RESTAURANT ANALYSIS

- ## Introduction

  In this hypothetical scenario, the client is a food delivery service similar to Zomato or Uber Eats. They are interested in gaining an understanding of the restaurant 'landscape' in Toronto so that they can effectively target the most suitable restaurants to sign up for their service. The restaurants will be categorised by postal code, meaning there may be multiple neighbourhoods per postal code. However, using geographical distancing form one venue to another, it may also be possible to group restaurants in a way other than by geographical or postal code region. By putting restaurants into catchment areas such as these, it may be possible to enable more efficient (i.e. quicker) deliveries than if restaurants are simply categorised by catchment area. There may also be scope for analysing socioeconomic data of residents in each neighbourhood. The data from this analysis will be used to build up an understanding of the restaurant business in Toronto to allow our client to make an informed choice.
- ▯ Find out the number of restaurants by postal code. Postal code will be used instead of neighbourhood as it will be used to categorise delivery areas more efficiently.
- ▯ The data will be sourced as follows:
- o Postal code regions will be web scraped from Wikipedia (https://en.wikipedia.org/iki/List_of_postal_codes_of_Canada:_M)
- o Restaurant data, including data conering ratings, food time and respective geographical distances will be acquired from FourSquare, using their API developer functionality.
- o Geographical data will be accessed from Geospatial Cognitive Class (https://cocl.us/Geospatial_data), including postal codes, neighbourhoods and boroughs as well as longitudinal and latitudinal data.
- ▯ In each postal code region, restaurants will be grouped initially by postal code area. We will then overlay, via means of a map, the numbers and types of each restaurant by
- ▯ DBSCAN clustering will be used to assess distances between restaurants to see if they can be grouped according to metrics other

than post code. This may result in more efficient delivery times. These areas will be referred to as ‘delivery zones’.

- ▯ Using this data, we will then overlay geographic and postal code regions with the new ‘delivery zones’ to see how they align and recommend the best course of action for the client.

- **Data and Methodology.**

The reason for clustering data, regardless of the tools used, is that today there is an abundance of data for venues across a given city, as used but companies such as Google and FourSquare. Therefore, we will be looking to cluster food venues into geographically logical catchment areas. DBSCAN clustering will be used to cluster venues into catchment areas.
DBSCAN offers several advantages over k-means clustering when it comes to categorising geographical data. Firstly, k-means minimises variance rather than geographical distance. Further away from the equator (at high-latitudes), there is substantial distortion amongst groupings. The DBSCAN algorithm works better with arbitrary distances and can be implemented into scikit-learn’s version of the DBSCAN algorithm. DBSCAN clusters a spatial data set based on two parameters: a

physical distance from each point, and a minimum cluster size. This method works much better for spatial latitude-longitude data. Additionally, DBSCAN generates the number of cluster for us, rather than requiring us to specify the number as with the k-means approach.

> The DBSCAN method first calculates the centroid's coordinates. Then I use Python's built-in min function to find the smallest member of the cluster in terms of distance to that centroid. The key argument does this with a lambda function that calculates each point's distance to the centroid in meters, via geopy's great circle function. Finally, the coordinates of the point that was the least distance from the centroid are returned to us. To use this function, we map it to series of clusters from out pandas dataframe. In other words, for each element (i.e., cluster) in the series, it gets the center-most point and then assembles all these center-most points into a new series called *centermost_points*. Then I turn these center-most points into a pandas dataframe of points which are spatially representative of my clusters (and in turn, the full data set.

DATA SECTION REPORT

•Clustering the Districts:

Finally, we try to cluster these 5 districts based on the venue categories and use dbscan clustering. So our expectation would be based on the similarities of venue categories, these districts will be clustered.

DBSCAN offers several advantages over k-means clustering when it comes to categorising geographical data. Firstly, k-means minimises variance rather than geographical distance. Further away from the equator (at high-latitudes), there is substantial distortion amongst groupings. The DBSCAN algorithm works better with arbitrary distances and can be implemented into scikit-learn's version of the DBSCAN algorithm. DBSCAN clusters a spatial data set based on two parameters: a physical distance from each point, and a minimum cluster size. This method works much better for spatial latitude-longitude data. Additionally, DBSCAN generates the number of cluster for us, rather than requiring us to specify the number as with the k-means approach.

The DBSCAN method first calculates the centroid's coordinates. Then I use Python's built-in min function to find the smallest member of the cluster in terms of distance to that centroid. The key argument does this with a lambda function that calculates each point's distance to the centroid in meters, via geopy's great circle function. Finally, the coordinates of the point that was the least distance from the centroid are returned to us. To use this function, we map it to series of clusters from out pandas dataframe. In other words, for each element (i.e., cluster) in the series, it gets the center-most point and then assembles all these center-most points into a new series called *centermost_points*. Then I turn these center-most points into a pandas dataframe of points which are spatially representative of my clusters (and in turn, the full data set.

- **Results and Discussion**

The result of this analysis means that each venue in Toronto is assigned to a specific DBSCAN cluster, which can then be provided to our client so that an optimal delivery service can be devised. Whilst the final implementation of this will be up to the client, we can recommend that they assign delivery riders/drivers to a specific cluster. A full presentation of all venues is not really feasible however as there are too many to list visually. Instead, these will have to be maintained in some form of database this information can be updated. Any new venues, or changes to existing venues can then be re- categorised in the DBSCAN model and added to the client's venue roster. Finally and of note is that the central region of Toronto (or more specifically, the central cluster) is somewhat isolated from the rest of the city. This is due to this area being the most densely populated.

- **Conclusion**

Outcome of this job will be the function wich use Forsquare API data, geo data for particular city and category of veneu Customer want to analyze and find best place. I can try this function to find best places in Toronto.