Comparing Toronto with the most important South American cities

Which Toronto's neighborhoods are more like Sao Paulo (Brazil), Buenos Aires (Argentina), Asuncion (Paraguay) and Montevideo (Uruguay)?

Lucas Mielke

November 20, 2019

Summary

1. Introduction

Toronto is a very cosmopolitan city. In this city, there are several immigrants who have moved to it, either temporarily for a trip, taking a medium-term course, graduating or even permanent changes due to work or marriage.

Among the immigrants of this city, there are a considerable number of people from South America.

Migration is not always easy. South American immigrants face language, cultural, climate barriers. In this context, living in a slightly more similar place with your place of origin may be helpful.

In this context, this project aims to compare 4 major South American cities with Toronto neighborhoods and find those that are most like these cities.

The cities analyzed will be:

- Sao Paulo (Brazil),
- Buenos Aires (Argentina),
- Asuncion (Paraguay) and
- Montevideo (Uruguay).

2. Data

The analysis of this project will include the following data:

- Foursquare API, which will provide the most common establishments from all analyzed locations (Toronto, Canada + 4 South American Cities described at introduction): https://developer.foursquare.com/
- List of Neighborhoods and Postcodes in Toronto, provided by wikipedia, that will serve as the basis of Toronto's analysis: https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M
- Toronto geospatial data, Where will we have the coordinates of each neighborhood of Toronto: https://cocl.us/Geospatial_data.
- Geopy Geocoders, which will be used to convert the location of the 4 south American cities into geographic coordinates: https://pypi.org/project/geopy/

For this project, we will create a database that will contain the neighborhoods of Toronto and the 4 South American cities analyzed, with their respective commercial establishments and geographic information. This base will be used in a machine learning algorithm to create clusters (kmeans - https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html) and as a result we will have combinations of South American cities with the neighborhoods of Toronto.

As a first step, let's capitalize Wikipedia's neighborhood data and neighborhood geospatial data, do some cleaning and treatment, and combine these two bases into one called "toronto_data".

Second step, let's create 4 databases with 4 South American cities (Sao Paulo, Buenos Aires, Asuncion, and Montevideo) with latitude and longitude using Geopy. The bases will be "saopaulo_data", "buenosaires_data", "asuncion_data" and "montivideu_data".

In the third step we will combine the above 5 bases into one. This base, which will contain a list of neighborhoods and geographic information (latitude and longitude) will be the basis for the Foursquare API.

In the fourth step we will create a function of capturing establishments based on geographic information and we will apply them to all locations.

In the fifth step the data with the establishments of each locality will be normalized and used to create cluster using Kmeans.

In the sixty step we will analyses the clusters and were the South American cities are classified.

Methodology

Methodology Part 1a - Creating Toronto Data - Scrape Wikipedia List of postal codes

We scrape a table in Wikipedia using pandas, following the instructions at https://stackoverflow.com/questions/55234512/how-to-scrap-wikipedia-tables-with-python and we removed rows that were “not assigned”. Then we grouped the neighborhoods of duplicated postal codes. The final row had 103 rows and 3 columns.

	Postcode	Borough	Neighborhood
0	M1B	Scarborough	RougeMalvern,
1	M1C	Scarborough	Highland CreekRouge Hill, Port Union,
2	M1E	Scarborough	GuildwoodMorningside, West Hill,
3	M1G	Scarborough	Woburn
4	M1H	Scarborough	Cedarbrae

```
table.shape
```

```
(103, 3)
```

Methodology Part 1b - Creating Toronto Data - Scrape Geospatial Data

Then we scraped a table from Geospatial data (https://cocl.us/Geospatial_data). This table have the geographic information (latitude and longitude) by postal code.

	Postal Code	Latitude	Longitude
0	M1B	43.806686	-79.194353
1	M1C	43.784535	-79.160497
2	M1E	43.763573	-79.188711
3	M1G	43.770992	-79.216917
4	M1H	43.773136	-79.239476

This Table was merged with the first table (in part 1a)

Methodology Part 2 - Adding the 4 South American cities to the Analysis

Using Geopy we got the Latitude and longitude of the 4 analyzed cities of south America. Example of Sao Paulo is shown below. The “Borough” was named “South America” to help identifying these cities.

The geographic information (Latitude and Longitude) is taken using the Geopy Librarie

We create a simple dataframe for Sao Paulo containing only the Latitude and Longitude from Geopy.

```
address = 'Sao Paulo, Brazil'
Borough = "South America"

geolocator = Nominatim(user_agent="ny_explorer")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print('The geograpical coordinate of Sao Paulo are {}, {}.'.format(latitude, longitude))

Data = {'Borough':[Borough], 'Neighborhood': [address],
        'Latitude': [latitude], "Longitude" : [longitude]
        }
saopaulo_data = DataFrame(Data,columns= ['Borough','Neighborhood', 'Latitude', "Longitude"])
```

```
The geograpical coordinate of Sao Paulo are -23.5506507, -46.6333824.
```

Methodology Part 3 - Combining all the data

All the data (Toronto’s neighborhoods with Latitude and Longitude, Sao Paulo’s, Buenos Aires’, Asuncion’s and Montevideo’s latitude and longitude) was combined in one dataframe having 107 rows and 5 columns.

We combine all the tables using pandas append.

```
alldata = toronto_data.append(saopaulo_data, sort=False)
alldata = alldata.append(argentina_data, sort=False)
alldata = alldata.append(asuncion_data, sort=False)
alldata = alldata.append(uruguay_data, sort=False)
alldata.shape
```

```
(107, 5)
```

Methodology Part 4 - Creating a function to connect to Foursquare API and using it

We used Foursquare API to get venues of all locations (107 rows at “alldata” dataframe - above). We add our Client ID and Password from Foursquare developer tool.

Define Foursquare Credentials and Version

```
CLIENT_ID = '###' # your Foursquare ID
CLIENT_SECRET = '###' # your Foursquare Secret
VERSION = '20180605' # Foursquare API version

print('Your credentails:')
print('CLIENT_ID: ' + CLIENT_ID)
print('CLIENT_SECRET:' + CLIENT_SECRET)
```

```
Your credentails:
```

We created an API as below.

```
LIMIT = 100
radius = 500
url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={}&v={}&ll={},{}&radius={}&limit={}'.format(
    CLIENT_ID,
    CLIENT_SECRET,
    VERSION,
    neighborhood_latitude,
    neighborhood_longitude,
    radius,
    LIMIT)
url
```

Then we created a function to get category type and venues.

get_category_type function from the Foursquare lab.

```
# function that extracts the category of the venue
def get_category_type(row):
    try:
        categories_list = row['categories']
    except:
        categories_list = row['venue.categories']

    if len(categories_list) == 0:
        return None
    else:
        return categories_list[0]['name']
```

clean the json and structure it into a *pandas* dataframe.

```
def getNearbyVenues(names, latitudes, longitudes, radius=500):

    venues_list=[]
    for name, lat, lng in zip(names, latitudes, longitudes):
        print(name)

        # create the API request URL
        url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={}&v={}&ll={},{}&radius={}&limit={}'.for
            CLIENT_ID,
            CLIENT_SECRET,
            VERSION,
            lat,
            lng,
            radius,
            LIMIT)

        # make the GET request
        results = requests.get(url).json()["response"]['groups'][0]['items']

        # return only relevant information for each nearby venue
        venues_list.append([(
            name,
            lat,
            lng,
            v['venue']['name'],
            v['venue']['location']['lat'],
            v['venue']['location']['lng'],
            v['venue']['categories'][0]['name']) for v in results])

    nearby_venues = pd.DataFrame([item for venue_list in venues_list for item in venue_list])
    nearby_venues.columns = ['Neighborhood',
                  'Neighborhood Latitude',
                  'Neighborhood Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']

    return(nearby_venues)
```

Then we get all the locations.

Write the code to run the above function on each neighborhood and create a new dataframe called *alldata_venues*.

```
alldata_venues = getNearbyVenues(names=alldata['Neighborhood'],
                                 latitudes=alldata['Latitude'],
                                 longitudes=alldata['Longitude']
                                )
```

```
RougeMalvern,
```

Then we put all the data in one dataframe as below.

Let's check the size of the resulting dataframe

```
print(alldata_venues.shape)
alldata_venues.head()
```

```
(2535, 7)
```

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	RougeMalvern,	43.806686	-79.194353	Wendy's	43.807448	-79.199056	Fast Food Restaurant
1	Highland CreekRouge Hill, Port Union,	43.784535	-79.160497	Royal Canadian Legion	43.782533	-79.163085	Bar
2	GuildwoodMorningside, West Hill,	43.763573	-79.188711	Swiss Chalet Rotisserie & Grill	43.767697	-79.189914	Pizza Place
3	GuildwoodMorningside, West Hill,	43.763573	-79.188711	G & G Electronics	43.765309	-79.191537	Electronics Store
4	GuildwoodMorningside, West Hill,	43.763573	-79.188711	Big Bite Burrito	43.766299	-79.190720	Mexican Restaurant

```
alldata_venues.groupby('Neighborhood').count()
```

The final dataframe has 292 unique categories of venues.

Finding out how many unique categories can be curated from all the returned venues

```
print('There are {} uniques categories.'.format(len(alldata_venues['Venue Category'].unique())))
```

```
There are 292 uniques categories.
```

Go back to index

Methodology Part 5 - Normalizing and creating cluster using Kmeans

All the data was normalized to be used in Kmeans.

```
alldata_grouped = alldata_onehot.groupby('Neighborhood').mean().reset_index()
alldata_grouped
```

	Neighborhood	Yoga Studio	Accessories Store	Afghan Restaurant	Airport	Airport Food Court	Airport Gate	Airport Lounge	Airport Service	Airport Terminal	...	University	Vegetarian / Vegan Restaurant	Vid Gar Sto
0	AdelaideKing, Richmond,	0.000000	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.020000	0.0000
1	Agincourt	0.000000	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	0.0000
2	Agincourt NorthL'Amoreaux East, Milliken, Stee...	0.000000	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	0.0000
3	Albion GardensBeaumond Heights, Humbergate, Ja...	0.000000	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	0.0000
4	AlderwoodLong Branch,	0.000000	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	0.0000
5	Asuncion, paraguay	0.000000	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	0.0000

And we created a dataframe that list the top 10 venues from each location.

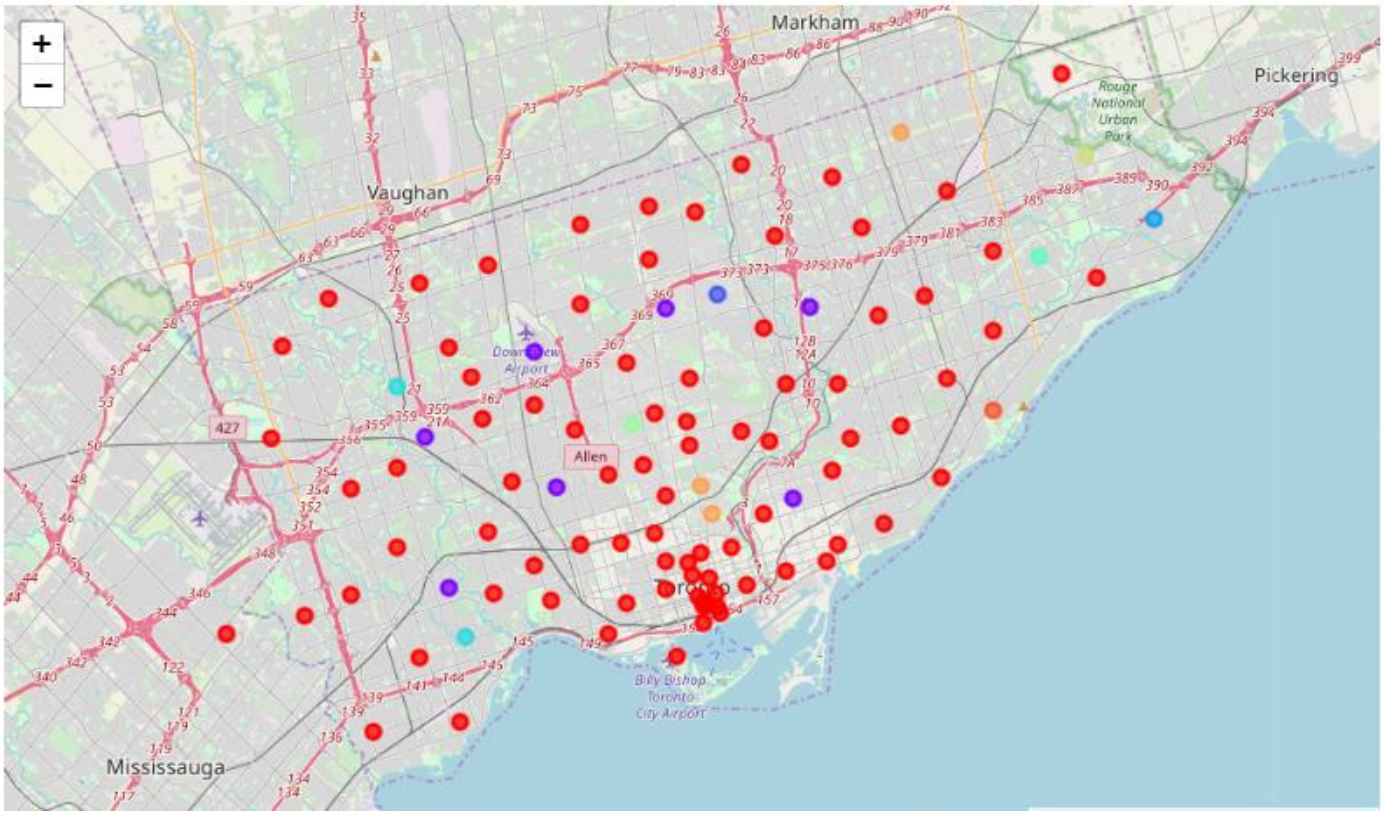
```
num_top_venues = 10

indicators = ['st', 'nd', 'rd']

# create columns according to number of top venues
columns = ['Neighborhood']
for ind in np.arange(num_top_venues):
    try:
        columns.append('{}{} Most Common Venue'.format(ind+1, indicators[ind]))
    except:
        columns.append('{}th Most Common Venue'.format(ind+1))

# create a new dataframe
neighborhoods_venues_sorted = pd.DataFrame(columns=columns)
neighborhoods_venues_sorted['Neighborhood'] = alldata_grouped['Neighborhood']

for ind in np.arange(alldata_grouped.shape[0]):
    neighborhoods_venues_sorted.iloc[ind, 1:] = return_most_common_venues(alldata_grouped.iloc[ind, :], num_top_venues)

neighborhoods_venues_sorted.head()
```

	Neighborhood	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
0	AdelaideKing, Richmond,	Coffee Shop	Café	Steakhouse	Bar	American Restaurant	Sushi Restaurant	Asian Restaurant	Thai Restaurant	Bakery	Breakfast Spot
1	Agincourt	Lounge	Latin American Restaurant	Skating Rink	Breakfast Spot	Electronics Store	Eastern European Restaurant	Dumpling Restaurant	Drugstore	Empanada Restaurant	Deli / Bodega
2	Agincourt NorthL'Amoreaux East, Milliken, Stee...	Playground	Park	Women's Store	Discount Store	Dance Studio	Deli / Bodega	Department Store	Dessert Shop	Dim Sum Restaurant	Diner
3	Albion GardensBeaumond Heights, Humbergate, Ja...	Pizza Place	Pharmacy	Beer Store	Fried Chicken Joint	Japanese Restaurant	Fast Food Restaurant	Discount Store	Sandwich Place	Grocery Store	Airport Terminal
4	AlderwoodLong Branch,	Pizza Place	Skating Rink	Pub	Coffee Shop	Gym	Pharmacy	Sandwich Place	Dance Studio	Deli / Bodega	Department Store

We ran the Kmeans model, classing the locations in 10 clusters and merged the clusters with the dataframe above, creating a list that classify each neighborhood in one location and lists the 10 most common venues.

	Postcode	Borough	Neighborhood	Latitude	Longitude	Cluster Labels	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 Mc Comm Ven
0	M1B	Scarborough	RougeMalvern,	43.806686	-79.194353	7.0	Fast Food Restaurant	Women's Store	Dog Run	Deli / Bodega	Department Store	Dessert Shop	Dim Su Restaur
1	M1C	Scarborough	Highland CreekRouge Hill, Port Union,	43.784535	-79.160497	3.0	Bar	Women's Store	Dance Studio	Department Store	Dessert Shop	Dim Sum Restaurant	Di
2	M1E	Scarborough	GuildwoodMorningside, West Hill,	43.763573	-79.188711	0.0	Electronics Store	Breakfast Spot	Intersection	Rental Car Location	Mexican Restaurant	Medical Center	Piz Pla
3	M1G	Scarborough	Woburn	43.770992	-79.216917	5.0	Coffee Shop	Korean Restaurant	Women's Store	Doner Restaurant	Department Store	Dessert Shop	Dim Su Restaur
4	M1H	Scarborough	Cedarbrae	43.773136	-79.239476	0.0	Hakka Restaurant	Caribbean Restaurant	Lounge	Bank	Bakery	Fried Chicken Joint	Th Restaur

The clusters in Toronto was as below

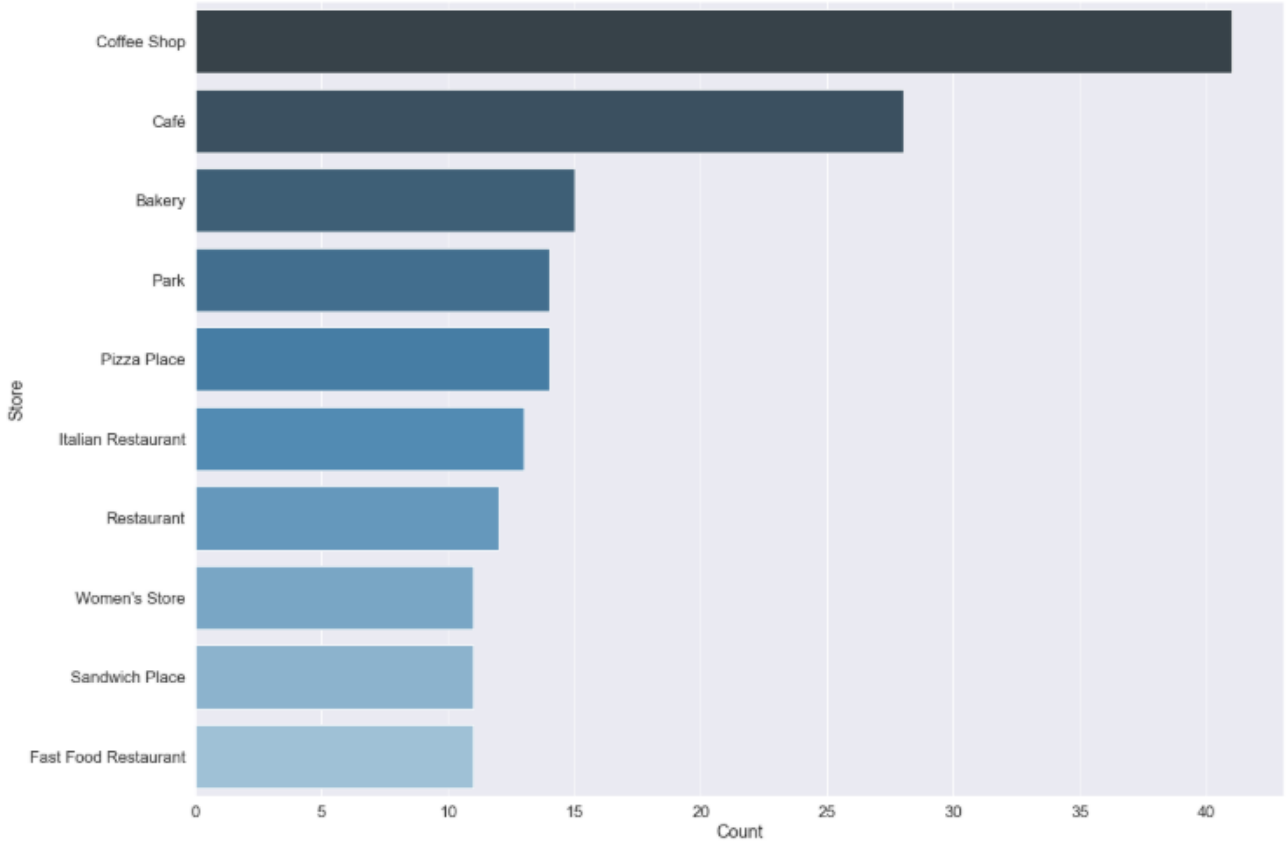


Most of the location was in cluster 1 (red)…

As the south American cities…

The four South American cities are in cluster 1, the biggest one. In this cluster are neighborhoods like GuildwoodMorningside, East Birchmount ParkIonview, Kennedy Park, West Hill, Clarks CornersSullivan,

Tam O'Shanter. Let's Plot the top 10 venues of the cluster 1, the cluster that contains the 4 south American cities. We can see below that the most common venue is Coffee Shop by far, followed Cafés, Bakery, Parks...



Conclusions

In this work we take public data from neighborhoods of Toronto, Canada (list of neighborhoods and geographic information) as well as the location of 4 major Latin American cities (Sao Paulo - Brazil, Asuncion - Paraguay, Buenos Aires - Argentina and Montevideo - Uruguay).

We combined this information into one database, which contained locations and geographic information. This combined base contained 107 locations to be analyzed.

A connection has been made to the Foursquare API and a function has been created to fetch Venues for each of the locations. Running this function in the combined database found 292 unique venues.

The combined database (with locations and venues) contains 2535 rows per 292 columns.

The database containing the neighborhood list and each venue type was normalized and used with Machine Learning. We ran a Kmeans algorithm that clustered all neighborhoods into 10 clusters based on the venues of every 1. Finally, we created a dataframe listing the 10 most common Venus by neighborhood and added the created clusters.

Cluster distribution was highly concentrated in cluster 1, which contained 89 neighborhoods, including the 4 South American cities. Cluster 1 is quite abundant in coffee shops and cafes and generally contains plenty of restaurants.