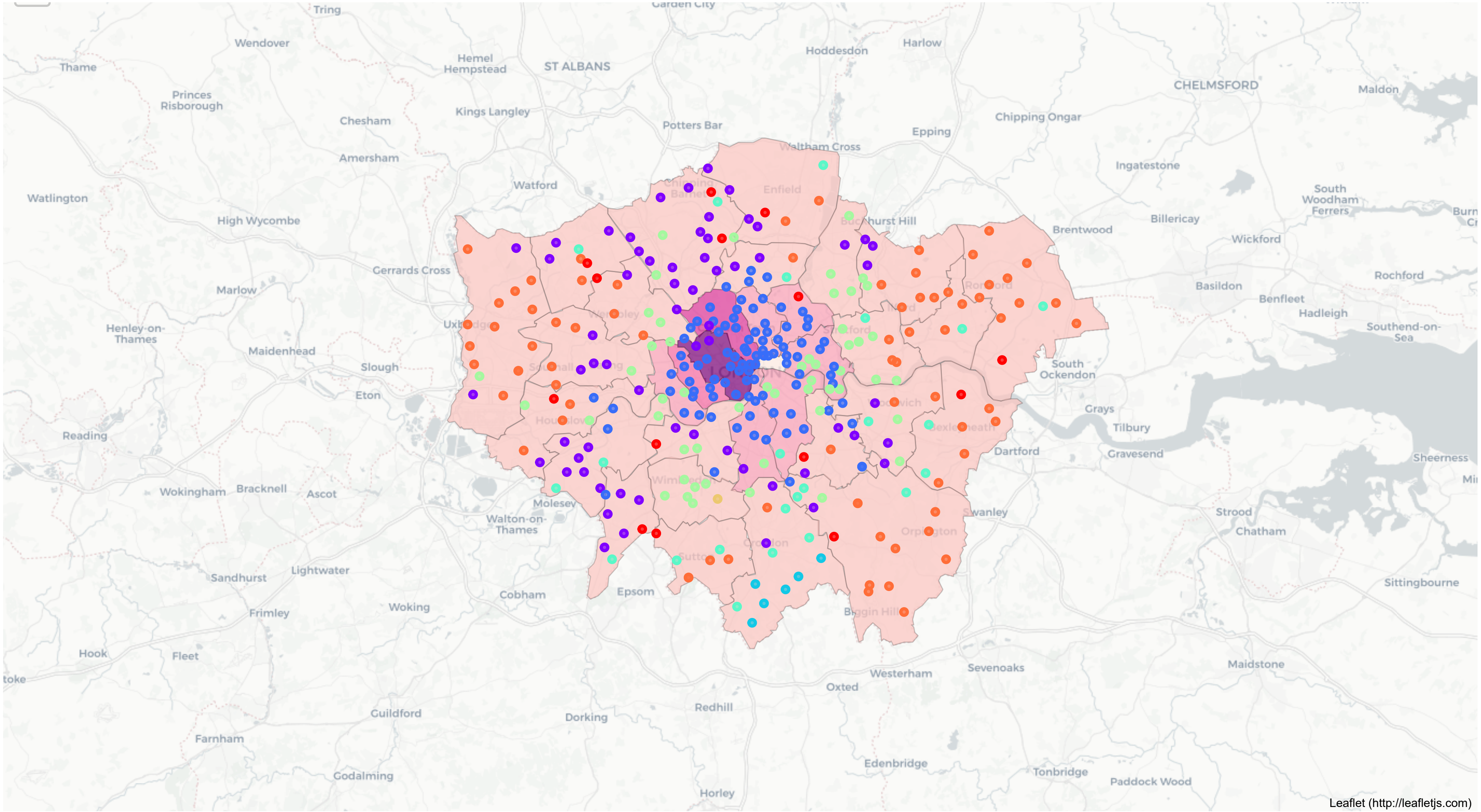




In [298…

```
# create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=10,    tiles ='cartodbpositron')
adjCrimeByBoroughMinor['Borough']=adjCrimeByBoroughMinor.index
# 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]

map_clusters.choropleth(
    geo_data=lnd_geo,
    data=adjCrimeByBoroughMinor,
    columns=['Borough','Burglary - Business and Community'],
    key_on='feature.properties.name',
    fill_color='RdPu',
    fill_opacity=0.7,
    line_opacity=0.2,
    legend_name='Offences per 10,000 over two years'
)

# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(london_merged['latitude'], london_merged['longitude'],  london_merged['Neighborhood'], london_merged['Cluster Labels']):
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    cluster= int(cluster)
    folium.CircleMarker(
        [lat, lon],
        radius=3,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7).add_to(map_clusters)

map_clusters
```

Out[298…

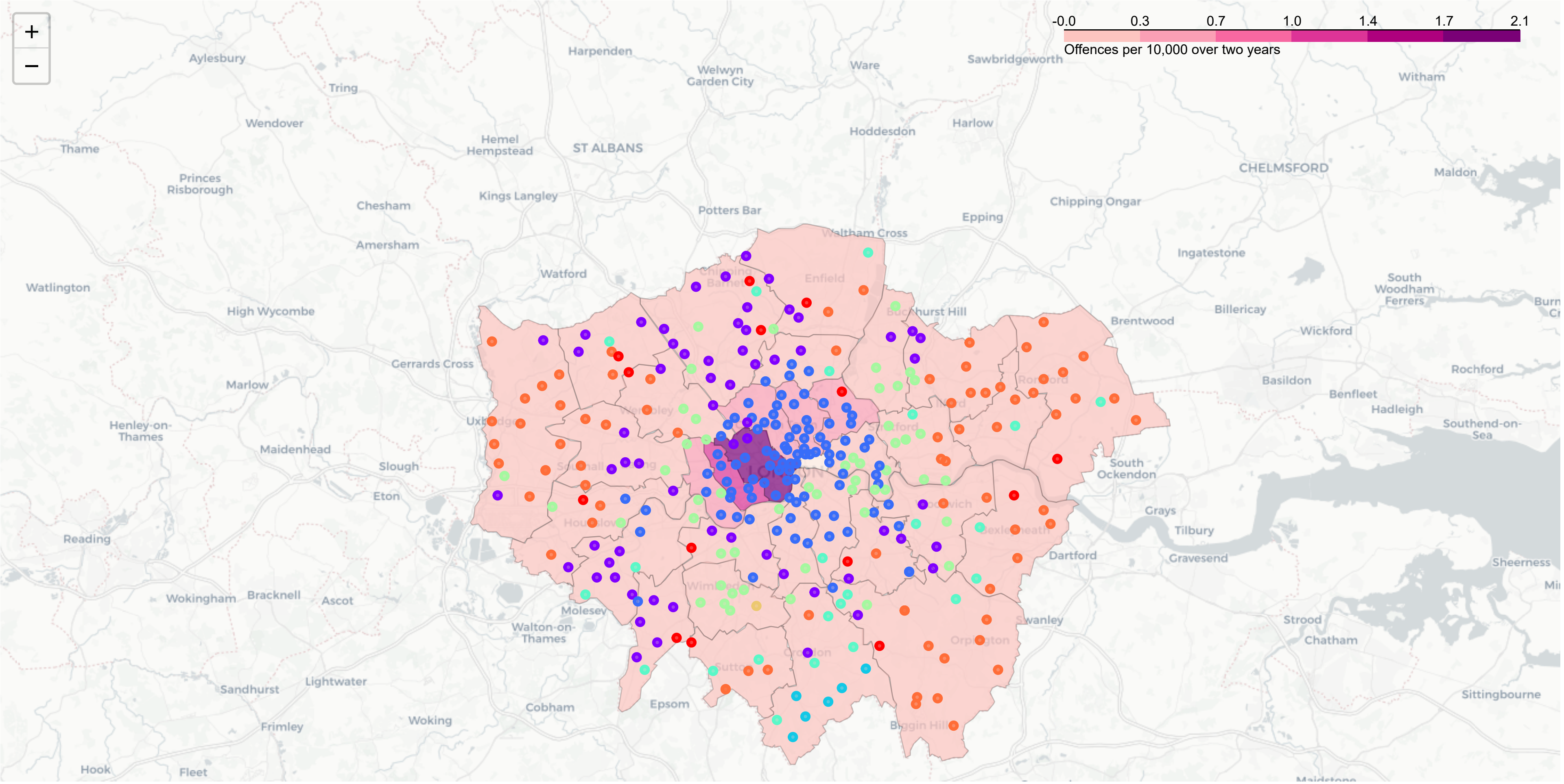


In [299…

```
# create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=10,    tiles ='cartodbpositron')
adjCrimeByBoroughMinor['Borough']=adjCrimeByBoroughMinor.index
# 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]

map_clusters.choropleth(
    geo_data=lnd_geo,
    data=adjCrimeByBoroughMinor,
    columns=['Borough','Violent Disorder'],
    key_on='feature.properties.name',
    fill_color='RdPu',
```

```
    fill_opacity=0.7,
    line_opacity=0.2,
    legend_name='Offences per 10,000 over two years'
)

# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(london_merged['latitude'], london_merged['longitude'],  london_merged['Neighborhood'], london_merged['Cluster Labels']):
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    cluster= int(cluster)
    folium.CircleMarker(
        [lat, lon],
        radius=3,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7).add_to(map_clusters)

map_clusters
```

Out[299…



Look at two of the clusters - cluster 2 the central blue cluster and cluster 5 the light green cluster

In [295…

```
london_merged.loc[london_merged['Cluster Labels'] == 2, london_merged.columns[[1] + list(range(5, london_merged.shape[1]))]]
```

Out[295…

	Unnamed: 0.1	Postcode district	Dial code	OS grid ref	latitude	longitude	Neighborhood	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
6	6	EC3	020	TQ334813	51.514248	-0.075719	Aldgate	2.0	Garden	Speakeasy	Bakery	Mediterranean Restaurant	Hotel Bar	Hotel	Office	French Restaurant	Beer Bar	Middle Eastern Restaurant
7	7	WC2	020	TQ307810	51.513103	-0.114920	Aldwych	2.0	Restaurant	Burger Joint	Gelato Shop	Opera House	Noodle House	Gym	Spanish Restaurant	Gym / Fitness Center	Steakhouse	Museum
10	10	EC1, N1	020	TQ345665	51.531842	-0.105714	Angel	2.0	Coffee Shop	Bookstore	Movie Theater	Caucasian Restaurant	Tapas Restaurant	Tea Room	Boutique	Cocktail Bar	Rock Club	Spa
18	18	EC1, EC2	020	TQ322818	51.520150	-0.098683	Barbican	2.0	Deli / Bodega	Cocktail Bar	Restaurant	Dance Studio	Roof Deck	Salad Place	Sandwich Place	Scenic Lookout	Concert Hall	Seafood Restaurant
26	26	N1	020	TQ305845	51.538935	-0.114735	Barnsbury	2.0	Restaurant	Brewery	Café	Spanish Restaurant	Gym	Burger Joint	Music Venue	Street Food Gathering	Breakfast Spot	Canal
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
490	490	E1	020	TQ345805	51.505436	-0.058729	Wapping	2.0	Theater	Street Food Gathering	Pilates Studio	Flea Market	Hotel	Salon / Barbershop	Bookstore	Movie Theater	Boutique	Tapas Restaurant
497	497	SW10	020	TQ253779	51.486976	-0.195185	West Brompton	2.0	Sports Bar	Hotel	Middle Eastern Restaurant	Cosmetics Shop	Farmers Market	Furniture / Home Store	Thai Restaurant	Pharmacy	Bookstore	Tapas Restaurant
503	503	NW6	020	TQ255855	51.546819	-0.189965	West Hampstead	2.0	Korean Restaurant	Thai Restaurant	Bar	Middle Eastern Restaurant	Grocery Store	Creperie	Pizza Place	Fish & Chips Shop	Hostel	Café
511	511	SW1	020	TQ295795	51.500444	-0.126540	Westminster	2.0	Theater	Performing Arts Venue	Hotel	Garden	Bookstore	History Museum	Boutique	Monument / Landmark	Tapas Restaurant	Tailor Shop
514	514	E1	020	TQ335815	51.518623	-0.062081	Whitechapel	2.0	Korean Restaurant	Burger Joint	Hotel	Flea Market	Salon / Barbershop	Movie Theater	Bookstore	Boutique	Cosmetics Shop	Music Venue

96 rows × 18 columns

In [301…

```
london_merged.loc[london_merged['Cluster Labels'] == 5, london_merged.columns[[1] + list(range(5, london_merged.shape[1]))]]
```

Out[301…

	Unnamed: 0.1	Postcode district	Dial code	OS grid ref	latitude	longitude	Neighborhood	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

	Unnamed: 0.1	Postcode district	Dial code	OS grid ref	latitude	longitude	Neighborhood	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
1	1	W3, W4	020	TQ205805	51.508140	-0.273261	Acton	5.0	Fast Food Restaurant	Brewery	Train Station	Hookah Bar	Food & Drink Shop	Bakery	Middle Eastern Restaurant	Bar	Chinese Restaurant
22	22	SW13	020	TQ225765	51.471896	-0.238744	Barnes	5.0	Pub	Grocery Store	Track	Flea Market	Trail	Train Station	Breakfast Spot	Historic Site	Gym
29	29	BR3, SE20	020	TQ375695	51.407094	-0.030318	Beckenham	5.0	Newsagent	Chinese Restaurant	Tennis Court	Tapas Restaurant	Movie Theater	Supermarket	Gym / Fitness Center	Breakfast Spot	Steakhouse
52	52	E14	020	TQ385805	51.507938	-0.007184	Blackwall	5.0	Restaurant	Fried Chicken Joint	Outlet Mall	Outdoor Sculpture	Canal Lock	Nightclub	Café	Newsagent	Grocery Store
82	82	SW13	020	TQ226776	51.485688	-0.233030	Castelnau	5.0	Soccer Field	Food & Drink Shop	Pharmacy	Trail	Convenience Store	Track	Bar	Basketball Court	Sporting Goods Shop
97	97	E4	020	TQ395945	51.630887	0.003996	Chingford	5.0	Café	Convenience Store	Thai Restaurant	Park	Gym / Fitness Center	Gym	Grocery Store	Coffee Shop	History Museum
107	107	SW19	020	TQ275705	51.418275	-0.177863	Colliers Wood	5.0	Café	Asian Restaurant	Thai Restaurant	Fast Food Restaurant	Tapas Restaurant	Grocery Store	Middle Eastern Restaurant	Park	Bar
116	116	TW5	020	TQ105765	51.480671	-0.409677	Cranford	5.0	Garden Center	Hotel	English Restaurant	Coffee Shop	Bakery	Bus Station	Fast Food Restaurant	Park	Gym / Fitness Center
128	128	E14	020	TQ385795	51.493668	-0.008243	Cubitt Town	5.0	Pub	Turkish Restaurant	Chinese Restaurant	Nail Salon	Convenience Store	Golf Driving Range	Pet Store	Bar	Middle Eastern Restaurant
143	143	SW18	020	TQ265735	51.446448	-0.189394	Earlsfield	5.0	Indoor Play Area	Burger Joint	Thai Restaurant	Gym	Grocery Store	Music Venue	Steakhouse	Brewery	General Entertainment
156	156	SE1, SE11, SE17	020	TQ319789	51.494888	-0.100573	Elephant and Castle	5.0	Nightclub	Garden	Theater	Thai Restaurant	Museum	Music Venue	Breakfast Spot	Brewery	Supermarket
252	252	NW9	020	TQ215888	51.584073	-0.241946	The Hyde	5.0	Pet Store	Supermarket	Park	Auto Workshop	Coffee Shop	Asian Restaurant	Hookah Bar	Hotel	Sushi Restaurant
256	256	TW7	020	TQ155755	51.468356	-0.326311	Isleworth	5.0	Middle Eastern Restaurant	Coffee Shop	Asian Restaurant	Gym / Fitness Center	Breakfast Spot	Hockey Field	Sandwich Place	Historic Site	Supermarket
260	260	NW10, NW6	020	TQ235825	51.530606	-0.224445	Kensal Green	5.0	Deli / Bodega	Electronics Store	Pizza Place	Train Station	Coffee Shop	Cocktail Bar	Bakery	Persian Restaurant	Thai Restaurant
283	283	E10, E15	020	TQ375865	51.569673	-0.015681	Leyton	5.0	Pub	Convenience Store	Discount Store	Metro Station	Hockey Field	Sporting Goods Shop	Pharmacy	Sports Bar	Fast Food Restaurant
284	284	E11	020	TQ395875	51.571078	0.006424	Leytonstone	5.0	Coffee Shop	Pizza Place	Mediterranean Restaurant	Bakery	Metro Station	Bar	Pharmacy	Fast Food Restaurant	Thai Restaurant
285	285	E14	020	TQ365815	51.512870	-0.039046	Limehouse	5.0	History Museum	Scenic Lookout	Hotel Bar	Hotel	Bakery	Bus Stop	Hostel	Convenience Store	Sandwich Place
304	304	SW19	020	TQ250695	51.407994	-0.201965	Merton Park	5.0	Donut Shop	Brewery	Pet Store	Modern European Restaurant	Tennis Stadium	Tennis Court	Sandwich Place	Flea Market	Movie Theater
307	307	NW7	020	TQ225925	51.615442	-0.233068	Mill Hill	5.0	Rugby Stadium	Soccer Field	Golf Course	Pizza Place	Athletics & Sports	Pharmacy	Grocery Store	Metro Station	Gym

	Unnamed: 0.1	Postcode district	Dial code	OS grid ref	latitude	longitude	Neighborhood	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
309	309	E14	020	TQ375785	51.493342	-0.021219	Millwall	5.0	Bus Station	Café	Thai Restaurant	Historic Site	Middle Eastern Restaurant	Boat or Ferry	Modern European Restaurant	Sushi Restaurant	Harbor / Marina
312	312	SM4	020	TQ255685	51.402761	-0.194755	Morden	5.0	Modern European Restaurant	Park	Farm	Bakery	Bar	Fast Food Restaurant	Grocery Store	Burger Joint	Gym / Fitness Center
318	318	NW2, NW10	020	TQ215855	51.554336	-0.250749	Neasden	5.0	Restaurant	Sushi Restaurant	Stadium	Music Venue	Halal Restaurant	Breakfast Spot	Brazilian Restaurant	Supermarket	Music Store
321	321	SE14	020	TQ365765	51.476371	-0.032624	New Cross	5.0	Pub	Pizza Place	Train Station	Sandwich Place	Coffee Shop	Cocktail Bar	Clothing Store	Thai Restaurant	Fast Food Restaurant
322	322	SE9	020	TQ440730	51.436230	0.068529	New Eltham	5.0	Sports Club	Fast Food Restaurant	Bar	Bus Stop	Park	Fish & Chips Shop	Gym	Grocery Store	Auto Garage
324	324	N11	020	TQ295925	51.613804	-0.142809	New Southgate	5.0	Kebab Restaurant	Hotel	Fish & Chips Shop	Mediterranean Restaurant	Park	Metro Station	Pizza Place	Middle Eastern Restaurant	Scenic Lookout
327	327	SW8	020	TQ295775	51.478743	-0.136263	Nine Elms	5.0	Sports Bar	Coffee Shop	Track Stadium	Hotel	Hookah Bar	Gastropub	Convenience Store	Flower Shop	Historic Site
330	330	SW16	020	TQ315695	51.411066	-0.122487	Norbury	5.0	Garden	Playground	Park	Bar	Grocery Store	Mediterranean Restaurant	Skating Rink	Pakistani Restaurant	Coffee Shop
338	338	E16	020	TQ435795	51.500407	0.064154	North Woolwich	5.0	Restaurant	Duty-free Shop	Bistro	Harbor / Marina	Pharmacy	Chinese Restaurant	Coffee Shop	Gym / Fitness Center	Exhibit
349	349	NW10	020	TQ216823	51.527949	-0.247089	Old Oak Common	5.0	Recreation Center	Pastry Shop	Moroccan Restaurant	Movie Theater	Halal Restaurant	Bowling Alley	Gym / Fitness Center	Brazilian Restaurant	Grocery Store
366	366	E13	020	TQ405825	51.531154	0.016683	Plaistow	5.0	Asian Restaurant	Coffee Shop	Sandwich Place	Market	Boutique	Hotel	Gym / Fitness Center	Fish & Chips Shop	Breakfast Spot
367	367	BR1	020	TQ405705	51.531154	0.016683	Plaistow	5.0	Asian Restaurant	Coffee Shop	Sandwich Place	Market	Boutique	Hotel	Gym / Fitness Center	Fish & Chips Shop	Breakfast Spot
381	381	SW20	020	TQ235685	51.408966	-0.230540	Raynes Park	5.0	Karaoke Bar	Supermarket	Golf Driving Range	Pub	Burger Joint	Grocery Store	Steakhouse	Gym Pool	Breakfast Spot
387	387	SE16	020	TQ358796	51.500291	-0.043632	Rotherhithe	5.0	Gym Pool	Hostel	Cheese Shop	Soccer Stadium	Grocery Store	Canal	Gym	Gym / Fitness Center	Sporting Goods Shop
398	398	E1	020	TQ355805	51.511250	-0.056924	Shadwell	5.0	Fried Chicken Joint	Event Space	Hostel	Market	Pizza Place	Breakfast Spot	Steakhouse	Sandwich Place	Coffee Shop
401	401	SE18	020	TQ435765	51.469226	0.066306	Shooter's Hill	5.0	Middle Eastern Restaurant	Indian Restaurant	Farm	Beer Bar	Fish & Chips Shop	Castle	Grocery Store	Park	Golf Course
404	404	E16	020	TQ415795	51.501363	0.038518	Silvertown	5.0	Steakhouse	Pharmacy	Park	Chinese Restaurant	Grocery Store	Gym / Fitness Center	Noodle House	Harbor / Marina	Café
407	407	E11	020	TQ395895	51.580997	0.021525	Snaresbrook	5.0	Diner	Mediterranean Restaurant	Bakery	Metro Station	Thrift / Vintage Store	Fast Food Restaurant	Sandwich Place	Pet Store	Tea Room

	Unnamed: 0.1	Postcode district	Dial code	OS grid ref	latitude	longitude	Neighborhood	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
417	417	SW19	020	TQ255705	51.415176	-0.192646	South Wimbledon	5.0	Noodle House	Gym / Fitness Center	Pizza Place	Train Station	Bar	Coffee Shop	Pet Store	Flea Market	Chinese Restaurant
423	423	SW18, SW19	020	TQ255735	51.445775	-0.206614	Southfields	5.0	Sporting Goods Shop	Metro Station	Track	Bar	Thrift / Vintage Store	Thai Restaurant	Tennis Stadium	Tennis Court	Gym
438	438	E1	020	TQ355814	51.517402	-0.046219	Stepney	5.0	Dessert Shop	Persian Restaurant	Concert Hall	Sandwich Place	Farm	Bar	Fast Food Restaurant	Beer Bar	Coffee Shop
442	442	E15	020	TQ385845	51.541289	-0.003547	Stratford	5.0	Eastern European Restaurant	Hockey Field	Burger Joint	Sandwich Place	Bubble Tea Shop	Supermarket	Historic Site	Rugby Pitch	Fish & Chips Shop
449	449	SE16	020	TQ356789	51.493421	-0.047832	Surrey Quays	5.0	Gym	Gym Pool	Chinese Restaurant	Soccer Stadium	Sporting Goods Shop	Café	Grocery Store	Pub	Bus Stop
480	480	E6, E13	020	TQ405837	51.535106	0.033984	Upton Park	5.0	Fast Food Restaurant	Steakhouse	Mediterranean Restaurant	Toy / Game Store	Pharmacy	Comfort Food Restaurant	Tapas Restaurant	Sandwich Place	Fish & Chips Shop
485	485	E17	020	TQ375865	51.584470	-0.018819	Walthamstow	5.0	River	Gym / Fitness Center	Gourmet Shop	Newsagent	Bus Station	Grocery Store	Gym	Burger Joint	Halal Restaurant
487	487	SE17	020	TQ325785	51.490114	-0.090660	Walworth	5.0	Gym	Supermarket	Hotel	Hostel	Pizza Place	Fish & Chips Shop	Sandwich Place	Thai Restaurant	Tennis Court
489	489	E11	020	TQ405885	51.575674	0.027799	Wanstead	5.0	Pub	Train Station	Pharmacy	Bakery	Metro Station	Bar	Coffee Shop	Thai Restaurant	Tea Room
498	498	UB7	01895	TQ065795	51.503513	-0.466270	West Drayton	5.0	Pub	Electronics Store	Thai Restaurant	Gym / Fitness Center	Fast Food Restaurant	Gym	Bed & Breakfast	Grocery Store	Coffee Shop
502	502	E13, E15	020	TQ405837	51.528097	0.004568	West Ham	5.0	Bulgarian Restaurant	Trail	Thai Restaurant	Gym	Boutique	Breakfast Spot	General Entertainment	Bus Station	Bus Stop
507	507	W14	020	TQ246783	51.490702	-0.205944	West Kensington	5.0	Pub	Museum	Hostel	Hookah Bar	Convention Center	Thai Restaurant	Tennis Stadium	Tennis Court	Convenience Store
508	508	SE27	020	TQ325715	51.434619	-0.103692	West Norwood	5.0	Italian Restaurant	Bus Station	Grocery Store	Movie Theater	Supermarket	Breakfast Spot	Nature Preserve	Fried Chicken Joint	Café
517	517	NW10	020	TQ227846	51.546622	-0.235866	Willesden	5.0	Pub	Halal Restaurant	Middle Eastern Restaurant	Fast Food Restaurant	Modern European Restaurant	Tapas Restaurant	Moroccan Restaurant	Sandwich Place	Movie Theater
518	518	SW19, SW20	020	TQ239709	51.421479	-0.206403	Wimbledon	5.0	Restaurant	Supermarket	Chinese Restaurant	Garden	Café	Noodle House	Gym / Fitness Center	Burger Joint	Brewery

Not clear from this how the clusters are formed

In [ ]:

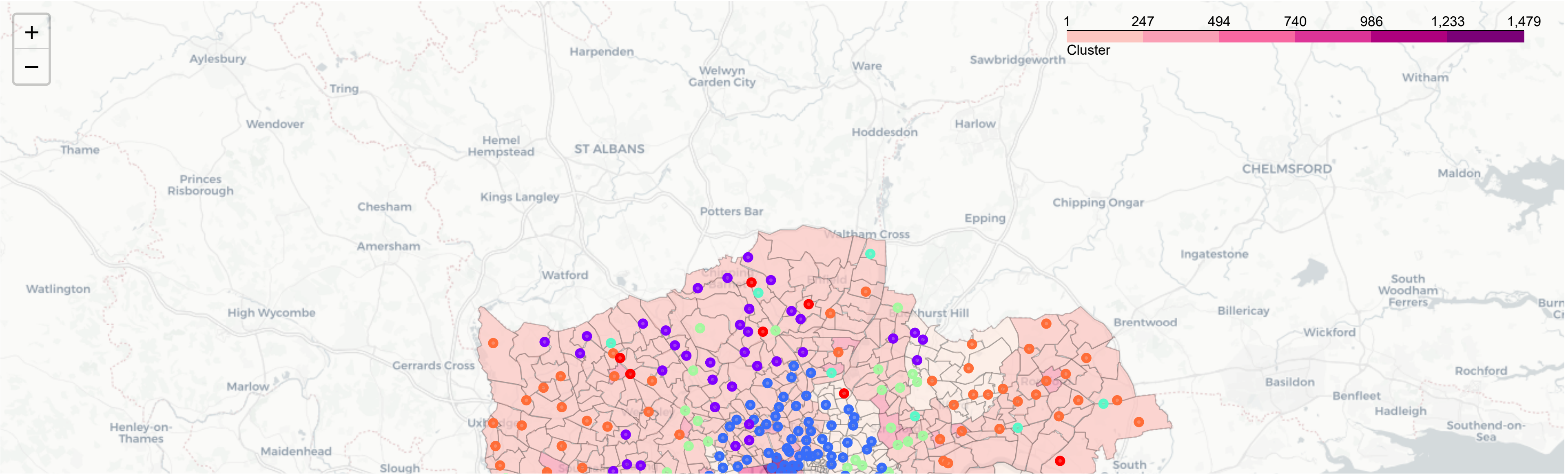
Ward level mapping

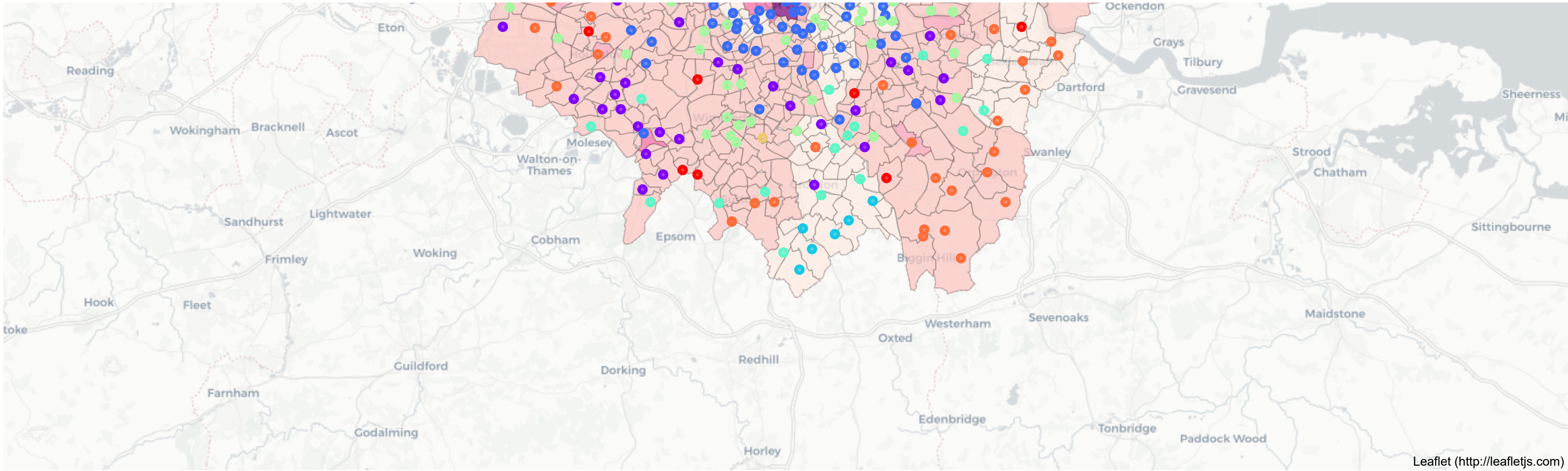
This is left in incase a more uptodate geojson is located - and to see if any pattern present in the wards mapped

In [302…

```
# create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=10,    tiles ='cartodbpositron')
lnd_ward_geo =  r'london-wards-2014.geojson'

CrimeByWardMajor['WardCode'] = CrimeByWardMajor.index
# 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]

map_clusters.choropleth(
    geo_data=lnd_ward_geo,
    data=adjCrimeByWardMajor,
    columns=['WardCode','Drug Offences'],
    key_on='feature.properties.gss_code_ward',
    fill_color='RdPu',
    fill_opacity=0.7,
    line_opacity=0.2,
    legend_name='Cluster'
    )

# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(london_merged['latitude'], london_merged['longitude'], london_merged['Neighborhood'], london_merged['Cluster Labels']):
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    cluster= int(cluster)
    folium.CircleMarker(
        [lat, lon],
        radius=3,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7).add_to(map_clusters)

map_clusters
```

Out[302…

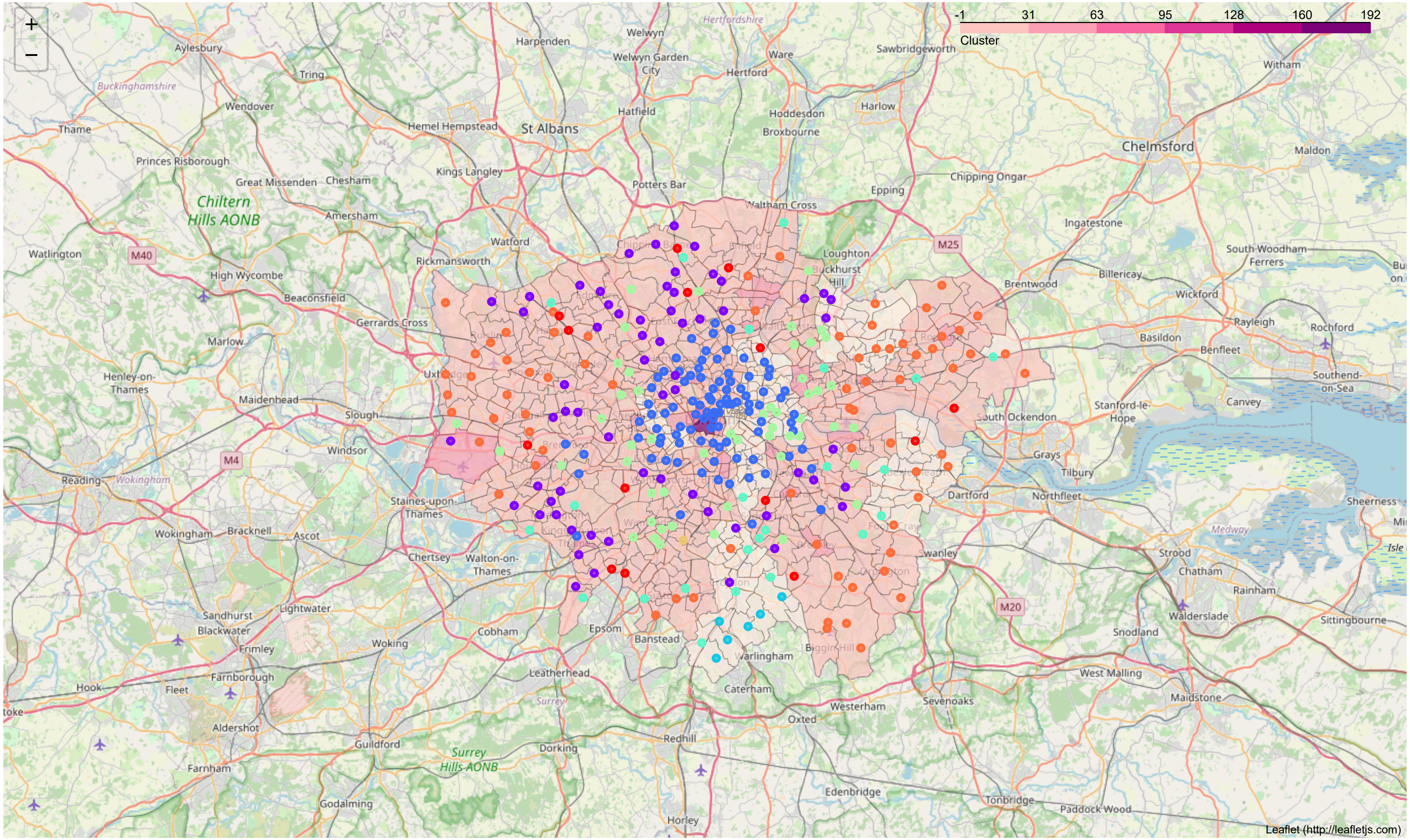




In [303…

```
# create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=10)

# 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]

map_clusters.choropleth(
    geo_data=lnd_ward_geo,
    data=adjCrimeByWardMajor,
    columns=['WardCode','Possession of Weapons'],
    key_on='feature.properties.gss_code_ward',
    fill_color='RdPu',
    fill_opacity=0.7,
    line_opacity=0.2,
    legend_name='Cluster'
)

# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(london_merged['latitude'], london_merged['longitude'],  london_merged['Neighborhood'], london_merged['Cluster Labels']):
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    cluster= int(cluster)
    folium.CircleMarker(
        [lat, lon],
        radius=3,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7).add_to(map_clusters)

map_clusters
```

Out[303…



Let's try remvoing the hotspots, and focussing on the crimes most correlated with facilities

In [304…

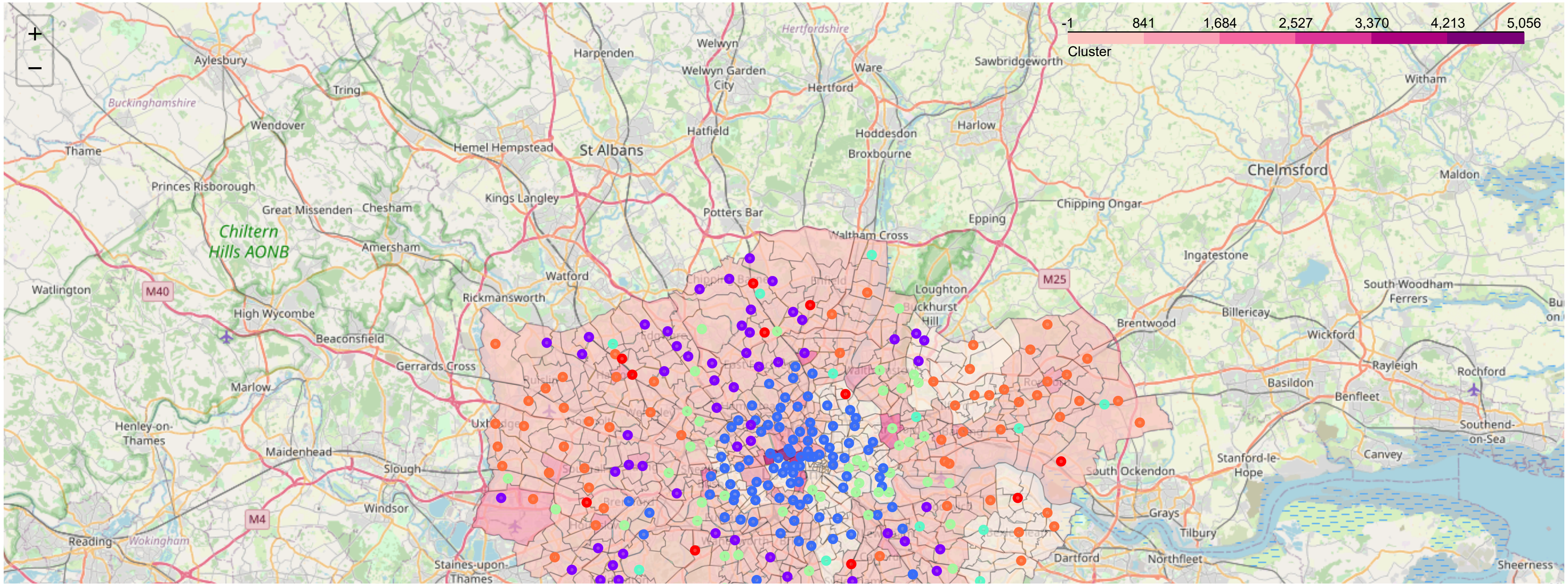
```
CrimeByWardMajorNoHotSpot = adjCrimeByWardMajor.drop(['E05000649', 'E05000644'], axis=0)
CrimeByWardMinorNoHotSpot = adjCrimeByWardMinor.drop(['E05000649', 'E05000644'], axis=0)
```

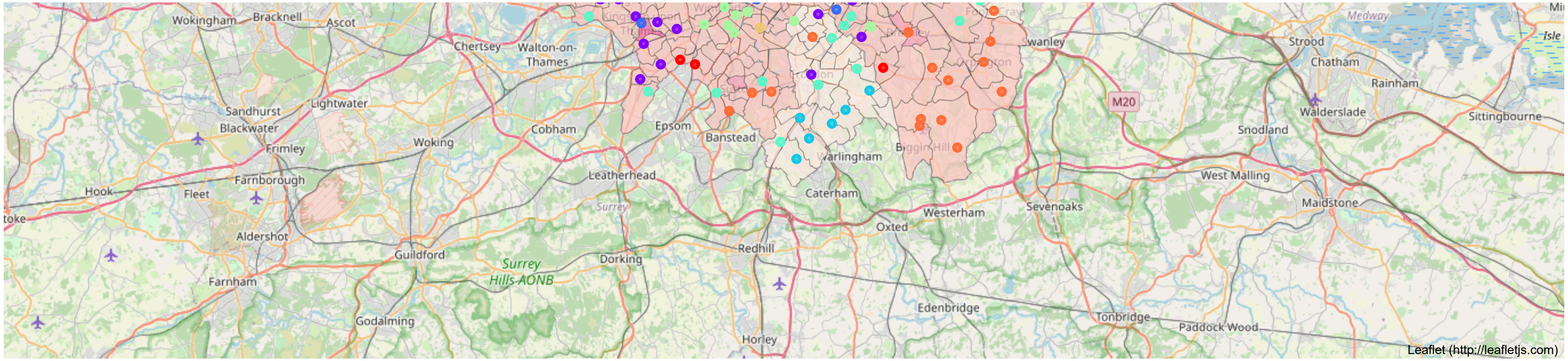
In [305…

```
# create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=10)

# 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]

map_clusters.choropleth(
    geo_data=lnd_ward_geo,
    data=CrimeByWardMajorNoHotSpot,
    columns=['WardCode','Theft'],
    key_on='feature.properties.gss_code_ward',
    fill_color='RdPu',
    fill_opacity=0.7,
    line_opacity=0.2,
    legend_name='Cluster'
)

# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(london_merged['latitude'], london_merged['longitude'],  london_merged['Neighborhood'], london_merged['Cluster Labels']):
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    cluster= int(cluster)
    folium.CircleMarker(
        [lat, lon],
        radius=3,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7).add_to(map_clusters)

map_clusters
```

Out[305…

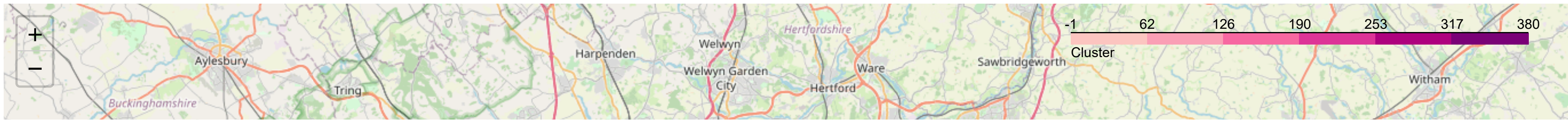


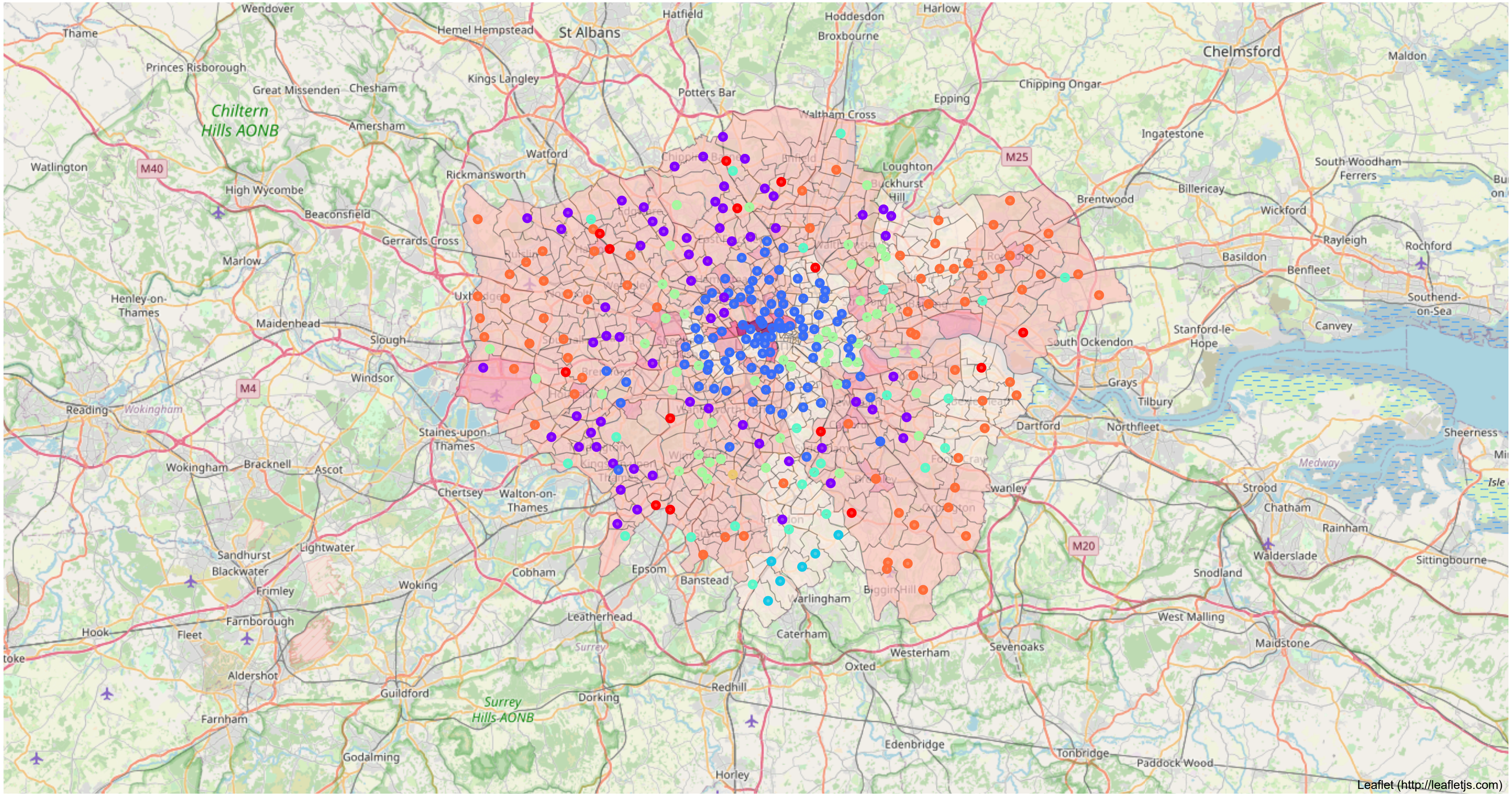


In [306…

```
# create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=10)

# 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]

map_clusters.choropleth(
    geo_data=lnd_ward_geo,
    data=CrimeByWardMinorNoHotSpot,
    columns=['WardCode','Burglary - Business and Community'],
    key_on='feature.properties.gss_code_ward',
    fill_color='RdPu',
    fill_opacity=0.7,
    line_opacity=0.2,
    legend_name='Cluster'
)

# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(london_merged['latitude'], london_merged['longitude'],  london_merged['Neighborhood'], london_merged['Cluster Labels']):
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    cluster= int(cluster)
    folium.CircleMarker(
        [lat, lon],
        radius=3,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7).add_to(map_clusters)

map_clusters
```

Out[306…





No evidence of a pattern

Additional fact finding for blog

LEt'd find a boring neighbourhood (seen in scratch pad analysis)

In [308…

```
neighborhoods_venues_sorted[neighborhoods_venues_sorted['1st Most Common Venue']=='Gas Station']
```

Out[308…

	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	9th Most Common Venue	10th Most Common Venue
214	4	Penge	Gas Station	Outdoor Sculpture	Fast Food Restaurant	Hardware Store	Sculpture Garden	Gym / Fitness Center	Tapas Restaurant	Grocery Store	Train Station	Tram Station

	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	9th Most Common Venue	10th Most Common Venue
280	7	Thornton Heath	Gas Station	Bus Stop	Home Service	Sandwich Place	Auto Garage	Coffee Shop	Tram Station	Mediterranean Restaurant	Chinese Restaurant	Pizza Place

In [311…

```
neighborhoods_venues_sorted[neighborhoods_venues_sorted['1st Most Common Venue']=='Bus Stop']
```

Out[311…

	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	9th Most Common Venue	10th Most Common Venue
93	4	Enfield Lock	Bus Stop	Hotel	Fast Food Restaurant	Gym / Fitness Center	Clothing Store	Trail	Train Station	Pet Store	Coffee Shop	Grocery Store
175	2	Lower Clapton	Bus Stop	Chinese Restaurant	History Museum	Historic Site	Bookstore	Breakfast Spot	Brewery	Gym / Fitness Center	Grocery Store	Noodle House
276	1	Swiss Cottage	Bus Stop	Coffee Shop	Hostel	Convenience Store	Pharmacy	Moroccan Restaurant	History Museum	Sandwich Place	Bookstore	Multiplex

In [312…

```
neighborhoods_venues_sorted[neighborhoods_venues_sorted['1st Most Common Venue']=='Fast Food Restaurant']
```

Out[312…

	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	9th Most Common Venue	10th Most Common Venue
1	5	Acton	Fast Food Restaurant	Brewery	Train Station	Hookah Bar	Food & Drink Shop	Bakery	Middle Eastern Restaurant	Bar	Chinese Restaurant	Pastry Shop
212	1	Palmers Green	Fast Food Restaurant	Thrift / Vintage Store	Train Station	Trail	Mediterranean Restaurant	Bakery	Greek Restaurant	Bar	Sandwich Place	Mexican Restaurant
290	5	Upton Park	Fast Food Restaurant	Steakhouse	Mediterranean Restaurant	Toy / Game Store	Pharmacy	Comfort Food Restaurant	Tapas Restaurant	Sandwich Place	Fish & Chips Shop	Coffee Shop

In [ ]: