Università di Pisa

Computer Science Department

Data Mining project report

A.Y. 20-21

Alessio Russo, Michele Zoncheddu

January 4, 2021

Contents

1 Introduction

The aim of the report is to describe, based on the use of data mining tools, the analysis of a given dataset, *supermarket.csv*, following the *KDD* process in order to determine the customer profile and discover frequent patterns. The analysis follows a specific sequence of phases. First it is executed the data understanding phase, a preliminary study about the overall dataset for each attributes (data distributions, data quality, etc). Before proceeding with the application of data mining tools, that address the given tasks, a data preparation phase is executed. The aim of this phase is to prepare the data, according the task, to better help the algorithms that we want to use to solve the task. The tasks are: clustering analysis, predictive analysis and mining frequent sequential patterns.

2 Data Understanding and Data Preparation

The dataset consists of 471910 rows and 8 columns. Each row gives information about a single purchase of a product during a shopping session of a given customer: the ID of the basket, the date of the purchase, the

price of the item, the ID of the customer, the country of the customer, the ID of the product, the description of the product and the quantity of the items bought.

2.1 Data type and semantic

In this section is illustrated the type and the semantic of each attribute.

- *BasketID*: a categorical attribute that describes the shopping session. A group of entries with the same BasketID belongs to the same shopping session. There are 3 types of BasketID: 6-digit numbers, eventually starting with a letter: we assume that the ones starting with letter 'A' are aborted transactions, and the ones starting with letter 'C' are canceled transaction. We verified that if one entry has BasketID starting with 'A' or 'C', all the others with the same BasketID start with 'A' or 'C';
- *BasketDate*: an interval attribute that describes the date of the purchase in the format *yyyy-mm-dd hh-mm*. The analyzed time interval is between 2010-12-01 and 2011-12-09;
- *Sale*: a numerical attribute that describes the unitary price of the product bought by the customer in a single shopping session. The prices are VAT-free: we noted that because some gift cards of 10£, 20£ and others, have prices by 16.6% lower than the one claimed in their description.
- *CustomerID*: a categorical attribute that uniquely identify the customer during its shopping sessions;
- *CustomerCountry*: a categorical attribute that represents the nationality of the customer;
- *ProdID*: a categorical attribute that uniquely identifies the product bought by a customer;
- *ProdDescr*: a string that briefly describes the product;
- *Qta*: a numerical value that represents the quantity of products per transaction;

2.2 Data quality

To assess the quality of the data, we performed some variable-wise checks:

- *BasketID*: there are no missing values;
- *BasketDate*: there are no missing values;
- *Sale*: by visualizing the data distribution, we noted that there some negative and huge amount of sales which differ from the mean. The negative sale is related to the BasketID that have a leading 'A' and so, as mentioned before, we consider them as "aborted transaction". There are no missing values;
- *CustomerID*: The 13% of CustomerIDs are missing. Analyzing the data, we found this implication: (ProdDescr = null) $\Rightarrow$ (CustomerID = null $\wedge$ Sale = 0), and almost holds also in the other way (the majority of the records with CustomerID = null and Sale = 0 have ProdDescr = null).
- *CustomerCountry*: there are no missing values, but 340 records have "Unspecified" as country. The remaining 37 values are all real countries;
- *ProdID*: after a deeper analysis we've detected some unusual and erroneous ProdIDs in terms of length and meaning, such as "M", "POST" or "CRUK" (charitable organization);
- *ProdDescr*: 0.15% of the descriptions are missing. Among the present ones, we found some unusual and erroneous descriptions, such as "POSTAGE" or "OOPS ! adjustment";

- *Qta*: by plotting the data, we noted that there some negative and huge amount of quantities that differ with respect to the mean. All the records with negative quantities have BasketIDs with a leading 'C', that we assume means "canceled order". There are no missing values.

2.3 Data cleaning

The following actions illustrates how we decide to manage the quality issues, emerged in the previous phase, in order to improve our data.

- *CustomerID*: for trying to recover some missing CustomerID, we checked if any of the purchases belonging to the same basket have at least one non-null value to be assigned to the other ones, but we didn't find any. Since one of the task is to analyze the customer behavior, and we are not able to understand if two sessions belong to the same customer, we decided to drop these values. In this way, because of the aforementioned implication, we removed also all the rows with null ProdDescr;

- *Sale*: we removed the two aborted transactions (because of negative sale), few (972 records) with Sale = 0 because the 97% have also ProdDescr = null or CustomerID = null, and so less meaningful. We also removed 3 records with Sale < 0.01 because it's not a valid value;

- *ProdID, ProdDescr*: since the previous analysis on ProdID and ProdDescr revealed unusual descriptions, we decided to get rid of that records: these descriptions and IDs represent artificial purchases, probably to remove items that had problems in the warehouse (water damages, broken products, ...), hence there is no customer behind these purchases.

- *Canceled orders*: there are few canceled orders (15% of the total). For the following tasks, we will consider only the not canceled orders, leaving the canceled ones for further analysis.

2.4 Features creation and variable transformation

Since the dataset is a set of transactions, we decided to reorganize it to better understand the behavior of the customers: we grouped the transactions belonging to the same shopping session (information given by the BasketID), and we computed the sum of the total sale. Then, we computed different attributes over the grouped data, as explained below. We focused on extracting new features for analysis:

- *BasketDay*: a categorical attribute that represents the day of the week;

- *Canceled*: a binary attribute that represents if an order was canceled;

- *Total*: a numerical ratio attribute that represents the total price of every single product in a shopping session considering the quantities.

We compute some indicators in order to profile the customers and understanding their purchase behavior:

- *Recency*: a numerical attribute that indicates the number of days since the last purchase;

- *Delta*: a numerical attribute that indicates the number of days of the period of observation of a given customer: it's computed by considering the difference in days between the date of the first purchase and the last purchase; in this way each customer has its own period of time that can help to better discriminate its behavior;

- *Relative frequency*: the numerical ratio between the number of purchases and the period of observation of a customer;

- *Relative monetary*: the numerical ratio between the total amount of money spent by a customer and its period of observation;
- *Median*: a numerical attribute that represents the median of the money spent in each shopping sessions by a customer;
- *Relative total items*: a numerical ratio between the total number of items bought by a customer and its period of observation;
- *Total distinct items*: a numerical attribute that represents the number of distinct items bought by a customer;
- *Max items*: a numerical attribute that represents the maximum number of items bought by a customer in a shopping session;
- *Entropy*: a numerical attribute that represents the Shannon entropy on the purchasing behaviour of the customer, based on the days of the week of the purchases;

The recency, frequency and monetary concepts come from the RFM method, a customer segmentation technique widely used in marketing research. Instead of using the absolute values of frequency e monetary, we decided to keep into account the period of observation of each customer (*delta*, the temporal range between its first and last purchase): imagine two customers A and B, both with 4 purchases, but A made them in the double of the time of B: we want to consider better B than A, and dividing the frequency by the delta (obtaining the relative frequency), we have this behavior; the same reasoning applies to the monetary attribute.

Using box plots and quantiles, we identified possible outliers. We found outliers in Median, RFrequency and RMonetary: their values were too high and only very few customers had those value (most probably wholesalers), so we decided to remove them.

2.5 Relevant feature selection

After the creation of the new indicators, we have quite an amount of attributes. So we determined the most interesting features based on the correlation matrix in order to identify redundant attributes and based on their meaning.

We decide to keep in consideration according our tasks: delta, recency, relative frequency and relative monetary for the following reasons: more recent the purchase, the more responsive the customer is (e.g. new sales or promotions); the more frequently the customer buys, the more engaged and satisfied it is; the relative monetary value differentiates heavy spenders from low-value spenders; the delta can be useful to understand the type of customers and its is the period of observation, more information (possibly) we have about the customer. By looking at the correlation matrix, we can see that all of this attributes are not highly correlated. We decide to discard the rest, except for the median and max items, that are not redundant.

2.6 Attributes distribution

In this section, we will analyze the distribution of some particular attributes, showing interesting statistic plots.

To conclude the data understanding, we report some interesting information about some attributes:

- *Sale*: the top-20% customers ensure the 74% of the total income, almost following the 80/20 rule;

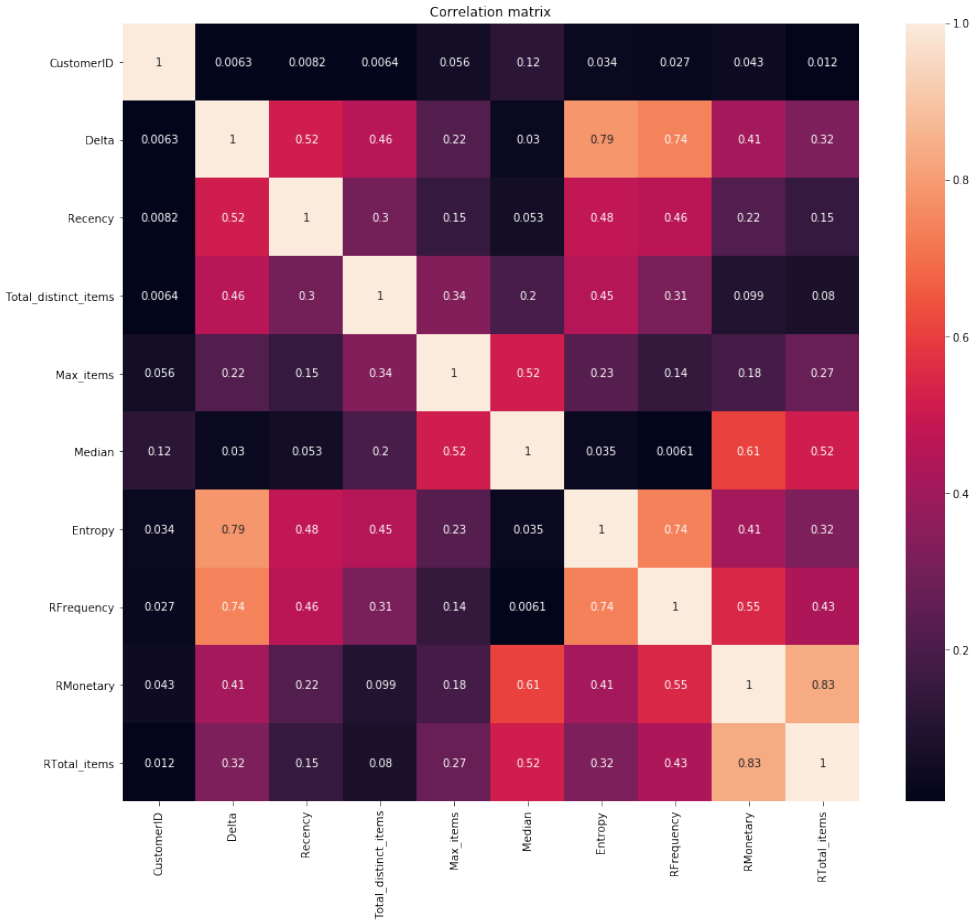
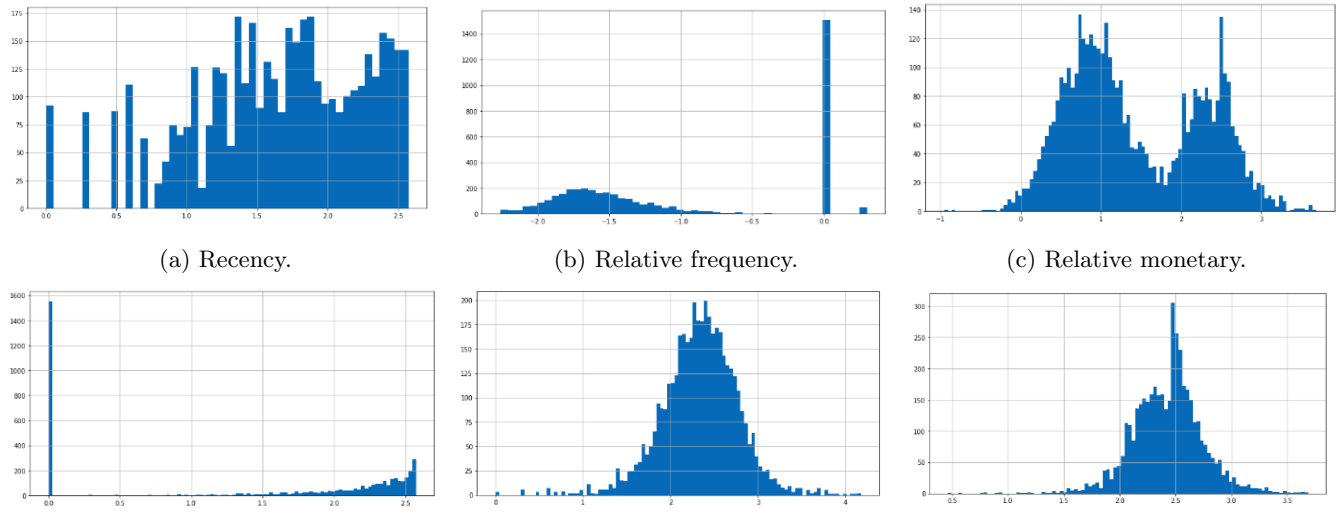


Figure 1: Correlation matrix.

- *BasketDay*: we noted that there are no sales on saturday. In general, the most number of sales are in the middle of the week;
- *Hour*: we observe that lunch time, especially from 12pm to 1pm, is preferred for the online shopping;
- *Top 10 frequent products*: we have identified the top 10 frequent products purchased. This information can be used in the sequential pattern mining task;
- *Customer country*: The 89% of the customers come from United Kingdom, and there are 340 (0.08%) unspecified records' countries;
- *Entropy on shopping date*: most of the customer have 0 entropy (they made every purchase only on one day of the week), but this comes from the fact that a lot of them have only one purchase.

3 Clustering

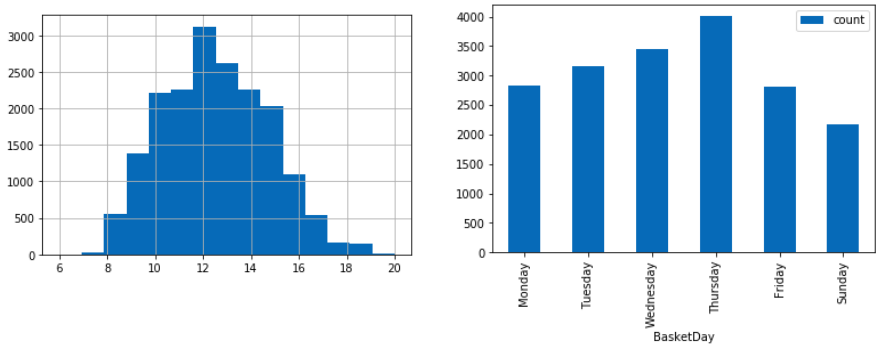
In this section, we describe the three main clustering algorithms applied to the dataset (K-means, DBSCAN and hierarchical), and their results. In addition, we performed some experiments on two extra algorithm: X-means and SOM-SC.



(a) Recency. (b) Relative frequency. (c) Relative monetary.

(d) Delta. (e) Max items. (f) Median.

Figure 2: The independent axis is log transformed in order to have better plots.



(a) Shopping hours. (b) Shopping days.

Figure 3: Temporal analysis of the shopping sessions.

3.1 Preprocessing

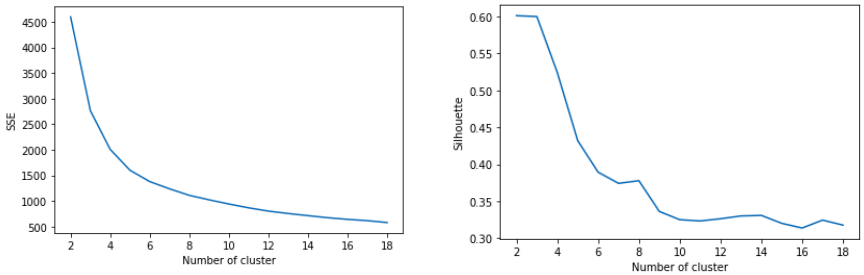
To work well with this algorithms, e.g. K-means, some transformation are required. Since some variables have an high positively skewed distribution, we applied a log transformation to have unskewed data. Then we standardize to the same average value and scale to the same standard deviation. Since the goal is to describe the customers' behavior, we focused only on the attributes that can give information about money spent and information related to the time, such as time interval in which the customers were active or the last purchase that can help to understand if there are some frequent pattern behavior such as customer that made only one purchase and then disappear. The chosen attributes are: relative monetary, delta and recency.

3.2 K-means

The following sections present the analysis of the results of the K-means clustering algorithm.

3.2.1 Identification of the best k

For choosing the optimal number of clusters, we followed the approach of the "knee" or "elbow" method.



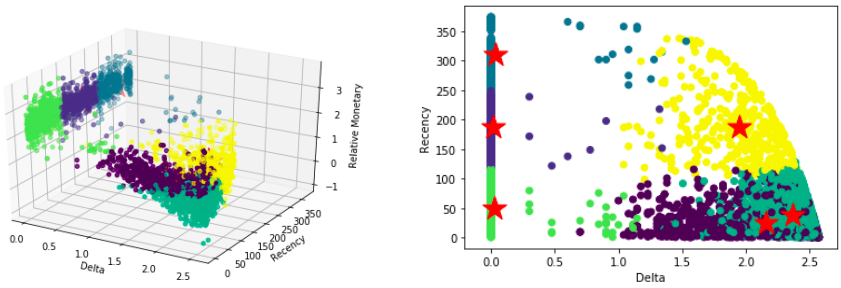
(a) SSE. (b) Silhouette.

Figure 4: For $k = 6$, SSE.1379.00 and Silhouette: 0.38

Looking at the curvature of the curve, we could choose between $k \in [4, 6]$. We decided to choose the largest value of k in the range ($k = 6$) as a possible solution to initial centroids problem . Then in the post processing, if needed, we could eliminate small clusters or merge clusters with small SSE.

3.2.2 Description of the best clustering

According to the clusters given by K-means, we classified the customers in the following categories:



(a) Clusters plotted with delta, recency and relative monetary. (b) Clusters plotted with delta and recency.

- *One shot customers* (dark teal and violet, bad customers): customers that have bought only one time and then disappeared. In the plot, we can see that this behavior is frequent and it represented by customers that have high or medium recency and delta equal to 0 (period of observation of only 1 day). In this case, the relative monetary is not relevant;
- *New customers* (green, potentially bad or good): despite the period of observation of 1 day, these customers are recent and so they are for sure new customers, and their future actions can put them in the one shot customer category or in the others. Also in this case, the relative monetary is not relevant;
- *Ex habitual customers* (yellow, at risk): this cluster describes customers that were good in the past in terms of relative monetary and delta but at certain point they stopped. Maybe this category can be identified as the category of customers that probably we are going to lose, or we've already lost.

- *Habitual customers* (teal, good customers): this cluster describes customers with quite recent purchases because of their low recency, and with a medium monetary with respect to the delta.
- *Best customers* (purple, very good customers): customers that have bought most recently and generated the most revenue with respect to the delta.

We can notice that K-means with this dataset and attributes, is good enough in finding the bad customers clusters, while it is less good to distinguish between good and best customers. But probably, since K-means has some limit related to the clusters' shapes, some habitual and best customers are put in the wrong cluster.

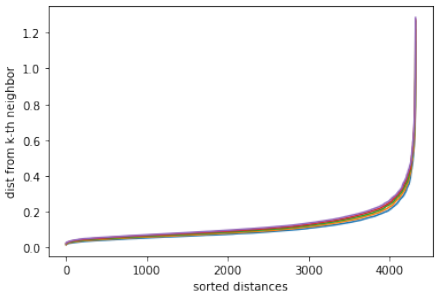
	RMonetary	**Delta**	**Recency**	**Behavior**
Mean	0.6	2.4	37.4	Habitual (teal)
	1.2	2.1	25.0	Best (purple)
	0.8	1.9	186.5	Ex habitual (yellow)
	2.3	0.03	311.0	One shot
	2.4	0.01	188.0	One shot
	2.4	0.02	50.1	New (green)

3.3 DBSCAN

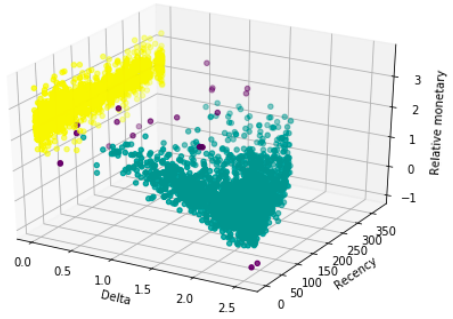
In this section, we present the analysis of the results of the DBSCAN clustering algorithm. We decided to follow the same reasoning used for K-means, thus we attempted to do the clustering over the same set of attributes.

3.3.1 Parameters selection

In order to choose the right ϵ and min points, we executed a grid search where $minpoints \in [5, 45]$ and $\epsilon \in [0.1, 1]$ using a proper increment for searching on those intervals. The resulting table shows for a given ϵ and min points, the number of clusters obtained and the mean noise points distance. In addition, to facilitate the choice for the ϵ we also used the "knee" method by plotting the distance to the k-th nearest neighbour with $k \in [5, 10]$. By visually inspecting the results of the curves and the grid search table, we selected the right hyperparameters for attempting the clustering with DBSCAN.



(a) K-th nearest neighbour.



(b) DBSCAN clustering with min points = 5 and $\epsilon = 0.5$. Purple points are noise points.

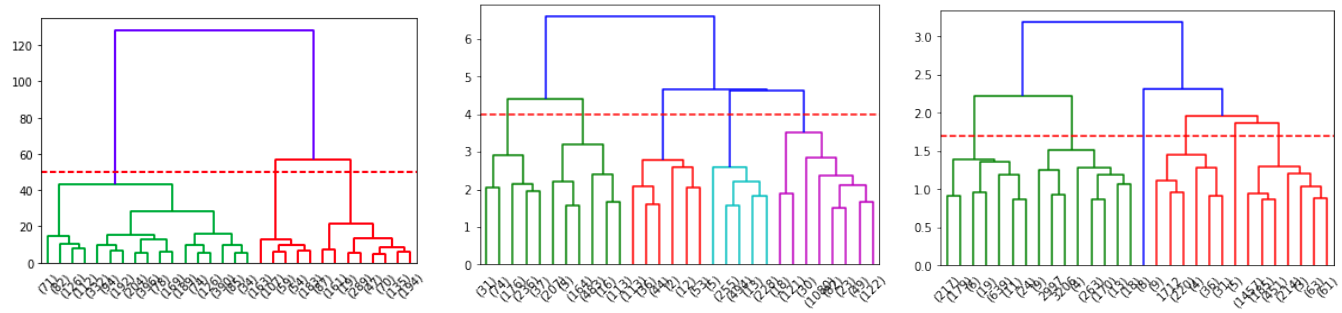
3.3.2 Description of the best clustering

Similarly to K-means, DBSCAN, tuned the best setting min points = 5 and $\epsilon = 0.4$, successfully identified the clusters of one shot customers. This is due to the properties of DBSCAN that is able to detect high or low density area. But it is unable to identify: best, good and ex habitual customers that are put in the same cluster. The main reason is that we have different density distributions inside this data cloud and this kind of behaviour represents the conditions under which DBSCAN performs worse.

3.4 Hierarchical

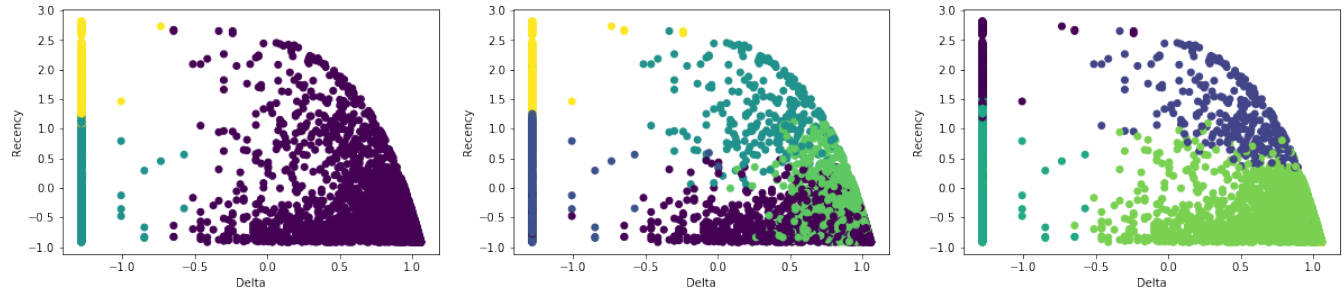
In this section, we present the analysis of the hierarchical clustering algorithm. We decided to use the same set of attributes. We performed the hierarchical clustering using the Euclidean distance as a metric, and we tried the single, complete, average linkage, and the Ward's method.

The visual results of those clustering are shown in the figure below, while their respective dendrograms. The clusters obtained are very similar to K-means.



(a) Ward method. (b) Complete linking method. (c) Average linking method.

Figure 7: Dendrogram plot.



(a) Agglomerative clustering with Ward method. (b) Agglomerative clustering with Complete method. (c) Agglomerative clustering with Average method.

Figure 8: Agglomerative clustering plots.

Methods	Silhouette
K-means	0.31
Ward	0.58
Complete	0.38
Average	0.43
Single	-0.1

Table 1: Silhouette table.

3.4.1 Comparisons and best clustering approach

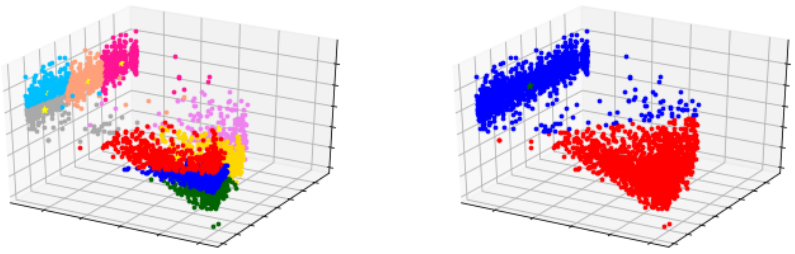
In conclusion, each clustering result found two main behavior of the customers in this dataset. Expect for DBSCAN, the other approaches found very similar results and allows us to understand the specific behavior inside this two main categories. The six categories found by K-means could be reduced, as we can see from the last method, and this could help to better generalize, e.g. put together good and best customers like the agglomerative clustering with average method that find 4 main clusters that can be identified as: good customers, ex habitual customers, one-shot buyers: inactive and new.

3.5 Optional algorithms

We explored the opportunity to use alternative clustering techniques in the library *PyClustering*, an open source data mining library written in Python and C++ that provides a wide range of clustering algorithms and methods, including bio-inspired oscillatory networks. Since the two main problems that we faced in this task, related to the algorithm used, are to find the correct number of clusters without manually managing it and find the best parameters, we looked for, inside this library, algorithm that automatically manage (or in part) these problems. We chose two explore two algorithm: X-Means and SOM-SC.

3.5.1 X-Means

We decided to choose this algorithm because: it is an extension of K-Means; we have seen during the lectures of this course; it starts with the assumption of having a minimum number of clusters, and then dynamically increases them. In particular, the X-Means goes into action after each run of K-Means, making local decisions about which subset of the current centroids should split in order to better fit the data. X-means uses specified splitting criterion to control the process of splitting clusters. We also used K-Means++ for calculation of initial centers.



(a) Splitting criterion: Bayesian Information Criterion.

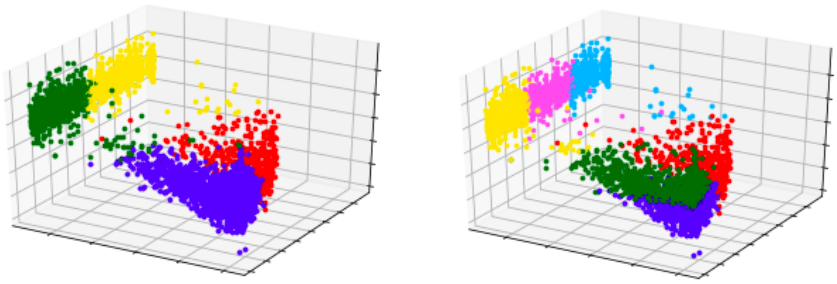
(b) Splitting criterion: Minimum Noiseless Description Length.

Figure 9: X-Means axis x, y, z: delta, recency and relative monetary.

In the figure above, we reported the best results. All the runs are executed starting with two clusters. As we can see, the execution with BIC gives similar results to K-means. In particular, it tends to split into several clusters the habitual (good) and best customers. Other runs of X-Means with BIC tended to find more small clusters, but they were not so interesting. We experiment another splitting criterion: Minimum Noiseless Description Length, and we found similar results to DBSCAN.

3.5.2 SOM-SC (Self-Organized Feature Map for Simple Clustering)

SOM-SC is adaptation of SOM for cluster analysis in simple way. The basic idea is that the amount of clusters that should be allocated define the amount of neurons in the self-organized map.



(a) N. clusters: 4 (b) N. clusters: 6

Figure 10: SOM-SC axis x, y, z: delta, recency and relative monetary.

The results that we obtained are very similar to results presented above.

4 Predictive analysis

In this section, we show how we created the customer profile and how we applied the classification algorithms (decision trees, SVM, KNN, random forests, neural networks and naive Bayes) to predict the customer behavior.

The predictive analysis phase is split in two: in the first part, we will assign the "high-spending", "medium-spending" and "low-spending" labels based on statistical analysis; in the second part, we will work on labels assigned on the bases of the clustering analysis.

4.1 Preprocessing

For this phase, the most important data transformation is the normalization: we opted for a Min-Max normalization. Oversampling and undersampling techniques will be discussed in their respective sections for the two types of labels assignment.

4.2 Statistical-based classification

4.2.1 Labels assigmnent

For the labels assignment, we started from the idea of the RFM scoring: we considered the relative monetary, the delta and the recency attributes, to keep track of the 3 main aspects of a customer behavior (the money spent, the frequency of the purchases and its last activity); then, for each attribute we assigned a discrete

score from 1 to 3, based on the position of each customer in the $0.0 - 0.3$ quantile, $0.3 - 0.7$ quantile, or $0.7 - 1.0$ quantile. Note that higher values mean better customers for the monetary and delta attributes, but worse customers for the recency.

Then, another score is computed over the weighted sum of the three single scores: $Score = \alpha \cdot RMonetaryScore + \beta \cdot DeltaScore + \gamma \cdot RecencyScore$. We chose for α, β and γ values of 0.3, 0.4 and 0.3, respectively, to prefer a little more customers with longer periods of observation.

Over this score, the customers belonging to the $0.0 - 0.2$ quantile will be assigned the "low-spending" label, the ones in the belonging to the $0.2 - 0.7$ quantile will be assigned the "medium-spending" label, and the "high-spending" label will be assigned to the remaining top-30%.

In this way, the majority of the customers will be classified as medium, and that's what we expect in a real scenario. The problem is that now the data is unbalanced (2059 medium, 1036 low and 1227 high), and low and high classes are underrepresented: we will deal with this problem with oversampling and undersampling the unbalanced classes.

4.2.2 Attributes for prediction

For the prediction of the classes, we chose only two attributes: the relative frequency and the recency, leaving out the monetary attribute. In this way, only one attribute used for the labeling (the recency) is being used for the prediction, while the other one, the relative frequency, has a correlation of 0.74 with the delta attribute. Full inference was left for the monetary value.

4.3 Clustering-based classification

4.3.1 Labels assignment

To not be biased by the one-shot customers, due to the hypershpere-shaped clusters that K-means generates, we temporarily removed them from the dataset, and applied K-means on the remaining data. Then we merged the obtained clusters in three supergroups: high (purple), medium (teal), low (yellow). After that, we put back the one-shot customers, assigning to all of them the low label.

In the left plot, we can note that the one-shot customers are divided into medium and low spending classes: the new ones (low recency) are medium spending, and without new purchases they'll become yellow, so low spending customers, otherwise they will move to the right cluster, in a sort of customer lifecycle.

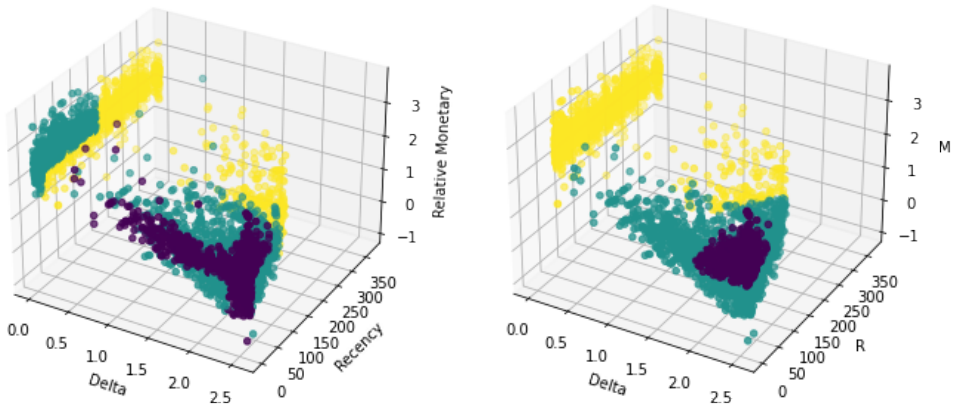


Figure 11: Customer classes distributions: statistical (left) and clustering-based (right).

After this labels assignment, we note that the classes are again a little unbalanced: 1690 customers for the medium class, 1250 for the low, and 912 for the high. Because of this, also in this clustering-based classification phase, we will apply both oversampling and undersampling to the data, to see the impact on the results of the classification algorithms.

4.3.2 Attributes for prediction

We recall that the clustering has been performed on the relative monetary, delta and recency attributes.

In this second phase of the classification task, instead of using the relative frequency and the recency, we decided to keep the delta and the median: again, we have an attribute, the delta, that has been used for the labels assignment, and another one, the median, that have a correlation of 0.61 with the relative monetary. For this reason, we are going to expect more difficulties in this phase of classification, compared to the previous one.

4.4 Models analysis

Dataset splitting The dataset has been divided in two parts: the 70% is used for the training and the validation of the models, and the remaining 30% is used for the testing of the prediction performances. The sampling is stratified on the labels, to reflect the original labels distributions.

4.4.1 Decision trees

The selection of the best model has been performed through a randomized grid search on the following hyperparameters distributions:

Hyperparameter	**Values**
max_depth	2, 3, 5, 6, 7, 10, 12
min_samples_split	[10, 50]
min_samples_leaf	[10, 50]
criterion	entropy, gini
splitter	best, random

Table 2: Hyperparameters distributions for decision trees.

	Statistical			**Clustering**		
	Unbalanced	Oversampled	Undersampled	Unbalanced	Oversampled	Undersampled
High	0.87	0.91	0.88	0.71	0.81	0.77
Medium	0.90	0.82	0.85	0.80	0.72	0.73
Low	0.95	0.98	0.96	0.94	0.93	0.93
Average	0.91	0.90	0.90	0.82	0.82	0.81

Table 3: F1 scores for test predictions with decision trees.

This decision tree is the simplest to explain; it has been computed with the following hyperparameters: criterion: gini, max_depth: 3, min_samples_leaf: 35, min_samples_split: 38, splitter: best.

The relative frequency and the recency attributes are used in a quite balanced way (four and three times, respectively), also allowing to achieve high values of accuracy in the prediction, as shown also in the table.

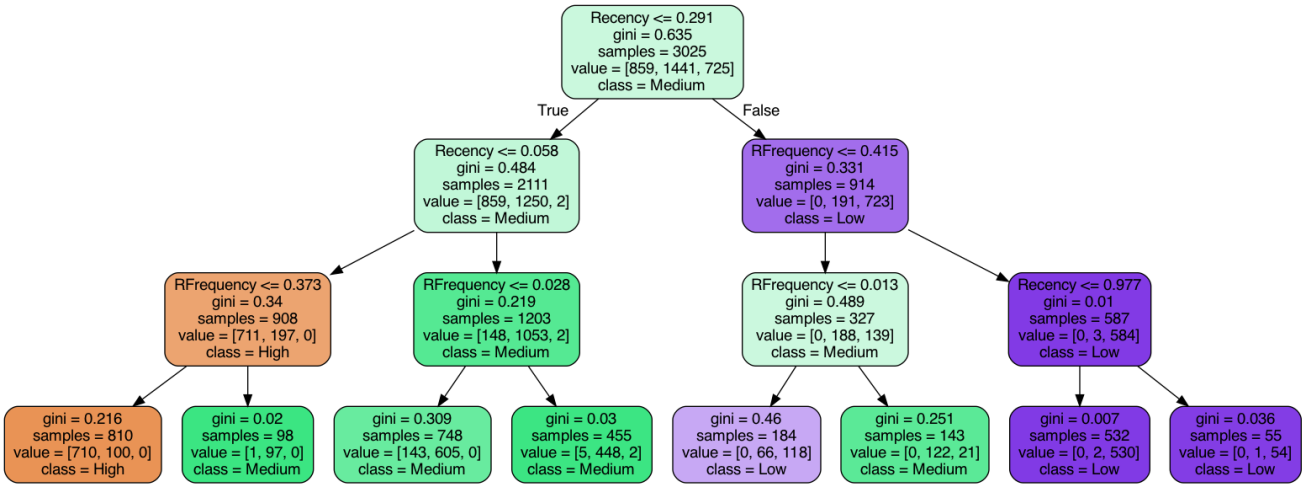


Figure 12: Decision tree with unbalanced classes.

4.4.2 SVM

Hyperparameter	Values
kernel	linear, poly, rbf, sigmoid
gamma	scale, 1e-2, 1e-3, 1e-4, 1e-5
C	0.001, 0.10, 0.1, 0.5, 1, 10

Table 4: Hyperparameters distributions for grid search with SVM.

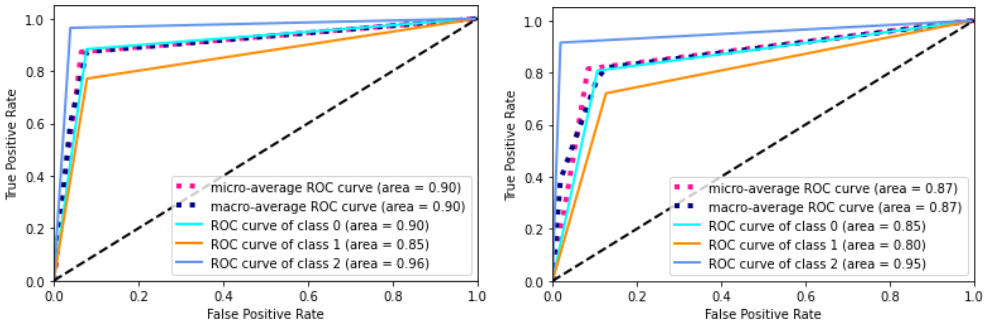


Figure 13: SVM ROC curves: statistical (left) and clustering-based (right).

In the two ROC curves, we can note that the ratio between true positives and false positives is similar, with less accuracy in the clustering-based approach. This behavior will appear in every model, probably because of the attributes chosen for the prediction.

4.4.3 KNN

In the model selection, we noted that the ball_tree algorithm always outperformed the kd_tree one. The optimal number of neighbors has been found to be 15 (statistical) and 10 (clustering).

	Statistical			Clustering		
	Unbalanced	Oversampled	Undersampled	Unbalanced	Oversampled	Undersampled
High	0.83	0.88	0.85	0.71	0.79	0.80
Medium	0.85	0.82	0.78	0.80	0.71	0.70
Low	0.89	0.95	0.94	0.90	0.92	0.92
Average	0.86	0.88	0.86	0.80	0.81	0.81

Table 5: F1 scores for test predictions with SVM.

	Statistical			Clustering		
	Unbalanced	Oversampled	Undersampled	Unbalanced	Oversampled	Undersampled
High	0.84	0.88	0.88	0.71	0.83	0.77
Medium	0.87	0.82	0.90	0.80	0.70	0.68
Low	0.92	0.95	0.93	0.92	0.93	0.91
Average	0.88	0.88	0.87	0.81	0.82	0.79

Table 6: F1 scores for test predictions with KNN.

4.4.4 Random forests

Despite the hyperparameters were similar to the ones used for the decision trees, the accuracy and the F1 scores of the random forest are slightly lower.

Hyperparameter	**Values**
max_depth	2, 3, 5, 6, 7, 10, 12
max_features	[1, #features]
min_samples_split	[10, 50]
min_samples_leaf	[10, 50]
bootstrap	yes, no
criterion	entropy, gini
class_weight	balanced, <high: 0.25, medium: 0.50, low: 0.25>

Table 7: Hyperparameters distributions for grid search with random forests.

	Statistical			Clustering		
	Unbalanced	Oversampled	Undersampled	Unbalanced	Oversampled	Undersampled
High	0.85	0.89	0.87	0.66	0.82	0.76
Medium	0.87	0.85	0.84	0.80	0.76	0.69
Low	0.93	0.96	0.96	0.93	0.94	0.94
Average	0.88	0.90	0.89	0.80	0.84	0.80

Table 8: F1 scores for test predictions with random forests.

4.4.5 Neural networks

For the model selection of the neural networks, we searched for the simplest model that was able to fit the data, being aware that the overfitting was really easy to achieve, especially in the oversampled data.

For both datasets, we chose the same model: two layers of 16 neurons and another one of 3, alternated with dropout layers, set with a probability of 20%. The activation functions are relu for the first two layers, and softmax for the last one, while the best optimization algorithm was found to be adam.

	Statistical			Clustering		
	Unbalanced	Oversampled	Undersampled	Unbalanced	Oversampled	Undersampled
High	0.82	0.84	0.83	0.67	0.78	0.81
Medium	0.82	0.74	0.64	0.73	0.65	0.62
Low	0.89	0.91	0.92	0.87	0.90	0.89
Average	0.84	0.82	0.80	0.75	0.78	0.77

Table 9: F1 scores for test predictions with neural networks.

4.4.6 Overall considerations

The naive Bayes classifier has not been listed above because of the lack of hyperparameters and its poor performances (it performed the worst among all the models), especially on the clustering-labeled data.

Despite the variety of prediction models that have been tried, the dataset with the labels assigned from the clustering phase revealed to be harder to classify in any trial, especially for the high and medium classes. These results might be affected by assumptions that we made to label customers, in the first place and by attributes used for the prediction.

5 Sequential pattern mining

In this section, we consider the problem of mining frequent sequential pattern in order to better understand some information hidden in the dataset. The basic idea behind market basket analysis is to find pairs or sets of products that are jointly observed in large samples of baskets. To perform this kind of analysis, we had to prepare the data.

5.1 Preprocessing

First, we have to define the main components. An item set X is a set of items such that $X \subseteq \{ProdIDs\}$. A sequence is an ordered list of item sets, where the order is temporal, according BasketDate. The structure of the dataset is represented in the table. For simplicity, for each distinct ProdID we have done a mapping to an integer.

Customer ID	**Temporal sequence of baskets**
12347	{26, 10}, {5, 1, 7}, {4}
...	...
18287	{18, 50, 99}, {78, 34}

Table 10: Items between curly brackets represent an item set. This sequence indicates that a customer 12347 purchased items 26 and 10 at the same time, then bought items 5,1,7 at the same time, and finally purchased 4.

Various measures can be used to assess how interesting a subsequence is. We focused on the support measure and in particular we want to retrieve all patterns (subsequences) that have a support no less than the minimum support threshold. We decided to exploit two algorithms: GSP, based on the Apriori algorithm for frequent itemset mining, and PrefixSpan. Both algorithm give the same correct results.

5.2 GSP

As a first attempt, we used the GSP algorithm on the entire database, but the execution required a lot of time, because it needs: multiple database scans to calculate the support of candidate patterns; generation of

patterns that may be not present in the database; maintaining the huge number of candidates in memory.

After more than 48 hours of computation with no results, we performed some experiments with only a portion of the database, trying different support thresholds, but the results were unsatisfactory, because we didn't manage to get sequences with more than one item. So, we switched to a more efficient algorithm, PrefixSpan.

5.3 PrefixSpan

One of most popular pattern-growth algorithm for sequential pattern mining is PrefixSpan. PrefixSpan proceeds exploring the search space of sequential patterns using a depth-first search. It starts from sequential patterns containing a single item and explores larger patterns by recursively appending items to patterns to create larger patterns. The pattern-growth approach of PrefixSpan has the advantage that it only explores patterns appearing in the database.

5.4 Experiments on PrefixSpan

The number of patterns in the search space depends on how the support threshold is set and on how similar the sequences are in a sequence database. The properties of the database are:

- *Number of input sequences*: 4333
- *Total number of events*: 18400
- *Number of distinct products*: 3658

5.4.1 Results

After the execution, we found 2306 frequent sequences with minimum support threshold ≥ 100. In the following table, we reported the most interesting patterns with the highest support and we identified four categories of the frequent products: ornaments, bags, bakery and party accessories.

	Pattern	**Support**
Description	REGENCY CAKESTAND 3 TIER	881
	WHITE HANGING HEART T-LIGHT HOLDER	856
	PARTY BUNTING	708
	ASSORTED COLOUR BIRD ORNAMENT	678
	SET OF 3 CAKE TINS PANTRY DESIGN	640
	JUMBO BAG RED RETROSPOT	635
	PACK OF 72 RETROSPOT CAKE CASES	635
	PAPER CHAIN KIT 50'S CHRISTMAS	613
	NATURAL SLATE HEART CHALKBOARD	587
	BAKING SET 9 PIECE RETROSPOT	581

Table 11: List of the most frequent individual items.

ProdID	Description	Category
85123A	WHITE HANGING HEART T-LIGHT HOLDER	Ornaments
84879	ASSORTED COLOUR BIRD ORNAMENT	
22469	HEART OF WICKER SMALL	
22086	PAPER CHAIN KIT 50'S CHRISTMAS	
22457	NATURAL SLATE HEART CHALKBOARD	
20725	LUNCH BAG RED RETROSPOT	Bags
22382	LUNCH BAG SPACEBOY DESIGN	
22383	LUNCH BAG SUKI DESIGN	
20727	LUNCH BAG BLACK SKULL	
23209	LUNCH BAG DOILEY PATTERN	
85099B	JUMBO BAG RED RETROSPOT	
23203	JUMBO BAG DOILEY PATTERNS	
22197	SMALL POPCORN HOLDER	
22384	LUNCH BAG PINK POLKADOT	
20728	LUNCH BAG CARS BLUE	
22423	REGENCY CAKESTAND 3 TIER	Bakery accessories
23245	SET OF 3 REGENCY CAKE TINS	
22720	SET OF 3 CAKE TINS PANTRY DESIGN	
21212	PACK OF 72 RETROSPOT CAKE CASES	
22138	BAKING SET 9 PIECE RETROSPOT	
47566	PARTY BUNTING	Party accessories
23298	SPOTTY BUNTING	

Table 12: Description of the most frequent products.

6 Conclusions

We used different data mining techniques to analyze this dataset following a specific process.

In the Data Understanding section, we addressed issues related to data semantics, quality and extracting knowledge from the data distributions for the next tasks.

In the Clustering section, we focused on finding groups of customers with similar behavior. Despite the obstacle of various densities and shapes of the clusters, we identified different interesting behaviors.

In the Predictive section, first we found a realistic classification of the customers into: "high-spending", "medium-spending" and "low-spending" based on statistical and clustering analysis. Then, we compared different classifiers with different hyperparameters and different sampling techniques. The general behavior was the same among most of the trials, with the medium class being the most difficult to detect, probably for being between the other two and not so easily divisible.

In the Sequential pattern mining section, we obtained results that reflect the shopping behavior of the majority of the customers: most of the items are cheap ones, and so the items belonging to common patterns of frequent items. An interesting fact it that they belong to only four categories, mostly house accessories and bags.

	Pattern	Support	Category
ProdID	85123A, 85123A	407	Ornaments
	85123A, 85123A, 85123A	224	
	84879, 84879	291	
	22469, 22469	201	
	20725, 20725	276	Bags
	20725, 22382	207	
	20725, 22383	206	
	20725, 20727	212	
	20725, 23209	213	
	85099B, 20725	200	
	85099B, 85099B	312	
	85099B, 23203	236	
	22197, 22197	200	
	22384, 20725	201	
	22383, 22383	226	
	20728, 20725	209	
	20728, 20728	205	
	20727, 20725	208	
	20727, 20727	229	
	23209, 23209	206	
	23203, 85099B	204	
	23203, 23203	232	
	22423, 22423	330	Bakery accessories
	22423, 23245	206	
	22720, 22720	225	
	47566, 47566	280	Party accessories
	23298, 23298	210	

Table 13: List of the most frequent set of items that occur in events