PERSON RELOCATION EXERCISE

Coursera Data Science Capstone

Abstract

Using KMeans to classify multi-city neighborhoods into like-clusters

Chingis Tsytsik

BUSINESS PROBLEM

This project stems from my own personal interests, as well as the interests of a few of my close friends. We have always discussed among ourselves where we would want to move if we were to leave the city of Boston. We generally drifted to a few cities: New York, Denver, Seattle, and Montreal. Consequently, I had the idea of using the Foursquare location data from the IBM Data Science capstone to help guide some of our discussions.

For the capstone project, I wanted to push this analysis a bit further. My friends and I live in Boston, but we live in different neighborhoods and different parts of the city. What if I could investigate the neighborhoods of other cities and see which ones more closely resembled the ones here in Boston. I had the idea of taking myself, and two of my buddies, choosing their neighborhoods, and seeing how they would fit in two other cities: Denver and Seattle.

With regards to how this would be relevant from a business case perspective; we can imagine ourselves to be recruiters, who are trying to figure out the best and most satisfactory way of moving talent from one city to another. Say, as a recruiter, I have a client who lives in relatively popular part of Boston, what would be the best neighborhood to suggest they move to in Denver? We can then take this example and apply it to all our clients, assuming they want to move to a neighborhood that closely resembles the one they are moving from.

In this specific business case, I will be looking at three individuals from three neighborhoods. They all live in Boston, and then I will recommend similar neighborhoods for these individuals in Denver, CO and Seattle, WA. I will run the K-means algorithm to group neighborhoods first within cities, then compare them across cities. The end result will be that the individuals will be able to move to a new city to a neighborhood that is similar to the one they left.

DATA

The data that I will be using for this business case is location data gathered from the Foursquare API. It can be used to accurately group locations together into coherent and similar neighborhoods. In that, I will be able to significantly differentiate one part of town from another. In the case of Boston, I will be able to classify and say that the Cambridge area is different from Back Bay. While South Boston will be different from a western suburb.

This method will be useful to make meaningful distinctions intra-city as well as inter-city. Once I can classify the neighborhoods in Boston, I will then be able to classify the and compare the neighborhoods in other cities. The foursquare data, with its abundance of information about nearby shops, businesses, and attractions, will allow us to complete this analysis and return a meaningful recommendation to our client.

METHODOLOGY

Discuss and describe

- exploratory data analysis that you did
- any inferential statistical testing that you performed
- what machine learnings were used and why

In order to begin the analysis, I had to create a list of neighborhoods of the Boston, Denver, and Seattle city areas. To this end, I utilized the Wikipedia neighborhood information that was previously used in my Coursera class (Link: https://en.wikipedia.org/wiki/Category:Lists_of_neighborhoods_in_U.S._cities). I scraped the websites using a python script and formatted them into pandas dataframes. One of the first issues that I came across was the

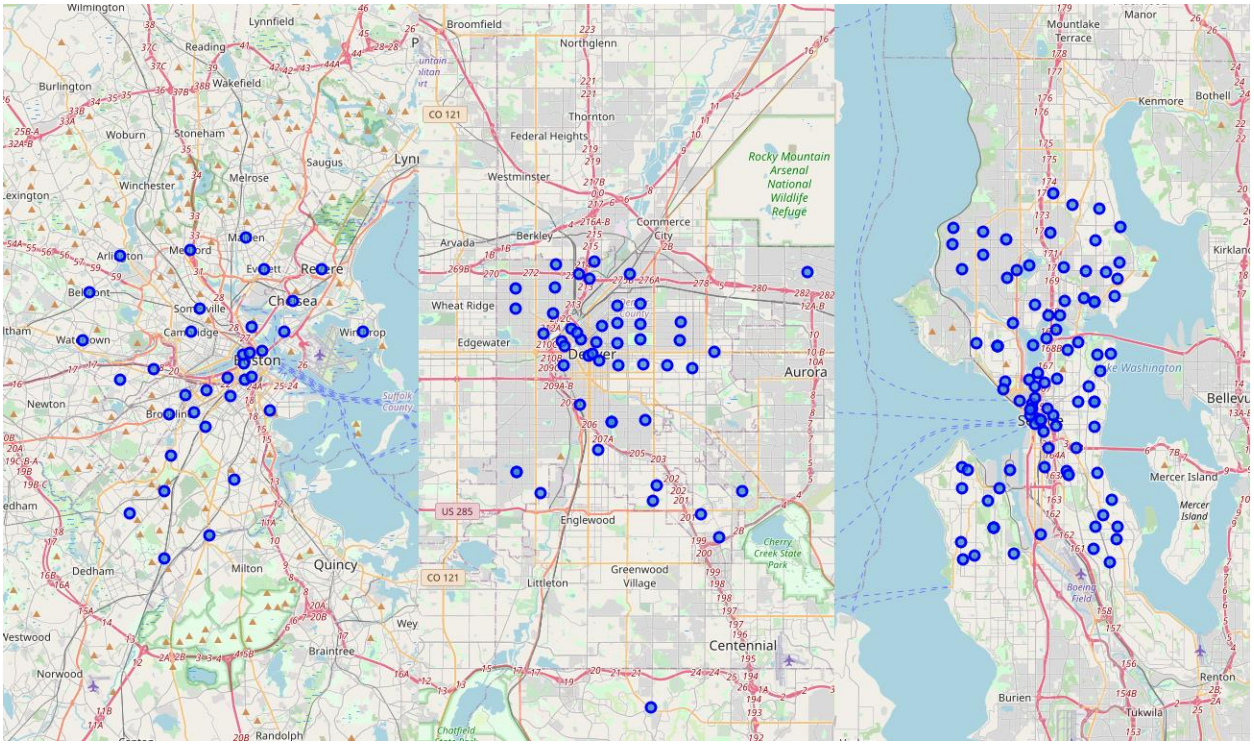
relative scarcity of neighborhoods in Boston compared to Denver and Seattle: 23, 69, and 127 respectively. I thought this might skew the results and so, being from Boston, I added a few more of the surrounding cities to the list to better the analysis. At the end of the project, I did a comparison of the results with the original against the expanded lists and was very glad I added this step.

The next step was to geocode the data; i.e. add longitude and latitude information all of the neighborhoods and cities. For this I used the Geopy library and passed on some identifying information to their API. Some of the neighborhood inputs returned either blanks or incorrect longitude and latitude information. Since they were a minority (only a few entries) I decided that for this relatively simple analysis I would remove them entirely. At this point, I had everything I needed to map the neighborhoods. See below for an example table of data.

[9]:

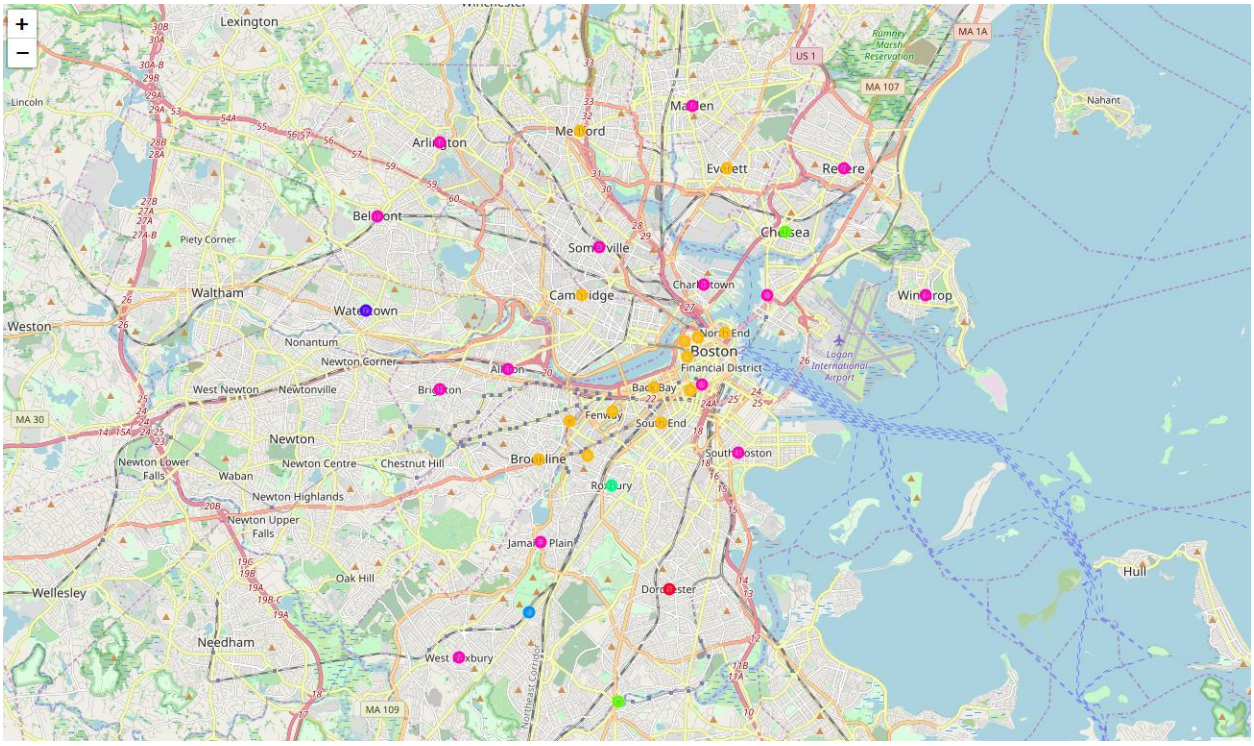
	Neighborhood	City	State	Name	Location_Detail	Latitude	Longitude
0	Allston	Boston	Massachusetts	Allston, Boston, Massachusetts	(Allston, Boston, Suffolk County, Massachusett...	42.355434	-71.132127
1	Back Bay	Boston	Massachusetts	Back Bay, Boston, Massachusetts	(Back Bay, Boston, Suffolk County, Massachuset...	42.350707	-71.079730
2	Bay Village	Boston	Massachusetts	Bay Village, Boston, Massachusetts	(Bay Village, Beach Street, Chinatown, Financi...	42.350011	-71.066948
3	Beacon Hill	Boston	Massachusetts	Beacon Hill, Boston, Massachusetts	(Beacon Hill, Boston, Suffolk County, Massachu...	42.358708	-71.067829
4	Brighton	Boston	Massachusetts	Brighton, Boston, Massachusetts	(Brighton, Boston, Suffolk County, Massachuset...	42.350097	-71.156442

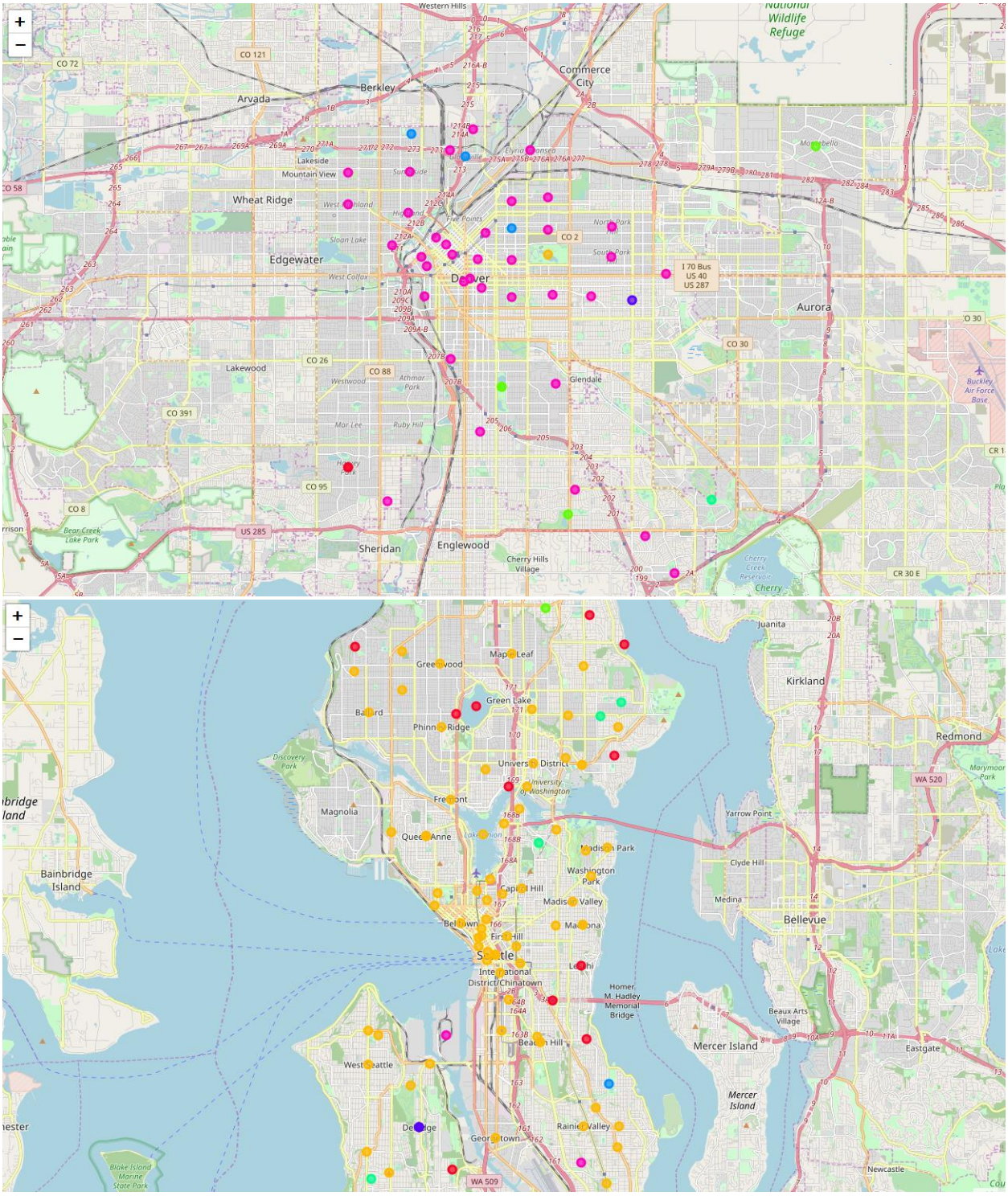
Finally, as an extra precaution, I also removed outliers in the data. I kept all the Boston data, kept 10-90 percentile for Denver, and 5-95 percentile for Seattle. This was to prevent data points that were too far away from the center of the city. See below for an image of the mapped neighborhoods in order of Boston on the left, Denver in the center, and finally Seattle on the right.



At this point I was happy with the maps and decided to move on to the next step of querying Foursquare. One way of grouping location data, which I utilized in the Coursera class, is to associate every point with its local businesses. This will allow us to compare locations according to the similarity of the nearby businesses. For this reason, I used

the Foursquare API to retrieve the closest 100 businesses within 500 meters of the neighborhood center. This gave me a detailed list of the businesses that I will feed into the machine learning algorithm. A few detailed coding lines later, I was able to build business feature data for each location, run it through the Kmeans algorithm, and spit out clustering information. In this case, I decided that 7 was the max number of clusters that I wanted to create inside of a city. Anymore than that and neighborhoods become a too discrete; any less than that and neighborhoods are grouped together that shouldn't. See the below output for Boston, then Denver, and finally Seattle.

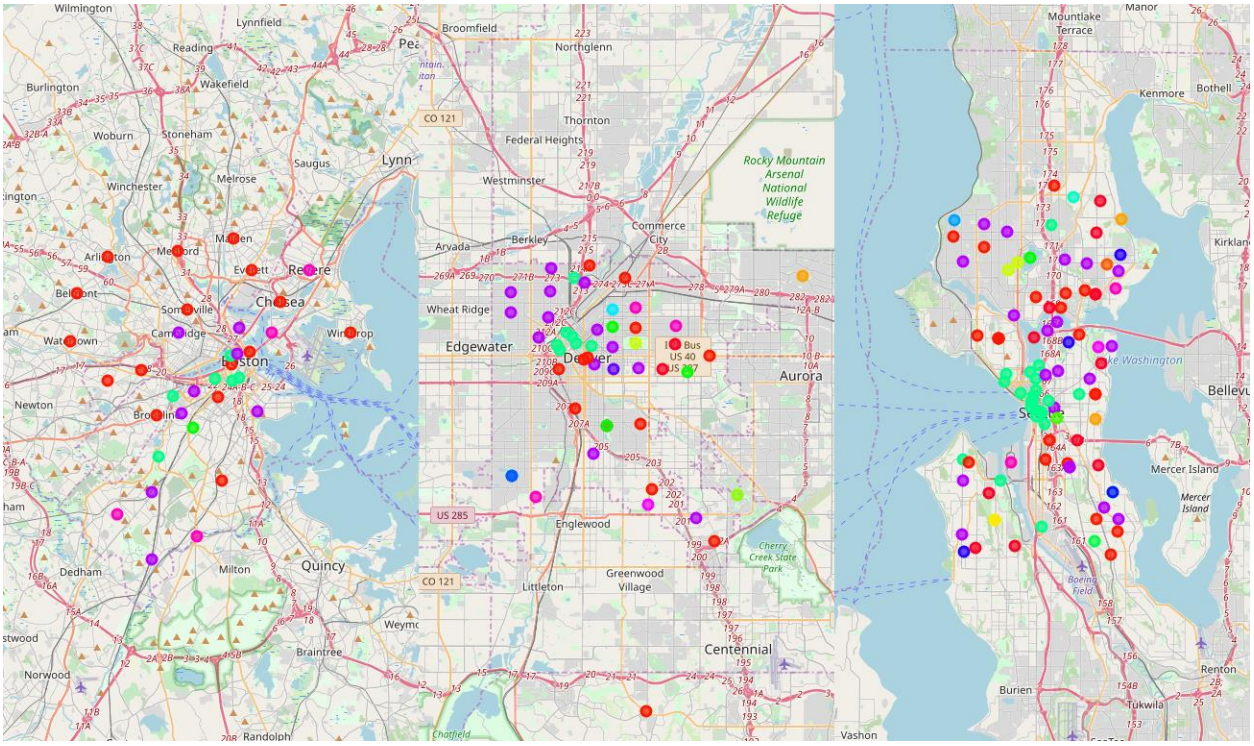




If you are familiar with the cities and their neighborhoods, you'll be able to speak to their accuracy (or inaccuracy). I'm from Boston, so I can look at the Boston map and see that it fits relatively well. I think a few of the neighborhoods are mischaracterized, but that is a feeling based on my subject interpretation of the "vibes" of the area. Since I am looking at business data, I can only think that the algorithm is accurately placing the neighborhoods into accurate groups. And finally, no analysis is going to be perfect and all algorithms will err;

however, that doesn't mean that the outputs are not useful for our purposes. Additionally, I asked a friend from Seattle if the map of his city was correct, and he has the same reaction as I did. Most of the feel was right, except for a few mismatches.

In the final step of this exercise, I wanted to see how the groups would look if they were all put into the same pool. How would the algorithm cluster neighborhoods if it could pull and pool areas from different cities together? The output of this question is below. In this exhibit, I tripled the clusters since I was tripling the size of the pool; therefore, there are 21 different clusters instead of 7. For clarity's sake, one can focus on the colors when looking at the maps. If you have lived in a red neighborhood in Boston, then a red neighborhood would the most similar in another city, according to the model.



RESULTS

The usefulness of the results of the clustering is up for debate. While the clustering may be useful for some people, others will find that relying on its results is misleading. For instance, internally to Boston, some of the neighborhoods in the total clustering exercise are not very similar to each other. I would not group Dorchester with all the other suburbs around Boston. And while I do not know the feel of the neighborhoods in other cities, I can confidently say that the same kind of erroneous grouping is occurring there as well. Denver neighborhoods aren't perfect, and neither are Seattle's neighborhoods.

Therefore, the results of the clustering are only semi-useful. It could be used a starting place to get an idea of where to look but cannot be used to make the ultimate choice. One possible use could be to reference the clusters to narrow down the search of neighborhoods for other cities. Let's say that I live in a purple cluster from Boston, I would use the above graphic to narrow down my search to only purple neighborhoods in Denver and Seattle. I can

be confident that the corresponding neighborhoods have something in common with my home, however I would have to do more (and different) research in order to come to a final decision.

DISCUSSION

In order for this write-up to be complete, I believe that touching upon its limitations and flaws is a must. Consequently, I would characterize this analysis as a good first step in a long process of building a relocation methodology. See below for a few quick points about where the model has room for improvement.

- Inaccuracy – While the model does get the neighborhood 'feel' right most of the time (>50%), I would say it also misses the mark quite often: considering my familiarity with the Boston area. This is a critical mistake and must be tackled first in the next iteration.
- Inconsistency – When rerunning the code, the clustering pattern of the neighborhoods changes because of the nature of the Kmeans model. In its original/general form, the model chooses semi-random starting centers and moves them from there. This results in clusters that can change wildly for each run of the code. One way around this would be to run the model many different times and choose the result that happens the most. Let's say I run the model 100 times. For 40 of those, the clustering pattern is relatively the same, while 60 of them are off. I would then choose the 40 runs as the true clustering result and discard the 60 others.
- Additional factors – For this exercise, I used data from foursquare to populate information about neighborhoods in the cities; specifically, local business information. But if I add data into the mix that is orthogonal, I could get a better result from the algorithm. From the top of my head I believe that adding demographic information about the residents and average rent/mortgage information would significantly improve the result. The former can be retrieved from census data and the latter from real estate companies like Zillow.

CONCLUSION

While the model can be optimized and the results can be cleaned, I think that this is a good start for a methodology for how to move people from one city to another. It is a strong start to analyze a neighborhood according to the shops that are located therein. It makes a decent approximation to the 'feel' of a neighborhood. As touched upon in the discussion section, it would be worth while in the future to add in a rent approximator and demographic information. Thanks for checking out my first foray into the data science world; its been a journey.

As to where I and two of my friends can move, feel free to copy my code into Jupyter Lab, pick a starting location, run the code, and pick a place in another city.