What countries to Travel in a Limited Time

Athar Kamal

4th November, 2019

1. Introduction

Background

People all around the world have ideas of travelling to Europe in the summers for a “Euro trip”. Students can take their time because of summer breaks, and enjoy the continent’s 44 countries. However, people who work have limited time and cannot leave their places of work for extended period of time, but the need to enjoy most of Europe is still with them. These travelers want to maximize the number of cities they can travel, and enjoy the different cultures.

Problem & Interest

This project aims to cluster the most populous and famous European cities by estimating the population density as well as the closeness of the airport to the city center so that travelers (who have limited time) can enjoy and maximize different cities and countries.

2. Data Acquisition and Sorting

Data Sources

The initial data about the European countries and cities has been acquired from the Wikipedia page with the list of urban areas. The data was scraped through the *beautifulsoup4* tool and saved into an initial dataframe. Using *GeoPy* , the latitude and longitude of each city were imported to form a dataset. Furthermore, Foursquare API was used to search the cities for airport terminals and the distance of the airport from the city.

Data Cleaning

Scraping through the Wikipedia page gave us these five columns (Rank, City, Country, Population and Density). We got a dataframe of size 86 rows x 5 columns. The data was further enhanced by finding the latitude and longitude of each city. However, 3 of the cities did not have any latitude and longitude and were thus removed. The cities were mapped on Europe through the folium library, which revealed that 4 more cities

had wrong latitude and longitude and these were subsequently removed as well. More cities were dropped using the Foursquare API, to estimate whether an airport near a city existed within a radius of 5 kilometers. Thus, our final dataset contained 42 cities.

3. Data Analysis & Methodology

This section provides the step by step methods to finding the clusters of the European cities to travel to fast.

Step 1: Scrape data

The first part of the methodology was to scrape the Wikipedia page. This was done using beautifulsoup4 tool, by finding the table and extracting the required data from it. The resulting dataframe was as follows:

```
Rank                 int64
City                object
Country             object
Population           int64
Density per km^2     int64
dtype: object
RangeIndex(start=0, stop=86, step=1)
```

[11]:

	Rank	City	Country	Population	Density per km^2
0	1	Paris	France	10950000	3800
1	2	London	United Kingdom	10470000	5900
2	3	Ruhr	Germany	6670000	2800
3	4	Madrid	Spain	6610000	4600
4	5	Milan	Italy	5280000	2800
...	...	...	...	...	...
81	85	Aachen	Germany	545000	1500
82	87	Bologna	Italy	530000	3400
83	89	Grenoble	France	515000	985
84	90	Saarbrücken	Germany	510000	2200
85	91	Douai	France	510000	1100

86 rows × 5 columns

Step 2: Finding Co-ordinates of cities

The second step was for us to find the latitude and longitude of the cities, so that it could be used in the Foursquare API. The Geopy too/library was used to find the latitude and longitude of each city and was merged with the dataframe developed in step 1. The following dataframe was established as a result:

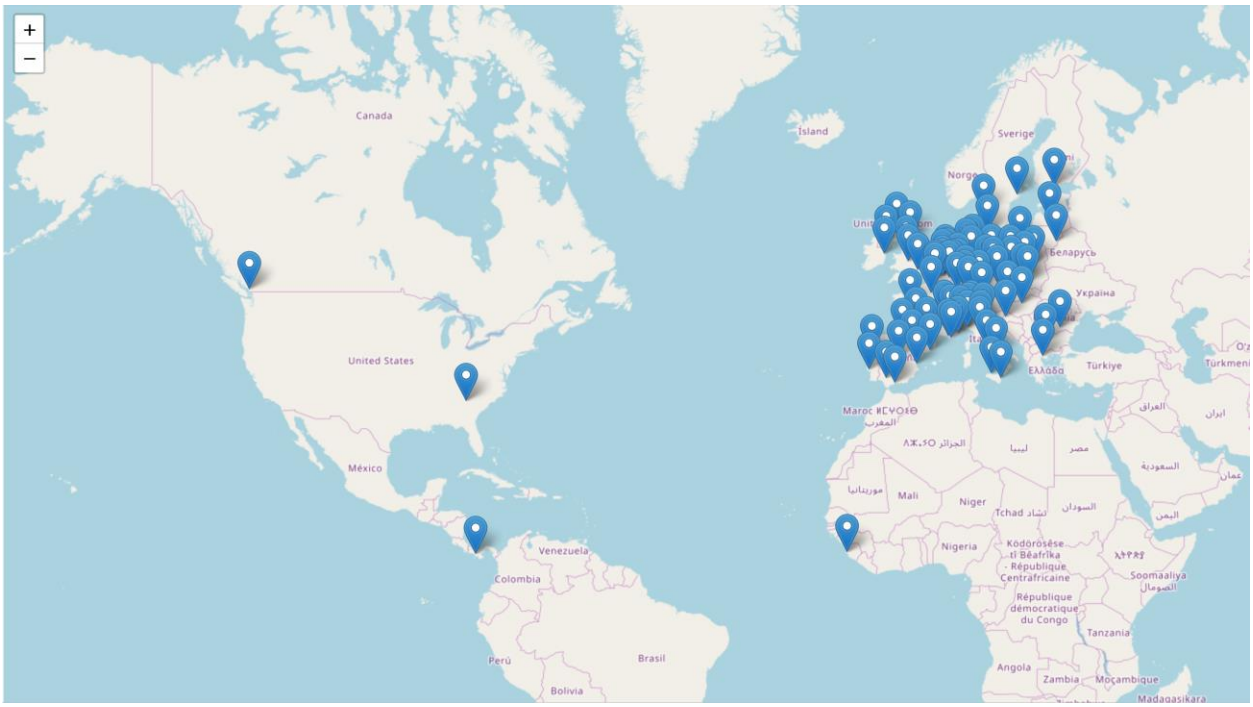
[12]:

	Rank	City	Country	Population	Density per km^2	Latitude	Longitude
0	1	Paris	France	10950000	3800	48.856610	2.351499
1	2	London	United Kingdom	10470000	5900	51.507322	-0.127647
2	3	Ruhr	Germany	6670000	2800	51.517518	7.143918
3	4	Madrid	Spain	6610000	4600	40.416705	-3.703582
4	5	Milan	Italy	5280000	2800	45.466800	9.190500
...	...	...	...	...	...	...	...
78	85	Aachen	Germany	545000	1500	50.776351	6.083862
79	87	Bologna	Italy	530000	3400	44.493671	11.343035
80	89	Grenoble	France	515000	985	45.187560	5.735782
81	90	Saarbrücken	Germany	510000	2200	49.234362	6.996379
82	91	Douai	France	510000	1100	50.370368	3.076138

83 rows × 7 columns

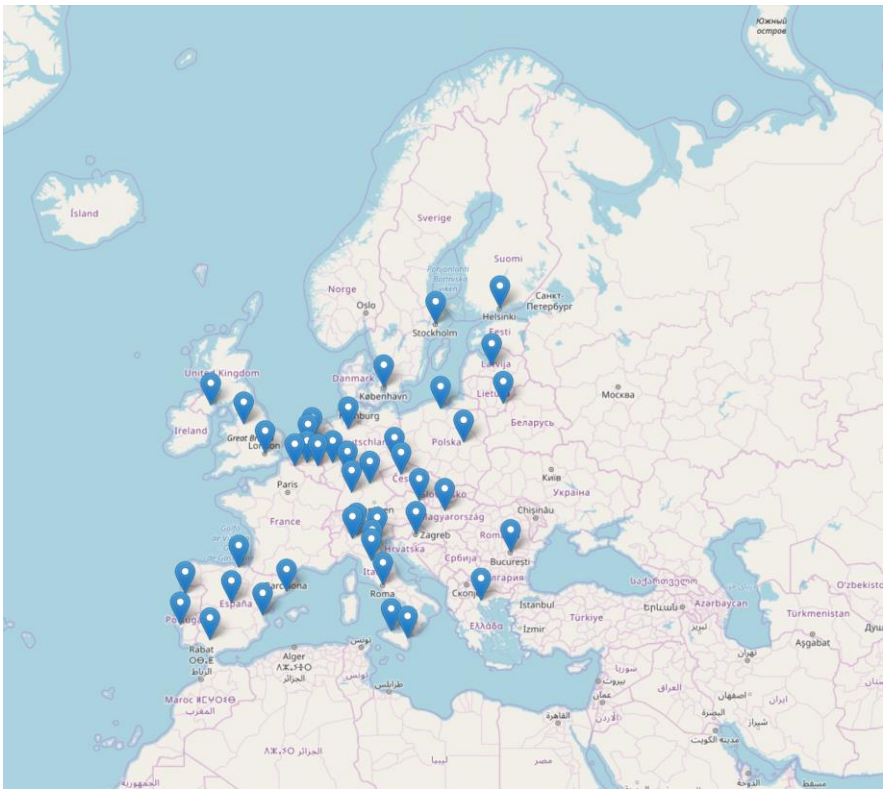
Step 3: Plotting using Folium

The dataframe developed in step 2, was then plotted with the help of folium library for visual investigation. This investigation revealed that 4 cities had wrong latitude and longitude and thus were dropped from the data set. The plotted map was as follows:



Step 4: Using Foursqaure API to get location data for Airports

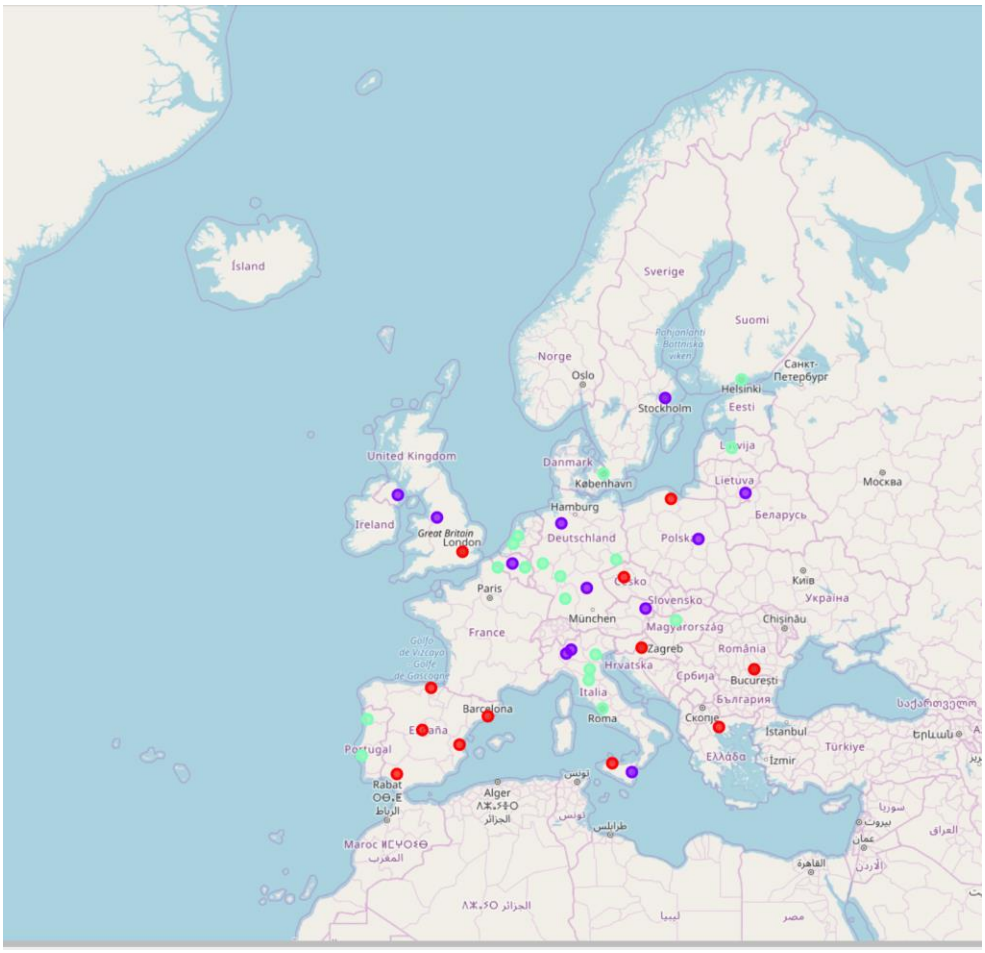
The Foursquare API was used to access the location data of cities and whether airports existed within a radius of 5 kilometers from the city. Results revealed that only 42 cities had airport near them and thus this was the dataframe used for k-means clustering. Map of 42 cities is as follows:



4. Results

Performing k-Means Clustering

K-means clustering was performed in these cities to estimate the clusters which have the closest airports to the city and depending on the population densities of each cities. This clusters were divided into 3 groups and the resulting map was as follows:



5. Discussion and Conclusion

The visual analysis reveals that two (red and purple) of the three clusters are optimal for maximizing the travel in Europe. Whereas the green colored cluster is concentrated close to Germany and does not cover as many countries. Analysis of the clusters through dataframe (Figure 1 and Figure 2 below) reveals that the red cluster is optimal for faster travels as airports in the cities are closest. This could be because, these cities are some of the most famous tourist destinations. However, there is a downside, which is that these cities are with some of the highest population densities, and may have traffic issues. The purple clusters on the other hand have greater airport to city travel distance, but these are less dense.

To conclude I suggest, that travelers who want to follow the crowd, follow the red cluster, but for those people who want to enjoy the countryside of Europe follow the purple cluster. The green cluster could better be ignored.

Figure 1: Red Cluster

[24]:

	City	Density per km^2	Latitude	Longitude	url	Distance
0	London	5900	51.507322	-0.127647	https://api.foursquare.com/v2/venues/explore?&...	608
1	Madrid	4600	40.416705	-3.703582	https://api.foursquare.com/v2/venues/explore?&...	2148
3	Barcelona	4300	41.382894	2.177432	https://api.foursquare.com/v2/venues/explore?&...	841
9	Prague	4600	50.087465	14.421254	https://api.foursquare.com/v2/venues/explore?&...	1327
13	Bucharest	6500	44.436141	26.102720	https://api.foursquare.com/v2/venues/explore?&...	3151
17	Valencia	5700	39.469901	-0.375951	https://api.foursquare.com/v2/venues/explore?&...	223
23	Sevilla	5600	37.388630	-5.995340	https://api.foursquare.com/v2/venues/explore?&...	1846
24	Gdańsk	5000	54.347629	18.645232	https://api.foursquare.com/v2/venues/explore?&...	1140
28	Thessaloniki	4300	40.640317	22.935272	https://api.foursquare.com/v2/venues/explore?&...	2243
29	Bilbao	5800	43.263005	-2.934992	https://api.foursquare.com/v2/venues/explore?&...	4166
32	Palermo	6000	38.111227	13.352443	https://api.foursquare.com/v2/venues/explore?&...	3631
33	Zagreb	4400	45.813177	15.977048	https://api.foursquare.com/v2/venues/explore?&...	1576

Figure 2: Purple Cluster

[27]:

	City	Density per km^2	Latitude	Longitude	url	Distance
2	Milan	2800	45.466800	9.190500	https://api.foursquare.com/v2/venues/explore?&...	2845
6	Greater Manchester Urban Area	4200	53.385561	-2.340603	https://api.foursquare.com/v2/venues/explore?&...	4371
10	Warsaw	3200	52.233717	21.071411	https://api.foursquare.com/v2/venues/explore?&...	4363
12	Brussels	2600	50.846557	4.351697	https://api.foursquare.com/v2/venues/explore?&...	4173
15	Vienna	3900	48.208354	16.372504	https://api.foursquare.com/v2/venues/explore?&...	3699
18	Stockholm	4300	59.325117	18.071093	https://api.foursquare.com/v2/venues/explore?&...	3347
26	Bergamo	3300	45.694495	9.669873	https://api.foursquare.com/v2/venues/explore?&...	3894
31	Catania	2900	37.502235	15.087380	https://api.foursquare.com/v2/venues/explore?&...	3785
34	Nuremberg	3000	49.453872	11.077298	https://api.foursquare.com/v2/venues/explore?&...	4513
35	Bremen	2400	53.075820	8.807165	https://api.foursquare.com/v2/venues/explore?&...	2988
36	Belfast	3500	54.596441	-5.930276	https://api.foursquare.com/v2/venues/explore?&...	3804
40	Vilnius	2500	54.687046	25.282911	https://api.foursquare.com/v2/venues/explore?&...	4903