CAPSTONE PROJECT – WHERE TO MOVE DUE TO BREXIT?

IBM / Coursera Certification, Dennis Breuer 12 / 2019

Table of contents

Business problem 2

Data 2

Methodology 4

Analysis 4

Results and discussion 9

Conclusion 9

Business problem

This project deals with the question to which EU city banks should move their HQ to, if the current location of their HQ is in the UK (more specifically London as it is the financial centre of the UK). The reason why banks are asking themselves this question is linked to the upcoming Brexit, which means that banks have higher hurdles to overcome when wanting to be involved in the EU market.

The target audience therefore is clearly consisting of decision-makers at the boards of these banks.

The main question which will be tackled through this project is: "which candidate cities offer a similar environment compared to London?".

Data

In order to be able to answer this question, data was gathered mainly from the eurostat database, which offers publicly available data on various topics related to the EU (https://ec.europa.eu/eurostat/data/database).

The following parameters were identified to be relevant for the project scope:

- The percentage of country inhabitants with an education level of 5-8 acc. the International Standard Classification of Education (ISCED) (2018)
- The labor costs in the respective country [€] (2018)
- The average monthly rent in the candidate city [€/m^2] (2018)
- Consumer prices acc. to the Harmonized Index of Consumer Prices (HICP) (2018)
- Crime rate / 100'000 inhabitants for the respective country
- The number of universities in the city area
- The difference in time zones compared to London
- The life satisfaction index (2018) for the respective country
- The number of active companies for the respective country (2017)

The following candidate cities were evaluated as potential new locations for banks, all of them being locations of stock exchanges:

- Dublin, Ireland
- Frankfurt, Germany
- Paris, France
- Warsaw, Poland
- Madrid, Spain

- Lisbon, Portugal
- Stockholm, Sweden
- Helsinki, Finland
- Milan, Italy
- Brussels, Belgium
- Amsterdam, Netherlands
- Copenhagen, Denmark
- Vienna, Austria
- Prague, Czech Republic

First, we need to import all relevant libraries.

```
# import all relevant libraries
import pandas as pd
import numpy as np
from sklearn import preprocessing
from sklearn.cluster import KMeans
from sklearn.cluster import DBSCAN
import json

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

import requests
from pandas.io.json import json_normalize

import matplotlib.cm as cm
import matplotlib.colors as colors

!conda install -c conda-forge folium=0.5.0 --yes
import folium

print('Libraries imported.')
```

Then we get the data file prepared for the analysis. To do so, we clone my GitHub repository.

```
# Clone the entire repo.
!git clone -l -s git://github.com/dennisb1239/Coursera_Capstone.git cloned-repo
%cd cloned-repo
!ls
```

Methodology

In this project we will try to find an alternative city, to which banks with their current HQ in London / UK can move to, in order to avoid negative consequences due to the upcoming Brexit. To do so, we followed the steps below:

1. We first imported a prepared data set through the cloning of my git repository, and transformed it into a pandas DataFrame
2. We created a second DataFrame, and store the geographical coordinates of the evaluated cities in it
3. A first map is created, to visualize the geographical locations of the cities
4. 2 different clustering algorithms are tested on the first Dataframe: KMeans & DBScan
5. The results of the 2 cluster analyses are shown on separate maps, in order to make it easier to identify the alternative candidates

Analysis

The data is saved in a pandas DataFrame and we check the dataframe. The data is already gathered in one table, and normalized. This was done in Excel to speed up the preprocessing process.

```
In [6]: df = pd.DataFrame()
        df = pd.read_excel("Data_C9W5_normalized.xlsx", index_col = 0)
        df
```

Out[6]:

City	EducationLevel	LaborCost	AvMonthlyRent	HICP	Crimes	NoUniversities	DiffTimeZones	LifeSatisfactionIndex	ActiveCompanies
London	0.948718	0.517964	0.795322	0.904255	1.000000	1.000000	0.0	0.642857	0.642911
Dublin	1.000000	0.658683	0.888304	0.000000	0.102203	0.148148	0.0	1.000000	0.011347
Frankfurt	0.346154	0.733533	0.260819	0.567376	0.167737	0.037037	0.5	0.500000	0.712623
Paris	0.670940	0.769461	1.000000	0.496454	0.032920	0.925926	0.5	0.428571	0.975433
Warsaw	0.431624	0.000000	0.155556	0.319149	0.000000	0.555556	0.5	0.785714	0.513589
Madrid	0.722222	0.338323	0.304094	0.471631	0.033625	0.037037	0.5	0.428571	0.777025
Lisbon	0.230769	0.122754	0.000000	0.460993	0.002395	0.888889	0.0	0.000000	0.179570
Stockholm	0.854701	0.793413	0.004678	0.765957	0.193409	0.222222	0.5	0.785714	0.149558
Helsinki	0.863248	0.703593	0.483626	0.287234	0.052930	0.111111	1.0	1.000000	0.019193
Milan	0.000000	0.541916	0.087719	0.301418	0.082759	0.037037	0.5	0.285714	1.000000
Brussels	0.807692	0.886228	0.070175	1.000000	0.600960	0.407407	0.5	0.642857	0.122702
Amsterdam	0.679487	0.772455	0.713450	0.393617	0.041654	0.037037	0.5	0.714286	0.263401
Kopenhagen	0.666667	1.000000	0.799415	0.177305	0.082741	0.000000	0.5	0.785714	0.000000
Vienna	0.555556	0.715569	0.177193	0.817376	0.057598	0.518519	0.5	0.928571	0.050220
Prague	0.196581	0.074850	0.154971	0.762411	0.028409	0.851852	0.5	0.500000	0.223478

Now let's have a look at the data types of the features in the dataframe.

```
df.dtypes
```

```
EducationLevel           float64
LaborCost                float64
AvMonthlyRent            float64
HICP                     float64
Crimes                   float64
NoUniversities           float64
DiffTimeZones            float64
LifeSatisfactionIndex    float64
ActiveCompanies          float64
dtype: object
```

In order to have a starting point for the visualization, we request the geographical coordinates of a center position in Europe. For this the city of Munich is chosen.

```
address_Muc = 'Munich, Germany'

geolocator = Nominatim(user_agent="to_explorer")
location = geolocator.geocode(address_Muc)
latitude_Muc = location.latitude
longitude_Muc = location.longitude
print('The geograpical coordinate of Munich are {}, {}.'.format(latitude_Muc, longitude_Muc))
```

```
The geograpical coordinate of Munich are 48.1371079, 11.5753822.
```

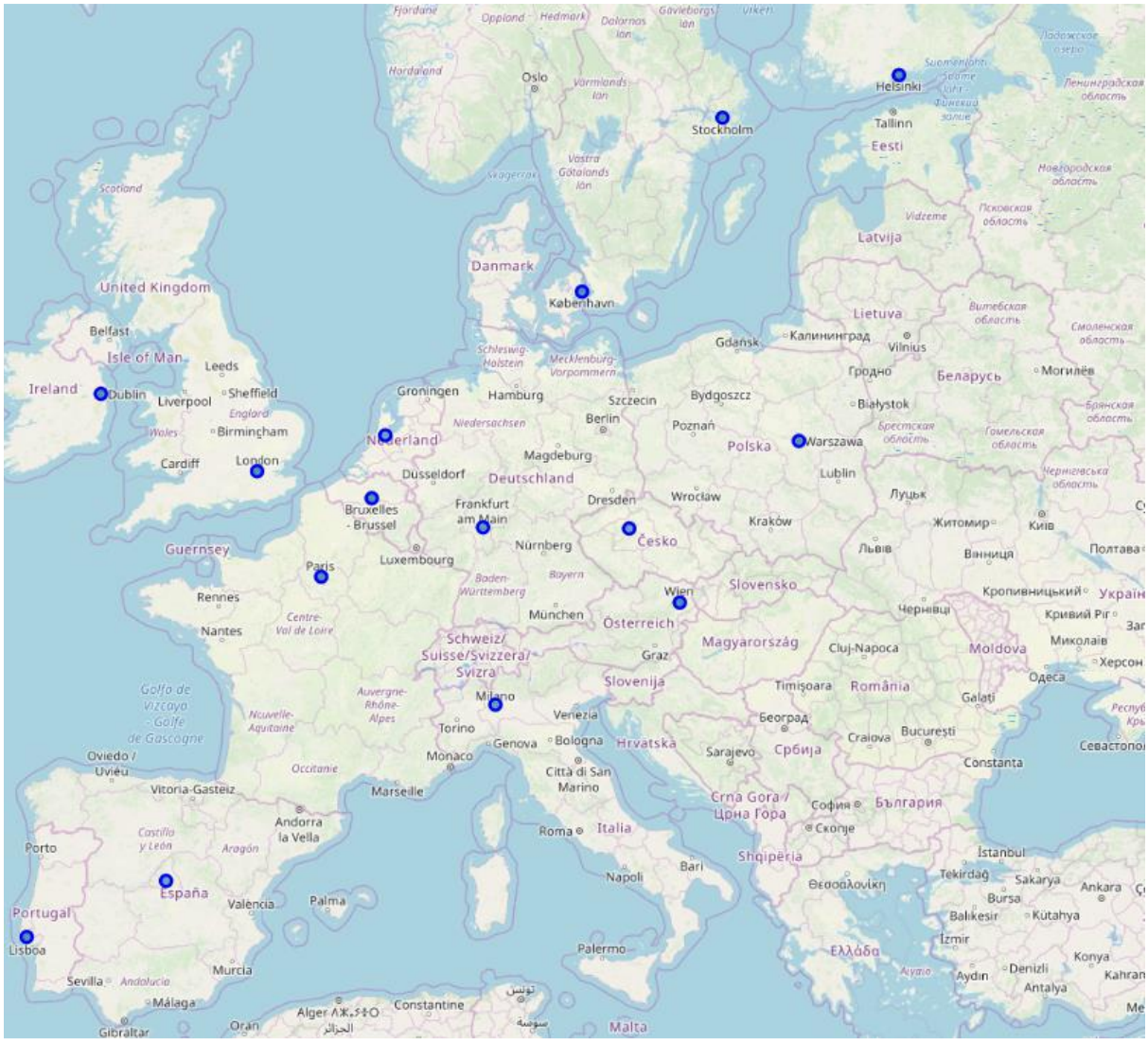
Now we to get the get the geographical coordinates of the cities which are to be evaluated.

```
cities = ['London, England', 'Dublin, Ireland', 'Frankfurt, Germany', 'Paris, France', 'Warsaw, Poland', 'Madrid, Spa
city_locs = pd.DataFrame(columns = ('city','latitude','longitude'))

for city in cities:
    address = city
    location = geolocator.geocode(address)
    latitude = location.latitude
    longitude = location.longitude
    data = {'city':[city], 'latitude':[latitude], 'longitude':[longitude]}
    data = pd.DataFrame(data = data)
    city_locs = city_locs.append(data)

city_locs
```

	city	latitude	longitude
0	London, England	51.507322	-0.127647
0	Dublin, Ireland	53.349764	-6.260273
0	Frankfurt, Germany	50.110644	8.682092
0	Paris, France	48.856610	2.351499
0	Warsaw, Poland	52.233717	21.071411
0	Madrid, Spain	40.416705	-3.703582
0	Lisbon, Portugal	38.707751	-9.136592
0	Stockholm, Sweden	59.325117	18.071093
0	Helsinki, Finland	60.167409	24.942568
0	Milan, Italy	45.466800	9.190500
0	Brussels, Belgium	50.843671	4.367437
0	Amsterdam, Netherlands	52.374540	4.897976
0	Copenhagen, Denmark	55.686724	12.570072
0	Vienna, Austria	48.208354	16.372504
0	Prague, Czech Republic	50.087465	14.421254

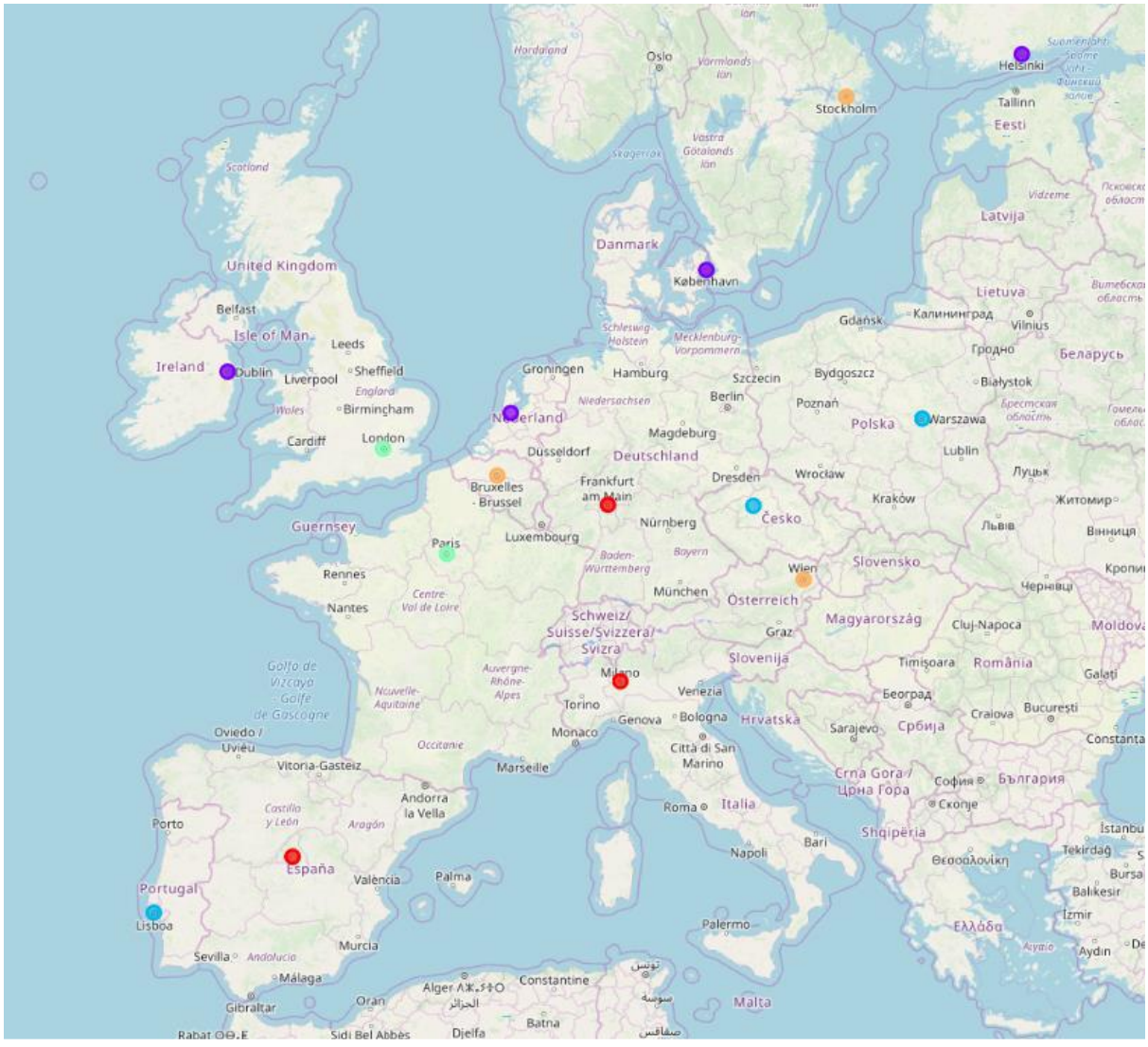
Then we create the first map to visualize the geographical positions of the evaluated cities.

```
# create map of Toronto using latitude and longitude values
map_muc = folium.Map(location=[latitude_Muc,longitude_Muc], zoom_start=5)

# add markers to map
for city ,latitude, longitude, in zip(city_locs['city'], city_locs['latitude'], city_locs['longitude']):
    #label = '{}, {}'.format(city)
    label = city
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [latitude, longitude],
        radius=5,
        popup=label,
        color='blue',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.7,
        parse_html = False).add_to(map_muc)

map_muc
```



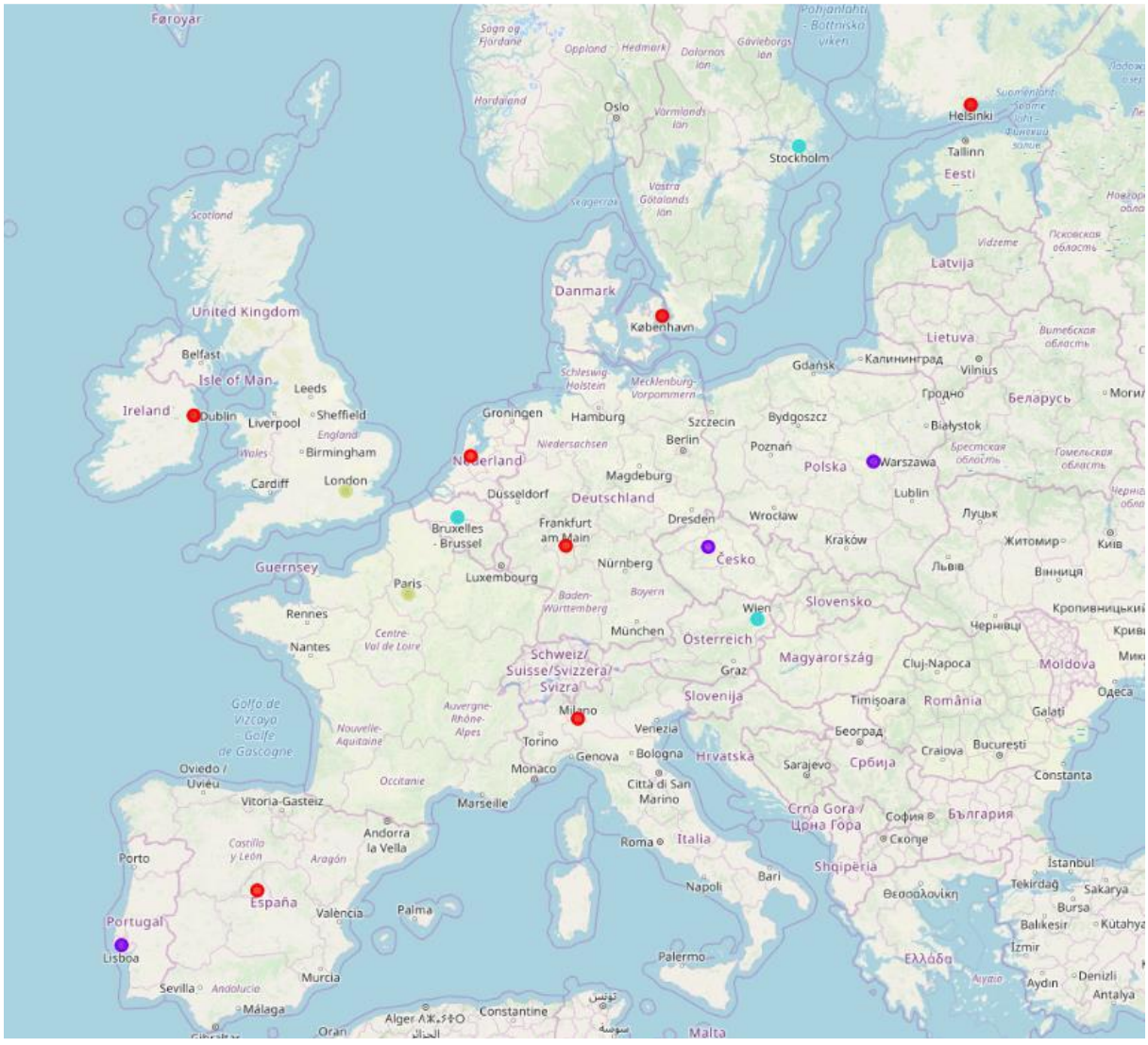
Then we conduct the first cluster analysis using the KMeans algorithm. The results are displayed on a separate map.

```
kclusters = 5
kmeans = KMeans(n_clusters=kclusters, random_state=0).fit(df)

cluster_km = []
for i in kmeans.labels_:
    cluster_km.append(i)

city_locs['cluster_km'] = cluster_km

# create map
map_km_clusters = folium.Map(location=[latitude_Muc, longitude_Muc], zoom_start=5)

# 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 city, lat, lon, cluster_km in zip(city_locs['city'], city_locs['latitude'], city_locs['longitude'], city_locs['c
    label = folium.Popup(str(city) + ' Cluster ' + str(cluster_km), parse_html=True)
    folium.CircleMarker(
        [lat, lon],
        radius=6,
        popup=label,
        color=rainbow[int(cluster_km)-1],
        fill=True,
        fill_color=rainbow[int(cluster_km)-1],
        fill_opacity=0.7).add_to(map_km_clusters)

map_km_clusters
```



Then we conduct the second cluster analysis using the DBSCAN algorithm. The results are displayed on a separate map.

```
dbscan = DBSCAN(eps = 0.8, min_samples = 2).fit(df)

cluster_db = []
for i in dbscan.labels_:
    cluster_db.append(i)

city_locs['cluster_db'] = cluster_db

# create map
map_db_clusters = folium.Map(location=[latitude_Muc, longitude_Muc], zoom_start=5)

# set color scheme for the clusters
x = np.arange(len(cluster_db))
ys = [i + x + (i*x)**2 for i in range(len(np.unique(cluster_db)))]
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 city, lat, lon, cluster_db in zip(city_locs['city'], city_locs['latitude'], city_locs['longitude'], city_locs['c
    label = folium.Popup(str(city) + ' Cluster ' + str(cluster_db), parse_html=True)
    folium.CircleMarker(
        [lat, lon],
        radius=5,
        popup=label,
        color=rainbow[int(cluster_db)-1],
        fill=True,
        fill_color=rainbow[int(cluster_db)-1],
        fill_opacity=0.7).add_to(map_db_clusters)

map_db_clusters
```



Results and discussion

The analysis shows that the only candidate city which can be considered similar enough is Paris, France.

Using the KMeans algorithm, various groupings can be obtained, with only a small range of k clusters giving reasonable results. For only 2 clusters, the cities of Lisbon, Madrid, Paris, Frankfurt, Milan, Prague and Warsaw are considered to be similar to London. As this result is not helping a decision, it is not used for the decision process. Within the range of 3-5 clusters, only Paris is considered to be an alternative to London. For a number of clusters higher than 5, there is no similar city in the evaluated candidate group. Therefore, the only reasonable configuration to use for the decision process is with 3-5 clusters.

For DBSCAN, the number of minimum samples was set to 2 samples, as this would be enough to provide at least one candidate city which is similar to London. Therefore, the other parameter changed was the epsilon parameter. Reducing this parameter from 2 to 0.1 in steps of 0.05, it becomes clear that values above 1.2 do not provide a similar city. Between 0.8 and 1.2, again Paris is the only city being considered an alternative to London. At 0.75, the next city added to the potential locations is Dublin. Below 0.75, further cities are added to this list, so that a reasonable recommendation cannot be given.

Conclusion

Given the results discussed above, it is clear that only 2 cities can be considered a realistic alternative compared to the current location in London: Paris and Dublin. This recommendation can be used for a in detail review of locations, involving other factors which were not included in this investigation, e.g. taxes to pay, infrastructure etc.