

Middle East Technical University NCC

CNG 514 – Data Mining Spring 2023

Assignment 3

Student: Due Date:

Hevra Petekkaya May 25th, 2023

2456200

# Table of Figures

[Figure 1: Fetching data from txt file 3](#_Toc135922266)

[Figure 2: Preparation for Apriori 3](#_Toc135922267)

[Figure 3: Generation of frequent item sets using Apriori 4](#_Toc135922268)

[Figure 4: Itemsets with support of 1 5](#_Toc135922269)

[Figure 5: Frequent 1-itemset with support of 1 5](#_Toc135922270)

[Figure 6: Maximum Frequent Itemsets 6](#_Toc135922271)

[Figure 7: Effect of changing min\_sup 6](#_Toc135922272)

[Figure 8: Data preparation 8](#_Toc135922273)

[Figure 9: Combining fixations & filtering the location features. 8](#_Toc135922274)

[Figure 10: DBSCAN clustering algorithm on location data of fixations 8](#_Toc135922275)

[Figure 11: Clustering being plotted based on location information. Hyperparameters are Min\_samples = 10 & eps = 20 9](#_Toc135922276)

[Figure 12: Cluster info based on Min\_samples = 10 & eps = 20 9](#_Toc135922277)

[Figure 13: Grid Search DBSCAN 10](#_Toc135922278)

[Figure 14: DBSCAN for 22 clusters 10](#_Toc135922279)

[Figure 15: Plotting for 22 clusters. 10](#_Toc135922280)

[Figure 16: DBSCAN for 5 clusters 11](#_Toc135922281)

[Figure 17: Plotting for 5 clusters. 11](#_Toc135922282)

# Part 1: Association Rule Mining

In part 1, we are required to find the frequent item sets given the dataset that is a sequency of fixations for 36 people. The file is made up of a list of lists, where each list has the fixations of a single person. Figure 1 shows the process of reading the file and creating the users\_fixations list.

Since we do not need any “time” information for this task, we can safely discard that part of the data and only take the ID of the fixations into account. This can be considered preprocessing the data because we do not want to carry unnecessary data with us that would lead to inefficiency and increased complexity.

A screenshot of a computer

Description automatically generated

Figure : Fetching data from txt file

Now that the data is ready, we can start the process of finding frequent item sets. The fit\_transform method of the TransactionEncoder class is responsible for converting transaction data into a suitable format for the Apriori Algorithm as can be seen in Figure 2. It takes a list of transactions, where each transaction is represented as a list of items and encodes them into a binary matrix. The matrix has a column for each unique item in the dataset, which is the ID of fixations, and each row represents a transaction. If an item is present in a transaction, its corresponding cell value is set to 1; otherwise, it is set to 0.

A screenshot of a computer

Description automatically generated

Figure : Preparation for Apriori

It is given that the min support is 75%, meaning that a minimum of 75% of the user fixations need to include the given itemset, which is a set of fixation IDs, in their bucket for the given itemset to be considered frequent. Otherwise, if less than 75% of the user fixations include the itemset, then it won’t be considered as frequent. Note that even though we are also provided with the value of the minimum confidence, this input does not affect the generated frequent item sets in any way; Apriori does not need minimum confidence as an input.

Minimum confidence value is needed to assess the strength of association rules that can be derived from the frequent item sets; however, finding association rules is not required for this part.

A screenshot of a computer

Description automatically generated

Figure : Generation of frequent item sets using Apriori

As can be seen in the code provided in Figure 3, we were able to generate the frequent item sets, which count to 1406. Now, let’s analyze them. It is important to remember what support value represents; it is calculated by dividing the number of user fixations in which the specific set of fixations occurs by the total number of user fixations, which is 36 for our dataset. Hence, support of value 1 indicates that the given set of fixations appear in every user fixation, meaning that all the users did look at the given set of fixation points. There are 31 set of fixations with support of 1 as can be seen in Figure 4.

A screenshot of a computer

Description automatically generated A screenshot of a computer

Description automatically generated

Figure : Itemsets with support of 1

A screenshot of a computer

Description automatically generated

Figure : Frequent 1-itemset with support of 1

As can be seen from Figure 5, fixations with only 1 itemset that have a support of 1 are {M}, {P}, {Q}, {R}, and {S}. When we look at the layout of the webpage, this result does make sense because {M}, {P}, {Q}, {R}, and {S} are the fixations that appear on the center of the webpage. Regarding fixation with ID of M, there is a sudden change of color from blue to green, hence, it is expected to be eye catching.

In order to find the maximum frequent itemset, we can have a look at Figure 6. Usually, the maximum frequent itemset refers to the largest itemset in terms of size that satisfies the minimum support threshold in a given dataset, which is {‘P’, ‘H’, ‘L’, ‘N’, ‘M’, ‘Q’, ‘K’, ‘S’, ‘R’, ‘I’}. However, we can also find the maximum frequent itemset with an additional condition of having support of 1 and that is {‘P’, ‘M’, ‘Q’, ‘S’, ‘R’}.

A screenshot of a computer

Description automatically generated

Figure : Maximum Frequent Itemsets

Given that {‘P’, ‘M’, ‘Q’, ‘S’, ‘R’} is located at the center, has support of 1, and it the largest possible frequent itemset that satisfies these conditions, one can realize how important the center of a webpage is. Hence, for example, if a company that has a website is willing to advertise other companies/businesses on their website, they can charge them according to where they put the advertisement. To elaborate more, they should probably charge the most for the companies that are willing to be advertised with ads that appear in the center of a webpage because that is where most if not all the users stare or look at and hence will be exposed to the ad more. Consequently, the fees for putting on advertisements should not only be dependent on how much space they are taking in the webpage but also and more importantly where in the webpage they are located it; it is a matter of balancing quantity and quality.

A screenshot of a computer

Description automatically generated

Figure : Effect of changing min\_sup

One last question that we can investigate is what happens to the frequent itemsets when we change the value of minimum support and minimum confidence. As previously mentioned, changing the minimum confidence will have no affect on the frequent itemsets that were generated by Apriori, because minimum confidence plays a role during the generation of association rules and not the frequent itemsets. On the other hand, changing the value of minimum support plays a critical role in the generation of frequent itemsets. Reducing the value of the minimum support will increase the number of frequent itemsets; whereas, increasing the number of minimum support will decrease the number of frequent itemset as can be seen in Figure 7.

# Part 2: Clustering

As required for this assignment, I will be using DBSCAN to cluster the fixations points together. DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a density-based clustering algorithm commonly used in machine learning and data mining. It is designed to discover clusters of data points based on their density distribution in the feature space. It has 2 hyper-parameters: Eps (Maximum radius of the neighborhood) and Minpts (Minimum number of points in an Eps-neighborhood of that point).

As usual, before performing any machine learning algorithm, one must prepare, and preprocess data as needed. For the given dataset, some files that are supposed to hold the fixation data of a users are empty. This could have several reasons. For example, the eye-tracker software might have been non-functional during the scanning process of a user. Hence, these files should be cleaned out. Additionally, some attributes that are not needed can also be discarded. For example, the stimuli name can be discarded because it is the same for all users since they are looking at the same webpage. This attribute was probably there because user fixations were tested on several webpages; however, the dataset provided to us only has the user fixation data for one webpage. Since its value is constant and not changing, we can’t arrive to any conclusion that is based on stimuli.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Figure : Data preparation

Now, let’s do some feature engineering. We can perform clustering on different combinations of features. In order to warm up and get a feel of how DBSCAN performs on eye-tracking data, we can start with a very basic combination features, namely, the X and Y coordinate of the fixation, which makes up only the location information as can be seen in Figure 9.

A screenshot of a computer

Description automatically generated

Figure : Combining fixations & filtering the location features.

As can be seen in Figure 10, an instance of the DBSCAN clustering algorithm is initiated and fed with the features we chose for now that are location data of fixations.

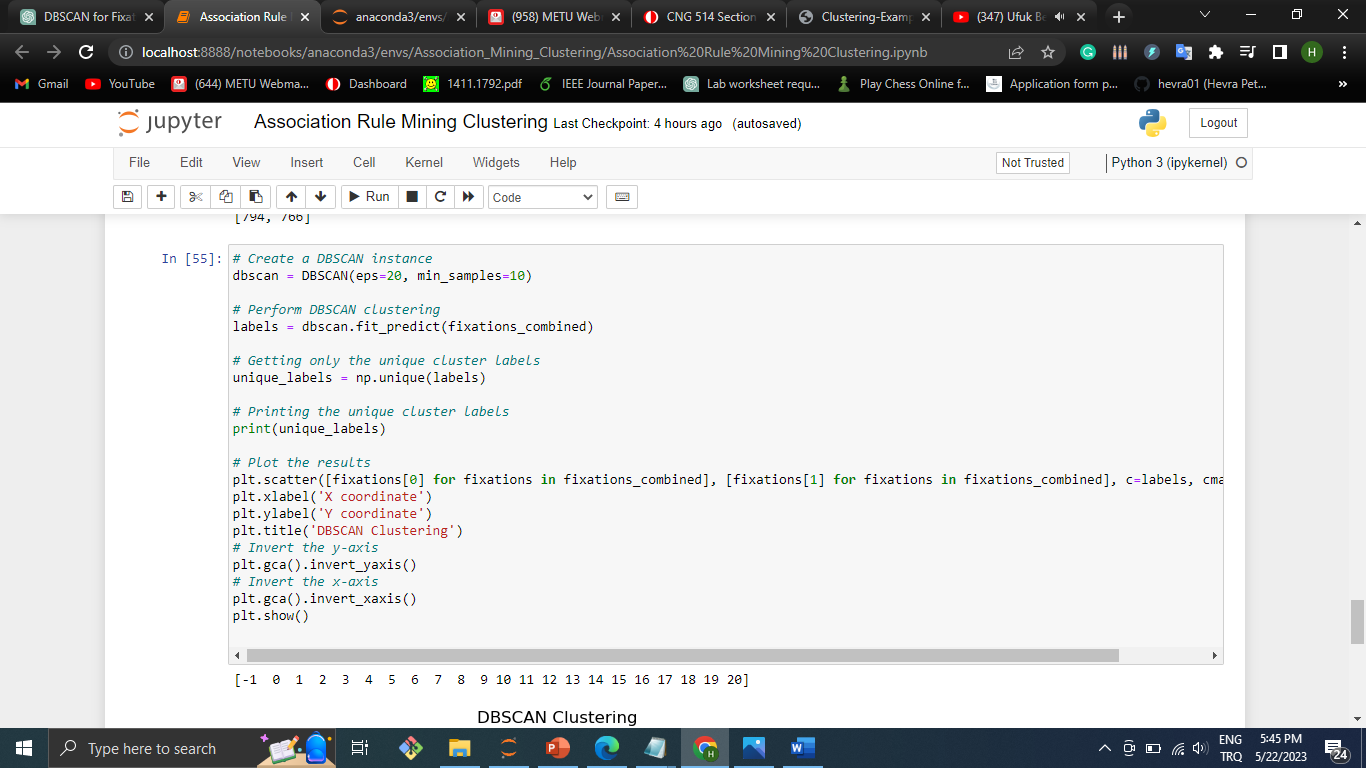


Figure : DBSCAN clustering algorithm on location data of fixations

As can be seen in Figure 11, when only the location data is used, we get a rough representation of the layout of the webpage where clusters are formed at the fixations specified in BABYLON\_segmented.png file. For example, we can see P, Q, R, and S fixations as clusters appearing above y=600. Moreover, we can see fixation, M, that has the “download” text on appearing somewhere between 200 < y < 400. Lastly, keep in mind that there are 2 hyperparameters, min\_samples and eps, that play a critical role in the generation of the clusters.

A screen shot of a computer screen

Description automatically generated with medium confidence

Figure : Clustering being plotted based on location information. Hyperparameters are Min\_samples = 10 & eps = 20

Based on Figure 12, we have 2604 fixations that don’t belong to any cluster, hence classified as outliers. Additionally, it can be seen that there are 21 clusters formed.

A screenshot of a computer

Description automatically generated

Figure : Cluster info based on Min\_samples = 10 & eps = 20

In total there are 3622 fixations and around 72% of them are labelled as outliers, not belonging to any cluster, which is a significant amount. Additionally, only 21 clusters have been found but, the Babylon page has 22 clusters. Hence, the DBSCAN has found 1 less cluster than the reality.

Now, let’s perform grid search to optimize for the best values of hyperparameters as can be seen in Figure 13. We pass a possible list of values that the two hyperparameters can have, the fixations combined, and the target number of clusters we wish to find. Note that we do not always know how many clusters we expect to find. In other words, we sometimes use clustering algorithms to find and explore clusters. Nevertheless, for this assignment, I assume that there is a specific number of clusters that we want to find and categorize fixations into. The number of clusters that we want to find can change but there is always a specific number that we want to find. I made this assumption because in the assignment we were told that the users were asked to follow a specific sequence of segments, namely, M, P, Q, R, and S and their fixations were recorded accordingly.

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

Figure : Grid Search DBSCAN

A screenshot of a computer

Description automatically generated

Figure : DBSCAN for 22 clusters

A screenshot of a computer screen

Description automatically generated with medium confidence

Figure : Plotting for 22 clusters.

It is important to realize that the best min\_samples increased from 10 to 20 and best eps reduced from 20 to 10 when the number of clusters changed from 22 to 5. The lesser the number of clusters we want to find the higher the value of min\_samples because it should be harder to form clusters when you are limited to 5 clusters. Moreover, the lower the eps value the lesser clusters we obtain because again it becomes harder to form clusters. Consequently, when the number of clusters is 5, we get clusters specifically where the users were asked to look at M, P, Q, and S because these regions have the highest number of fixations: the densest ones. However, we missed the point R.

A screenshot of a computer

Description automatically generated

Figure : DBSCAN for 5 clusters

A screenshot of a computer screen

Description automatically generated with medium confidence

Figure : Plotting for 5 clusters.

Coming to the question whether it is possible to identify the segments users looked at using DBSCAN or not, it depends on our expectations. For example, just because the algorithm missed segment R, in my opinion that does not make the technique inappropriate to be used. That is because overall it was able to identify the segments. The reason for missing segment R could be due to human behavior. In other words, the users might have been impatient or slightly bored after they were done looking at segment Q and wanted to quickly end the experiment and hence, looked at segment S without spending enough time at segment R.

Hence, instead of finding the algorithm/technique inappropriate or inadequate, one can rather see it as an opportunity to understand human behavior and organize webpages accordingly. In other words, the algorithm’s inability to identify the segment R, could mean that even though the center of a webpage is a critical region, a webpage designer shouldn’t put a lot of important content together one after the other. For example, there are some blank spaces to the left of segment O and to the right of segment T, hence, expanding the space between P, Q, R, and S, can be a good idea to avoid stacking them very close to each other.

To further explain the problem, even though the users might have been asked to look at M, P, Q, R, and S, they might have not done it properly. Hence, not enough fixations were formed to create a cluster/segment. Consequently, the algorithm behaved as it should have behaved (it formed clusters in dense regions) but the source that data is being generated can be prone to noise and unpredictability.