Insights on 2020s Hotspot: Texas

IBM ML Capstone

Shivank Vatsal - December 2020

1. Introduction

Among 2020's fastest growing American cities, Texas made its mark with 6 cities ranking in the top 20. Dubbed the "Exodus to Texas," low unemployment rates, attractive housing prices, financial pressure from the pandemic, among many other factors have influenced people to move to the lone star state, especially Californians. Tax rates have been equally as inviting. Pioneered by Elon Musk, Tesla and other silicon-valley companies/startups have been relocating as well.

For those late to hop on the trend, households and businesses alike, knowing where house prices are still low, the types of businesses in the area and other information can help make decisions about moving. Machine learning algorithms which cluster fast-growing cities in Texas alongside other data visualization/analysis methods can speed up the process of searching for the right home. These tools may also be highly beneficial to investors who seek to invest in fast-growing cities which feature both a diverse competitive landscape and low house prices.

2. Data Collection

2.1 Sources

Data in this project includes a geoJSON file from open data made available by the city of Denton, Texas (link). This JSON file contains names of all counties in Texas including centroid coordinates as well as coordinates to define each county's boundaries. Median house prices by county from November 2020 are accessed from Texas A&M's Real Estate Research Center (link). Data on Texas' fastest growing cities, such as name and sociodemographic rank, is retrieved from an article published by Rice University's Institute for Urban Research and copied into an excel file (link). Population data of counties in Texas is retrieved by web scraping texas-demographics (link). Finally, geographic coordinates of cities and counties as well as local venue data are obtained by making calls to Python's geocoding library and the Foursquare API.

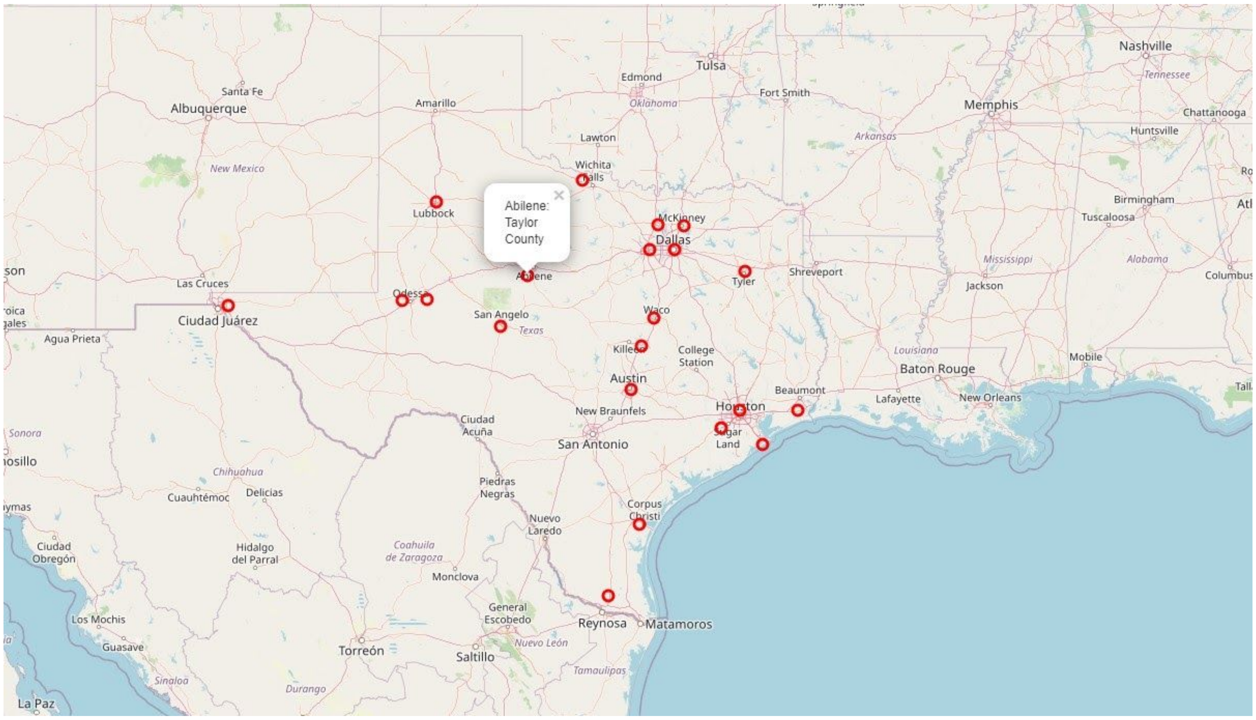
2.2 Transformation and Cleaning

Data was collected using a combination of Python's web-scraping and json-reading modules, Panda and BeautifulSoup, as well as written to an excel file and downloaded in the case of graphical data. Since this project concerned the fastest growing cities in Texas, any counties not containing a city from this list were cut. SD (sociodemographic) ranks for the cities were selectively pulled from a nationwide dataset, meaning the cities also had to be re-ranked since they were no longer ranked consecutively.

In regard to housing prices, while mean house prices were available by county, potential outliers rendered this option risky. Instead, median house prices were downloaded.

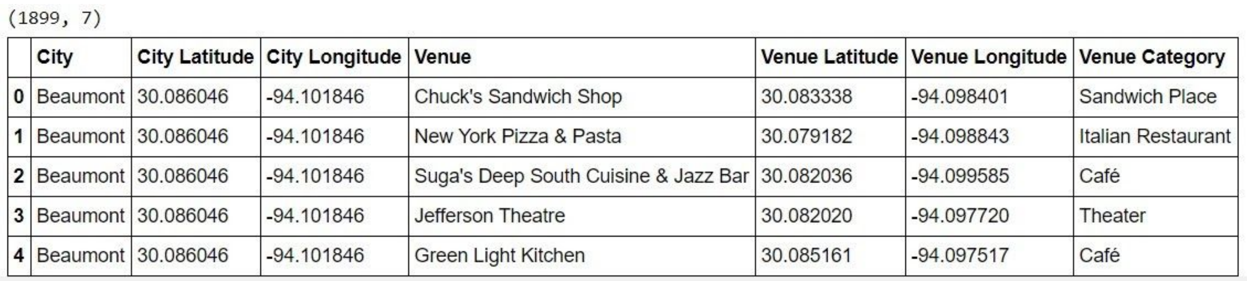
In the case of web scraping with BeautifulSoup, data returned was quite messy and included unrelated information. Any unnecessary zeroes, 'NaN' values, signs and commas were taken care of using applied lambda functions; inserting the remaining data into an array and reshaping it allowed for visualization as a data frame.

Finally, Python's geocoding library was used to assign coordinates (latitude, longitude) to each of Texas' growing cities. This data was then merged into the existing data frame. Before accessing the Foursquare API for local venues surrounding each city, preliminary geographic data was visualized using folium:



2.3 Foursquare API

The Foursquare API provides data on local venues based on geographical coordinates and a given radius. After defining a function with a radius of 1200m (~ 0.75 miles) to iterate through the list of cities, a data frame containing 1899 local venues alongside their locations was produced.

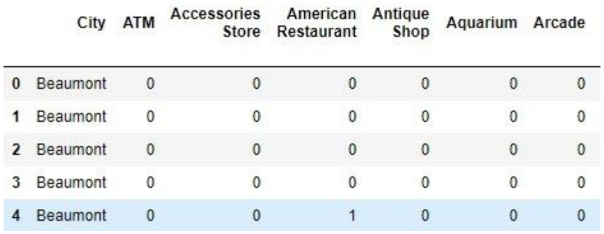


```
(1899, 7)
```

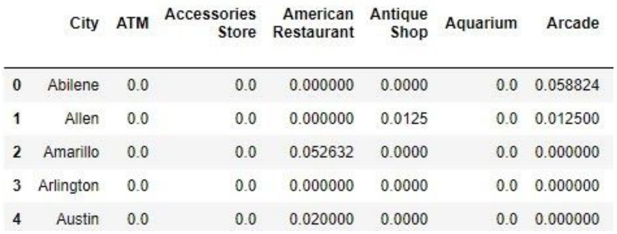
	City	City Latitude	City Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Beaumont	30.086046	-94.101846	Chuck's Sandwich Shop	30.083338	-94.098401	Sandwich Place
1	Beaumont	30.086046	-94.101846	New York Pizza & Pasta	30.079182	-94.098843	Italian Restaurant
2	Beaumont	30.086046	-94.101846	Suga's Deep South Cuisine & Jazz Bar	30.082036	-94.099585	Café
3	Beaumont	30.086046	-94.101846	Jefferson Theatre	30.082020	-94.097720	Theater
4	Beaumont	30.086046	-94.101846	Green Light Kitchen	30.085161	-94.097517	Café

2.4 One Hot Encoding

In order to prepare the venue data for clustering, the categorical data needed to be transformed into something the algorithm could interpret: numbers. To do this, ‘dummies’ of the venue category column were created in binary format to convey the presence of a venue in that category as a ‘1,’ and an absence as a ‘0.’ This is also known as one-hot-encoding. By calculating the mean of the binary data and grouping by city, the frequency of a certain venue category in any city became apparent. These became helpful in identifying the prominent characteristics of each city. To make the frequencies easier to understand, an additional data frame was created displaying the ten most common venue categories by city. Segments of each data frame can be seen below.

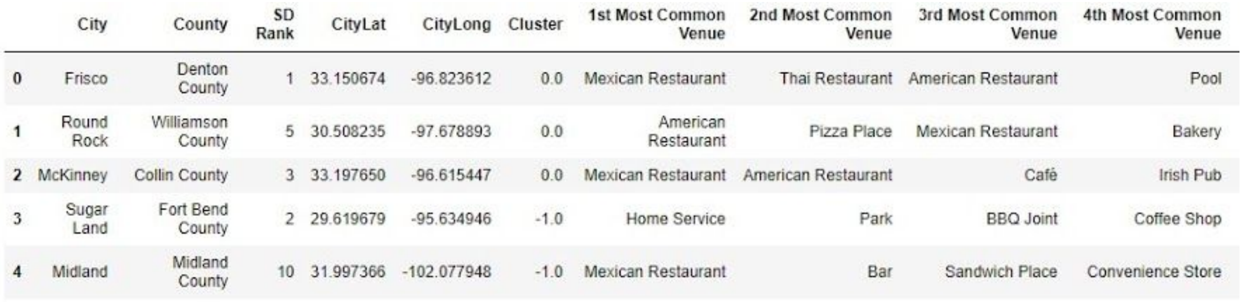


	City	ATM	Accessories Store	American Restaurant	Antique Shop	Aquarium	Arcade
0	Beaumont	0	0	0	0	0	0
1	Beaumont	0	0	0	0	0	0
2	Beaumont	0	0	0	0	0	0
3	Beaumont	0	0	0	0	0	0
4	Beaumont	0	0	1	0	0	0



	City	ATM	Accessories Store	American Restaurant	Antique Shop	Aquarium	Arcade
0	Abilene	0.0	0.0	0.000000	0.0000	0.0	0.058824
1	Allen	0.0	0.0	0.000000	0.0125	0.0	0.012500
2	Amarillo	0.0	0.0	0.052632	0.0000	0.0	0.000000
3	Arlington	0.0	0.0	0.000000	0.0000	0.0	0.000000
4	Austin	0.0	0.0	0.020000	0.0000	0.0	0.000000

One-Hot-Encoded → *Aggregated + Averaged*



	City	County	SD Rank	CityLat	CityLong	Cluster	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue
0	Frisco	Denton County	1	33.150674	-96.823612	0.0	Mexican Restaurant	Thai Restaurant	American Restaurant	Pool
1	Round Rock	Williamson County	5	30.508235	-97.678893	0.0	American Restaurant	Pizza Place	Mexican Restaurant	Bakery
2	McKinney	Collin County	3	33.197650	-96.615447	0.0	Mexican Restaurant	American Restaurant	Café	Irish Pub
3	Sugar Land	Fort Bend County	2	29.619679	-95.634946	-1.0	Home Service	Park	BBQ Joint	Coffee Shop
4	Midland	Midland County	10	31.997366	-102.077948	-1.0	Mexican Restaurant	Bar	Sandwich Place	Convenience Store

Most Common Venues

3. Methodology

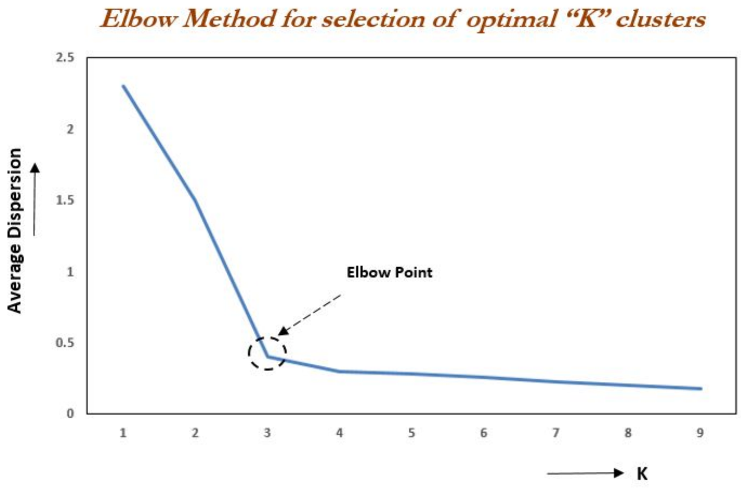
3.1 K-Means Clustering

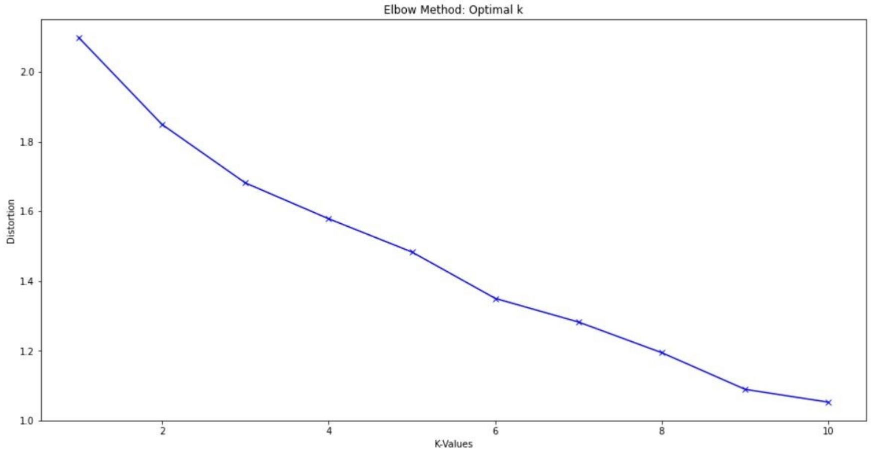
In order to cluster Texas’ fastest growing cities by their venue categories, the first algorithm of choice was K-Means. This unsupervised learning algorithm has a straightforward approach of using the distance between data points as a measure of similarity. After running the algorithm on the cities and their respective venue frequencies, a set of labeled data points was returned. However, the validity of these results was largely dependent upon how well the algorithm was able to distinguish clusters. In order to test this, the elbow method was employed.

The elbow method works by iterating the K-Means algorithm through a set of K-values (number of clusters) until there is a negligible effect on distortion reduction. Distortion can be interpreted as the average squared distance from sample points to their cluster centers. In other words, how incohesive the clusters are. At the point where distortion stops falling appreciably, an ‘elbow’ is visually apparent in the graph. This point is the optimal K-Value, or the point that represents the least amount of clusters with the least amount of distortion. Although the algorithm may still return labeled data, the lack of an elbow means K-Means is not able to identify distinctive clusters. In this situation, an elbow was not apparent.

Example of An Elbow Present

(Dangeti)



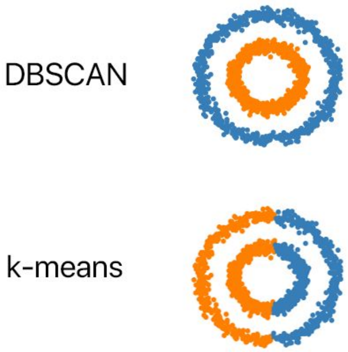


No Elbow Present

In This Scenario

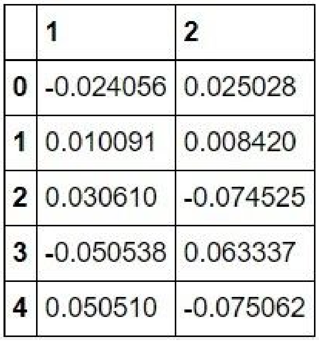
3.2 DBSCAN Algorithm + PCA

Given K-Means was unable to locate any defined clusters, it was likely that the nature of any potential clusters were not distance-related. Clusters may also be density-related, such as a group of data points that form concentric circles. In a scenario like this, DBSCAN, or density-based spatial clustering of applications with noise, may be more appropriate. DBSCAN works by clustering areas of high density that are separated from other clusters by areas of low density.



(Mattt, 2020)

Before making use of the algorithm, it was noted that the dimensionality of the dataset was very high. There were multiple venue category columns with different frequency values for each city's row. It was entirely possible that this impacted K-Means clustering since processing data in higher dimensions is more challenging for a number of reasons. The dimensionality was reduced using PCA, or principal component analysis. This method works to represent as much variance in the data as possible with a select number of transformed features from the original dataset. A more in-depth explanation of PCA can be found here. Upon using PCA, the original data was represented using a two-dimensional data frame.

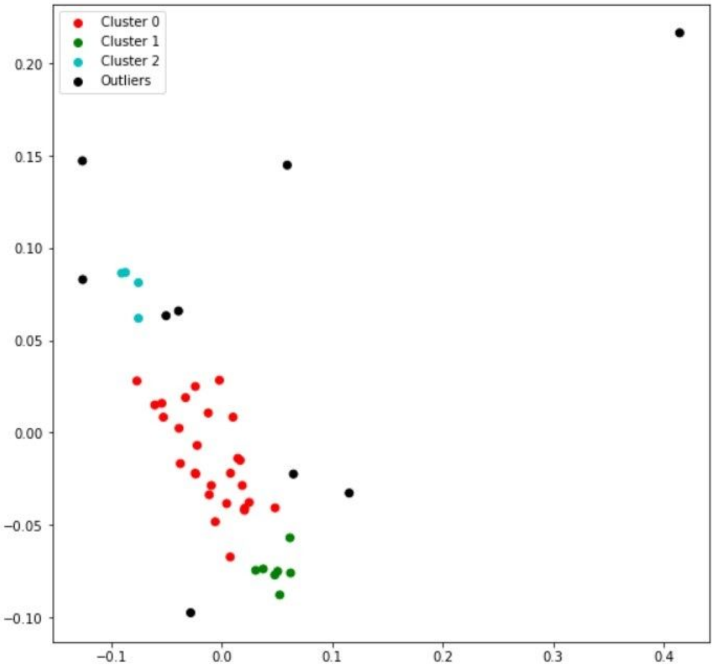


	1	2
0	-0.024056	0.025028
1	0.010091	0.008420
2	0.030610	-0.074525
3	-0.050538	0.063337
4	0.050510	-0.075062

In order to run the DBSCAN algorithm, two parameters are always necessary: ε (epsilon) and minimum-samples. Minimum-samples refers to the least number of data points necessary to form a cluster. It is important to note that DBSCAN does not cluster all data and identifies remaining points as outliers. As a consequence, inputting a low minimum-samples value, such as two, may result in the algorithm assigning a few outliers to their own cluster. When working with two-dimensional data, it is typically a good idea to input a value of four.

The epsilon parameter on the other hand refers to the maximum distance allowed between two points that belong to the same cluster. In other words, the maximum distance to classify two points as neighbors. An optimal epsilon value is unique to its respective dataset. While there are methods known to help assist in finding the optimal, in this scenario, it was found using trial and error.

Upon running the DBSCAN algorithm with an epsilon of 0.02377 and 4 minimum samples, each data point was assigned a label: 0, 1, 2 or



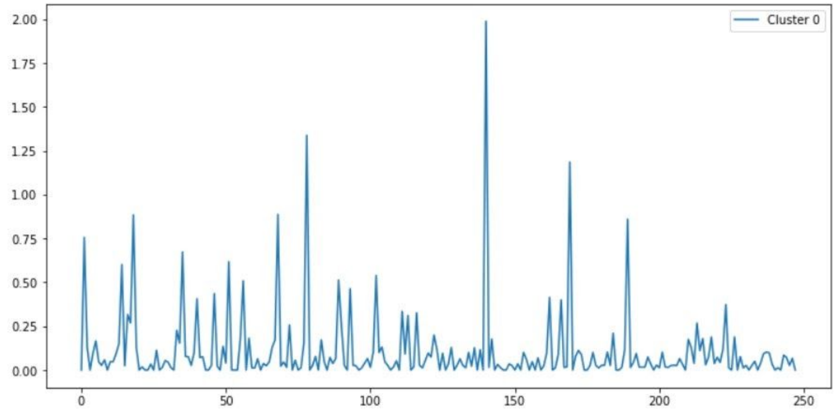
-1. A value of -1 indicated that the point was an outlier, while the other values represented one of three clusters the point was assigned to. Aside from the high dimensionality of the original data, a visual of the results revealed how K-Means may have struggled to cluster points in such close proximity. This is precisely why DBSCAN was a better candidate for the situation.

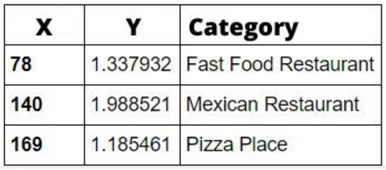
4. Results

4.1 Cluster Characterization

With cluster labels for each city, there was still no information on the nature of these clusters. What made one different from the other? Why were the cities put into these clusters? Fortunately, venue category frequencies were able to answer these questions. After creating a data frame containing each city alongside its cluster label and venue category frequencies, the city column was dropped. After grouping by cluster label and aggregating the frequencies, it was then easy to see the venue category occurrences in each cluster. These results were visualized as a graph, and with a little filtering, a table displaying the most common venue categories by cluster was produced. Finally, each cluster was titled based on their respective characteristics.

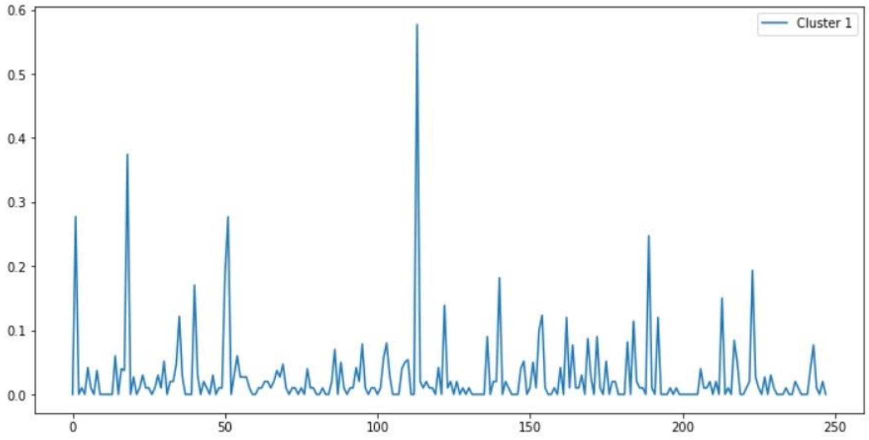
Note: Y represents Venue Category Frequency

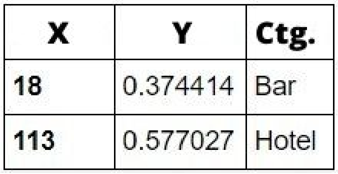




X	Y	Category
78	1.337932	Fast Food Restaurant
140	1.988521	Mexican Restaurant
169	1.185461	Pizza Place

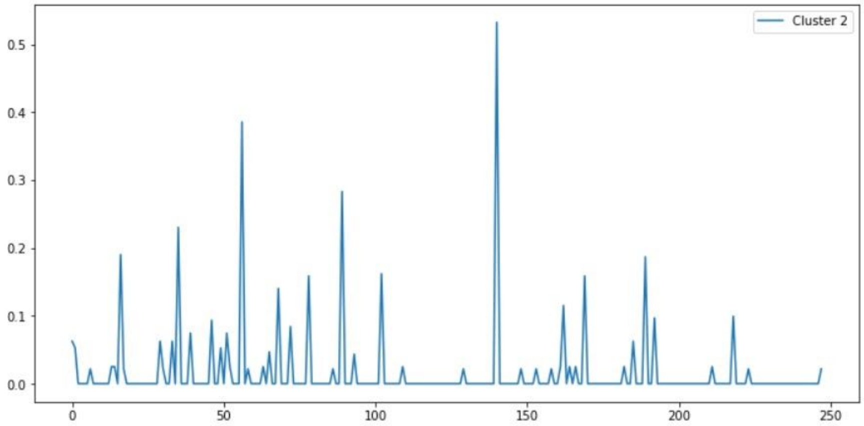
Cluster 0 *would feature ‘Fast Food and Mexican Cuisine.’*

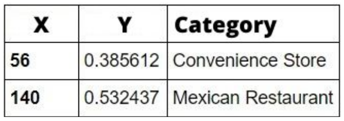




X	Y	Ctg.
18	0.374414	Bar
113	0.577027	Hotel

Cluster 1 *would feature 'Hotels and Social Venues.'*



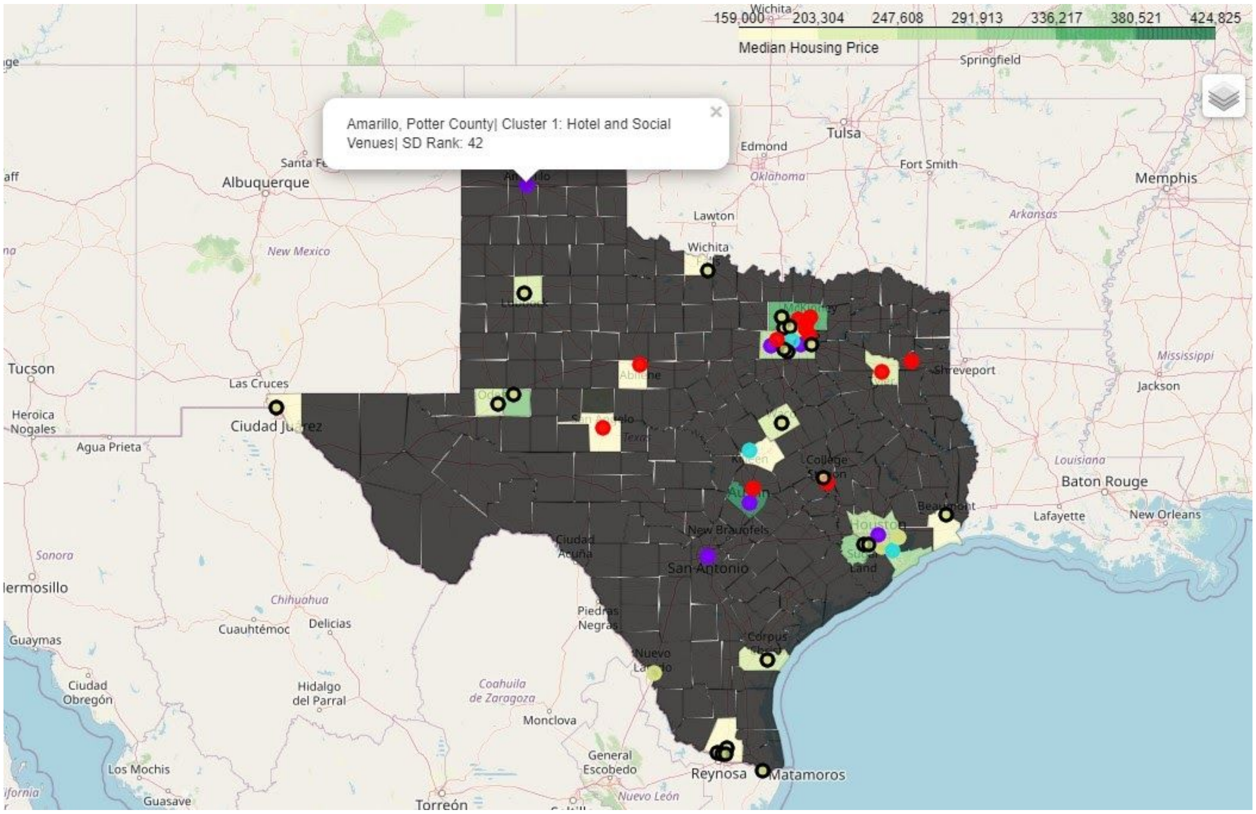


X	Y	Category
56	0.385612	Convenience Store
140	0.532437	Mexican Restaurant

Cluster 2 *would feature 'Convenience Stores and Mexican Cuisine.'*

4.2 Choropleth Map

After merging the cluster labels, county/city coordinates, median house prices and other relevant data into one data frame, a choropleth map with interactive pop ups was produced. As the legend indicates, darker colors translate to more expensive median house prices, whereas lighter colors indicate cheaper median house prices. Each popup's color indicates whether it belongs to a certain cluster, or is an outlier, which is denoted by black. By clicking on a city's popup, its title, SD Rank, and cluster characteristics are also displayed.



5. Discussion

Evidently, this project somewhat successfully clustered the fastest growing cities in Texas. However, the extent to which the final visual may help investors and moving households is limited. In order to help investors more, updated city-specific housing prices would be necessary, alongside the names of businesses present in the city as opposed to only their venue categories. For moving households, the same is applicable in addition to the statistics that contribute to each city's sociodemographic rank, such as crime rate, school ratings, etc.
Some issues can be addressed by making additional calls to the Foursquare API, however, other information that is not readily available on the internet would require a lot more research.

6. Conclusion

In brief, there may have been other well-suited clustering methods and steps that could have been taken to improve/supplement the results of DBSCAN, K-Means and PCA. Nevertheless, this project is a foundation for approaching the problem of deciding where to invest or move with so many factors at hand. Products like these may especially have applications in areas where people are dealing with foreign territory.

GitHub Repo and Blog:

Source code can be found here. Blogspot can be found here.

More information:

PCA ~ https://towardsdatascience.com/how-exactly-does-pca-work-5c342c3077fe

DBSCAN ~ https://www.mygreatlearning.com/blog/dbscan-algorithm/

K-Means ~ https://towardsdatascience.com/how-does-k-means-clustering-in-machine-learning-work-fdaaaf5acfa0

Image References:

Dangeti, Pratap. “Statistics for Machine Learning.” *O'Reilly Online Learning*, Packt Publishing, www.oreilly.com/library/view/statistics-for-machine/9781788295758/c71ea970-0f3c-4973-8d3a-b09a7a6553c1.xhtml.

Mattt. Feb, 2020. DBSCAN. Portland (OR): Github; [accessed 2021 Jan 30].https://github.com/NSHipster/DBSCAN.