Gebze Technical University
Computer Engineering

CSE 454 – 2021 Spring
PROJECT REPORT

Dilara KARAKAŞ

171044010

Contents

Chapter 1

Data Understanding

The dataset includes 471910 transactions performed by some customers in a supermarket. Each transaction is defined by a set of 8 attributes (initially) that provide information about both the customer and related products. All statistics and evaluations were made before the data cleaning phase.

1.1 Data semantics

This section contains a brief description of each attribute that appears in the original dataset.

Attribute	Type	Description
BasketID	Numerical	Unique alphanumeric ID assigned to a *basket*, defined as a set of transactions.
BasketDate	Numerical	The date on which the transaction was made.
Sale	Numerical	Cost of one unit of product.
CustomerID	Numerical	The ID of the customer.
CustomerCountry	Categorical	The Customer's country of origin.
ProdID	Numerical	The ID of the product.
ProdDescr	Categorical	A description of the product.
Qta	Numerical	The number of units bought in the transaction.

1.2 Data distributions and statistics

```
[ ] dataset_frame.isnull().sum(axis = 0)

BasketID              0
BasketDate            0
Sale                  0
CustomerID        65080
CustomerCountry       0
ProdID                0
ProdDescr           753
Qta                   0
TotalSale             0
dtype: int64
```

Since I needed to define a customer's profile, I modified the original dataset by removing all records with null values in the CustomerID attribute: they cannot contribute to our goal and also helps reduce the size of the dataset (65080 transactions without the CustomerID). Further testing revealed that 753 processes also had a missing ProdDescr attribute, but they were all included in the previous ones. Cleaning took us to a total of 406830 records.
I then analyzed the dataset and saw the presence of many records (ie the presence of empty Sales and negative Qtas) that would not help achieve the main objective of this study. The former does not benefit our work because "zero sales" is not economically relevant to describe a customer's profile. Instead, when the Qta quantity of products sold is negative, it can mean many things: I assumed it was a returned product (for many possible reasons, such as broken, defective, etc.) Items that are in no way related to how a customer behaves. This led to the removal of 1279 records in the first case and 9752 records in the second case.

1.2.1 Time-related

The stored records range from the Minimum date: 2010-12-01 08:26:00 to the Maximum date: 2011-12-09 12:50:00, so just a bit longer than a whole year. As shown in Figure 1.1, the sales are kind of constant throughout the year and increased in the proximity of Christmas. In fact, the day with more sales was 2011-12-05 and the day with fewer sales was 2010-12-22. A simple way to justify this would be that the store started its activity that first Christmas and steadily increased its clientele from time to time.

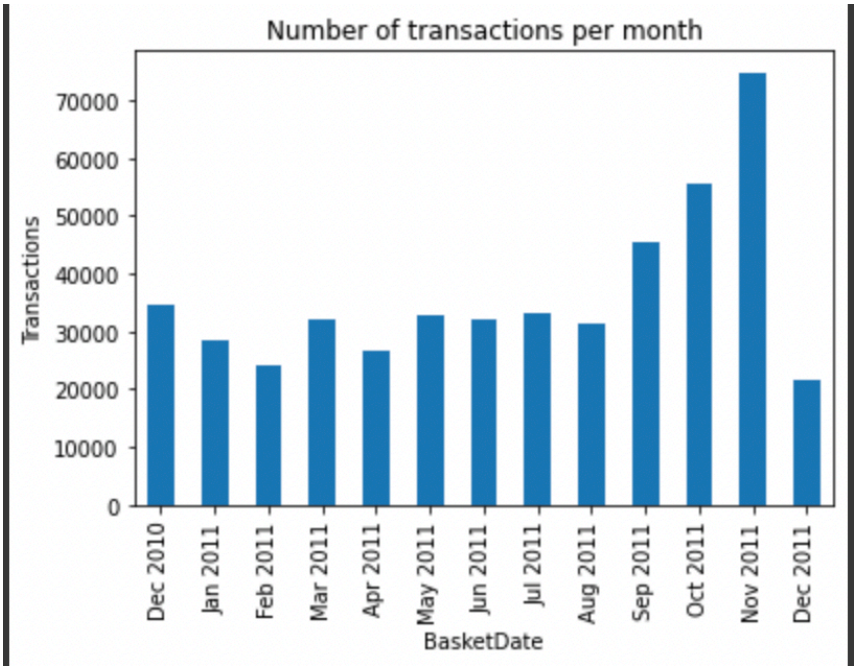


Figure 1.1

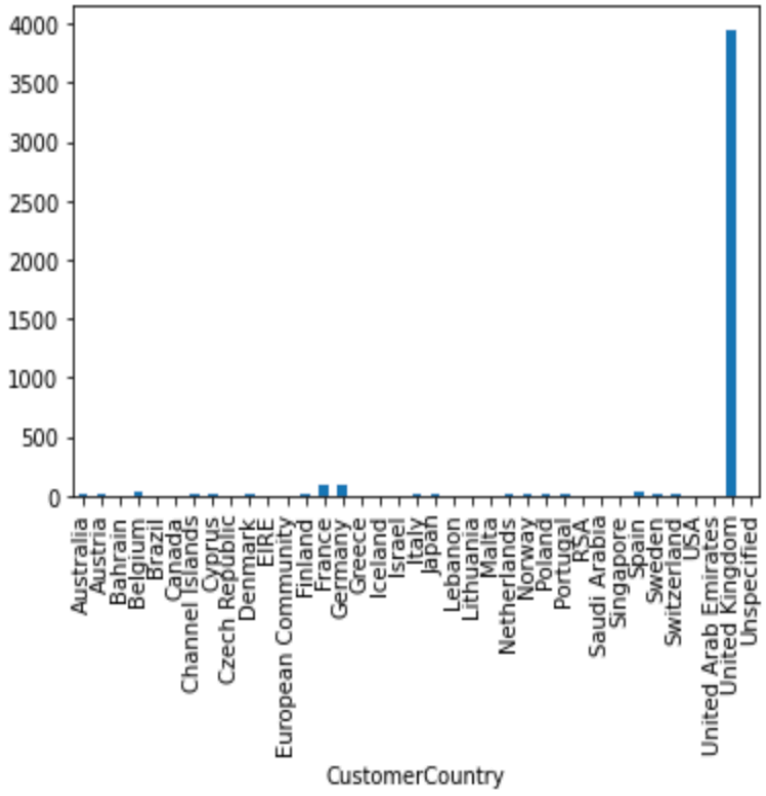


Figure 1.2

1.2.2 Geolocalization-related

As shown in Figure 1.2, most of the customers come from the United Kingdom. This suggests that the store is probably located there, but no customer-side information is relevant here and this attribute will be no longer considered for this reason.

1.2.3 Products-related

There are a total of 3684 different products that were sold. Simple analyses are:

- **Best seller:** The best-selling item. ID: 85123A, WHITE HANGING HEART T-LIGHT HOLDER.
- **Worst seller:** The worst-selling item. ID: 22146 EASTER CRAFT IVY WREATH WITH CHICK.

```
prodID_counts = dataset_frame['ProdID'].value_counts()
print(prodID_counts.describe())

best_seller_id = prodID_counts.index[0]
best_seller_descr = dataset_frame.loc[dataset_frame['ProdID'] == best_seller_id]['ProdDescr'].iloc[0]
print("Best seller: {} ({})".format(str(best_seller_id), str(best_seller_descr)))

worst_seller_id = prodID_counts.index[-1]
worst_seller_descr = dataset_frame.loc[dataset_frame['ProdID'] == worst_seller_id]['ProdDescr'].iloc[0]
print("Worst seller: {} ({})".format(str(worst_seller_id), str(worst_seller_descr)))
```

```
count    3684.000000
mean      110.431596
std       167.529851
min         1.000000
25%        12.000000
50%        49.000000
75%       139.000000
max      2077.000000
Name: ProdID, dtype: float64
Best seller: 85123A (WHITE HANGING HEART T-LIGHT HOLDER)
Worst seller: 22146 (EASTER CRAFT IVY WREATH WITH CHICK)
```

Figure 1.3

1.2.4 Extra attributes

We also added a new numerical attribute to help us make some studies:

Total Sale	Numerical	The amount of the transaction, calculated by multiplying the Sale by the Qta.

1.3 New features

To describe in a better way the customer profile and also improve data quality, I extracted some features for each customer, creating a new dataset indexed by their CustomersIDs.

Attribute	Type	Notes
ObservationItems	Numerical	The total number of items purchased by a customer during the period of obser vation.
ObservationDistinctItems	Numerical	The number of distinct items bought by a customer in the period of observation.
ObservationMaxItem	Numerical	The maximum number of items purchased by a customer during a shopping session.
BasketNum	Numerical	The number of distinct baskets for each customer.
SumExp	Numerical	The total expenditure for each customer.
AvgExp	Numerical	The entire expenditure for each customer divided by the number of baskets.

```
[ ] dataset_frame_customer.describe()
```

	ObservationItems	ObservationDistinctItems	ObservationMaxItem	BasketNum	SumExp	AvgExp
count	4372.000000	4372.000000	4372.000000	4372.000000	4372.000000	4372.000000
mean	93.053522	61.211116	31.867795	5.075480	1898.463818	315.884437
std	232.471568	85.425119	31.225805	9.338754	8219.344627	361.237024
min	1.000000	1.000000	1.000000	1.000000	-4287.630000	-4287.630000
25%	17.000000	15.000000	12.000000	1.000000	293.362500	151.991250
50%	42.000000	35.000000	23.000000	3.000000	648.075000	236.987500
75%	102.000000	77.000000	41.000000	5.000000	1611.725000	370.816071
max	7983.000000	1794.000000	542.000000	248.000000	279489.020000	6207.670000

Figure 1.4

1.4 Outliers

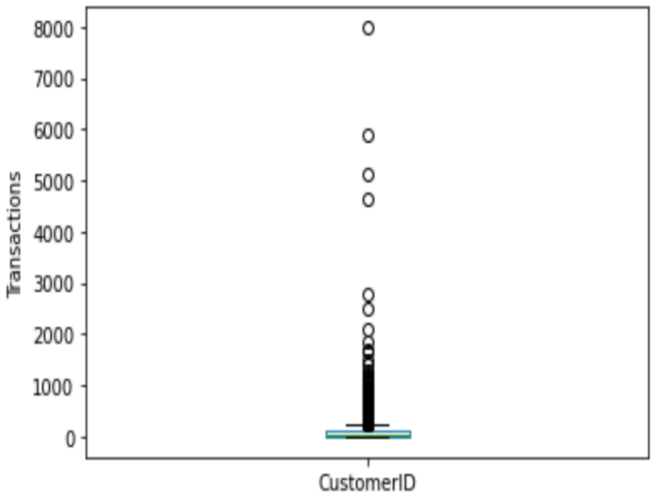


Figure 1.5

The final step of the data quality assessment was the detection of outliers for all new features only).

For analysis and removal of outliers, I decided to use the Z-Score metric, an important metric that tells how many Standard Deviations are above or below a number from the mean of the dataset.

According to the empirical rule, I considered all data with an absolute Z-score value above 3 as outliers and removed them.

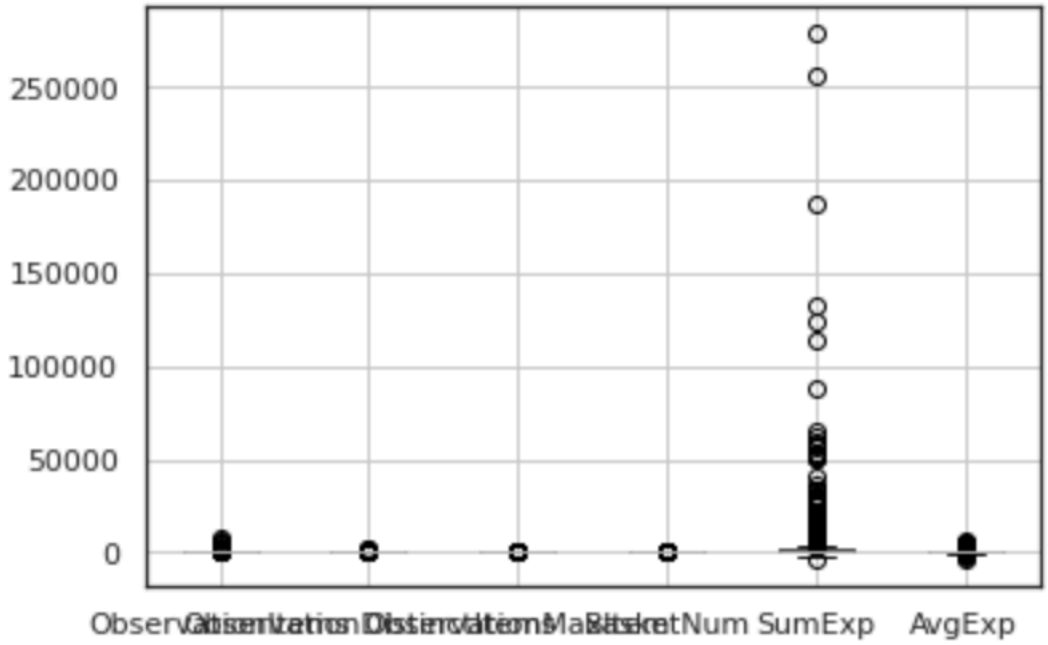


Figure 1.6

Once outliers were removed, the total number of customers has dropped down from **4372** to **4158** (**214** "outliers" customers).

1.5 Correlation

As shown in Figure 1.7, the correlation in the original dataset is not high in most of the pairs considered. Exceptions are:

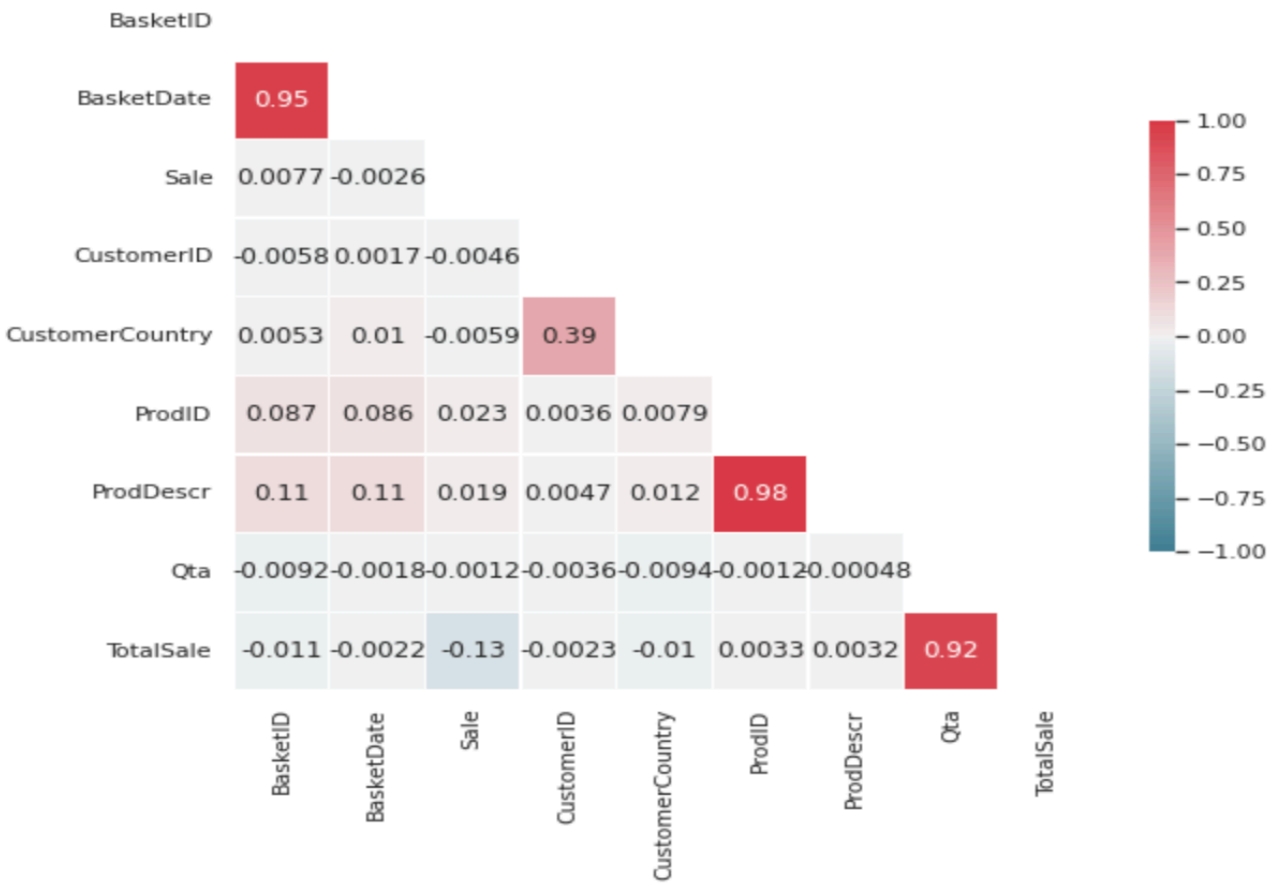


Figure 1.7 Original Data Set

- **BasketIDs and BasketDates:** All transactions belonging to the same basket are usually made on the same date.
- **ProdID and ProdDescr:** the same item has (usually) the same description.

So, due to the high correlation score (0.98) for descriptions and items, I can safely assume that ProdDescr is a superfluous attribute and so it can be dropped in future studies.
On the other hand, our new attribute Amount has of course a very high correlation with the Qta attribute, simply because they're directly proportional.

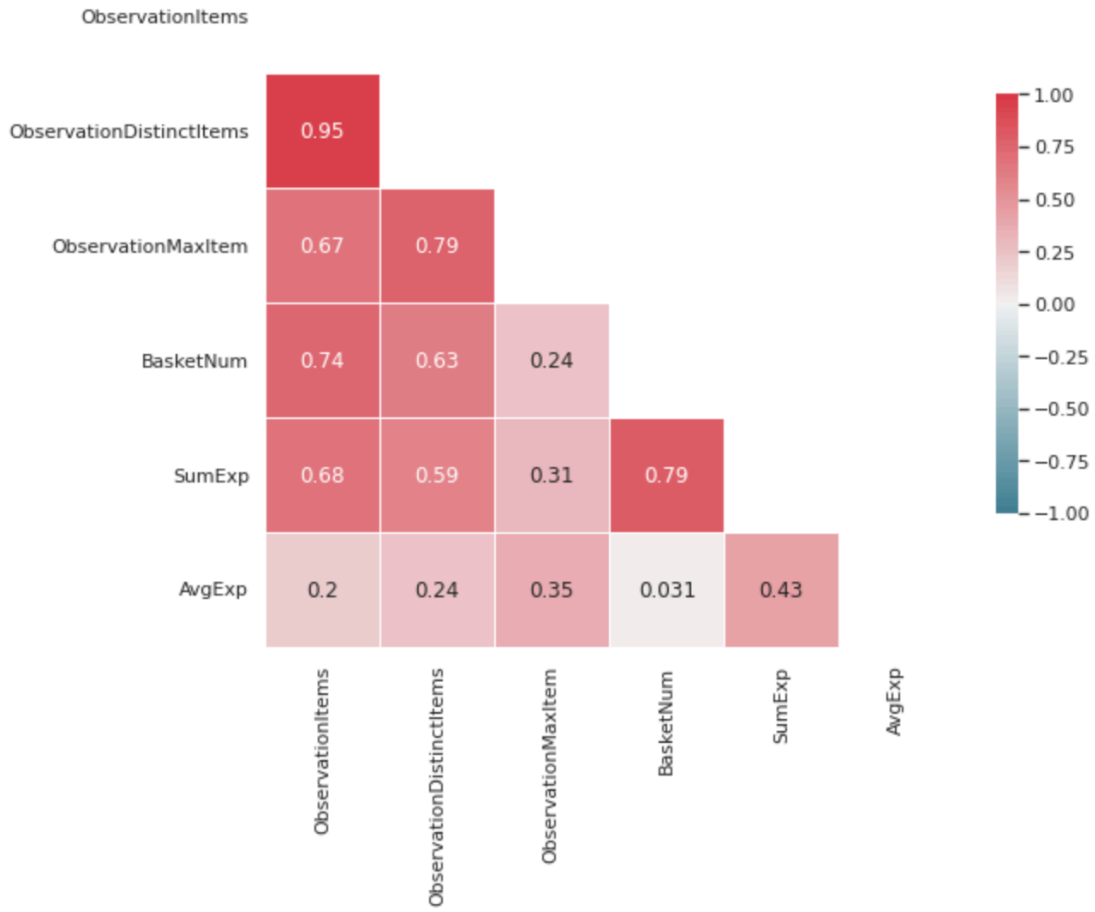


Figure 1.8 New Data Set

In I new dataset, all the numeric attributes are strictly correlated as shown in Figure 1.8. Two main examples are:

- **ObservationDistinctItems and ObservationItem**, because they both refer to bought items.
- **SumExp and BasketNum,** because the more a customer comes to the store, the more he will eventually spend.

For reference, a standard describe() has been called on the new dataframe and results can be seen in the Figure 1.9.

	ObservationItems	ObservationDistinctItems	ObservationMaxItem	BasketNum	SumExp	AvgExp
count	4158.000000	4158.000000	4158.000000	4158.000000	4158.000000	4158.000000
mean	71.440356	52.205147	28.607504	4.242665	1223.584777	281.482805
std	88.671079	53.796519	22.830264	4.682463	1793.110829	200.922765
min	1.000000	1.000000	1.000000	1.000000	-1165.300000	-611.860000
25%	17.000000	15.000000	12.000000	1.000000	279.812500	149.785000
50%	39.000000	33.000000	22.000000	3.000000	609.520000	230.555833
75%	92.000000	71.000000	39.000000	5.000000	1415.215000	355.347500
max	756.000000	316.000000	125.000000	33.000000	21086.300000	1395.795000

Figure 1.9

Chapter 2

Clustering

It's a good practice in clustering to normalize data to avoid biases. Two possible approaches are the StandardScaler (also called Z-Score) and the Min-MaxScaler. I used the last one because the obtained results were more interesting.

I were able to use my whole new dataset because every attribute is numerical and has a very high pairwise correlation, as explained previously in Section 1.5.

2.1 DBSCAN

2.1.1 Parameters and distance function

I still used the MinMax scaler and the Euclidean Distance, because it's a good metric with numeric values. DBSCAN uses these two parameters:

- **epsilon:** the maximum distance between two samples for them to be considered as in the same neighborhood.
- **min_pts:** the number of samples in a neighborhood for a point to be considered as a core point. This includes the point itself.

Since there's no automatic way to decide the value of these parameters, I did some research and found that a standard setup is to choose min samples as twice the dimensionality of the dataset (min_pts = 2*6 = 12). About epsilon, I used a technique that exploits again the Elbow method. This technique calculates the average distance between each point and its k-nearest neighbors, where k = min_pts value. The average k-distances are then plotted (as in Figure 2.1) in ascending order. I found the optimal value for epsilon at the point of maximum curvature, so at epsilon = 0.17229769941409642.

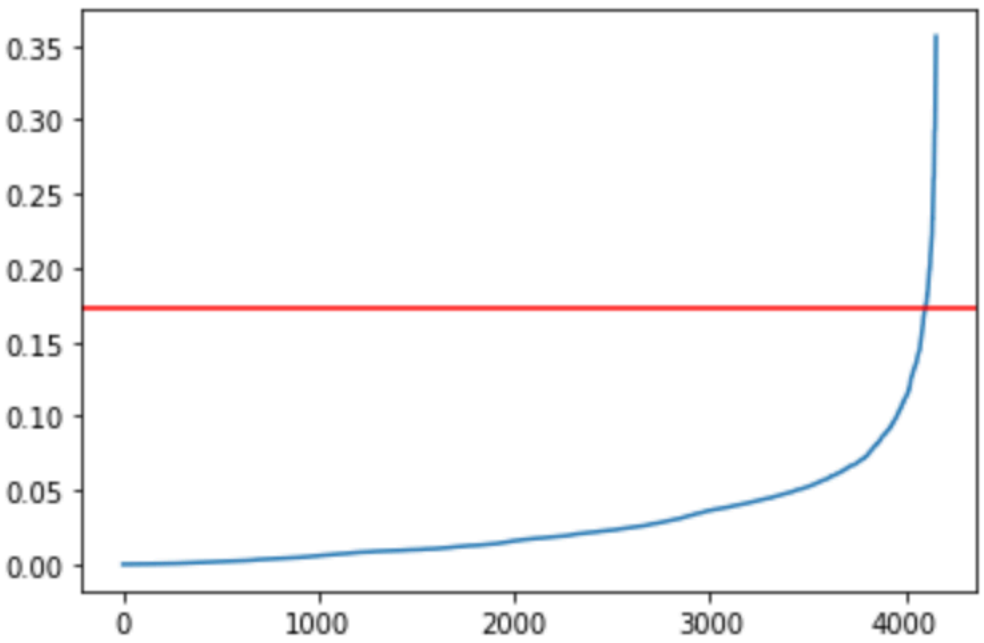


Figure 2.1

2.1.2 Cluster analysis

After executing the algorithm with the selected parameters as described earlier (Section 2.2.1), the algorithm returned two sets, one almost empty (144 elements) and the other with the rest (4021 elements). After this failure, I tried to change these parameters manually, but neither the number of clusters nor the data distribution does not change significantly. The results are shown in Figure 2.2.

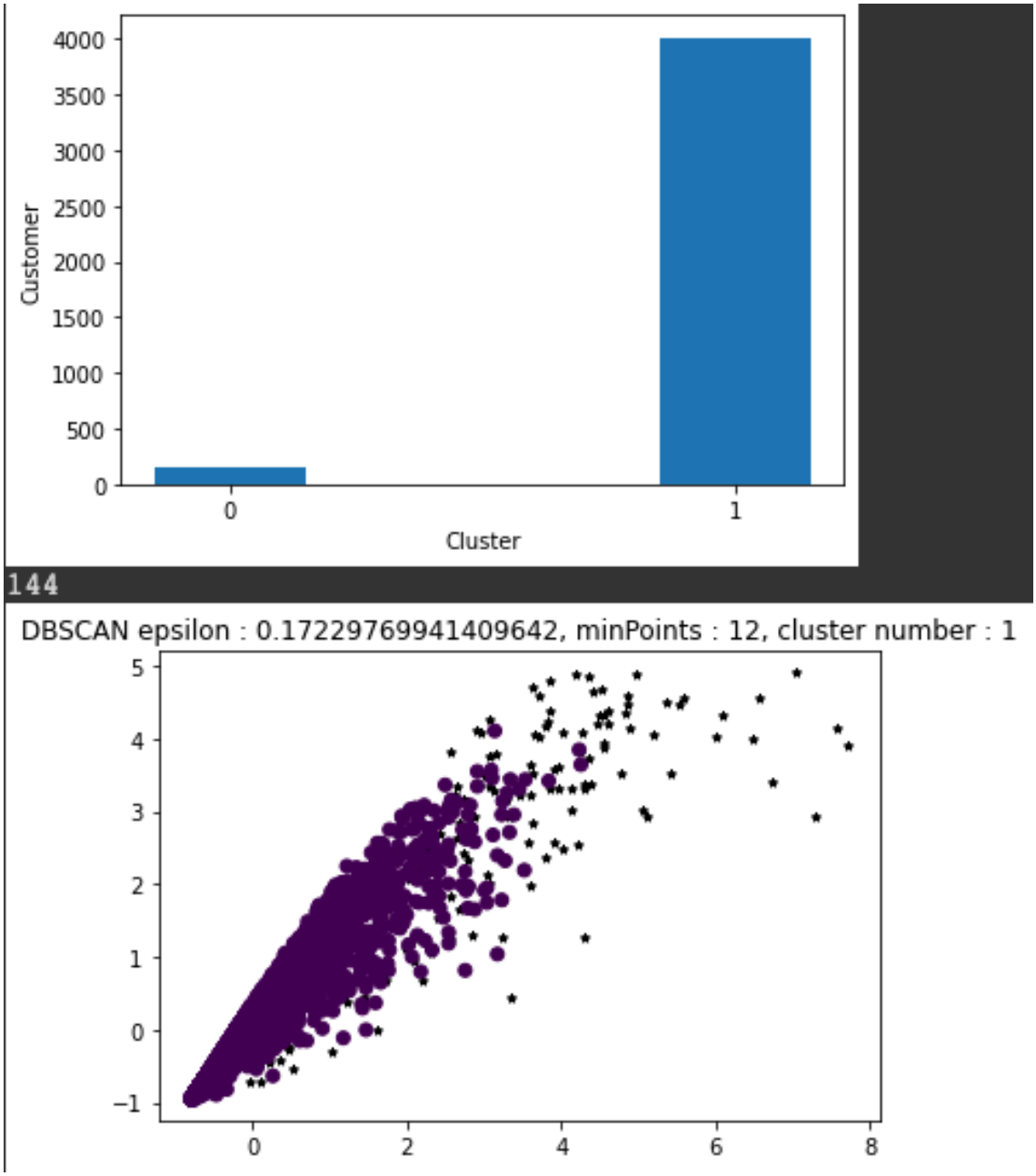


Figure 2.2

I used ready-made DBSCAN to compare the accuracy of my DBSCAN algorithm that I implemented. Figure 2.2 is the results of the ready DBSCAN algorithm.

2.1.3 DBSCAN that I implement's test result

First, I tried with the correct values that I found in the ready DBSCAN algorithm. As seen in Figure 2.3, my implementation is correct. It gave the same results.

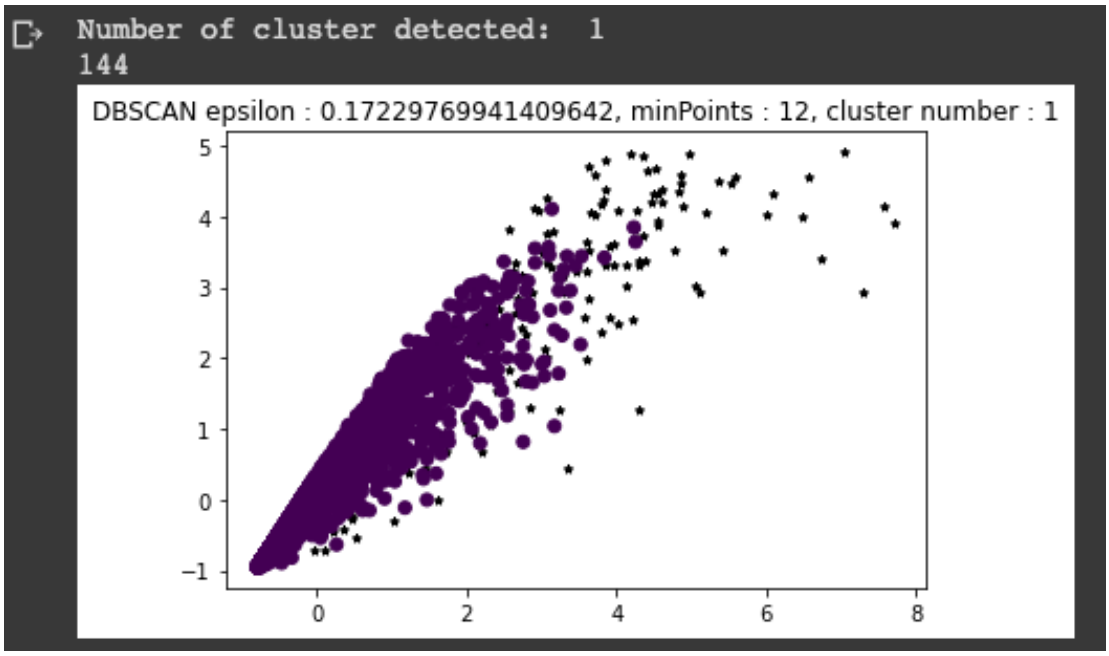
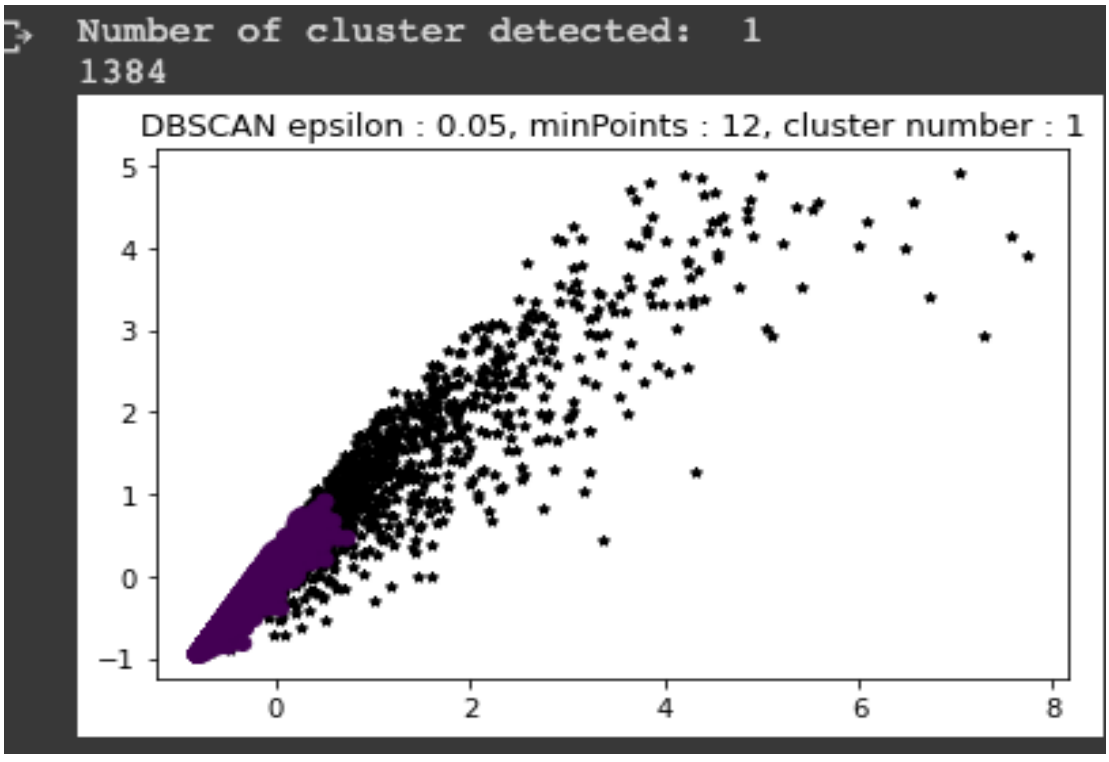


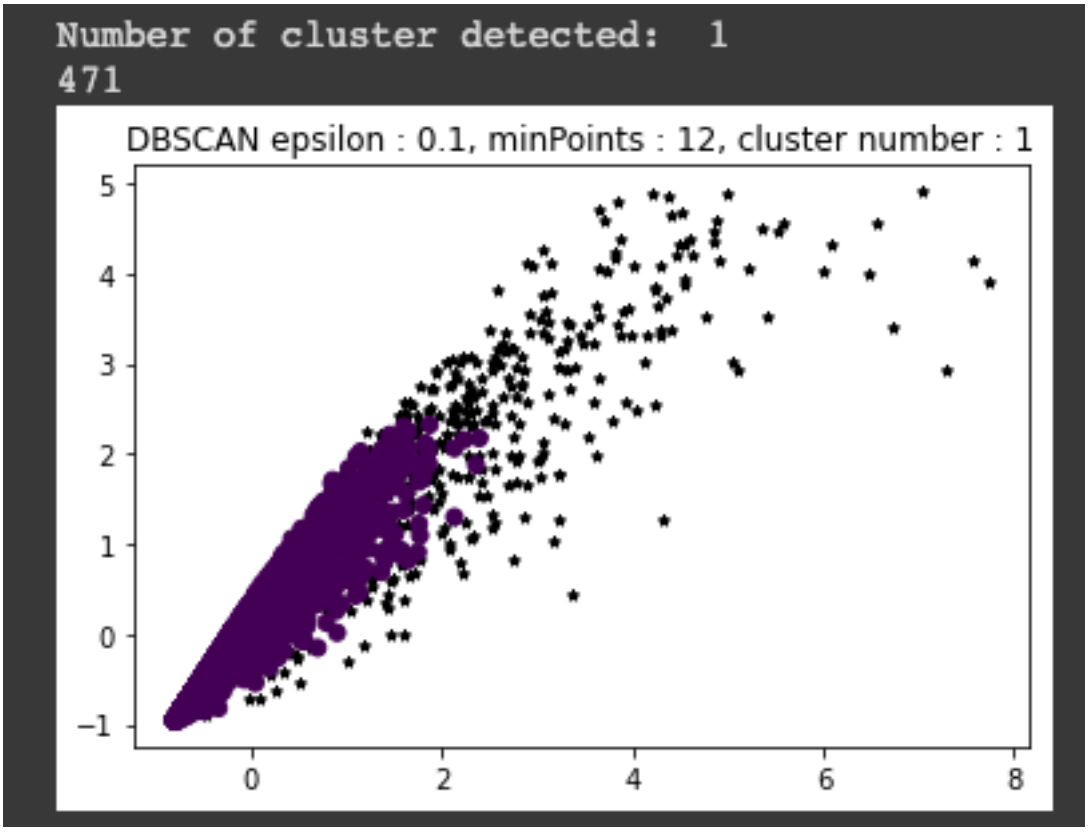
Figure 2.3

As I mentioned in Section 2.1.2, when bad results came out, I did experiments. These experiments were carried out in order to find a good result. It did not give good results in any of the experiments. Test results are below.

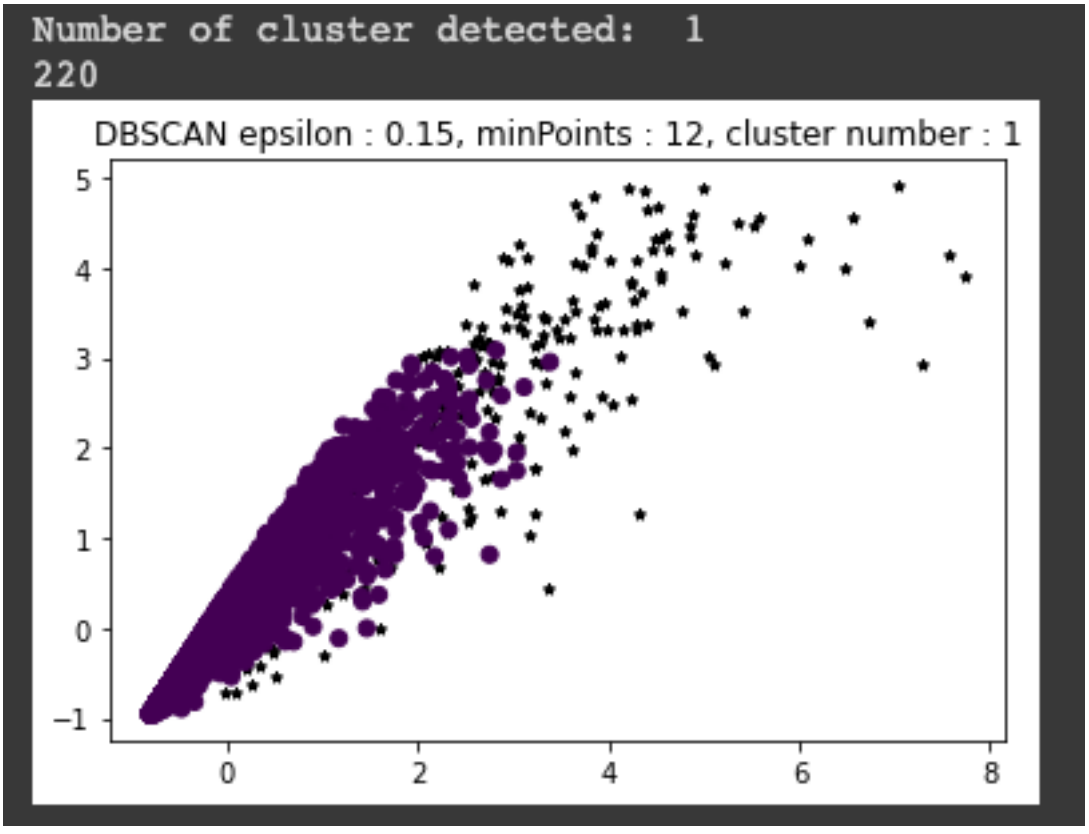
- Epsilon: 0.05



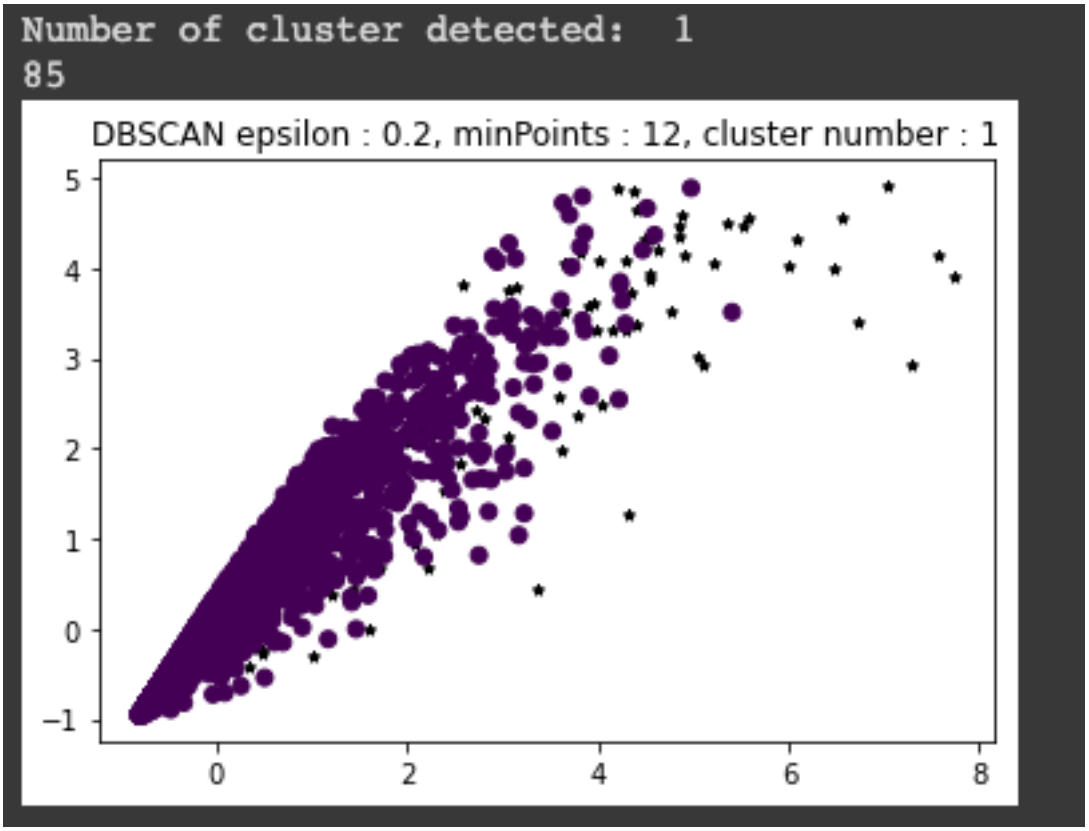
- Epsilon: 0.1



- Epsilon: 0.15



- Epsilon: 0.2



I did not change the minPoints value as seen in the experiments. On the research I did, the minPoints value was taken as dimensional*2 as the optimum value.

2.2 Hierarchical clustering

2.2.1 Parameters and distance function

Out of all the possibilities, I've chosen Single, Complete, Average, and Ward methods and I applied them to different metrics (Euclidean Distance, Cosine Similarity, and Manhattan Distance).

2.2.2 Dendrograms analysis

I've run the hierarchical clustering algorithm with all the different combinations of methods and metrics and will now list the most relevant ones for brevity.

- The **Single method** wasn't a good one because it creates a very large cluster (more than 99% of items) and spreading the rest in one-element clusters, like what happened in Figure 2.4 where there is a huge cluster of 4157 elements and three tiny clusters with 1 or 2 points.
- The **Complete method** returned very interesting results with the Euclidean Distance metric: this confirms that it's a really good metric with numerical values. The resulting dendrogram is shown in Figure 2.5.
- The **Average method** behaved almost exactly like the Single method, just retrieving a lower number of clusters, as shown in Figure 2.6.

- Finally, the **Ward method** didn't retrieve a suitable data distribution. As shown in Figure 2.7, it always found two clusters and in some cases (like the shown one) these are almost empty as well.

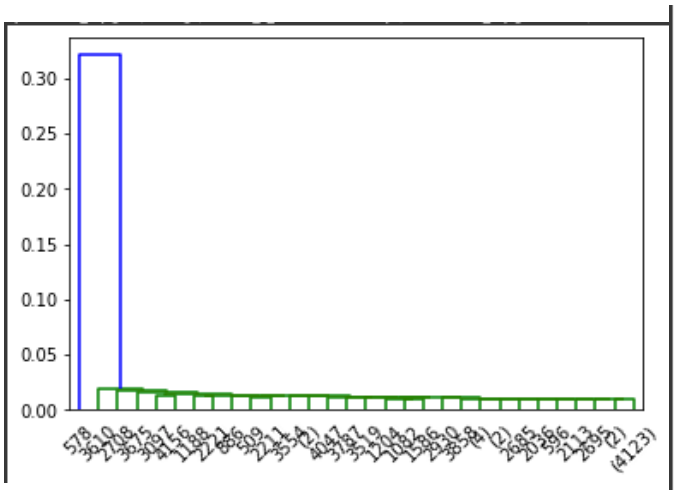


Figure 2.4

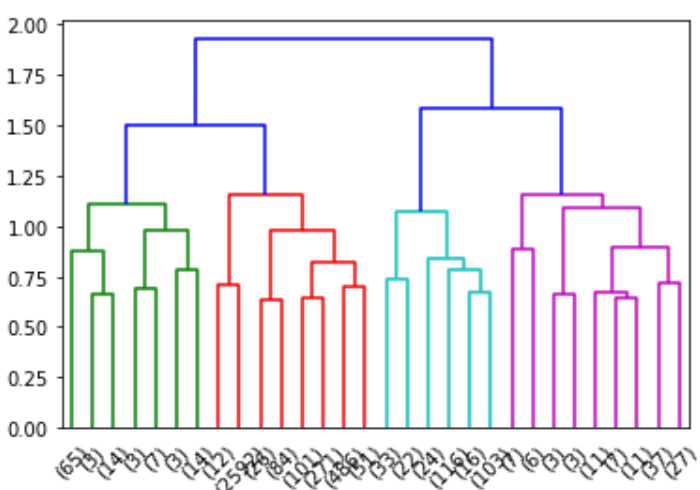


Figure 2.5

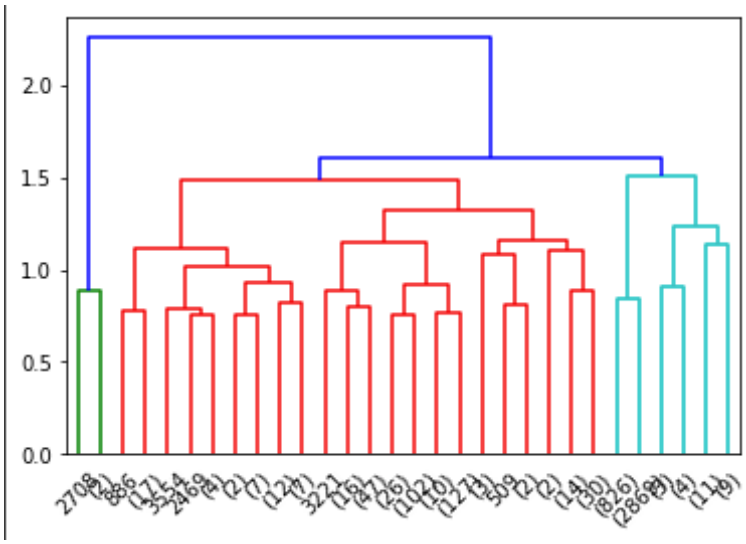


Figure 2.6

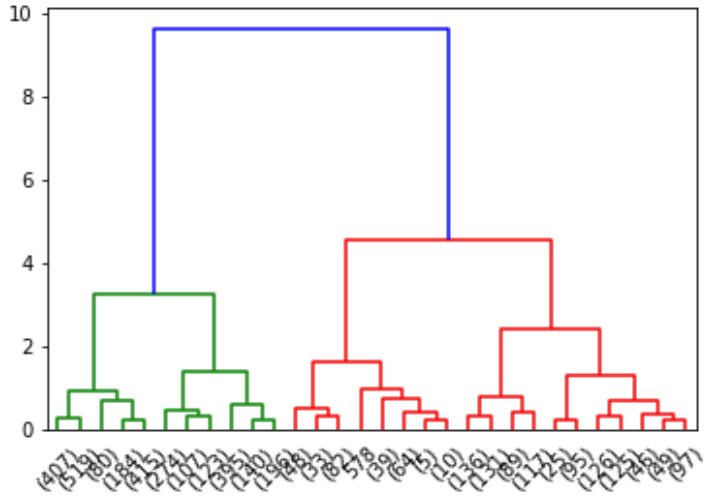


Figure 2.7

2.3 Evaluation and comparison of clustering approaches

To conclude the clustering section, I'll now compare the different clustering algorithms and the obtained clusters.

- **DBSCAN** , no matter what the values of the initial parameters, I couldn't obtain a good clustering because it always produced a huge cluster with the majority of the points and another negligible one.
- **The Hierarchical approach** produced very unbalanced clusters with some linking methods (eg Single, Ward). Instead, when using other combinations (like the Euclidean Distance Exact Method), it produced a better distribution for my data.

Chapter 3

Literature Review

Reviewed article: Dongwoo Won, Bo Mi Song, Dennis Mcleod, “An Approach to Clustering Marketing Data”

3.1 Article summary

Currently, large market basket data is collected by many organizations because this data is considered the best information for marketing analysis. Each row of data contains a transaction containing a set of items and customer information. Mining association rules are often used to identify important patterns in such a transaction database. However, association rules mining tends to generate too many rules to extract relevant and useful information because of its high. In turn, clustering association rules were introduced to reduce the overall number of rules required. Pruning, grouping, and merging are performed to create more general rules. One disadvantage of these approaches is that they still create a large number of rules. It also tends to be difficult to understand the relationship and relevance between clusters.

This paper discusses three convergent approaches for determining association rules. First, two methods were proposed: pre- and post-operative methods. Using ontologies allows them to have preliminary and final information about the item set. Depending on the field, ontologies provide a means to represent knowledge or knowledge that includes key concepts and the interrelationships between them. They applied this idea by generalizing and reducing the set of elements that ultimately produced fewer but more closely related rules. The next second approach is hierarchical subspace clustering. Subspace clustering searches for relevant attribute sets in the market basket data and localizes the search space to find clusters that allow for overlapping subspaces. They discover the most relevant attributes using clustered association rules at each level of the domain ontologies. To reduce the total number of rules, they aggregate similar association rules by clustering association rules.

As a result, this hierarchical subspace clustering approach can analyze market basket data efficiently and accurately. Finally, they propose a new set sorting function. Since most previous work has been in the field of document sorting, few attempts have been made in the past to sort clusters. In this article, they enumerate the cluster resulting from subspace clustering.

The paper also states that an attempt has been made in the field of molecular biology to sort subspaces of interest for high-dimensional data into cluster data. However, this effort is mathematically very complex and difficult to implement in a marketing application.

https://drive.google.com/drive/folders/1jPeqJkELJX9TL_BKG6VWe8EGu-Tu-PG_