Coursera Capstone

IBM Applied Data science Capstone

**Multicultural Cities Similarity Detection and Visualization**

# Introduction:

Predict and visualize similarity between multicultural cities like New York/Toronto across the world. If a person from New York migrates to a new city like Toronto and if he wants to start his living in the similar neighbourhood where he was living earlier OR if he wants to start a business in similar area, what place can we recommend?

Basically in this problem we will collect data about New York neighborhoods, and pull location data from Foursquare to understand variety of venues. Same exercise would be done for city of Toronto and both the data sets would be combined and clustered using K-means clustering, and similar neighborhoods would be visualized on the world map

With the growth of smart-phones, and location-based social networks, data is being generated like never before. There are nevertheless many practical questions in urban computing that require the comparison across cities. For example, a job seeker with may wish to focus her search on a single city with jobs that best match her qualifications, rather than dispersing her search efforts across multiple cities. Likewise, a large corporation looking to expand its locations might perhaps select cities it wishes to expand into before considering particular sites or neighborhoods. Additionally, many within-city computations might be aided by modelling a city’s relationship to other cities. For example, a person buying or renting a home in a new city might want to be able to compare the neighborhoods of the city to other neighborhoods in different cities. One issue in comparing spatial regions such as cities is the normalization of absolute data, since often raw data different contexts are incomparable.

# Data:

Newyork and Toronto neighbourhood data would be captured from wikipedia. Also data about longitude and latitude of the data can be extracted from Web.Venue data from Foursquare would be extracted for each neighbourhood's 5km radius. Both the data sets would be merged for comparison and clustering. Number of venues for each neighbourhood would be aggregated using pandas dataframe which can becomes base for similarity between the locations.

Exploring these ideas empirically using real world data re-quires that we gather both the description, or the categorizations, of the venues in a city, as well as information about the city’s municipal neighborhood boundaries. For the venues , we collected data from the widely used location-based Social Network (LBSN) Foursquare. Users of Foursquare “checkin” to their current location on their mobile device by selecting it from a list of nearby named venues. Their check-in is then broadcast to their social connections. Foursquare also specify a hierarchical categorical description to avenue, such as “Restaurant” and “Mexican Restaurant”. We call higher-level (more general) categories the primary category, and we call lower level (more specific) categories the secondary category. An interesting side-benefit of such collaborative tagging is that as the system’s user base increases ,an accurate and up-to-date crowd-sourced representation of the venues types within a city is naturally accumulated. In this work we exploit this natural data for our empirical study. The venue information is easily accessible through a public API, and all venues are annotated with categories of different granularities which represent a natural semantic grouping for venues.

# Methodology:

This problem involves data about various neighbourhoods in the city. Various venues and its categories are extracted using Foursquare APIs. Pandas dataframes are used for data understanding and wrangling wherever required. Exploratory data analysis to understand correlation between various fields was not much required as problem we are solving belongs to unsupervised learning category.

Various neighbourhoods and and its venue categories are merged together in pandas dataframe and mean of number of venues for each neighborhood is captured before applying clustering algorithm.

K-means clustering:

Kmeans algorithm is an iterative algorithm that tries to partition the dataset into Kpre-defined distinct non-overlapping subgroups (clusters) where each data point belongs to only one group. It tries to make the intra-cluster data points as similar as possible while also keeping the clusters as different (far) as possible. It assigns data points to a cluster such that the sum of the squared distance between the data points and the cluster’s centroid (arithmetic mean of all the data points that belong to that cluster) is at the minimum. The less variation we have within clusters, the more homogeneous (similar) the data points are within the same cluster.

The way kmeans algorithm works is as follows:

Specify number of clusters K.

Initialize centroids by first shuffling the dataset and then randomly selecting K data points for the centroids without replacement.

Keep iterating until there is no change to the centroids. i.e assignment of data points to clusters isn’t changing.

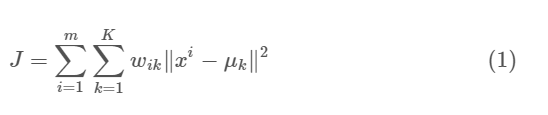
Compute the sum of the squared distance between data points and all centroids.

Assign each data point to the closest cluster (centroid).

Compute the centroids for the clusters by taking the average of the all data points that belong to each cluster.

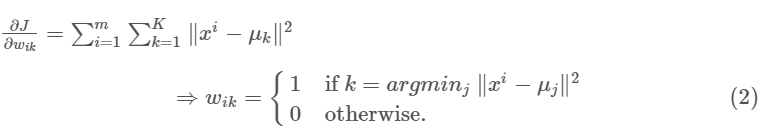
The approach kmeans follows to solve the problem is called Expectation-Maximization. The E-step is assigning the data points to the closest cluster. The M-step is computing the centroid of each cluster. Below is a break down of how we can solve it mathematically (feel free to skip it).

The objective function is:



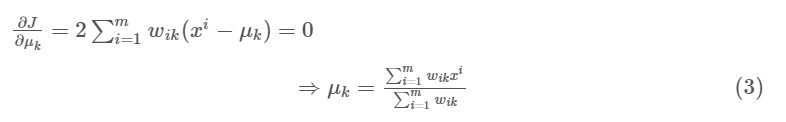
where wik=1 for data point xi if it belongs to cluster k; otherwise, wik=0. Also, μk is the centroid of xi’s cluster.

It’s a minimization problem of two parts. We first minimize J w.r.t. wik and treat μk fixed. Then we minimize J w.r.t. μk and treat wik fixed. Technically speaking, we differentiate J w.r.t. wik first and update cluster assignments (E-step). Then we differentiate J w.r.t. μk and recompute the centroids after the cluster assignments from previous step (M-step). Therefore, E-step is:



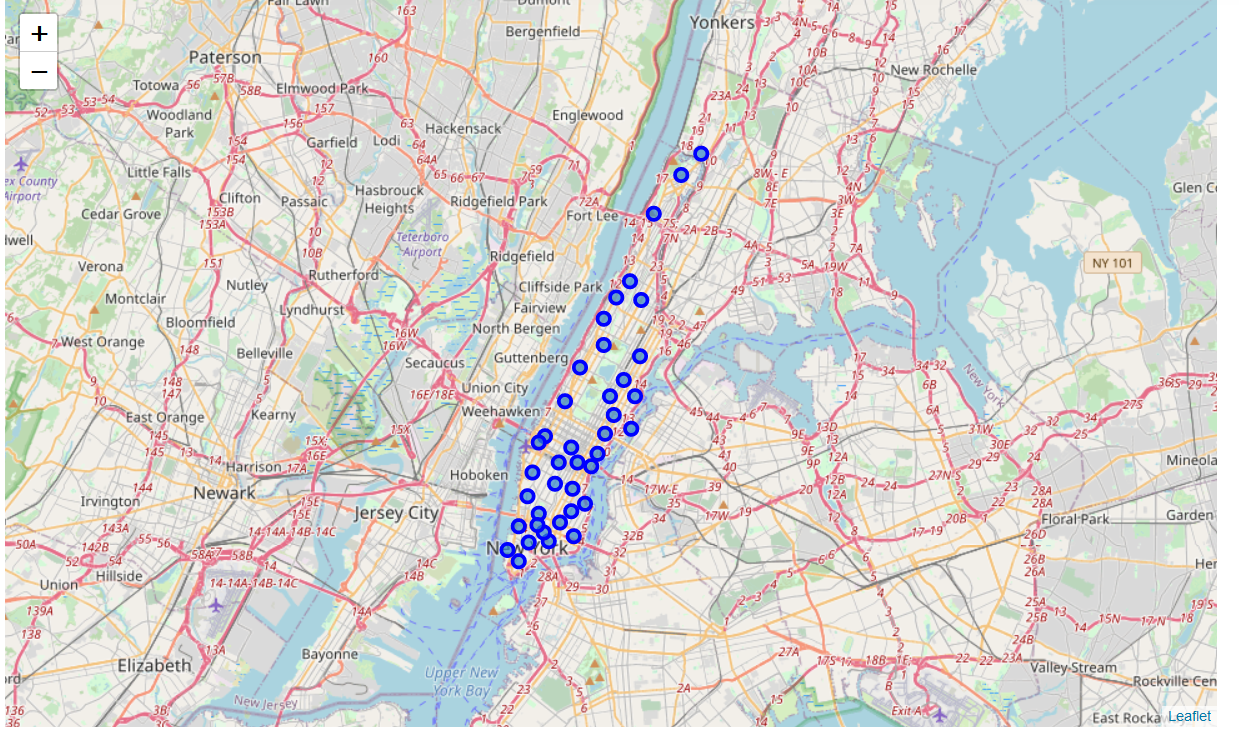
In other words, assign the data point xi to the closest cluster judged by its sum of squared distance from cluster’s centroid.

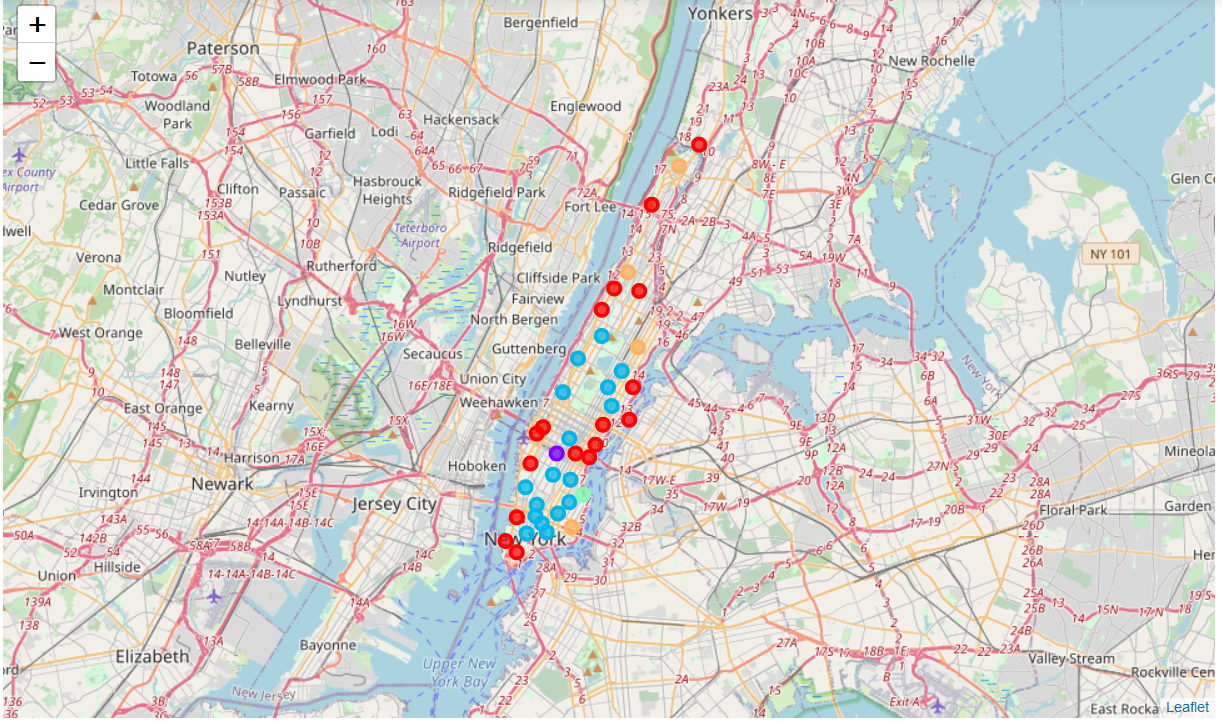
And M-step is:



Which translates to recomputing the centroid of each cluster to reflect the new assignments.

# Results:





# Conclusion:

In this preliminary investigation we have presented different methods for comparing cities as vectors of venue categories. We have identified and emphasized the choice of aggregation level and shown that it can have significant quantitative and qualitative effects for city-to-city comparisons. We have pre-sented the results of city similarities as well as an analysis on the frequencies of different venue categories and how each representation may affect them. In future work, we want to carry on a similar analysis between neighborhoods of cities,in order to identify similar neighborhoods across cities and to get a better understanding of cities as collections of individual neighborhoods. Finally, we hope that our work will motivate future studies into how to best characterize a city in terms of its venues