Best Houston Neighborhoods for People New to Area Based on Preferences

Joseph Gibbons

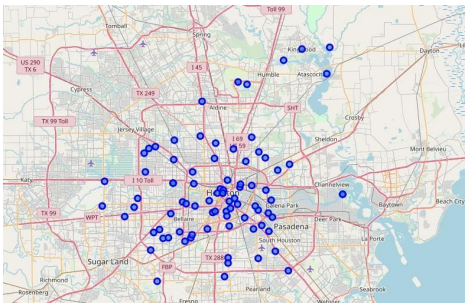
1 PROBLEM INTRODUCTION

Moving by itself is stressful, but even more so when moving to a city about which you know very little. Everyone has their preferences of what they would like nearby to where they live, and while some people prefer parks and greenery, others would rather have restaurants and bustling nightlife. My models will first use Foursquare data to cluster the 88 'Super Neighborhoods' of Houston, Texas based on the quantity of over 200 types of venues and then be able to predict which neighborhoods new Houston residents should look at to live based on their preferences and the cluster into which they fall.

In theory, this program will help people looking to move to Houston focus on specific neighborhoods to live instead of being faced with the daunting task of looking at the entirety of the city.

2 DATASET

To complete this task I will be scraping a list of the 'super neighborhoods' from wikipedia and then using the geopy library on Python to get the latitudes and longitudes of each neighborhood. Nominatim did not have data for every neighborhood, so I googled the remaining neighborhoods and manually placed them into my dataframe. This data resulted in the map of neighborhoods to the right.



Using the latitudes and longitudes I then utilized the Foursquare API to get a list of venues for each neighborhood. I set the radius for each neighborhood to 1 kilometer since Houston is a pretty spread out city and I think a 1 km walking radius is a decent maximum. I limited the number of venues for each neighborhood to 100 because I didn't want a few neighborhoods with a lot of venues to overpower those which had fewer. This gave me a dataframe of over 2,600 venues which looked like this:

	Neighborhood	NeighborhoodLatitude	NeighborhoodLongitude	Venue	VenueLatitude	VenueLongitude	VenueCategory
0	Willowbrook	29.660254	-95.456096	Kolache Factory	29.666938	-95.462662	Breakfast Spot
1	Willowbrook	29.660254	-95.456096	Emmit's Place	29.657188	-95.463080	Bar
2	Willowbrook	29.660254	-95.456096	Popeyes Louisiana Kitchen	29.664607	-95.463331	Fried Chicken Joint
3	Willowbrook	29.660254	-95.456096	Annie's Burgers	29.663315	-95.463785	Burger Joint
4	Willowbrook	29.660254	-95.456096	Hunter's Pub	29.666188	-95.462762	Dive Bar

From this dataframe I was able to count the venues in each neighborhood, organize them by category, and figure out the most popular type of venues in each individual neighborhood and their frequency as a percentage of the entire amount of venues in the neighborhood (represented as a decimal). An example of the top 10 types of venue in the 'Super Neighborhood' Central Southwest is to the right. This is the data I used for modeling.

```
----Central Southwest----
                          category  freq
0                            Hotel  0.09
1                      Coffee Shop  0.05
2               Mexican Restaurant  0.05
3                       Steakhouse  0.04
4                      Burger Joint  0.04
5                              Bar  0.04
6  Southern / Soul Food Restaurant  0.04
7                          Theater  0.03
8                      Pizza Place  0.03
9                      Cocktail Bar  0.03
```

3 METHODOLOGY

Before diving into creating my model, I wanted to get a better sense of the data I got from the Foursquare API and how it looks. First, I looked at how many distinct neighborhoods were in my dataframe. This was confusing at first because I knew from my webscraping that there are 88 neighborhoods in Houston, but I only got 87 unique neighborhoods from Foursquare. Upon further inspection, I noticed that one of the neighborhoods had no venues, so it was excluded from the results. I then wanted to see how many unique venue categories were in my data, which turned out to be 282. The majority of these were cafés and restaurants. Finally, I looked at how many venues were in each neighborhood. Only eight neighborhoods reached the limit of 100 venues and the average amount of venues per neighborhood was 30.02.

After exploring the data, I used three machine learning techniques to try and create a model which would solve my problem as explained in Part 1 of this report. I first needed to cluster the data and save the labels in order to then make a classification model.

KMeans++: I first tried the KMeans++ clustering model in order to create the labels for each neighborhood. My results for this model were very far from ideal:

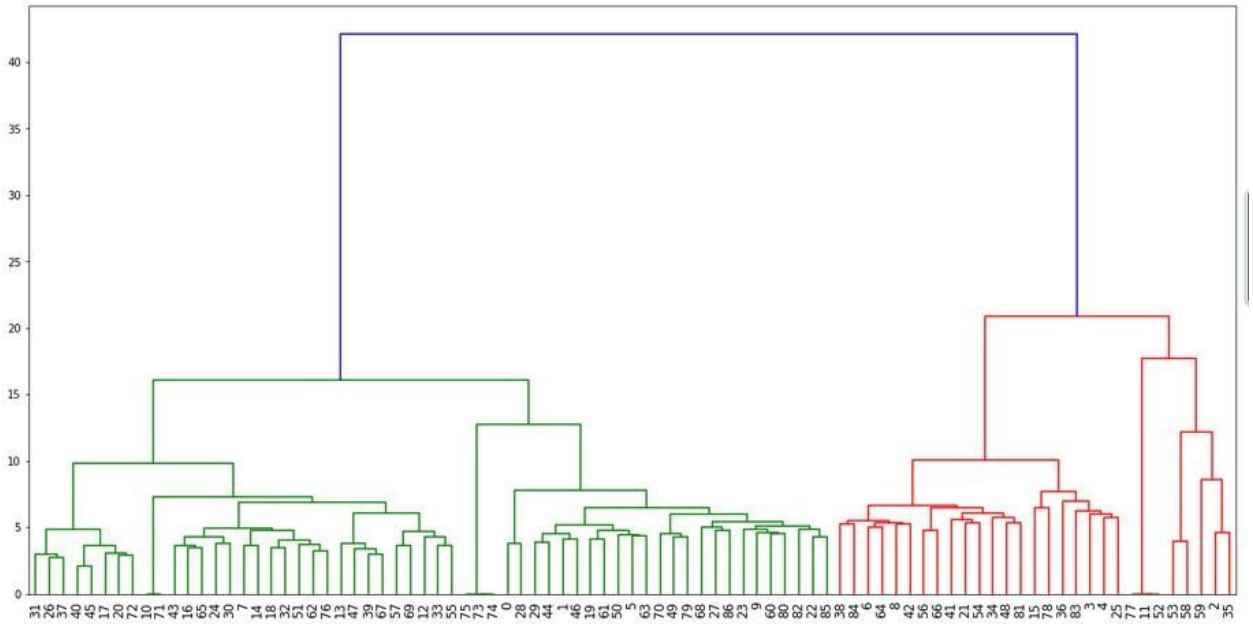
```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
       0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 4, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
      dtype=int32)
```

As you can see, the vast majority of the neighborhoods were placed into the same cluster, which would not help my users at all since the list of neighborhoods which match their preferences would still be roughly 90% of all the neighborhoods in Houston.

Agglomerative Clustering: I then decided to try agglomerative clustering since this builds clusters from the bottom up and so I can see if there is a better point at which the clusters are at a more manageable size. For some reason my model ignored my n_clusters parameter and gave me a label array of:

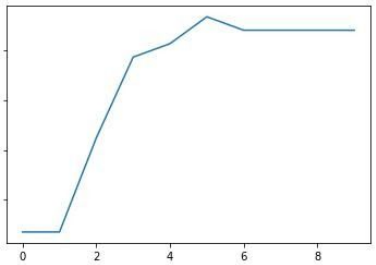
```
[2 0 0 0 0 0 0 0 0 0 3 0 0 2 0 0 8 1 0 0 6 0 0 2 2 0 0 0 2 0 0 0 0 0 0 0 0
 0 0 2 1 0 0 0 0 5 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 7 0 2 0 0 0 3 4 0
 0 0 0 0 0 0 0 2 0 0 0 0 0]
```

which is very similar to the label array I got as a result of my KMeans++ model. However, with Agglomerative Clustering I was able to make a dendrogram and then cut it to where there were 9 distinct clusters and I manually made lists of neighborhood indices for each cluster. The dendrogram looked like this:



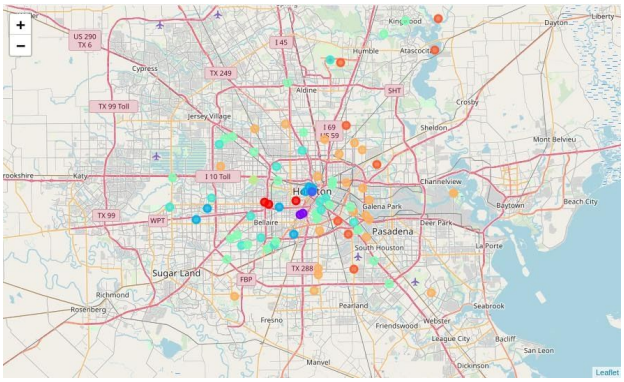
and I cut it at y = 9.

K Nearest Neighbors (KNN): With my neighborhoods clustered and a new 'labels' feature (based on the cluster number) added to my dataframe, I was able to train a KNN model on my data. First I split the data into a training set and a testing set and then I fit the model with the training set and used the test set to score the model. Using a for-loop, the graph of which is to the right, I found that the highest accuracy number of neighbors was 5, so I set my n_neighbors parameter to 5.



4 RESULTS

The agglomerative clustering labels showed that the neighborhoods could be split into 9 groups based on having a majority of:



0 - clothing/furniture stores
1 - zoo shops and zoo related venues
2 - hotels
3 - bars and stadium venues
4 - restaurants (especially mexican) and parks
5 - restaurants (wide variety)
6 - 'food' (the category was just 'food', which I assume could be restaurants, but also grocery stores or anything related to food. I am not sure, I would need to look into it much more)
7 - parks, historic sites, and theaters
8 - construction and discount stores

The final accuracy score of my KNN model was not even close to ideal, but it was still about 3 times better than if I had just randomly guessed the cluster (guessing would provide an 11% chance to be right while my model consistently predicted with 32% accuracy).

5 DISCUSSION

Although the final accuracy score was less than ideal, this was not entirely the KNN model's fault. There were some neighborhoods that were inherently more difficult to classify correctly because their clusters only had 2 or 3 neighborhoods total. If you account for the fact that if a neighborhood was in the test set to begin with, that means one less in the training set cluster which in turn makes it impossible for the cluster to be a majority because it could, at best, have 2 neighborhoods while my k was set to 5.

I wonder if I should have removed neighborhoods with significantly less venues or if maybe I should have removed (or joined together) the neighborhoods which made up the smaller clusters. I didn't do this because I wanted the potential user to have the option of any neighborhood. If I removed those with less venues, there would have been 11 neighborhoods which would never be an option for the user, and that doesn't seem fair.

6 CONCLUSION

Because my model was able to predict a cluster classification correctly with 3 times the accuracy of merely guessing, it could really help people narrow down the amount of neighborhoods to look at when deciding to move to Houston. I started with 88 ‘Super Neighborhoods’ and was able to split them up into nine more manageable clusters. A potential client would only need to select their preferences and then the KNN model would tell them which cluster is similar and provide them with a list of potential neighborhoods in which to live.

Future: In the future I think it might be best to train the models on only the top 10 venue categories of each neighborhood. I think this would help in two main ways: 1) The model would be easier to read and understand and 2) there would probably be less categories than the 282 originals for the user to look through in order to select their preferences. Also, as lightly touched upon above, I would try to figure out some other way to include the neighborhoods in the extra small clusters in order to make it more realistic for my KNN model to classify a user’s input as those potential neighborhoods.