$\text{Data}_S cience_C apstone_R eport$

schreiber.felipe

February 2020

1 Introduction

Imagine a traveler is looking for a neighborhood in Toronto to stay for a period of time, but he's in doubt which to choose. So with the proper data set we can build a recommendation system listing the most appropriate neighborhoods according to his preference.

2 Data section

For this purpose, we will need a data set that contains Toronto neighborhoods and it's venues. However, we cannot use one hot encoding purely. Imagine that neighborhood "A" have one coffee shop, and "B" 30 coffee shops. If we use purely one hot encoding to indicate whether or not a neighborhood has coffee shops, both will have the same weight. To solve this, we will put some weights according to the amount of venues of each type. For example, in the coffee shop column, the neighborhood "A" will have a less weight than "B".

3 Methodology

I will use the Foursquare API to get the data about the nearby venues for each neighborhood, and then sort them according to the 10 most popular ones in each neighborhood. Then, with this subset of most common venues, I will create another data set which columns are Neighborhood and all the venues that appear in the 10 most common venues data set. Finally I will put some weights according to the position of the venue in the respective neighborhood. Example: suppose that for neighborhood "A" we have the following venues and popularity's:

[hbt!]

1st	Coffee Shop
2nd	Cocktail Bar
3rd	Bakery
4th	Beer Bar
5th	Seafood Restaurant
6th	Farmers Market
7th	Cheese Shop
8th	Steakhouse
9th	Café
10th	Greek Restaurant

Then, putting the following weights according to it's position, the data frame will look like:

Neighborhood	Coffee Shop	Cocktail Bar	Bakery	Beer Bar	Seafood Restaurant
"A"	25	18	15	12	10

Farmers Market	Cheese Shop	Steakhouse	Café	Greek Restaurant
8	6	4	2	1

Other weights could have been selected, although. For simplicity, I picked up the points of Formula 1 according to racer position. Finally, an example how the Foursquare api calls looked like:

$https://api.foursquare.com/v2/venues/explore?client_id = client_secret = ll = 43.67635739999999, -79.2930312radius = 700limit = 100$

"ll" stands for latitude and longitude of the neighborhood in question.

We searched venues in a radius of 700 meters, with a limit of 100 venues. This was done for different neighborhoods and it's latitude/longitude coordinates that was collected previously.