Exploring Riyadh City neighborhoods

Applied Data Science Capstone

HICHEM Dridi

4/5/21

IBM Data Science Professional Certificate

Table of Contents

List of Figures

1 Introduction

1.1 Background

During the last course of the professional certificate, we've been using different tools to explore famous cities like *New York & Toronto*. The process was simple: gather data about the city, use *Foursquare API* to explore the venues in the neighborhoods and then group similar neighborhoods into different clusters.

In order to accomplish the final assignment, I've decided to explore also one of the popular cities in the world: ***Riyadh City***.

1.2 About the city

Riyadh ('The Gardens') is the capital and main financial hub of *Saudi Arabia,* and the largest city on the Arabian Peninsula. Located in the center of the an-Nafud desert, on the eastern part of the Najd plateau, the city sits at an average of 600 meters (2,000 ft) above sea level, and receives around ***5 million tourists*** each year, making it the ***forty-ninth most visited city in the world*** and the ***6th in the Middle East***.

Riyadh had a population of ***7.6 million people*** in 2019, making it the most-populous city in Saudi Arabia, 3rd most populous in the Middle East, and 38th most populous in Asia. (1)

1.3 Objective

Considering the mentioned characteristics about the city, the main goal of this project is to explore the city and answer the following questions:

- What are the most common venues in the city?
- Which neighborhoods in the city are similar?

The answers to these questions could be exploited by:

- Tourists that would like to come and explore the city.
- People moving to the city for a job offer, or looking for a better neighborhood to live in.
- Small investors who would like to open a restaurant, a market etc.

2 Data Acquisition and wrangling

The pre-step before starting to acquire data about the neighborhoods is to familiarize myself with the country rules and concepts, how they divide states and cities etc.

After that, in order to acquire a clean, well-structured data, the following steps were made:

2.1 Collect data about the city

The main information about the city are: **Municipality, Neighborhood and Postal Code**. These three features are necessary to identify each neighborhood in the city and to acquire further information in the data preparation part.

After a little research, I found all the mentioned data in this web page. The page has the name of each municipality in a header format and under each one land a table that contains neighborhoods' names and their corresponding postal code. Scrapping that page gives the results showed in Figure 1:

	Municipality_ar	Neighborhood_ar	Postal_code
0	بلدية العليا	حي العليا	12222
1	بلدية العليا	حي الازدهار	12486
2	بلدية العليا	حي المصيف	12466
3	بلدية العليا	حي التعاون	12475
4	بلدية العليا	حي الورود	12252

Figure 1. Web scrapping results for information about Riyadh city neighborhoods

2.2 Data wrangling

2.2.1 Data transformation

The above figure shows that municipalities and neighborhoods names are in Arabic. So, the first was translating these names to English.

To accomplish this task, I used *deep-translator* module, it's one of the modules that makes it straightforward to translate texts, and it provides support for multiple famous translators.

The last transformation step was to remove 'municipality', 'neighborhood' and 'district' from the translated names.

2.2.2 Data cleaning

The inspection of the data showed there is 16 municipalities and 203 neighborhoods in total with no nulls.

Checking for duplicates gives:

- No duplicate values in general terms.
- Two neighborhoods appeared in two different municipalities.
- Some neighborhoods share the same postal code.

To deal with the duplicates in the data set I followed these instructions:

- In *Saudi Arabia* two municipalities could share the same neighborhood. So, we'll merge the municipalities names.
- Neighborhoods with the same postal codes and the same municipality will be merged together.

- ➢ If a postal code has different neighborhoods and different municipalities, we'll keep one item of that postal code with merged names.
- ➢ For the exception (postal code number 14522), one neighborhood had the correct name (merged names of two neighborhoods), we'll keep it and drop the second one.

The cleaning process left us with 185 entry in the data set.

2.3 Data preparation

The goal of this project is to group neighborhoods based on their similarities. The main factor to accomplish that is the most common venues in each neighborhood. To explore venues, i.e. fetch related data about venues in each neighborhood (venue's name, category etc.) we'll use the *Foursquare API*. This API takes the latitude and the longitude of each neighborhood and return a descriptive list about the venues in it.

To prepare our data set for further work, we need the geo-coordination of each neighborhood. The results in Figure 2 were acquired using *Geocoder API*, it's an API that finds you the geo-coordinates of a known address, place, locality or administrative area, even if the query is incomplete or partly incorrect.

	Municipality_ar	Neighborhood_ar	Postal_code	Municipality_en	Neighborhood_en	Latitude	Longitude
0	بلدية العليا	حي العليا	12222	Olaya	Al Alia	24.69502	46.69004
1	بلدية العليا	حي الازدهار	12486	Olaya	Al-Izdihar	24.77904	46.72446
2	بلدية العليا	حي المصيف	12466	Olaya	Al-Masif	24.76386	46.68849
3	بلدية العليا	حي التعاون	12475	Olaya	Al-Taawon	24.76724	46.69640
4	بلدية العليا	حي الورود	12252	Olaya	Al Worood	24.72213	46.68584

Figure 2. Data collection final results

All the entries have their corresponding latitude and longitude values.

3 Methodologies

3.1 Data Preprocessing

3.1.1 Collecting neighborhoods' venues

The main factor of clustering the city neighborhoods is the type and the number of venues that exists in each one of them. For this task we'll count on the *Foursquare API*.

The *Foursquare Places API* provides location-based experiences with diverse information about venues, users, photos, and check-ins. The API supports real time access to places, Snap-to-Place that assigns users to specific locations, and Geo-tag. Additionally, *Foursquare* allows developers to build audience segments for analysis and measurement. JSON is the preferred response format. (2)

So, for each neighborhood in our dataset, we send a URL request that contains:

- The client credentials (ID and secret).
- The API version (set to 20180605).

- Neighborhood's geo-coordination.
- Radius (set to 500 meters)
- Limit (at least 100 venue per neighborhood)

The API sends the results of each query in a *json* file. The data is collected and stored in a table called *Riyadh_venues* as shown in Figure 3 below.

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Al Alia	24.69502	46.69004	Paskalia (بسكاليا)	24.697674	46.687086	Restaurant
1	Al Alia	24.69502	46.69004	Cacti Cafe	24.698523	46.688123	Coffee Shop
2	Al Alia	24.69502	46.69004	Starbucks (ستاربكس)	24.699015	46.689939	Coffee Shop
3	Al Alia	24.69502	46.69004	Tutti Cafè (أنواع القهوة)	24.697852	46.687181	Café
4	Al Alia	24.69502	46.69004	Adidas Originals	24.698398	46.687715	Sporting Goods Shop
5	Al Alia	24.69502	46.69004	Tim Hortons (تيم هورتنز)	24.698790	46.689771	Coffee Shop
6	Al Alia	24.69502	46.69004	Le Gourmet (الذواق)	24.691527	46.691695	Bakery
7	Al Alia	24.69502	46.69004	Zara Man (زارا مان)	24.698963	46.691498	Clothing Store
8	Al Alia	24.69502	46.69004	Fitness First Ladies	24.698749	46.691229	Gym / Fitness Center
9	Al Alia	24.69502	46.69004	Labouchee (لابوشيه)	24.691923	46.692852	Cupcake Shop
10	Al Alia	24.69502	46.69004	الرماح للأقمشة الرجالية	24.690697	46.689549	Men's Store
11	Al Alia	24.69502	46.69004	Dunkin' (دانكن)	24.698873	46.689581	Donut Shop
12	Al Alia	24.69502	46.69004	Gardenia	24.698984	46.689034	Flower Shop
13	Al Alia	24.69502	46.69004	PAUL (بول)	24.697306	46.686645	French Restaurant
14	Al Alia	24.69502	46.69004	Fire Grill (فاير جريل)	24.698120	46.687036	Mexican Restaurant

Figure 3. Collecting venues data

The process results in *2252* venues in the city with *239* unique categories.

3.1.2 Visualizing primary results

Now that we have a list of venues for each neighborhood and their related information from the *Foursquare API*, the following figures show the count of venues in each municipality (Figure 4) and for the top 10 neighborhoods (Figure 5).

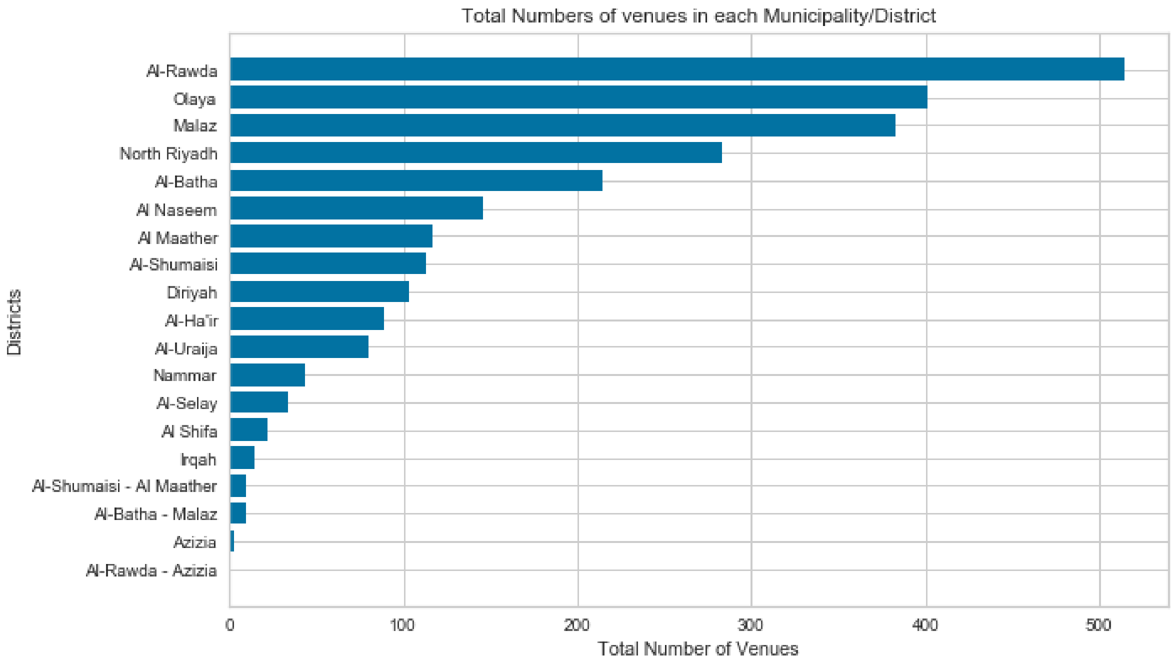


Figure 4. Total number of venues per municipality

From the bar plot above we can divide municipalities to three sections:

- Municipalities with crowded neighborhoods (more than *200* venues).
- Medium size municipalities: more than *80* and less than *200* venues.
- Small districts with less than *50* venues.

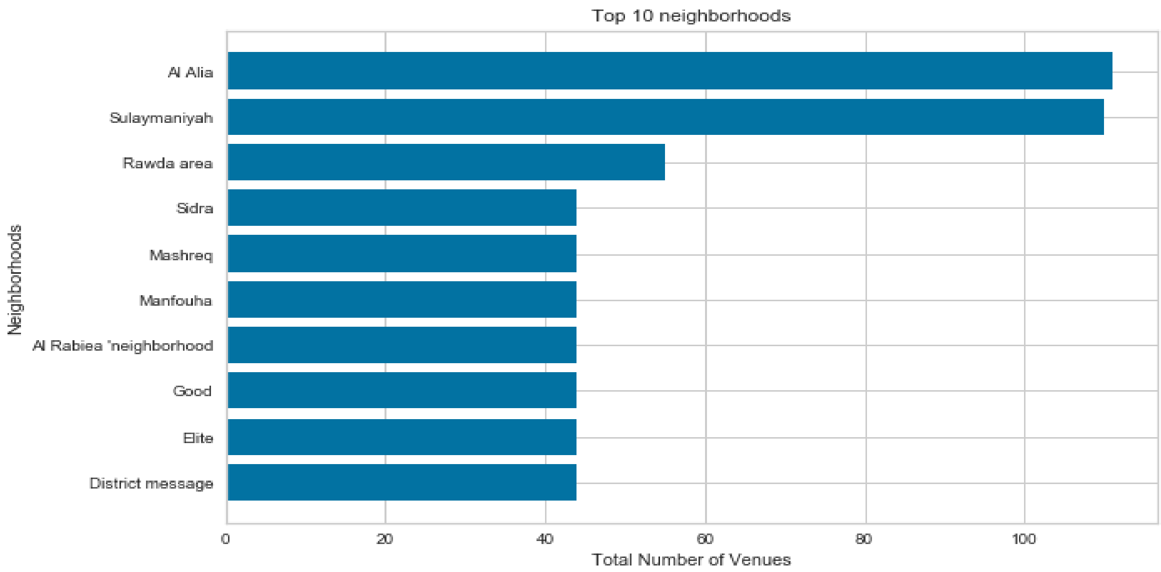


Figure 5. Top 10 neighborhoods

The neighborhoods of *'Al Alia'* and *'Sulaymaniyah'* differ a lot from the rest. They have more than *100* venues each one while the rest have less than *60*.

This indicates that these two neighborhoods are very crowded with habitants and workers.

3.2 Most common venues for each neighborhood

To get a clear picture about each neighborhood, I chose to get the list of the most *10 common venues*.

From the *Riyadh_venues* dataset, for each neighborhood, I took the list of venues category and computed the frequency of each one. The resulting table has the name of *neigh_grouped* and shown in Figure 6.

	Neighborhood	ATM	Accessories Store	Afghan Restaurant	African Restaurant	Airport Lounge	Airport Terminal	American Restaurant	Amphitheater	Antique Shop	...	Used Bookstore	Vacation Rental	Vegetarian / Vegan Restaurant	Vide Gam Stor
0	Ad-Durahimiyah	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	...	0.0	0.0	0.0	0.
1	Al Alia	0.0	0.0	0.0	0.0	0.0	0.0	0.009174	0.0	0.0	...	0.0	0.0	0.0	0.
2	Al Aqiq	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	...	0.0	0.0	0.0	0.
3	Al Faisaliah	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	...	0.0	0.0	0.0	0.
4	Al Fakhriyah	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	...	0.0	0.0	0.0	0.

5 rows × 240 columns

Figure 6. Venues frequency distribution for each neighborhood

Sorting the results in descending order, I end up with the results shown in Figure 7.

	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	Ad-Durahimiyah	Food Truck	Sandwich Place	Arts & Crafts Store	Yoga Studio	Film Studio	Filipino Restaurant	Fast Food Restaurant	Farm	Falafel Restaurant	Fabric Shop
1	Al Alia	Coffee Shop	Café	Cupcake Shop	Dessert Shop	Sporting Goods Shop	Men's Store	Clothing Store	Donut Shop	Bakery	Ice Cream Shop
2	Al Aqiq	Juice Bar	Mobile Phone Shop	Breakfast Spot	Fast Food Restaurant	Sporting Goods Shop	Burger Joint	Food Truck	Cafeteria	Café	Restaurant
3	Al Faisaliah	Coffee Roaster	Department Store	Furniture / Home Store	Auto Garage	Yoga Studio	Electronics Store	Flea Market	Film Studio	Filipino Restaurant	Fast Food Restaurant
4	Al Fakhriyah	Hookah Bar	Asian Restaurant	Coffee Shop	Pedestrian Plaza	Campground	Sports Bar	Health Food Store	Health & Beauty Service	Fast Food Restaurant	Farm

Figure 7. List of 10 common venues in each neighborhood

Now each individual in this table has information about the neighborhood's name and the most *10 common venues* in descending order.

3.3 Neighborhoods clustering

To perform neighborhoods segmentation based on similarities of venues, I used the data of the *neigh_grouped* table, it's already scaled (all values range between 0 and 1).

My first choice was to use the *K-Means* clustering algorithm. Because of my lacking of knowledge about the neighborhoods, choosing a value for k isn't arbitrary, so I tried to find what is the optimal value of k would be.

3.3.1 The optimal value of K for K-Means algorithm

K-means is a simple unsupervised machine learning algorithm that groups data into a specified number (k) of clusters. Because the user must specify in advance what k to choose, the algorithm is somewhat naive – it assigns all members to k clusters even if that is not the right k for the dataset. (3)

Searching for the best value for K is a challenge for data scientists. For this task I'll start with the most common one: *the elbow method*.

The *elbow method* runs *k-means* clustering on the dataset for a range of values for k (say from 1-10) and then for each value of k computes an average score for all clusters. By default, the *distortion score* is computed, the sum of square distances from each point to its assigned center. Other metrics can also be used such as the *silhouette score*, the mean silhouette coefficient for all samples or the *Calinski_harabasz score*, which computes the ratio of dispersion between and within clusters. (3)

To get quick results, I used the *Yellowbrick* library.

The *Yellowbrick* library is a diagnostic visualization platform for machine learning that allows data scientists to steer the model selection process. It extends the *scikit-learn API* with a new core object: The *Visualizer*. Visualizers allow visual models to be fit and transformed as part of the *scikit-learn* pipeline process, providing visual diagnostics throughout the transformation of high-dimensional data. (4)

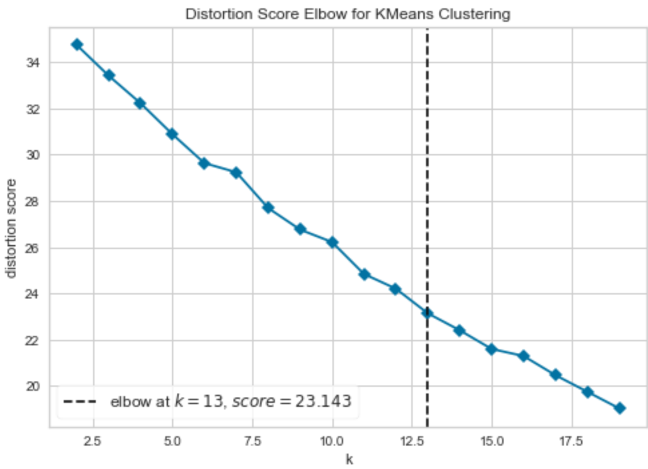


Figure 8. Distortion score Elbow for K-Means clustering

In general, increasing the value of K will always result in a decrease in the distortion score. We use the elbow method to find at which point there's a sharp decrease in the score, but the line above keeps decreasing gradually and indicates that the best value of K will be *13*.

Dividing our neighborhoods into 13 different groups will make the results interpretation a hard task. We need to try another method.

The *Silhouette Coefficient* is used when the ground-truth about the dataset is unknown and computes the density of clusters computed by the model. The score is computed by averaging the silhouette coefficient for each sample, computed as the difference between the average intra-cluster distance and the mean nearest-cluster distance for each sample, normalized by the maximum value. This produces a score between 1 and -1, where 1 is highly dense clusters and -1 is completely incorrect clustering. (5)

I've ran a little experiment to determine the optimal value of K. using the *SilhouetteVisualizer* function for different values of K that range from 2 to 15. Results are shown in appendix n°1.

Figure 9 shows the best result founded was with *6* clusters.

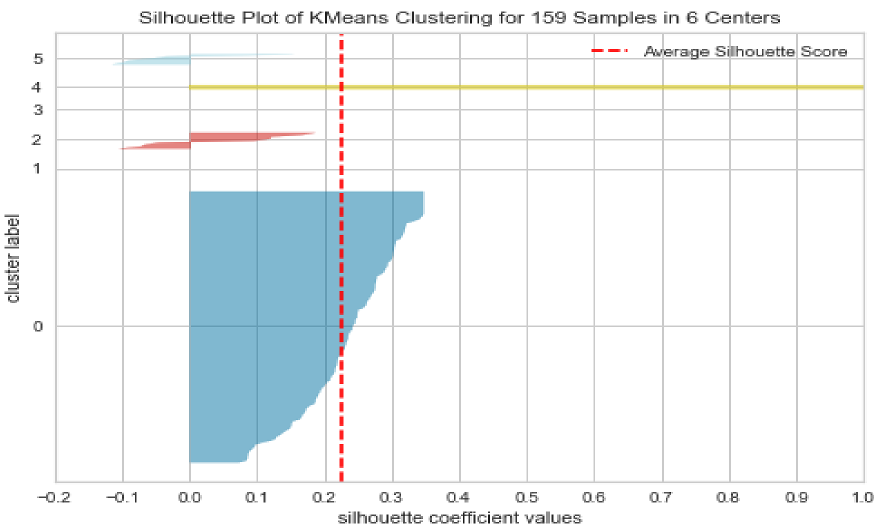


Figure 9. Silhouette plot of K-Means with 6 centers

The average silhouette score for 6 clusters is about *0.22*, it's the best score but we'll end up with two tiny clusters. The density of the blue zone shows that the cluster 0 have the majority of neighborhoods, and the negative scores of some elements in clusters 2 and 5 means that these elements were incorrectly classified.

The pros of clustering with six centers is that the silhouette score for cluster 4 is equal to 1, this means that neighborhoods within this cluster do really belong to it and they are very similar.

Observing the rest of the plots in appendix 1, we see that as the value of K increase two things happen:

- More tinier clusters appear.
- We have clusters with negative silhouette scores which means that it's an incorrect cluster.

Referring to the results of both the elbow method and silhouette score, I cannot determine which is the best value of k to pass to the K-Means algorithm. Even with the 6 clusters, having two tiny clusters and wrong clustering is not a good result.

3.3.2 DBSCAN clustering

DBSCAN, or Density-Based Spatial Clustering of Applications with Noise, is an unsupervised machine learning algorithm. Unsupervised machine learning algorithms are used to classify unlabeled data. In other words, the samples used to train our model do not come with predefined categories. In comparison to other clustering algorithms, DBSCAN is particularly well suited for problems which require:

1) Minimal domain knowledge to determine the input parameters (i.e. **K** in k-means and **Dmin** in hierarchical clustering).
2) Discovery of clusters with arbitrary shapes.
3) Good efficiency on large databases.

The algorithm works by computing the distance between every point and all other points. We then place the points into one of three categories.

Core point: A point with at least min_samples points whose distance with respect to the point is below the threshold defined by epsilon.

Border point: A point that isn't in close proximity to at least min_samples points but is close enough to one or more core point. Border points are included in the cluster of the closest core point.

Noise point: Points that aren't close enough to core points to be considered border points. Noise points are ignored. That is to say, they aren't part of any cluster. (6)

DBSCAN technique is one of the most common clustering algorithms which works based on density of object. The whole idea is that if a particular point belongs to a cluster, it should be near to lots of other points in that cluster. It works based on two parameters: *Epsilon* and *Minimum Points.*

- **Epsilon** determine a specified radius that if includes enough number of points within, we call it dense area.
- **Minimum_Samples** determine the minimum number of data points we want in a neighborhood to define a cluster. (7)

i. Find the best Epsilon:

Before we fit our data to the model, I need to find the best value for epsilon following these steps:

1) Calculate the distance from each point to its closest neighbor using the *NearestNeighbors* library. The *kneighbors* method returns two arrays, one which contains the distance to the closest n_neighbors points and the other which contains the index for each of those points.
2) Sort and plot results.
3) The optimal value for epsilon will be found at the point of maximum curvature.

Figure 10 shows the results.

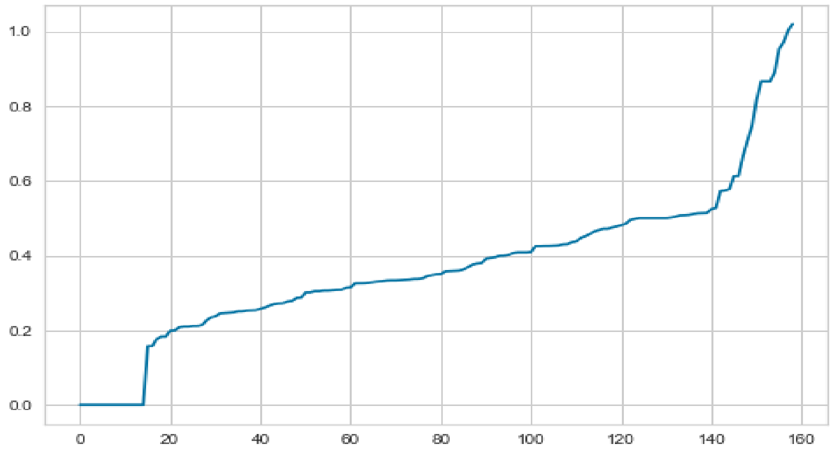


Figure 10. Optimal value for epsilon with elbow method

We observe that the best value for epsilon is around *0.5*.

ii. Run the model:

At this stage, the data in the *neigh_grouped* dataframe is fitted to the *DBSCAN* model with the following parameters: **eps** = 0.5 and **n_samples** = 3. The clustering results are merged with the original table (*Riyadh_df)* to get the results in Figure 11.

	Municipality_ar	Neighborhood_ar	Postal_code	Municipality_en	Neighborhood_en	Latitude	Longitude	Cluster_labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Con V
0	بلدية العليا	حي العليا	12222	Olaya	Al Alia	24.69502	46.69004	0.0	Coffee Shop	Café	Cupcake Shop	D
1	بلدية العليا	حي الازدهار	12486	Olaya	Al-Izdihar	24.77904	46.72446	0.0	Coffee Shop	Jewelry Store	Falafel Restaurant	Bik
2	بلدية العليا	حي المصيف	12466	Olaya	Al-Masif	24.76386	46.68849	0.0	Coffee Shop	Clothing Store	Hotel	E
3	بلدية العليا	حي التعاون	12475	Olaya	Al-Taawon	24.76724	46.69640	0.0	Coffee Shop	Donut Shop	Gift Shop	
4	بلدية العليا	حي الورود	12252	Olaya	Al-Worood	24.72213	46.68584	0.0	Coffee Shop	Pizza Place	Furniture / Home Store	Phar

Figure 11. Neighborhoods dataframe with clustering labels

Counting the number of elements in each cluster, I found:

- 131 neighborhoods in cluster 0.
- 3 neighborhoods in cluster 1.
- And 32 neighborhoods labeled as noise.

3.4 Saving the results

For easy access and further analysis in the future, the results are stored in an *Excel* spreadsheet. The first sheet contains the data about the neighborhoods (municipality, neighborhood, postal code and geo-coordination) and the second sheet has the final results (neighborhoods, cluster, top 10 common venues).

4 Results

4.1 Riyadh city map

At the beginning, I wanted to see how is the distribution of the clusters on the map. So using the *folium* library, the latitude and longitude values and the cluster label of each neighborhood, here is a plot in Figure 12 that shows the distribution of our neighborhoods geographically according to their corresponding clusters :

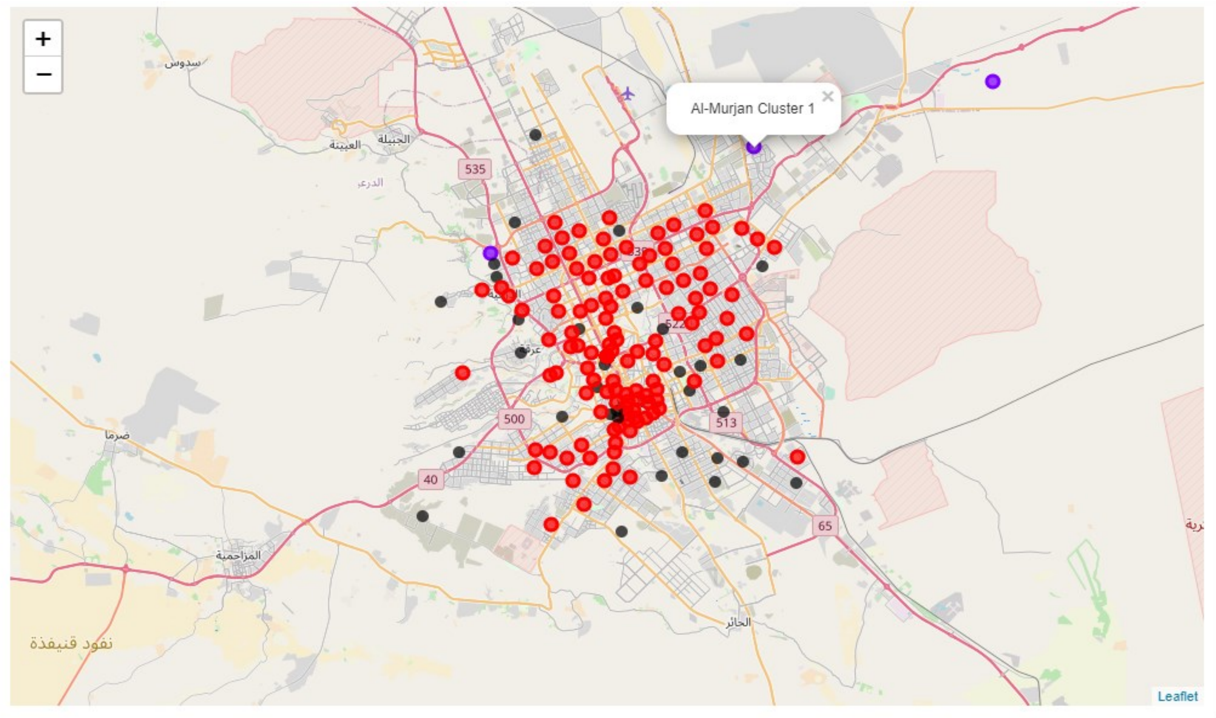


Figure 12. Riyadh city map with the distribution of neighborhoods according to their clusters

Points in black are the neighborhoods that our model identified as noise points, red points belongs to cluster 0 and purple points are neighborhoods of cluster 1.

4.2 Examining the clusters

Now that we have for each neighborhood in the in the dataset: name, geo-coordination, cluster label and the 10 most common venues, as shown in Figure 11, the final step in this project is to generate stacked bar charts for each cluster (even for the noise points) to understand what are the venues among the top 10 ones that appear the most.

For this purpose, I created a function that takes as inputs: the dataframe, the cluster label and the minimum number of occurrences for a venue to appear in the plot.

We start with the biggest cluster:

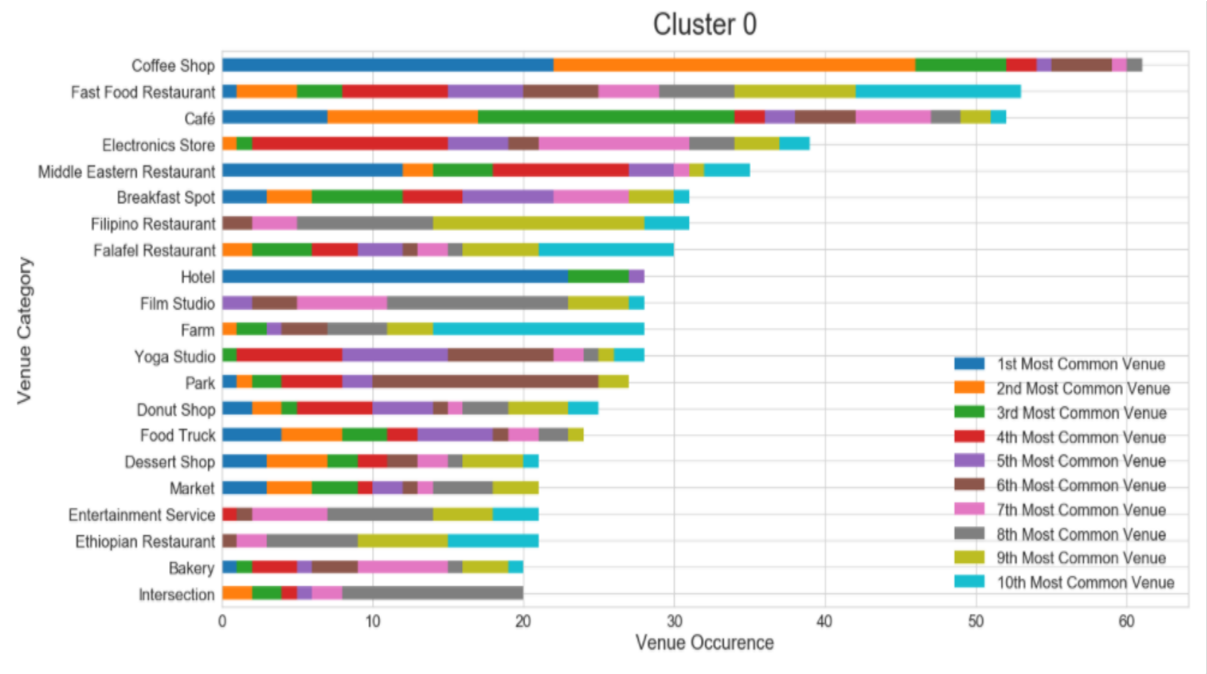


Figure 13. Cluster 0 most common venues

The 131 neighborhoods in this cluster (red points in the map) represent the modern city:

- Coffee shops appeared more than *60* times in the most *10* common venues, and more than *20* times as the first most common venue in cluster 0 neighborhoods.
- A lot of restaurants exist in these neighborhoods with different specialties (fast food, middle eastern, Filipino etc.)
- Hotels (*22* time as first common venue) and café places (more than 50 appearances) are also so common in these neighborhoods.

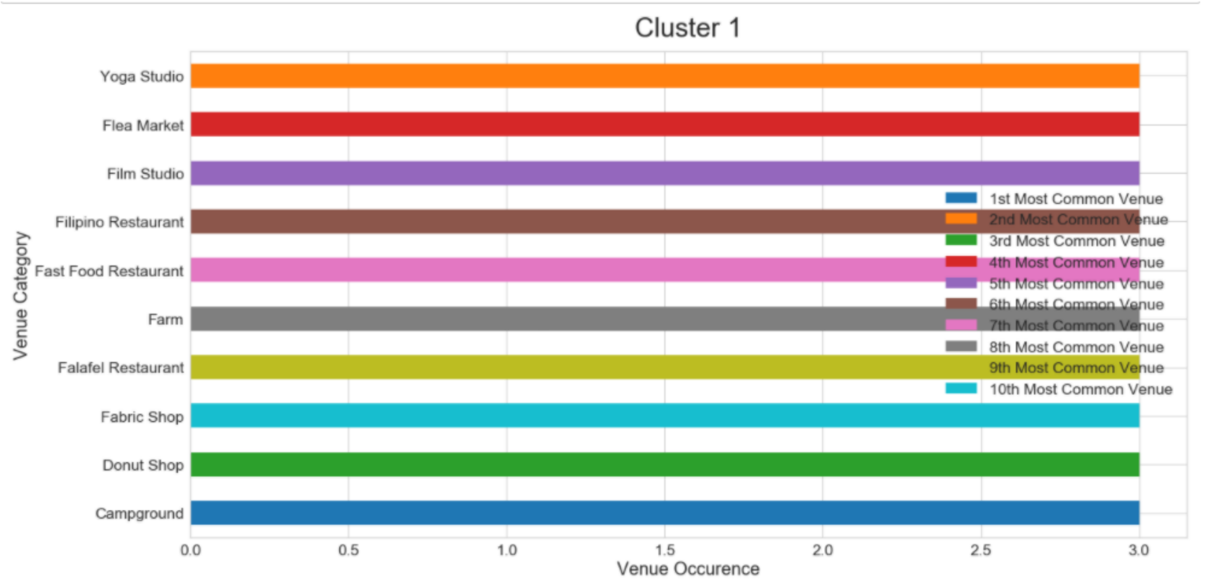


Figure 14. Cluster 1 most common venues

With only 3 neighborhoods in this cluster, we see:

- Campgrounds is the most common venue.
- Yoga studios is the second common venue and film studios is the fifth in all these neighborhoods.
- There exist some farms, some restaurants (with no fast food restaurants) and some shops in these neighborhoods.

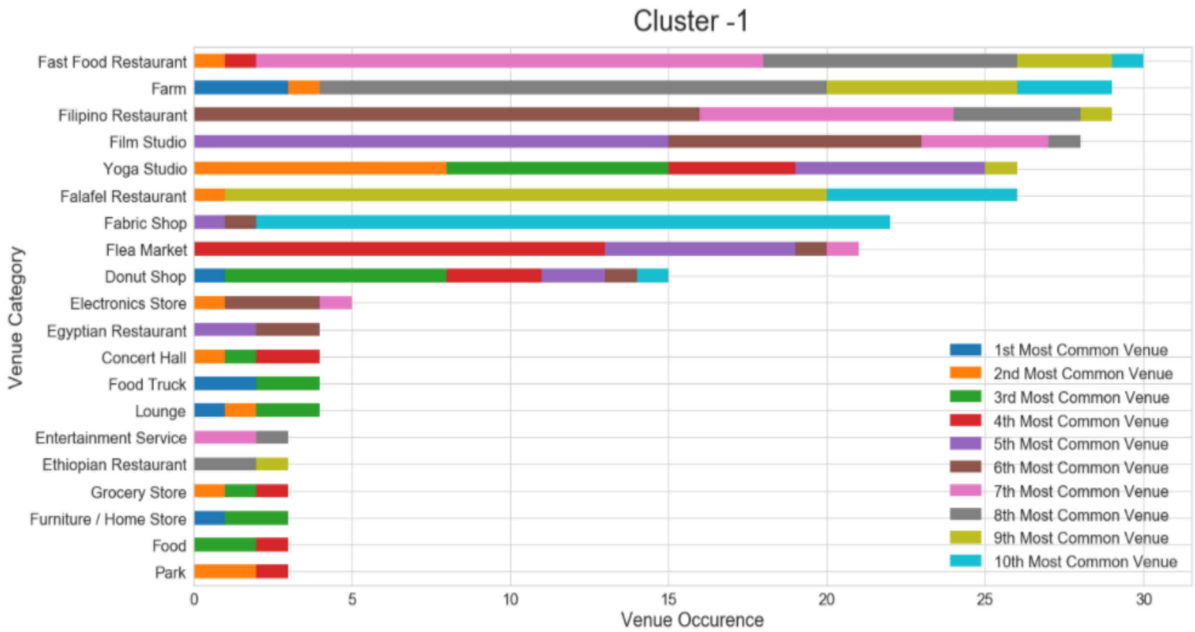


Figure 15. Top 10 common venues for neighborhoods clustered as noise

The advantage of *DBSCAN* clustering algorithm is its capability to recognize noise points, we observe from the above plot that neighborhoods clustered as noise have a lot of restaurants as common places but also studios, farms shops, parks, lounges, concert halls etc.

5 Discussion

5.1 Overall findings

Looking at the results of cluster 0 we conclude that:

- More than *70%* of the city neighborhoods shares the same characteristics of the modern city neighborhoods: a lot of coffee shops, café places, hotels, restaurants etc.
- The city is so populated, receives a lot of tourists each year and it contains a lot of companies, that explains the popularity of this kind of places.

The three neighborhoods in cluster 1 are extremely similar to each other, venues share the same ranking, that’s the power of clustering algorithms, they find the hidden patterns easily, even the *K-Means* algorithm found the same results (with 6 centers, cluster 4 have a small number of neighborhoods within with a score equal to 1).

These neighborhoods seem to be much quitter place than the ones of the first cluster, from the map plot, they reside outside the city (points in purple).

Rather than classifying the rest of the neighborhoods incorrectly, it is a better approach to identify them as noises and then try to understand for what reasons. From Figure 15, we see that they share some venues with both previous clusters but with a different ranking and a less number, some restaurants, shops, studios.

5.2 Recommendations

To make the findings of this project more exploitable, I recommend:

- Extract the neighborhoods labeled as noise and run further analysis and identify similarities between them.
- Build a little application that facilitate to a user to find results about a specified neighborhood.

6 Conclusion

In this project, I tried to study the city of *Riyadh*, trying to understand the reasons behind its popularity and to provide insights about its neighborhoods for tourists, habitants and investors.

Starting with collecting data about municipalities, neighborhoods and postal codes. Then acquiring latitude and longitude of each neighborhood in the city, that facilitates the task of collection information about venues using the *Foursquare API*.

Initial analysis shows remarkable differences between municipalities and neighborhoods in term of total venues.

Further analysis with machine learning algorithm *DBSCAN*, helped to identify the most populated and visited neighborhoods that contains venues like restaurants, coffee places, hotels etc. They represent over 70% of the neighborhoods in the city, and that confirms the fact that the city is the main financial hub of the country and the most populated.

I was also able to identify calm neighborhoods outside the city with campgrounds farms and studios.

Finally, I was able to identify which neighborhoods are unique (i.e. not similar to the others) that require further analysis.

References

1. **Wikipedia. *wikipedia.* https://en.wikipedia.org/wiki/Riyadh.**

2. Programmable Web. https://www.programmableweb.com/api/foursquare.

3. Yellowbrick elbow method. https://www.scikit-yb.org/en/latest/api/cluster/elbow.html#:~:text=The%20elbow%20method%20runs%20k,point%20to%20its%20assigned%20center..

4. About Yellowbrick. https://www.scikit-yb.org/en/latest/about.html.

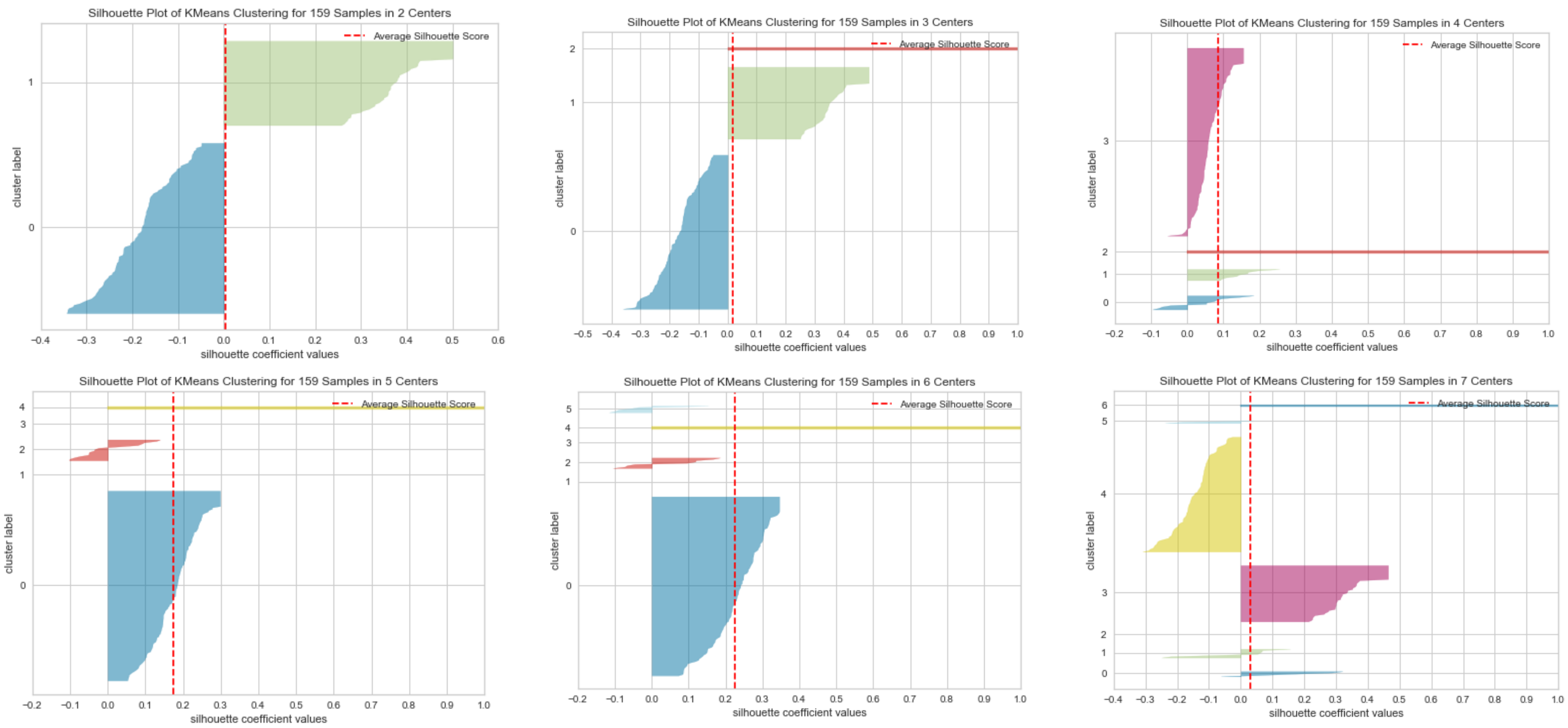
5. Yellowbrick silhouette Visualizer. https://www.scikit-yb.org/en/latest/api/cluster/silhouette.html.

6. DBSCAN, towardsdatascience. https://towardsdatascience.com/machine-learning-clustering-dbscan-determine-the-optimal-value-for-epsilon-eps-python-example-3100091cfbc.

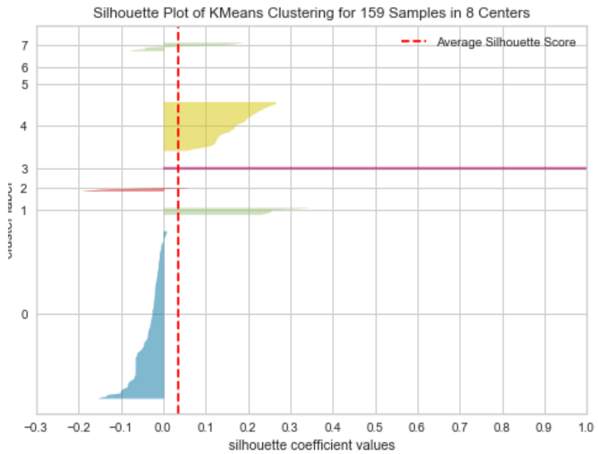
7. DBSCAN, sklearn. https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html.

Appendices

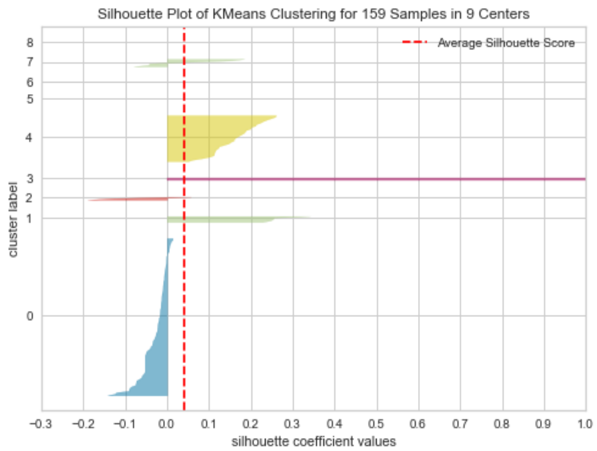
Appendix 1



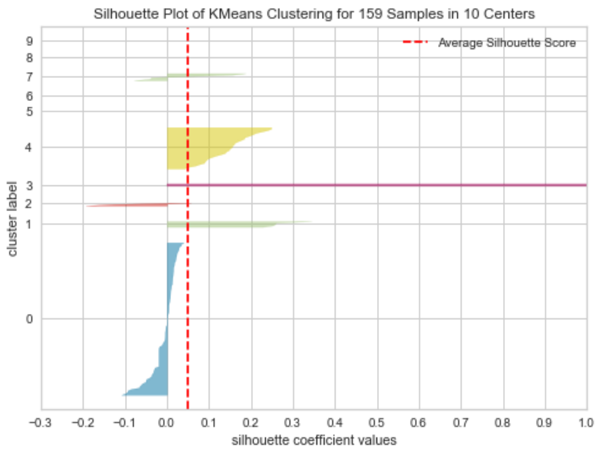
Appendices



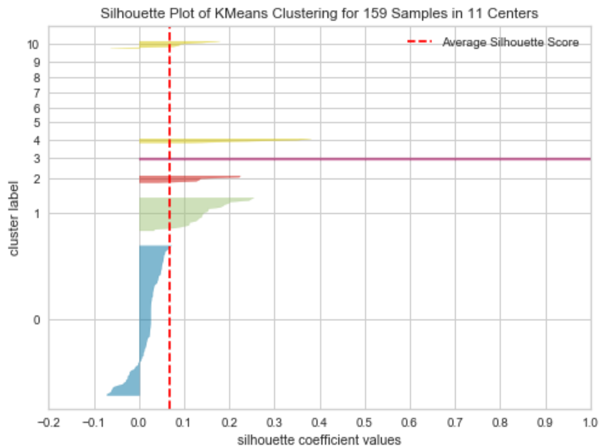
Silhouette Plot of KMeans Clustering for 159 Samples in 8 Centers
Average Silhouette Score
silhouette coefficient values



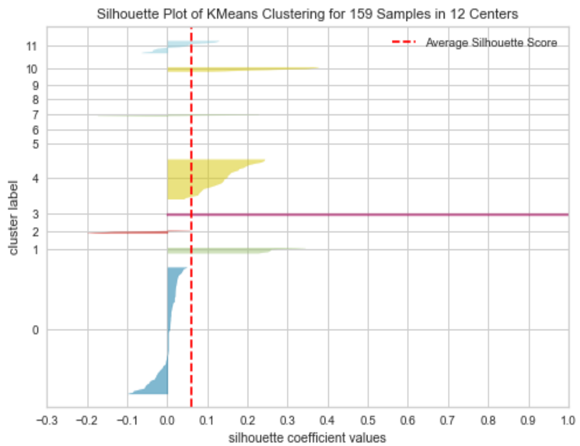
Silhouette Plot of KMeans Clustering for 159 Samples in 9 Centers
Average Silhouette Score
cluster label
silhouette coefficient values



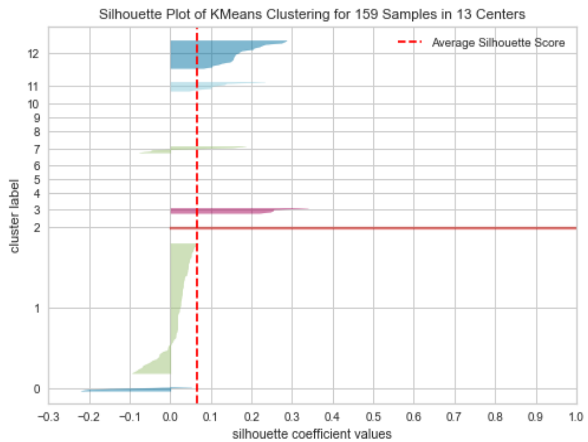
Silhouette Plot of KMeans Clustering for 159 Samples in 10 Centers
Average Silhouette Score
cluster label
silhouette coefficient values



Silhouette Plot of KMeans Clustering for 159 Samples in 11 Centers
Average Silhouette Score
cluster label
silhouette coefficient values



Silhouette Plot of KMeans Clustering for 159 Samples in 12 Centers
Average Silhouette Score
cluster label
silhouette coefficient values



Silhouette Plot of KMeans Clustering for 159 Samples in 13 Centers
Average Silhouette Score
cluster label
silhouette coefficient values