Moving to a similar neighborhood in Mexico City

Octavio Ivan Hernandez Salinas
honter1997@gmail.com

April 2020

Table of Contents

Introduction

Many people in Mexico City are moving to new houses every day, which is great but a lot of people have problems getting familiarized with the new neighborhood because they miss their old neighborhood and the venues they used to go to . For example: they miss their favorite tacos or their favorite coffee shop.

So, we can prevent people from spending many hours looking for a new neighborhood with the same kind of venues that they have nearby their old house. Instead, we can recommend neighborhoods with almost the same kind of venues as the old neighborhood.

Problem

People spend many hours looking for neighborhoods with similar kind of venues when they are buying a new house

Target audience: People who are looking for a new neighborhood to move in.

Solution

Let’s create a basic recommendation model, using clustering and the Foursquare API, to show people the neighborhoods that are similar to theirs.

Data

Data sources

We will use the following dataset, provided by the Mexico City State
https://datos.cdmx.gob.mx/explore/dataset/coloniascdmx/download/?format=csv&timezone=America/Mexico_City&lang=es&use_labels_for_header=true&csv_separator=%2C
The dataset has the following features:

Column name	**Data type**	**Description**
COLONIA	object	Neighborhood’s name

ENTIDAD	float64	State’s ID
Geo Point	object	Latitude and Longitude separated by comma
Geo Shape	object	JSON with the coordinates of the neighborhood’s area shape
CVE_ALC	int64	City’s ID
ALCALDIA	object	City’s name
CVE_COL	object	Neighborhood’s ID
SECC_COM	object	Data used for electoral purposes
SECC_PAR	object	Data used for electoral purposes

Let’s change the features name and drop other features in order to get a better result

Data cleaning

We will prepare the data in order to use it with the Foursquare API and the K-Means algorithm. The preprocessed dataset will have the following columns:

- neighborhood
- neighborhood_lat
- Neighborhood_lng
- city

First, let’s drop some features that are no relevant for us:

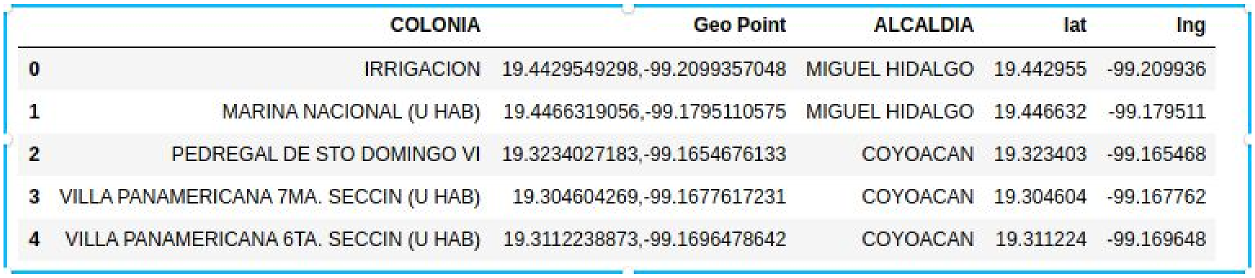
Dropped features	**Reason**
ENTIDAD	We will work with only one state so this feature is always the same
Geo Shape	We won’t use the area of the neighborhood so let’s drop this feature
CVE_ALC	We won’t use the city’s ID so let’s drop this feature
CVE_COL	We won’t use the neighborhood’s ID so let’s drop this feature

SECC_COM	We won’t use this feature so let’s drop this feature
SECC_PAR	We won’t use this feature so let’s drop this feature

After we dropped the features listed above, we have the following features:

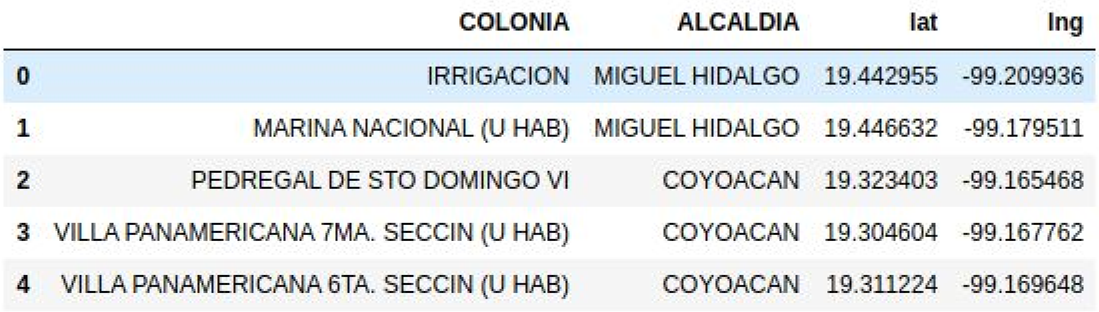
COLONIA	object
Geo Point	object
ALCALDIA	object

Now let’s remove all rows that contain NaN in ‘Geo Point’ column Convert the 'Go Point' feature into two new features: lat and lng



	COLONIA	Geo Point	ALCALDIA	lat	lng
0	IRRIGACION	19.4429549298,-99.2099357048	MIGUEL HIDALGO	19.442955	-99.209936
1	MARINA NACIONAL (U HAB)	19.4466319056,-99.1795110575	MIGUEL HIDALGO	19.446632	-99.179511
2	PEDREGAL DE STO DOMINGO VI	19.3234027183,-99.1654676133	COYOACAN	19.323403	-99.165468
3	VILLA PANAMERICANA 7MA. SECCIN (U HAB)	19.304604269,-99.1677617231	COYOACAN	19.304604	-99.167762
4	VILLA PANAMERICANA 6TA. SECCIN (U HAB)	19.3112238873,-99.1696478642	COYOACAN	19.311224	-99.169648

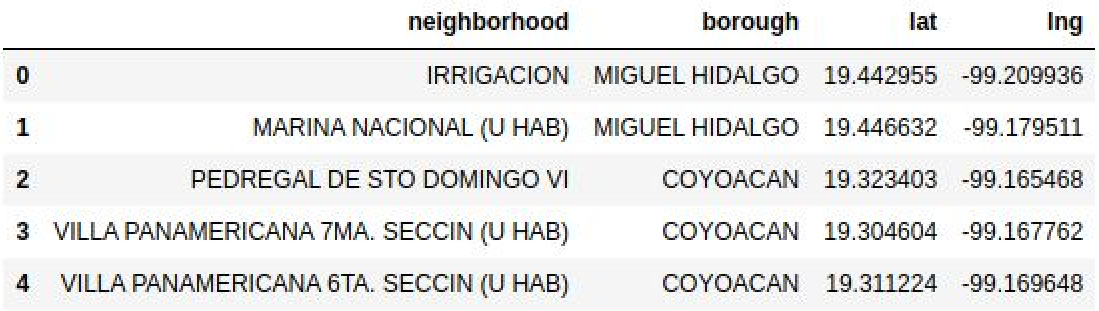
Then we need to drop the column ‘Geo Point’ so the our dataset should look like as follows:



	COLONIA	ALCALDIA	lat	lng
0	IRRIGACION	MIGUEL HIDALGO	19.442955	-99.209936
1	MARINA NACIONAL (U HAB)	MIGUEL HIDALGO	19.446632	-99.179511
2	PEDREGAL DE STO DOMINGO VI	COYOACAN	19.323403	-99.165468
3	VILLA PANAMERICANA 7MA. SECCIN (U HAB)	COYOACAN	19.304604	-99.167762
4	VILLA PANAMERICANA 6TA. SECCIN (U HAB)	COYOACAN	19.311224	-99.169648

Finally, Change the columns name so they are more descriptive:

- COLONIA -> neighborhood
- ALCALDIA -> borough



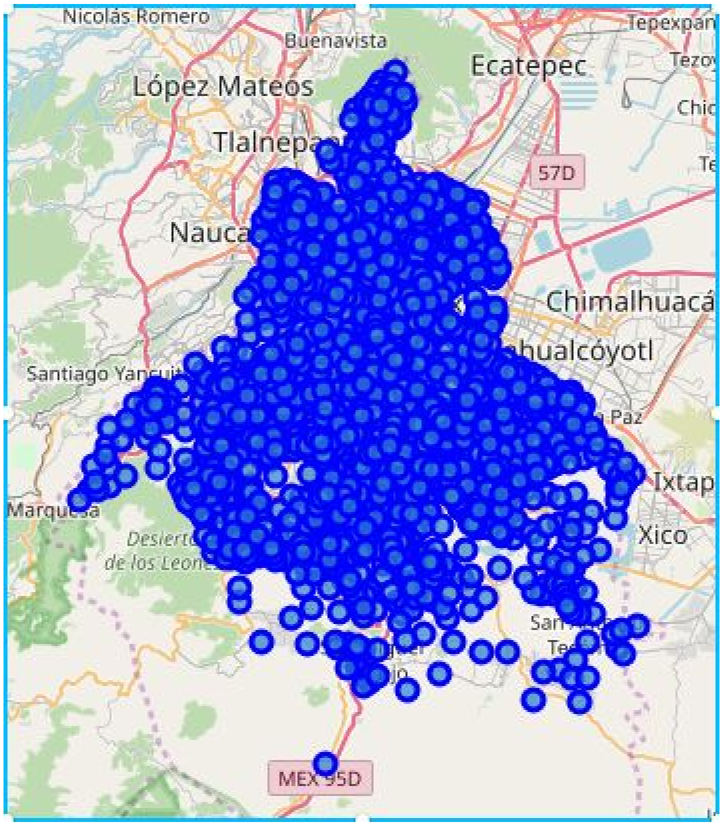
	neighborhood	borough	lat	lng
0	IRRIGACION	MIGUEL HIDALGO	19.442955	-99.209936
1	MARINA NACIONAL (U HAB)	MIGUEL HIDALGO	19.446632	-99.179511
2	PEDREGAL DE STO DOMINGO VI	COYOACAN	19.323403	-99.165468
3	VILLA PANAMERICANA 7MA. SECCIN (U HAB)	COYOACAN	19.304604	-99.167762
4	VILLA PANAMERICANA 6TA. SECCIN (U HAB)	COYOACAN	19.311224	-99.169648

We have cleaned our dataset and It’s ready to be used. In the next steps, we will request, to the Foursquare API, the nearby venues’ category for each neighborhood and get the mean value for each category so each row corresponds to each neighborhood and each column to a venue category and this will be the input of the K-Means algorithm.

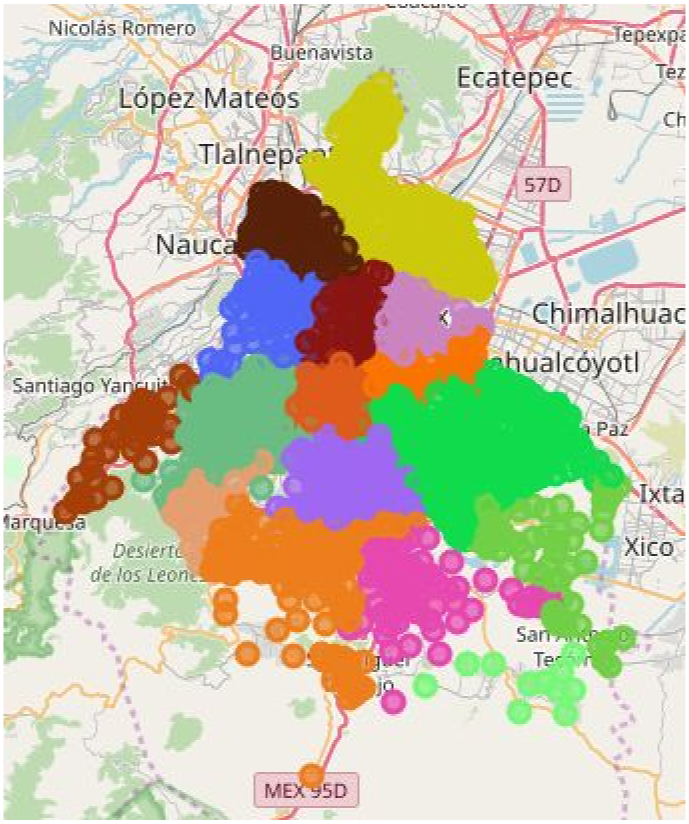
Methodology

K-means clustering is a type of unsupervised learning, which is used when you have unlabeled data (i.e., data without defined categories or groups). The goal of this algorithm is to find groups in the data, with the number of groups represented by the variable K. The algorithm works iteratively to assign each data point to one of K groups based on the features that are provided. Data points are clustered based on feature similarity. So, we will use K-Means clustering to cluster the neighborhoods that are similar based on their venues nearby.

First of all, let’s plot the neighborhoods in a map so we can look the distribution of the neighborhoods in Mexico City



It didn’t give us any information about the distribution or something else. Let’s divide our neighborhoods by borough and see how it looks.

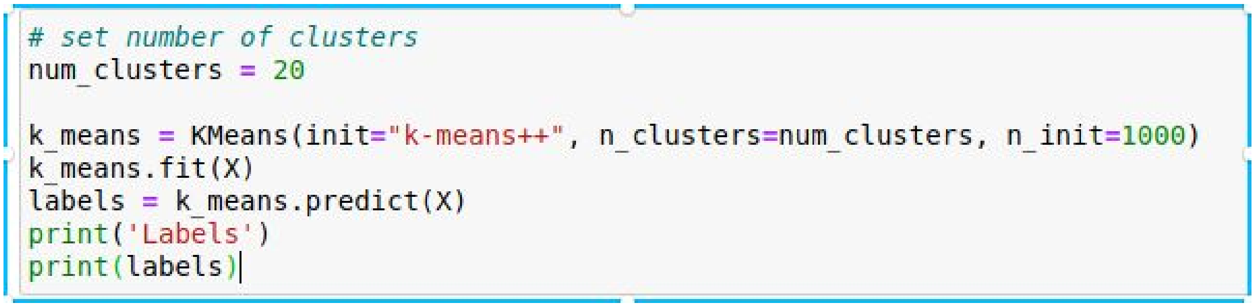


Now let's get venues for each Neighborhood. We have to make a request to the Foursquare API for each neighborhood. The request will retrieve the nearby venues so our job is to get the mean value for each venue category for each neighborhood and after that we will be ready to fit our K-Means model and predict the k-labels.
In other words, the steps to follow for each neighborhood are:

1. Request the neighborhood nearby venues
2. Get the main category for each venue that we’ve got
3. Get the distance from the neighborhood to the venue
4. One-hot encoding for the categories obtained
5. Use all rows to get the mean value for each category
6. Transform the data we’ve got to a DataFrame so each row corresponds to one neighborhood
7. Replace NaN with 0s

Finally, after doing the steps described above for all neighborhoods we can fit our k-Means algorithm. After trying different values for k, I found out that the best k is 20.

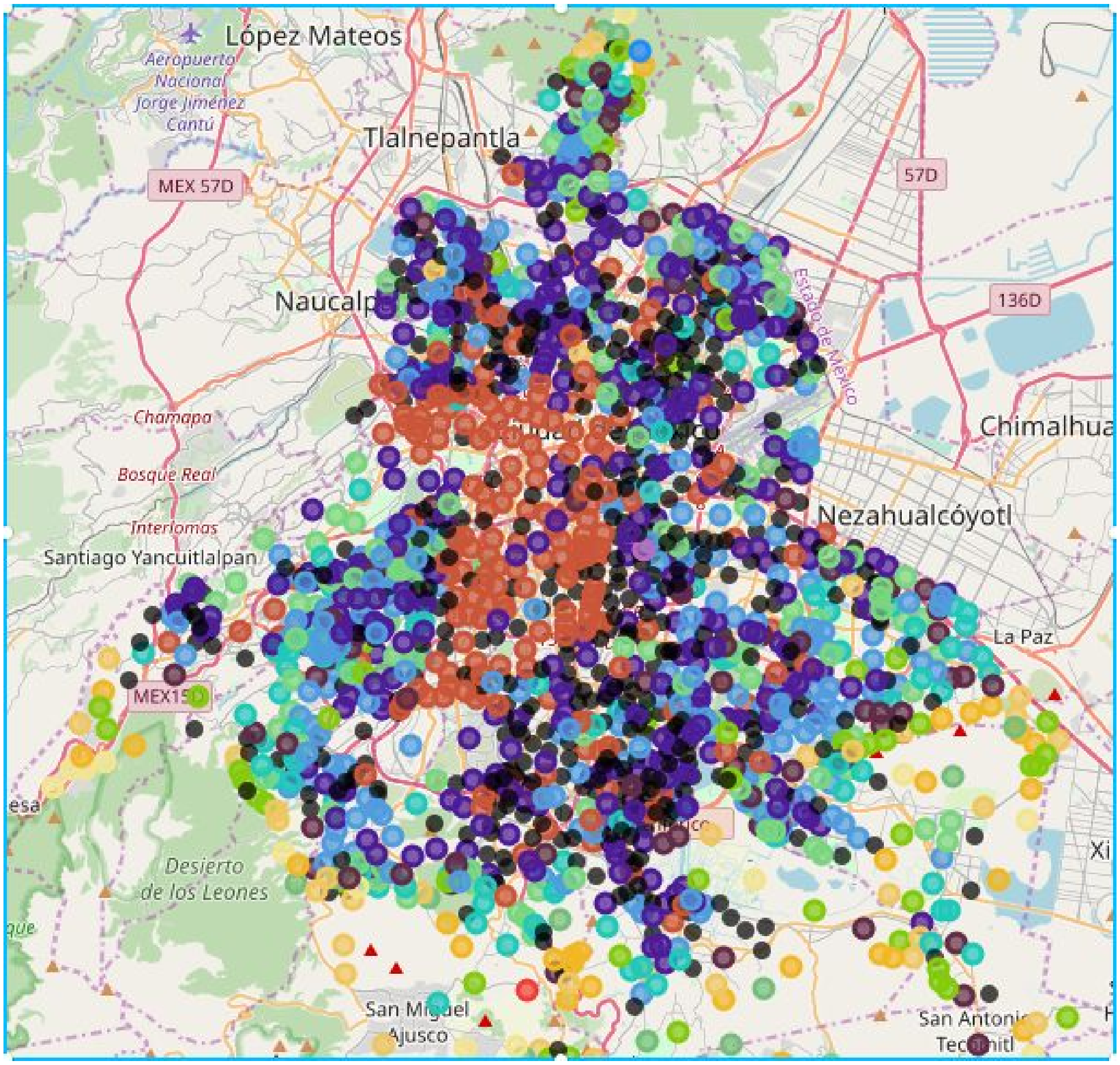
Results



```
# set number of clusters
num_clusters = 20

k_means = KMeans(init="k-means++", n_clusters=num_clusters, n_init=1000)
k_means.fit(X)
labels = k_means.predict(X)
print('Labels')
print(labels)
```

The next step is to visualize the result of labeling our neighborhoods and see how it looks like and how much it differs from the map divided by borough. I think that shows the intention of the project, It can be improved with more data, but for now, I think that it’s OK



Discussion

I think that we’ve got good results to start. In my opinion, the model can be improved with the following data: traffic data, criminal data and mean borough income. Unfortunately, in Mexico it’s difficult to get government data even if it’s public. It took me a couple of hours before I found the coordinates of the neighborhoods.

Conclusion

In conclusion, we looked at how the neighborhoods are distributed in Mexico City, we found that there are a lot of neighborhoods very close to each other. We found that the neighborhoods downtown are very similar. Finally we discussed that the model can be improved by adding data but unfortunately it’s difficult to get it.