CLUSTERING OF CITIES OF THE WORLD

By: Nirali Parekh
Github: nirali25parekh

Ideation:

This project focuses on clustering of similar cities types according to venues in the area.
Hence, by the end, we will be able to find and segregate similar cities based on categories of places and venues that are abundant in the cities.

Method:

Unsupervised Learning (Clustering) :

"**Clustering**" is the process of grouping similar entities together. The goal of this technique is to find similarities in the data point and group similar data points together.
In our case, we are obtaining data from various sources and using it, we are clustering or 'grouping' cities together.

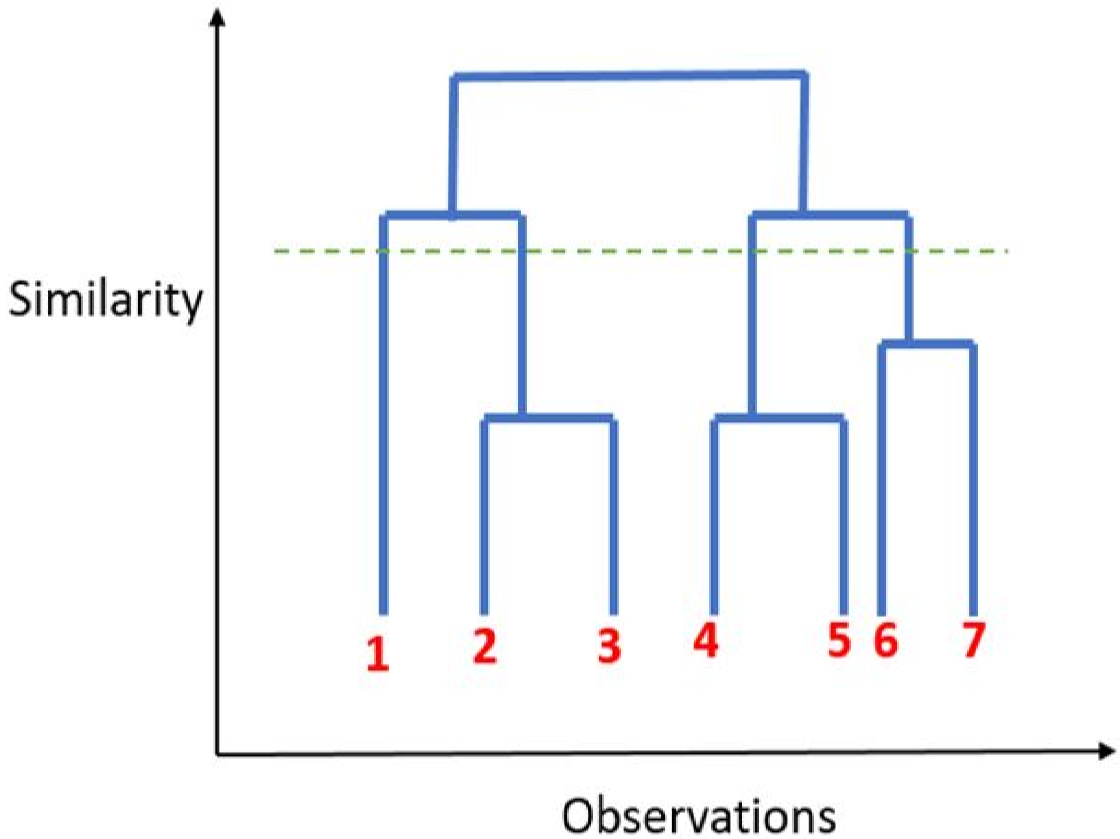

Fig: Clustering algorithm

Cluster analysis groups objects (observations, events) based on the information found in the data describing the objects or their relationships. The goal is that the objects in a group will be similar (or related) to one other and different from (or unrelated to) the objects in other groups. The greater the similarity (or homogeneity) within a group, and the greater the difference between groups, the ―better‖ or more distinct the clustering. Data mining is the process of analysing data from different viewpoints and summerising it into useful information. Data mining is one of the top research areas in recent days. Cluster analysis in data mining is an important research field it has its own unique position in a large number of data analysis and processing.

Relationship:

Upon observing, we see that the cities that have the most similar venues ie. categories like Cafe, Pubs, Parks, Theatres, etc.

Technologies / Libraries used:

- pandas (for dataframe related functionalities)
- geopy (to convert place to longitude latitude)
- matplotlib (for plotting and data visualization)
- sklearn (for KMeans algorithm)
- folium (for rendering map)

Process:

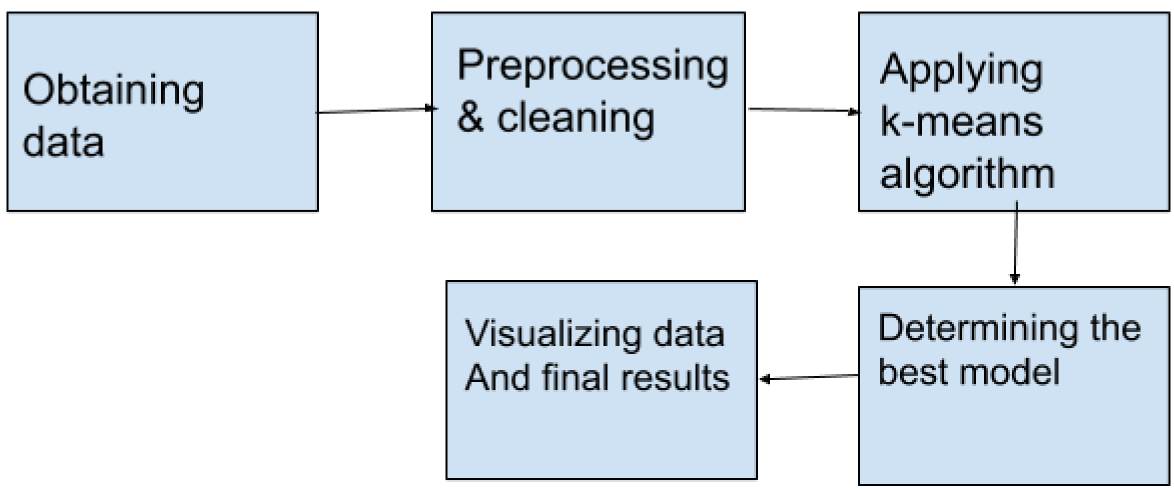

Fig: process of the cities clustering flow

Step 1.   Obtaining Data:

- The geographical data was obtained by the free 'Single Maps' Database. It gave us back about 15493 cities with its information.

Since that is too large of a database for computing efficiency, it’s better to use a subset of it. Now, we could use a fixed number random training samples, or filter examples based on some feature.
https://simplemaps.com/data/world-cities

shape of df (15493, 11)

	city	city_ascii	lat	lng	country	iso2	iso3	admin_name	capital	population	id
0	Tokyo	Tokyo	35.6850	139.7514	Japan	JP	JPN	Tōkyō	primary	35676000.0	1392685764
1	New York	New York	40.6943	-73.9249	United States	US	USA	New York	NaN	19354922.0	1840034016
2	Mexico City	Mexico City	19.4424	-99.1310	Mexico	MX	MEX	Ciudad de México	primary	19028000.0	1484247881
3	Mumbai	Mumbai	19.0170	72.8570	India	IN	IND	Mahārāshtra	admin	18978000.0	1356226629
4	São Paulo	Sao Paulo	-23.5587	-46.6250	Brazil	BR	BRA	São Paulo	admin	18845000.0	1076532519

Fig: The data obtained from simple Maps Database

- So,I used only cities with a population > 3,00,000 for the model. Hence, we obtained only 2051 cities which was adequate for training our model.
- It was found that there are 570 unique categories of venues obtained.

- The information from the database (considered as features) were:
  - City
  - Latitude
  - Longitude
  - Country
  - Population
  - Capital typel (if any)
- Here, FourSquare API was used to collect data of venues for each city. https://developer.foursquare.com/docs/

	City	City Latitude	City Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	New York, United States	40.6943	-73.9249	Sunrise/Sunset	40.693544	-73.922875	Cocktail Bar
1	New York, United States	40.6943	-73.9249	Hearts Coffee	40.692155	-73.926602	Coffee Shop
2	New York, United States	40.6943	-73.9249	Wonderville	40.692394	-73.927500	Bar
3	New York, United States	40.6943	-73.9249	Kichin	40.697706	-73.927023	Korean Restaurant
4	New York, United States	40.6943	-73.9249	Mood Ring	40.697888	-73.926955	Bar
5	New York, United States	40.6943	-73.9249	Broadway Pizza	40.693314	-73.928931	Pizza Place
6	New York, United States	40.6943	-73.9249	BK Bagels	40.693813	-73.929598	Bagel Shop
7	New York, United States	40.6943	-73.9249	Papelon Con Limon	40.693959	-73.930210	Food Truck
8	New York, United States	40.6943	-73.9249	The Buren Cafe + Bar	40.692019	-73.926045	Coffee Shop
9	New York, United States	40.6943	-73.9249	Santa Panza	40.694715	-73.930692	Pizza Place
10	New York, United States	40.6943	-73.9249	brooklyn whiskers	40.695585	-73.929244	Bakery
11	New York, United States	40.6943	-73.9249	The Gateway	40.690926	-73.924902	Music Venue

Fig: The venues information from FourSquare API

```
all_venues = getNearbyVenues(names=cities_data['City'], latitudes=cities_data['lat'],
longitudes=cities_data['lng'])

New York, United States 34
Mexico City, Mexico 17
Mumbai, India 5
São Paulo, Brazil 5
Delhi, India 7
Shanghai, China 33
Los Angeles, United States 2
Dhaka, Bangladesh 5
Buenos Aires, Argentina 50
Cairo, Egypt 15
Rio de Janeiro, Brazil 23
Ōsaka, Japan 4
Beijing, China 16
Manila, Philippines 16
Moscow, Russia 49
Istanbul, Turkey 33
Paris, France 50
Seoul, Korea, South 50
Jakarta, Indonesia 33
Guangzhou, China 22
```

Fig: The name of city and number of venues obtained

- Hence, we have all the data we need. It's time for preprocessing

Step 2: Preprocessing and Cleaning:

- This consists of getting rid of unwanted data (null values) and tweaking data according to our needs.
- Here we, used One-Hot-Encoding to convert our categorical values to digits.

	City	Accessories Store	Adult Boutique	American Restaurant	Argentinian Restaurant	Art Gallery	Bagel Shop	Bakery	Bar	Beer Garden	...	Taco Place	Tea Room	Theater	Thrift / Vintage Store
0	Beijing, China	0.000000	0.00	0.000000	0.00	0.0	0.00	0.0	0.00	0.0625	...	0.0	0.000000	0.000000	0.0
1	Buenos Aires, Argentina	0.000000	0.02	0.000000	0.02	0.0	0.02	0.0	0.02	0.0000	...	0.0	0.000000	0.040000	0.0
2	Cairo, Egypt	0.066667	0.00	0.000000	0.00	0.0	0.00	0.0	0.00	0.0000	...	0.0	0.066667	0.133333	0.0
3	Delhi, India	0.000000	0.00	0.142857	0.00	0.0	0.00	0.0	0.00	0.0000	...	0.0	0.000000	0.000000	0.0
4	Dhaka, Bangladesh	0.000000	0.00	0.000000	0.00	0.0	0.00	0.0	0.00	0.0000	...	0.0	0.000000	0.000000	0.0

	City	City Latitude	City Longitude	Venue	Venue Latitude	Venue Longitude	Accessories Store	Adult Boutique	American Restaurant	Argentinian Restaurant	...	Taco Place	Tea Room	Theater
0	New York, United States	40.6943	-73.9249	Sunrise/Sunset	40.693544	-73.922875	0	0	0	0	...	0	0	0
1	New York, United States	40.6943	-73.9249	Hearts Coffee	40.692155	-73.926602	0	0	0	0	...	0	0	0
2	New York, United States	40.6943	-73.9249	Wonderville	40.692394	-73.927500	0	0	0	0	...	0	0	0
3	New York, United States	40.6943	-73.9249	Kichin	40.697706	-73.927023	0	0	0	0	...	0	0	0

Fig: One Hot Encoding of the venue Categories as features

- Redundant columns are also removed.
  - We do not need country name, and capital info for our model.
  - City column is repeated hence, we remove one of them.

Step 3: K Means Algorithm:

- Now, we apply KMeans algorithm to our training data and obtain cluster labels.
- We model or algorithm for k in range 1 to 15 and then check which k gives the best performance.

```
1 - [0 0 0 0 0 0 0 0 0 0]
2 - [0 0 0 0 0 1 0 0 0 0]
3 - [1 1 1 1 2 0 1 1 1 1]
4 - [0 0 0 0 1 3 0 0 0 0]
5 - [0 0 0 0 2 1 0 0 0 0]
6 - [5 1 1 1 0 3 5 1 1 1]
7 - [0 0 0 4 2 6 0 0 0 1]
8 - [5 1 1 7 2 0 5 1 1 6]
9 - [5 4 1 8 3 2 5 1 1 7]
10 - [7 1 8 4 3 2 7 6 1 9]
11 - [ 1  3  0  8  4  2  1 10  9  7]
12 - [10  0  9  6  3  1  5  7 11  8]
13 - [ 6  3 10  8  4  1 12 11  7  5]
14 - [11  4  6  9  3  2  7 10  1  0]
```

Fig: KMeans Labels fortop 10 for k in range 1 to 14

Step 4: Choosing the best model:

- As seen in the image above, the squared error decreases as k increases, hence we pick the adequate value of 13.
- Hence, the classification labels are added to our data.

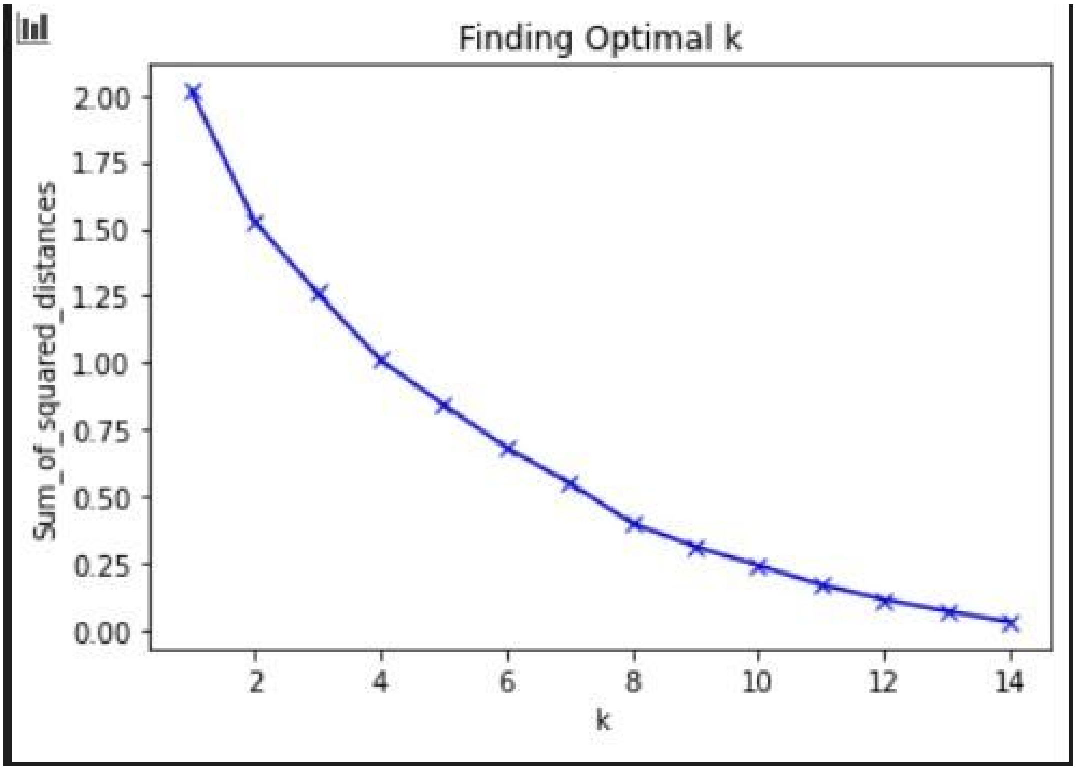

Fig: graph of squared error vs K

Step 5: Visualizing Data and Final Results:

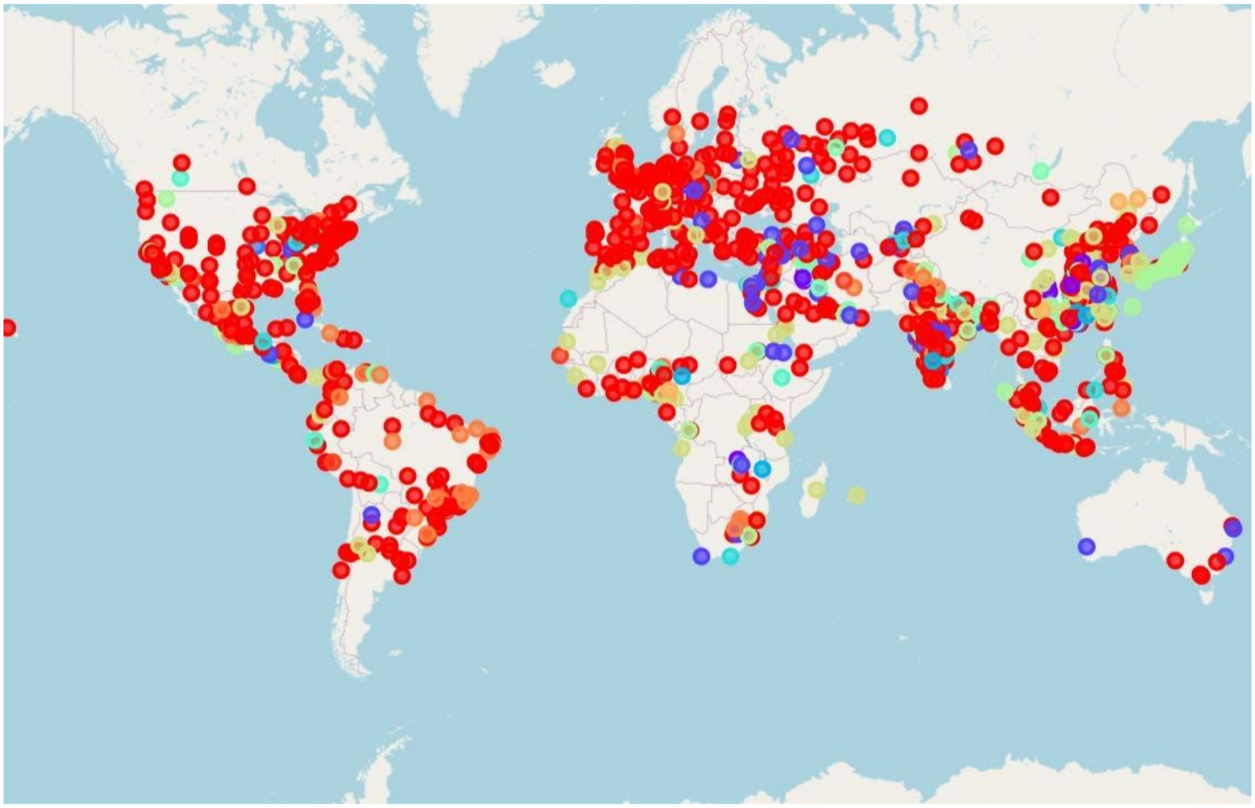

Fig: Map showing cities colored cities representing clusters

Application and Uses

1. It enables an individual to find and explore various cities similar to the ones they need.
   For eg. The users who might wish to migrate to other cities can find similar places to their taste of lifestyle and venues.
    Fig: top 10 venues for cities in a table

	City	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
13	Akola, India	ATM	Café	Asian Restaurant	Dessert Shop	Indian Restaurant	Food Court	Football Stadium	Food Truck	Food Stand	Food Service
16	Al Manşūrah, Egypt	Café	Flea Market	Train Station	Fish Taverna	Fishing Store	Fish Market	Flower Shop	Fondue Restaurant	Food	Fountain
19	Al ‘Ayn, United Arab Emirates	Café	Gym / Fitness Center	Fast Food Restaurant	Athletics & Sports	Ice Cream Shop	Bakery	Juice Bar	Hotel	Donut Shop	Cafeteria
30	Amman, Jordan	Café	Middle Eastern Restaurant	Bookstore	Hotel	Dessert Shop	Arts & Crafts Store	Italian Restaurant	Theater	Coffee Shop	Plaza
38	Ansan, Korea, South	Intersection	Café	Zoo Exhibit	Food Court	Forest	Football Stadium	Food Truck	Food Stand	Food Service	Food & Drink Shop

	City	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
23	Alexandria, Egypt	Park	Middle Eastern Restaurant	Pizza Place	Cosmetics Shop	Fish Taverna	Fish Market	Fishing Store	Flea Market	Flower Shop	Fountain
60	Az Zarqā’, Jordan	Park	Food & Drink Shop	Forest	Football Stadium	Food Truck	Food Stand	Food Service	Food Court	Food	Franconian Restaurant
76	Baotou, China	Park	Food & Drink Shop	Forest	Football Stadium	Food Truck	Food Stand	Food Service	Food Court	Food	Franconian Restaurant
147	Bytom, Poland	Park	Pool	Forest	Football Stadium	Food Truck	Food Stand	Food Service	Food Court	Food & Drink Shop	Zoo Exhibit
174	Changde, China	Park	Pier	Cosmetics Shop	Fish Market	Fish Taverna	Fishing Store	Flea Market	Flower Shop	Fondue Restaurant	Fountain