Authors: Diletta Goglia and Marco Petix

Data Mining Project 20/21

ANALYSIS OF THE "CUSTOMER_SUPERMARKET" DATASET

1 Data Understanding and Preparation

1.1 Description of the "customer_supermarket" dataset

The customer_supermarket dataset contains information about the transactions between customers around the world and an unspecified business entity. This data is classified among eight different attributes for a total of 471910 records.

1.1.1 The Semantics and Statistics of the dataset

We proceed by describing the aforementioned attributes and their distributions:

BasketID (String)

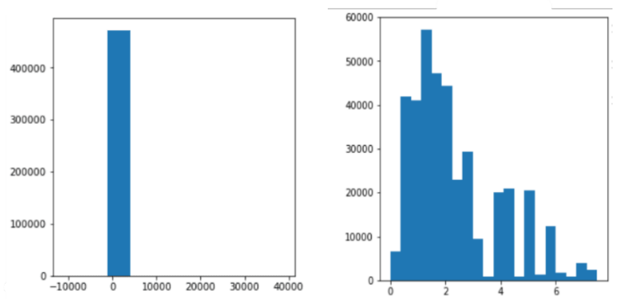
- A predominantly numeric attribute, recognized by pandas as an object due to the presence of a few alphanumeric strings (see **Dealing with the negative values of Qta** below).
- It identifies the purchase sessions initiated by a single customer.
- The attribute, presenting 24627 distinct values, accounts to the same amount of individual purchasing sessions. Said amount decreases to 18867 after dropping the records containing outliers.

BasketDate (String at loading, then Datetime)

- A datetime attribute, initially recognized as containing objects prior to conversion through the pandas casting function.
- It contains the date and time of each purchase session initiated by the customers.
- The observation period identified by BasketDate starts on the 1st of January 2010 and ends on the 10th of December 2011.

Sale (String at loading, then Float)

- A numerical attribute, initially recognized by pandas as object due the use of commas as delimiter for the decimals, then cast to Float.
- It represents the amount spent by the customer to purchase a single unit of the products bought during each transaction.
- Basic checks regarding the attribute's statistics revealed the presence of two records with negative values associated with the adjustment of debts.
- The general shape of the distribution of the attribute values was evaluated using the parameters of skew and kurtosis thus proving to be extremely positively asymmetric and leptokurtic. These conditions resulted greatly diminished after applying a process of outliers elimination within the dataset.
- In the picture on the right, the distribution in question is shown as it appeared before, and after, the aforementioned process.



Customer ID (Float at loading, then String)

- A numerical attribute later casted to object due to its role as identifier for the customers taking part in the transactions.
- The attribute accounts to 4373 individual customers. Said amount decreases to 3477 after dropping the records containing outliers.
- It also contains 65080 missing values whose replacement is addressed within the following section.

CustomerCountry (String)

- A categorical attribute representing a country associated with each customer.
- The attribute accounts to 38 distinct countries.
- The vast majority of the dataset population (90%) consists in records with 'United Kingdom' as value for CustomerCountry.

ProdID (String)

- An alphanumeric attribute acting as identifiers for the products purchased within a transaction.

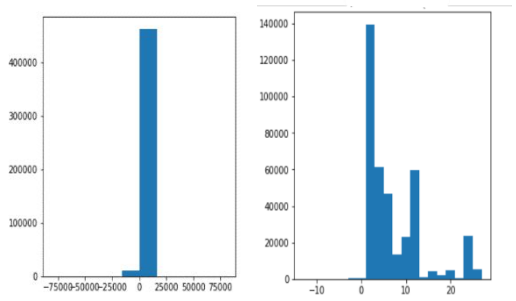
- The attribute accounts to the 3953 individual products. Said amount decreases to 3672 after dropping the records containing outliers.

ProdDescr (String)

- An textual attribute representing a brief description of the products purchased within a transaction.
- The attribute accounts to 4098 individual descriptions, meaning that some of the products accounted for by ProdID descripted in multiple different ways.
- It also contains 753 missing values whose replacement, however, is not addressed due to the low influence of said attribute with respect to our analysis.

Qta (Integer)

- A numerical attribute representing the amount of unity purchased within a transaction with respect to a single type of product.
- Basic checks regarding the attribute's statistics revealed the presence of several records with negative values, some of which further characterized by a BasketID containing C. According to our hypothesis, such records, while representing the legitimate cancellation of purchases, could mislead the calculation of some of the indicators developed in the second part of this task. To prevent this eventuality, a process aimed at eliminating, or at least balancing, the pairs of transactions representing the completed and canceled purchase was applied.
- As for the Sale attribute, the general shape of the distribution of the attribute values was evaluated using the parameters of skew and kurtosis thus proving to be mildly positively asymmetric while still extremely leptokurtic. Once again, these conditions resulted greatly diminished by the process of elimination of outliers within the dataset.
- In the picture on the right the distribution in question before and after the aforementioned process.



1.1.2 Assessing and improving the Data Quality of "customer_supermarket"

In order to improve the quality of the data contained in the dataset, and therefore also of the analysis to be carried out, we examined the dataset in order to identify duplicate records or records with missing values and outliers.
Here follows the results of our analysis in this regard and the measures we have taken to correct them.

1.1.2.1 Duplicates in "customer_supermarket"

Assuming, by personal interpretation, the presence of duplicate records as the result of errors during data entry, and not as the intentional addition of multiple transactions, we decided to search for them within the dataset and, eventually, drop them.

1.1.2.2 Extra: Dealing with the negative values of Qta

As already mentioned, studying the dataset statistics led to the discovery of 9084 records representing canceled purchases.
By realizing the misleading influence that these records, and their positive counterparts (representing the same purchases before cancellation), could have on the computation of the shopping behavior of the customers we implemented a process aimed at identifying, cancelling, or balancing, said couples.
Not all the negative records showed a counterpart of the opposite sign but, by applying the aforementioned process, we proceeded to eliminate 8009 of the former and an equivalent number of the latter.

1.1.2.3 Missing values in "customer_supermarket"

As for the identification of missing values, having already confirmed the presence of the same in the CustomerID and ProdDescr attributes, we have considered the option of analyzing the records with value 0 (often used as NaN for numeric attributes) for the Sale and Qta attributes.
After actually identifying several of these records, we decided to consider the value 0 as legitimate in order to represent aspects of particular transactions. No replacement process was therefore applied for the latter.
The only missing values subjected to a replacement process were those belonging to the CustomerID attribute. This is how said process was implemented:

1. We segmented the dataset based on the value for CustomerCountry in each record;

2. Within each of the 38 segments of the dataset, one for each country represented by the CustomerCountry attribute, we identified the most active customer by computing the mode on the CustomerID attribute;
    a. In this regard, it was necessary to create a custom CustomerID in order to make up for the lack of a non-null one by the population segment associated with the CustomerCountry 'Hong Kong'.
3. We used these CustomerID values as a replacement for the missing ones.

The process therefore worked by associating the transactions without a customer, but with a CustomerCountry value relating to a particular region, to the most active customer in the region itself.

1.1.2.4 Outliers in "customer_supermarket"

The process of identifying and deleting outliers resulted in the deletion of 69967 records from the dataset. Their presence, as well as being strongly suggested by the Kurtosis values for the Sale and Qta attributes, was ascertained through both the use of boxplots and the computation of the interquartile range (IQR).

1.1.2.5 Correlation and visualization with "customer_supermarket"

Explicitly analyzing the correlation between the dataset numerical attributes did not produce remarkable results.
The Sale and Qta attributes proved to be barely correlated and the visualization via scatter plot, also augmented by the introduction of categorical attributes such as CustomerCountry, did not lead to any interesting conclusions.
In order to build plots displaying the behaviour of Sale and Qta over period of times, we created an *ad hoc dataframe* by duplicating the original one and setting the BaskeDate attribute as index.
This led to the discovery of interesting correlations such as:

- the increase in transactions near the middle of the week, Thursday seems to be the busiest day from the point of view of purchases by customers;
- the presence of few purchases with very high prices during the first few hours of operation of the supermarket in question;
- the increase in the number of low-medium price purchases close to the central hours of the day, possibly due to the purchase of food items for lunch.

1.2 The "customers" dataset

A second dataset was then created, this time focused on the representation of the purchasing behavior of each individual customer, and then indexed through the values of the Customer ID.

1.2.1 The Semantics of the "customers" dataset

We proceed by describing the attributes computed for the aforementioned dataset:

- **I** (Integer). The total amount of products purchased by the customer.
- **Iu** (Integer). The total amount of distinct products purchased by the customer.
- **Imax** (Integer). The maximum amount of products purchased by the customer within a single shopping session.
- **E** (Float). The Shannon's Entropy computed on the ProdID values associated with the customer. It's a measure of variety, and disorder, in the products purchased by the customer.
- **Eb** (Float). The Shannon's Entropy computed on the BasketID values associated with the customer. It's a measure of variety, and disorder, in the transactions initiated by the customer. The aim is to distinguish the regular customers from one-time spenders.
- **Ew** (Float). Shannon's Entropy calculated on all days of the week associated with the user's transactions. It's a measure of the variety and clutter in the customer's purchase timeline. The goal is to distinguish the customers who visit the store regularly.
- **Em** (Float). Similarly to Ew, this is the Shannon's Entropy computed on the months associated with the user's transactions. Having a period of observation of slightly longer than a year, it's aim is to distinguish those who initiated transactions with the supermarket for a short period of time, like a single month.
- **Stot** (Float)**.** The total amount spent by the customer within the whole period of observation.
- **Smax** (Float)**.** The maximum amount spent by the customer within a single shopping session.
- **NSess** (Integer). The total number of shopping sessions initiated by the customer.

1.2.1.1 Additional Attributes designed for cluster characterization

- **Country** (String). The attribute Country associated with the Customer within the "customer_supermarket" dataset

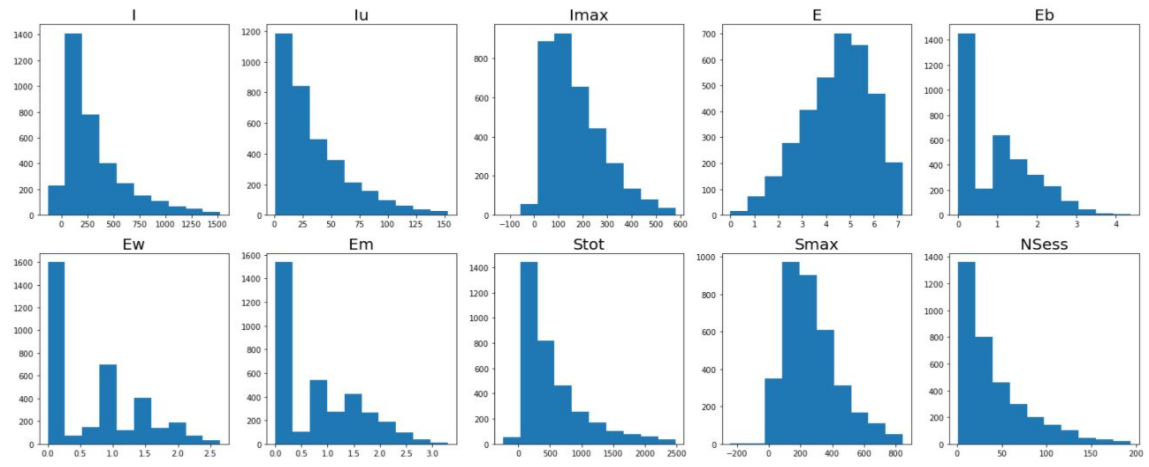
- **Weekly Behaviour** (**Monday**, **...**, **Sunday**) (7 Integers)
  - Seven attributes containing the total number of products purchased by the customer during each specific day of the week
  - The aim is to understand if the population of particular cluster tends to prefer a particular day for its purchases
- **Weekday** (String)
  - The day of the week during which the customer made more purchases overall
  - The aim is similar to the one stated for weekly behaviour but while experimenting with an attribute simpler to compute
- **Monthly Behaviour** (**January**, **...**, **December**) (12 Integers)
  - Twelve attributes containing the total number of products purchased by the customer during each specific month
  - The aim is to understand if the population of particular cluster tends to prefer a particular month for its purchases
- **Month** (String)
  - The month during which the customer made more purchases overall
  - The aim is similar to the one stated for monthly behaviour but while experimenting with an attribute simpler to compute

1.2.2 Data Statistics of the "customers" dataset

Once again we computed the skew and kurtosis parameters in order to obtain general information regarding the distributions of the new attributes. Attributes like I, Iu, Stot and Nsess resulted deeply skewed. Also, most of the attributes from Customers presented values of Kurtosis greatly over 3.

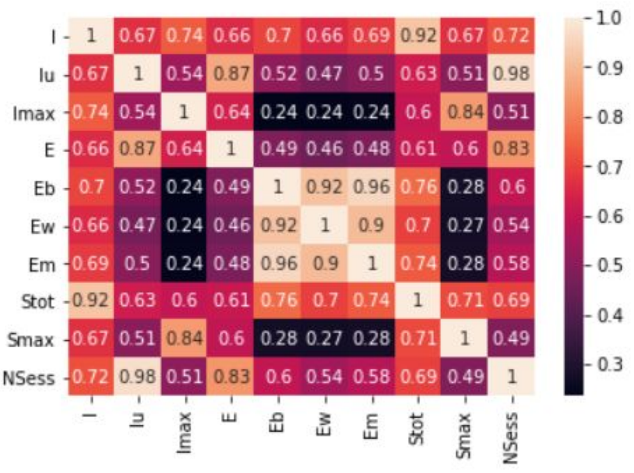
The presence of so many leptokurtik suggested an heavy presence of outliers. In order to deal with said records we again identified them, by computing the interquartile range (IQR), and subsequently dropped them.

These are the distributions of the attributes from Customers as they appeared after the aforementioned process.



1.2.3 Correlations within the new Dataset

We computed the correlation matrix between the main attributes of the new dataset in order to spot couples of highly correlated attributes potentially describing redundant informations.



	I	Iu	Imax	E	Eb	Ew	Em	Stot	Smax	NSess
I	1	0.67	0.74	0.66	0.7	0.66	0.69	0.92	0.67	0.72
Iu	0.67	1	0.54	0.87	0.52	0.47	0.5	0.63	0.51	0.98
Imax	0.74	0.54	1	0.64	0.24	0.24	0.24	0.6	0.84	0.51
E	0.66	0.87	0.64	1	0.49	0.46	0.48	0.61	0.6	0.83
Eb	0.7	0.52	0.24	0.49	1	0.92	0.96	0.76	0.28	0.6
Ew	0.66	0.47	0.24	0.46	0.92	1	0.9	0.7	0.27	0.54
Em	0.69	0.5	0.24	0.48	0.96	0.9	1	0.74	0.28	0.58
Stot	0.92	0.63	0.6	0.61	0.76	0.7	0.74	1	0.71	0.69
Smax	0.67	0.51	0.84	0.6	0.28	0.27	0.28	0.71	1	0.49
NSess	0.72	0.98	0.51	0.83	0.6	0.54	0.58	0.69	0.49	1

Here follows some of the conclusions obtained by analysing the matrix above:

- The high correlations between **I** and **Stot**, and **Imax** and **Smax**, both refer to the increase in cost being obviously proportional to the number of products bought.
- The correlation between **Iu** and **E** is expected due to both attributes being an indicator of variety (and disorder) within the customer's choice of products.

- The very high correlation between **Iu** and **NSess**, as the one between **E** and **NSess**, highlight a tendency to prioritise the purchase of new products, new sessions often leading to the purchase of products that are not familiar.

2 Task 2: Data Clustering

2.1 Dataset introduction

The following clustering analysis is performed on the *customers* dataset: it contains 3494 observations and describes the **purchase behaviour of each single customer.**
The "customers" dataset contains both numerical attributes directly involved in the clustering process, and some additional attributes, both categorical and numeric, used later for the clusters characterization and interpretation.
For the exhaustive list of indicators we refer to the previous dataset explanation section of task1.

2.2 Pre-processing

Before starting the clustering process, some operations must be done. The two main activities for this step are:

1. High-correlated features elimination

Clustering analysis and results can be affected by higher correlations: this because the semantic meaning of strongly correlated features is that they are basically capturing the same information.
Dropping redundant attributes also benefits the analysis by reducing the dimensionality of the dataset and raising the influence that more useful features could have on the whole clustering process.
For this purpose, we first examined the correlations between attributes to be clustered in order to identify the highly correlated couples. Then we fixed a maximum **threshold** value at 0.9 in order to identify highly correlated features and, subsequently, we selected which of these must be deleted.
As a result of this analysis, the indicators that we decided to drop because of too high correlation are: *Eb, Em, Ew, Nsess*
In particular, the high value of correlation for the three entropy-based indicators *Eb, Em, Ew* is due to the fact that they had very similar nature and so they were describing almost the same kind of information. Moreover, we had no problem in deleting them since they do not affect the clustering process.
On the contrary, even if the attribute *Stot* is highlighted as highly correlated with *I*, we decide not to drop them because they will further reveal themselves to be the most influential indicators that we have regarding the clustering process (see below “Identifying the most influential attributes” section).

2. Normalization

We have chosen the Min-Max normalization approach in order to have always-interpretable data so that we can avoid applying inverse transformation (needed instead in the Z-Score approach).
For this purpose we used the MinMaxScaler library from Scikit-learn.

2.3 Clustering by K-means

The K-means algorithm allows to divide a set of objects into *k* groups on the basis of their attributes.
For this project we used the KMeans library from Scikit-learn.
The value of *k* is the input of the algorithm, and it represents exactly the number of clusters that you want to find.

2.3.1 Identification of the best value of k

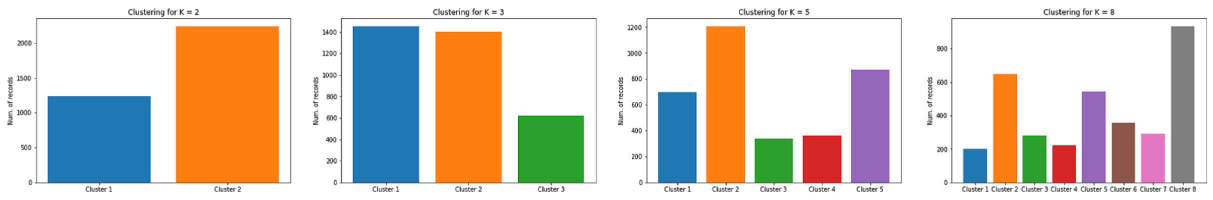
Clearly, the definition *a priori* (i.e. not knowing the data) of this number is one of the most delicate phases. Our aim in this context is to find the best value of *k* by running multiple executions. We started from an initial value of 2 and we went on computing the clusterization until we reached the value we chose for the maximum value of *k,* that is 20.

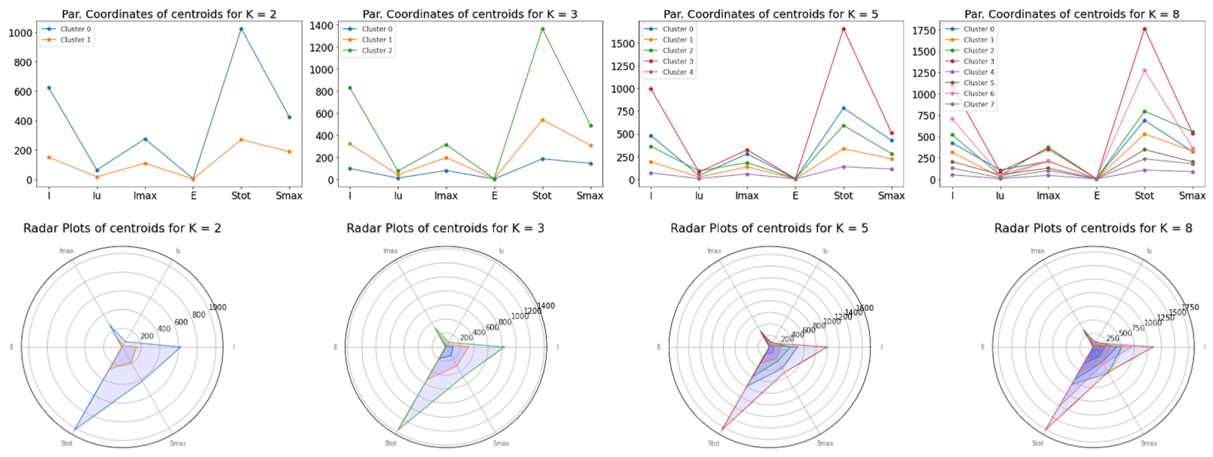
2.3.2 Evaluation
In order to evaluate the outcomes of this *k* value selection process we used the following three techniques, based on two different metrics:

1. **Elbow Method on the clusters Inertia:** plotting the SSE as a function of the number of clusters, and picking the elbow of the curve as the number of clusters to use.
2. **Average Silhouette Method** as an indicator of both separation and cohesion among clusters.
3. **Insights from Hierarchical Clustering** performed through the Ward method, which aims at the local optimization of the SSE function.

2.3.2.1 Comparing conclusions
According to the first two methods, the values 5 and 8 could be plausible candidates for the role of optimal *k*. The elbow method also suggests the value 3 and, while the value 2 stands as the most promising candidate according to Silhouette score. The 2 value is also supported by the dendrogram previously displayed.
Ultimately, **we decide to compute four different clustering runs**, each associated with a potential *k* value candidate, and to leave the choice for the best *k* to post-processing evaluation on the obtained results.

2.4 Characterization of the obtained clusters
After having executed the K-Means algorithm for all the four *k* values candidates, we now present an analysis aimed toward the study of the **clusters composition and population**, so that we can fully understand and interpret the results obtained by the k-means algorithm.



2.4.1 Identifying the most influential attributes


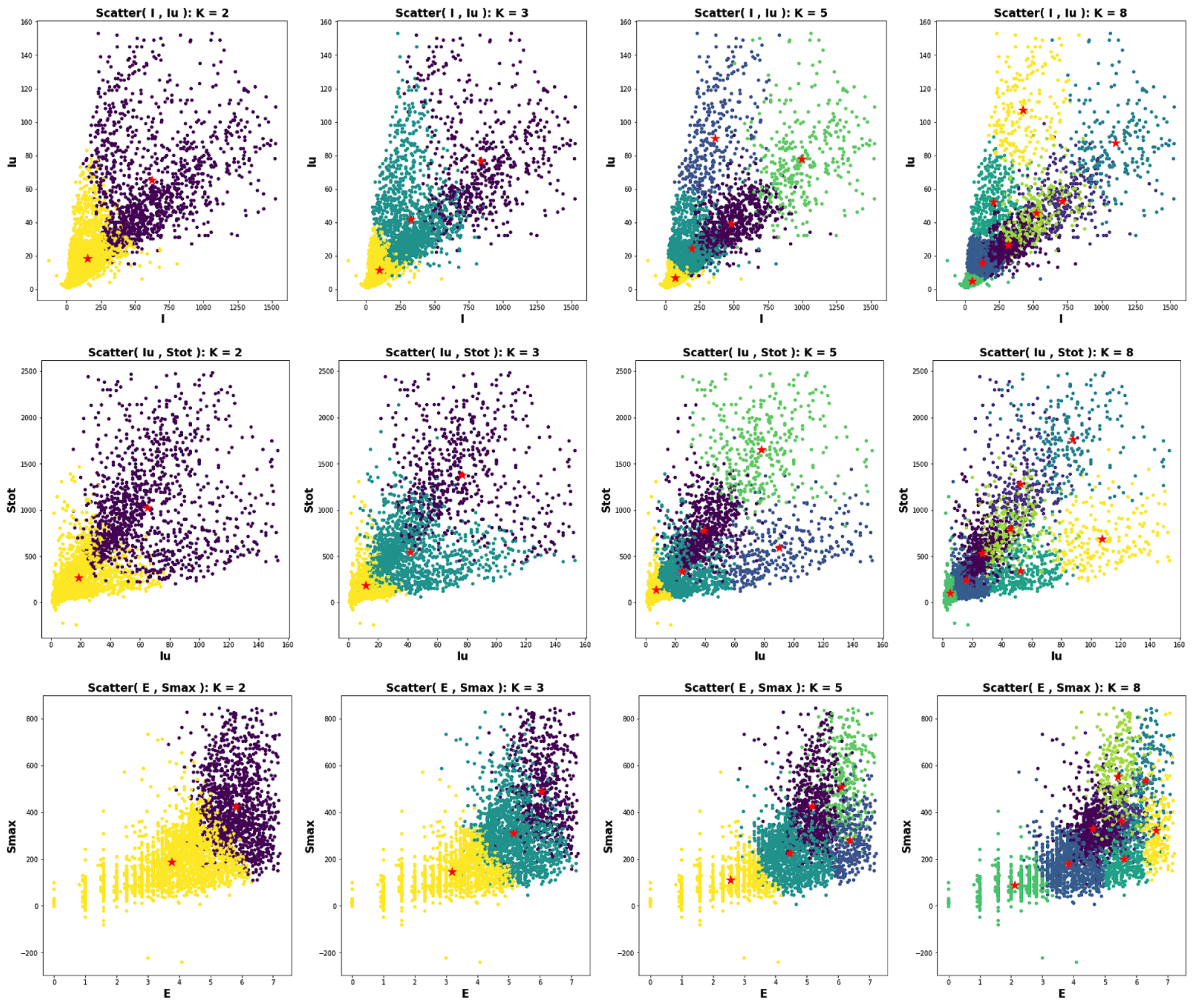
The coordinate plot and the radar plot show the differences for each attribute among the centroid of each cluster.

The same results seem to be shared among the four clustering: the centroids, and thus the population of each cluster, mostly differs in their values for I and Stot. Therefore, our decision to not drop them because of high-correlation value is justified.

2.4.2 Characterizing the clusters
The interpretation of clusters with respect to categorical labels led to several barplots and comparison. In this report we do not show them since they are not meaningful for any conclusion or interesting observation. As a matter of fact, as we will confirm later in this report, the process of building clusters is not following one of these categorical features. Instead, the algorithm performs divisions with respect to a completely different indicator, that we will discover and explain later.

2.4.3 Visualizing the clusters and the centroids

We now present a global view on the clusters and their population thanks to bidimensional visualization, i.e. scatter plots for specific correlations between pairs of features.
Here you can clearly see the cluster separation and assignment in different colors, and also the centroids in red, so that it is immediate to understand and visualize where the center (i.e. the average value) is located for any cluster.



The graphs above show a sparse region of points at the top of the visualization. Clustering such as that for k equal to 2, 3 or 5 group this region into single clusters and this will certainly have a negative impact on the cohesion index of the latter. Clustering like that for k equal to 8 instead covers this area with different clusters, thus gaining cohesion but showing a low separation index due to the overlaps in the central area.

2.4.4 Evaluation of the clustering results

At this point, we are going to use already-known metrics in order to evaluate the clustering obtained with our four versions of k-means. Doing this will grant us a numerical expression of the overall "goodness" of the arrangements of our clusters and, again, it will guide our choice toward which one is the best number of clusters for the optimal partition of this dataset.
The (internal) evaluation metrics that we are going to compute are:

- **SSE** to measure cohesion
- **Davies Bouldin Score** to measure separation
- **Silhouette Score** to measure them both

Clusters	**Cohesion (SSE)**	**Separation (Davies Bouldin Score)**	**Cohesion and Separation (Silhouette Score)**
K = 2	339.51	0.91	0.45
K = 3	247.53	1.05	0.36
K = 5	167.96	1.04	0.33
K = 8	120.46	1.07	0.31

As previously observed and expected, the SSE of our clusterings tends to decrease with the increase of k. A low value of SSE for k equals to 8 was therefore expected.
The SSE value for k equals to 5, however, appears relatively low if compared with the one from the other candidates. This could be an indicator of optimal cohesion within the clusters.
As already observed during our preliminary observations, the clustering for k equals to 2 seems the most promising in terms of Silhouette.

The latest results certainly make the choice of the optimal k rather conflicting. On the basis of what we have observed during the characterization of the various clusters, however, the choice of this value could be relatively irrelevant if not for reasons related to the field of application of the clustering process itself.

2.4.5 Final interpretation

By this we mean that, having assessed that the differences between the populations of the clusters for different values of k are mostly related to the **spending power of a customer**, the choice of one or the other value of k would simply represent the addition of intermediate classification intervals between the extremes linked to customers with small or large propensity / spending capacity.
This conclusion will be supported and reinforced by the DBSCAN clustering results.

2.5 K-means conclusion

What emerges from the evaluation phase with numerical metrics is that our hypothesis (coming from data visualization) that *k*=3 was the optimal choice is definitely confirmed by these numerical results.

2.5.1 Pros

As strongly sustained by literature, the k-means algorithm converges quickly. Despite our data not having a globular shape, the result of the partitions was pretty good.

2.5.2 Cons

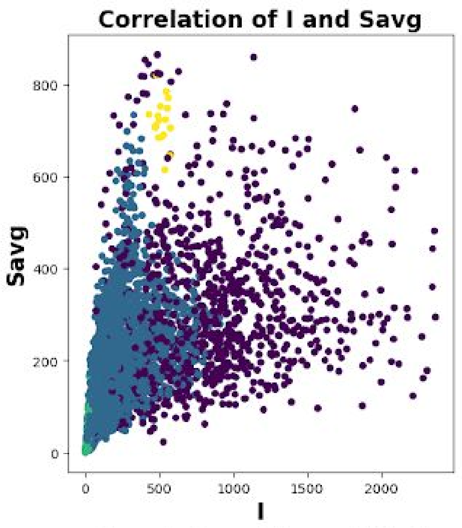
The main disadvantage of k-means is that it requires choosing *a priori* the number of clusters to be identified; if the data is not naturally partitioned you get strange results. This human-driven definition for *k* values definitely required a lot of analysis and adjustments: in case of autonomous establishment from the algorithm itself of the correct number of clusters (as we will see for DBSCAN), this entire process would not have been necessary.
In terms of the quality of the solutions, the algorithm does not guarantee the achievement of the global optimum: the quality of the final solution largely depends on the initial set of groups and can, in practice, obtain a solution much worse than the overall optimum. Since the algorithm is usually extremely fast, it is possible to apply it several times and choose the most satisfactory solution among those produced.

3 Density-based clustering

The application and execution of the DBSCAN algorithm was a very complex part due to the definition itself of density-based clustering. In fact, identifying dense areas in this dataset was not easy, because densities in the space were various and not uniform. For this reason, although the results of the algorithm are satisfactory, we argue that this is not the best clustering algorithm for this kind of data.

3.1 Parameters tuning and obtained clusters

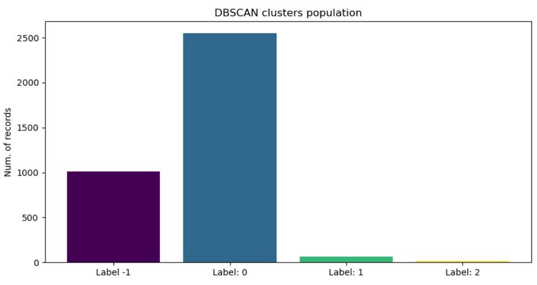
This phase was the longest and most delicate: we carried out many tests in order to decide which was the best combination of values for parameters *eps* and *number of neighbors*, with the aid of the elbow (or knee) method. We first fixed the first parameter and derived the second as a function of it, and then vice versa, in order to have a more stable and precise estimation. After many runs of the algorithm, we established what was the optimal configuration of the two parameters also by computing and analyzing other interesting and influential factors, such as: the amount of noise detected, the number of clusters found and the population of each one.



Collecting and storing all these steps in a summarizing table was really useful to keep track of the process and to notice how even the slightest change in values can change the configuration of the results.

As a final result we considered it appropriate the following combination of values: 10 for the number of neighbors and 0.11 for the *eps* parameter.

We report here, as an example of data visualization for DBSCAN clusters, the scatter plots derived from the correlation between ...



It is clearly visible that the number of clusters detected is 4, having strong differences in the number of populations. In the following section we are going to analyse them in order to interpret them and understand how the clustering process has been performed (i.e. what criterion has been followed to group different records). We anticipate that customers have been grouped in different clusters on the basis of their **spending behavior.**

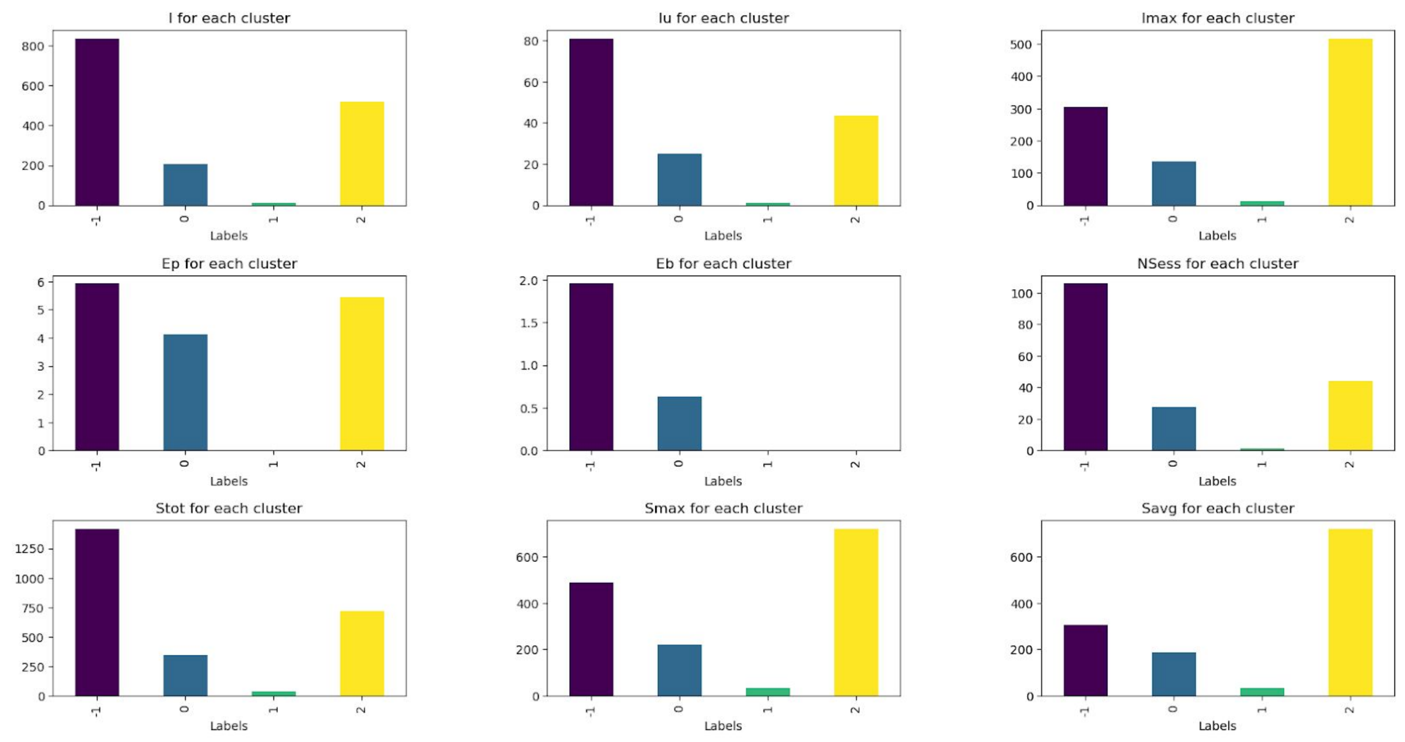
3.2 Post-processing analysis: DBSCAN clustering interpretation

"The validation of clustering structures is the most difficult and frustrating part of cluster analysis. Without a strong effort in this direction, cluster analysis will remain a black art accessible only to those true believers who have experience and great courage."

- Algorithms for Clustering Data, *Jain and Dubes*

By plotting the indicators distributions among all the clusters, we were able to perform a really exhaustive analysis and an in-depth study of cluster's composition.

In the barplots below you can see a first overview of what we discovered about the profile of the customers constituting each group, better explained and summarized in the table below.



Cluster	Customer type	Behavior
-1 (purple)	High-spending	They spent a lot by doing many shopping sessions. They buy different kinds of products.
0 (blue)	Medium-spending	Their expenses are average and both the number and the types of products they bought are average.
1 (green)	Low-spending	They spend a little, they buy very few types and quantities of products. They make very few shopping sessions.
2 (yellow)	Wholesale (grossisti)	They spent the most and bought the highest number of items. They buy different kind of products but in very few shopping sessions.

3.3 Additional evaluation measures

Unsupervised measures of cluster validity are often further divided into two classes:

- measures of cluster cohesion (compactness, tightness), which determine how closely related the objects in a cluster are
- measures of cluster separation (isolation), which determine how distinct or well-separated a cluster is from other clusters.

Unsupervised measures are often called internal indices because they use only information present in the data set (*Introduction to Data Mining (Second Edition)*, chapter 7.5: Cluster Evaluation, p. 573).

Silhouette and separation scores

The silhouette score describes not only the cohesion but also separation between clusters (it is able to capture if clusters are well separated or not). A value of silhouette equal to 1 means that there is a good cohesion and separation between clusters.

Notice that if there actually are subclusters that are not separated very well from one another, the silhouette score may provide only a coarse view of the cluster structure of the data.

- Silhouette score: 0.188
- Separation score: 0.900

The result shows that the two obtained clusters are well-separated, but not so cohesive.

It must be also said that **none of the unsupervised cluster validity measures does well for density-based clusters** (*Introduction to Data Mining (Second Edition)*, *General Comments on Unsupervised Cluster Evaluation Measures*, pp. 582-583).

We decide not to report here the evaluation of DBSCAN based on external indices and also the relative evaluation based on Entropy because they revealed not to be significant for the purpose of clusters interpretation. The same is for the clusters interpretation via categorical attributes.

3.4 Density based conclusions and final considerations

Points in low-density regions are classified as noise and omitted; thus, DBSCAN does not produce a complete clustering.

- Kumar Book, chapter 7, page 533

3.4.1 Two possible interpretations for noise in DBSCAN

In the previous analysis we have seen that the cluster identified as noise (-1 label) actually represents a different group with respect to shopping behavior: it represents those high-spending customers who, however, spread their purchase into many shopping sessions. So, since this seems to represent a proper cluster and not what we usually know as *noise*, we want to distinguish here two different perspectives by which noise labels in DBSCAN can be interpreted. Depending on these interpretations, both correct in our opinion, we can recognize two possible outcomes:

- *Interpretation 1*: DBSCAN assignes a -1 label to those records That we properly call *noise*, i.e. data points that are far from dense areas (so distant from other clusters).
- *Interpretation 2*: since DBSCAN is not able to recognize clusters with different densities, it cannot assign a separate cluster to the less-dense region and so it automatically labels that region with -1 : in this case, the meaning of this label is *not* representing the usual definition of *noise*, but just the **inability of DBSCAN to recognize two different clusters with different densities**.

In our case we are clearly in the perspective number 2: **since the less-dense area, reprensenting a cluster with a specific behavior, is difficult to be recognized properly as a cluster** (because of this intrinsic limitation of DBSCAN algorithm), it is still well distinguished and separated from the others, but **it has been given the *noise* label**. Coherently with this interpretation, -1 points are not what we properly call *noise*, but a less-dense cluster, and so it is the demonstration of this already known algorithmic limitation.

In support this hypothesis, please see the scatter plots in the data visualization section of the notebook (where it is easy to see the lower density of that area) and the interpretations in post-processing analysis section.

3.4.2 DBSCAN conclusions

Since "*density-based definition of a cluster is often employed when [...] noise and outliers are present*" (Kumar book, chapter 7, pp. 532-533), we argue that DBSCAN is **not** the best algorithm for clustering on this data. In fact we had previously performed a deep cleaning of the data itself, removing outliers and noise (see "Task1"). Therefore, the literature broadly supports that the optimal algorithm for a dataset like this one, already treated and cleaned up, is a contiguity-based approach. Confirming these theories we have in fact noticed that clustering with k-means returns the best results. This also perfectly explains why the DBSCAN algorithm has classified a specific group of customers (the high-spending ones) as noise (label -1).

Pros

- Deciding *a priori* the number of clusters is not needed: the algorithm is able to discover them by itself.
- Robust wrt noise: it detects noisy points and assigns them a -1 label (instead in K-Means they're assigned to the closest centroid)

- Is able to find clusters with any shape (not only globular / spherical shape as K-Means does), because there is no notion of *centroids* here.

Cons

Density-based clustering methods are somewhat **harder to tune** compared to parametric clustering methods like K-Means. Things like the epsilon parameter for DBSCAN are less intuitive to reason about compared to the number of clusters parameter for K-Means, so it's more difficult to pick good initial parameter values for these algorithms.

As a matter of fact, we had to do a lot of executions in order to tune parameters in a sufficiently good way (see above *Parameter tuning*).

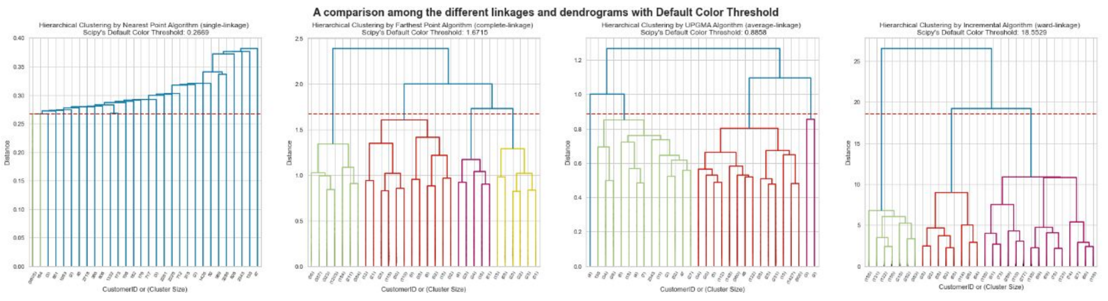
Another limit of DBSCAN algorithm is that it is not **able to recognize different densities**: "*DBSCAN can have trouble with density if the density of clusters varies widely*" (Kumar book, chapter 7, page 569). From the scatter plots above it is easy to see that DBSCAN does not work very well in a varying-density environment. For this reason we had to distinguish between two possible noise interpretations.

4 Hierarchical Clustering

Regarding the hierarchical clustering of the records contained in the dataset, the dendrograms of the four main linkage types (single, complete, average and ward) were plotted and analyzed. The main features of the aforementioned linkages were discussed on the notebook in relation to the expectations regarding the results from the clustering process.

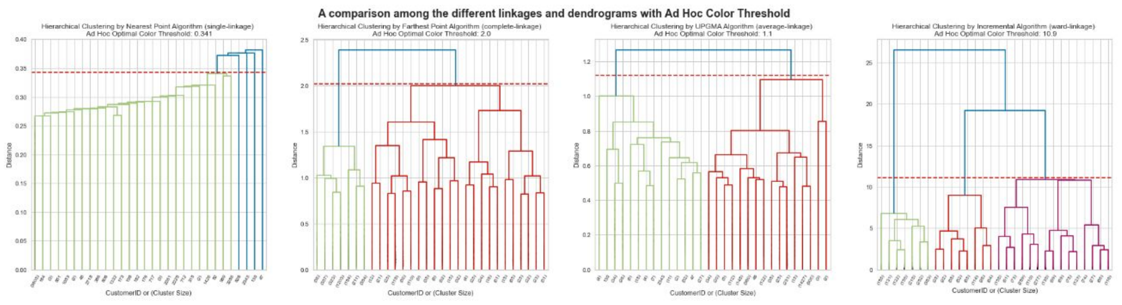
The first set of dendrograms was stained using the default value of the color_threshold parameter of the Scipy dendrogram function. This provided some insights into the main clusters within the dataset but, as we demonstrated shortly after, the default height calculation performed by color_threshold did not always coincide with the height of the "best cut" of the dendrograms, a cut that passes from the longest vertical segment not interrupted by horizontal lines.

These are the dendrograms from the aforementioned first set:



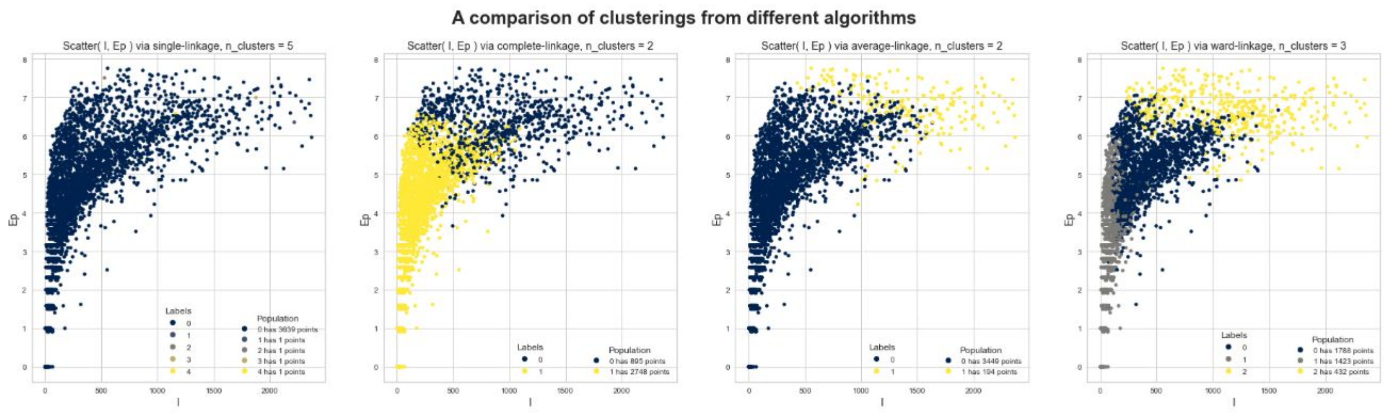
We then defined a function to identify the height of the optimal cut within each dendrogram. By iteratively cutting the tree at a height increased at each step. Each cut returned the given number of clusters identified by the structure under the cut itself and, by calculating the value for the number of clusters suggested by the greater number of cuts, we identified the longest uninterrupted vertical segment of the dendrogram.

These are the dendrograms from the second set displaying their optimal cuts:



While the dendrogram from the ward linkage with its three main clusters seemed mostly unaltered by the new cuts, the ones from the average and complete linakes both highlighted just two main clusters below their cuts.

The clusterings for each of these linkages were computed by using their optimal cuts as indices of the ideal number of clusters. These are the visual representations of the clusters obtained with each linkage.



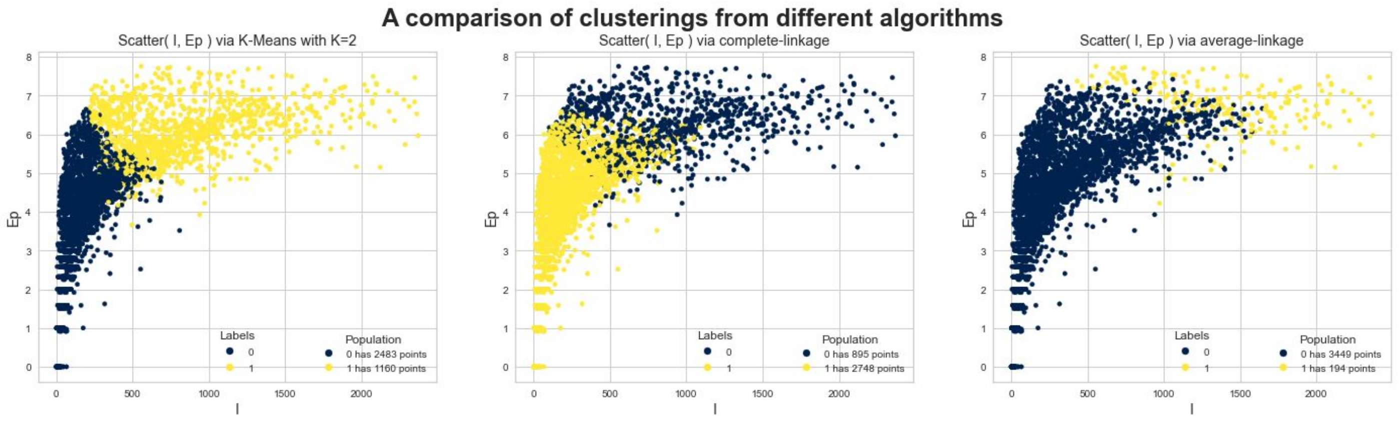
Each of the clustering were then evaluated via both well-known internal indices and the Cophenetic Correlation Coefficient.

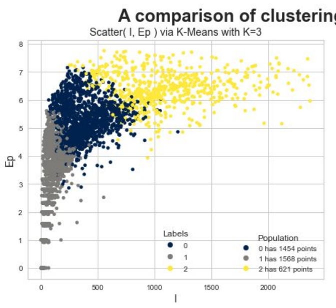
According to these metrics, the clusters from the average linkage (the second from the left in each picture) should be the best fitting in terms of performance and affinity with the dataset.

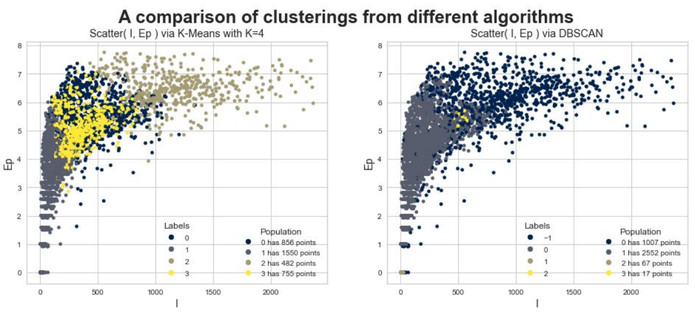
5 Clustering algorithms comparison

For all the DBSCAN's limitations and difficulties of interpretation, we state that the best algorithm to apply in this context and for this data, in order to have a clean, meaningful and clearly readable clustering result, is k-means algorithm.

Here we present a visual comparison between different clustering algorithms with, respectively, 2, 3 and 4 clusters.







6 Task 3: Data Classification

The aim is to perform a predictive analysis in order to classify our customers in three categories (high-spending, low-spending and medium-spending) on the basis of their shopping behaviour.

We run, compared and discussed the performance of **10 different models**: K-Nearest Neighbors, Radius-Neighbors, Gaussian Naive Bayes, Multinomial Naive Bayes, Support Vector Machine, Feed-forward Neural Network, Multi-layer Perceptron, Decision Tree, Random forest and Voting classifier.

After having discretized the categorical features we moved on to the definition and construction of the label.

We have chosen the **Savg attribute** as a starting point for the **construction of the label**, because it is the **most representative feature of the customers' shopping profile**: in fact it is computed on the basis of the average amount spent by each customer during a shopping session. This means that the relevance of said attribute remains significant, w.r.t. to the classification task, to both long-standing customers and brand-new ones without any prior interaction with the business entity.

Then we defined the three classes in the following way, using quantiles:

- If Savg < 0.33 → low-spending customer
- If Savg > 0.66 → high-spending customer
- If 0.33 < Savg < 0.66 → medium-spending customer

This division provides the most balanced distribution of the customers among the three classes. By doing so we also avoid incrementing the relevance (or representation) of a particular class with respect to the others. This allowed us to avoid using *class weights* as parameters for our models.

After dividing the available data into train and test sets, we defined and fitted all the models, and performed a prediction both on train and test set in order to be sure to avoid overfitting situations.

Where possible, we performed a grid search to find the best parameter configuration, and, where necessary, data preprocessing (MinMax normalization).

We then moved on to different steps for the evaluation.

For the **evaluation** of each single model we decided to present:

- Relevant **metrics** to measure the performance: accuracy, precision, recall, ...
- The plot of the **decision boundaries** drown by the classifier: in fact, classification can be viewed as the task of separating classes in feature space (i.e. find a hyperplane that separates the data).
- A multiclass **confusion matrix**: it is basically a table in which model predictions and actual data are represented in rows and columns. It makes it easy to visualize the model's classification errors and whether the predictions are more or less correct: our aim is to use it for the evaluation of each classifier and to determine how accurate and effective it is.
- The **scatter plots** between two indicators to highlight differences between real data and predicted ones. We choose two random indicators
- A plot for showing the **misclassifications** (in order to clearly display how many errors the model made).
- The **ROC curve**

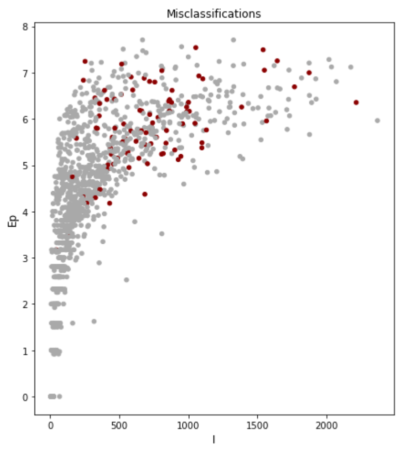
6.1 Classification results

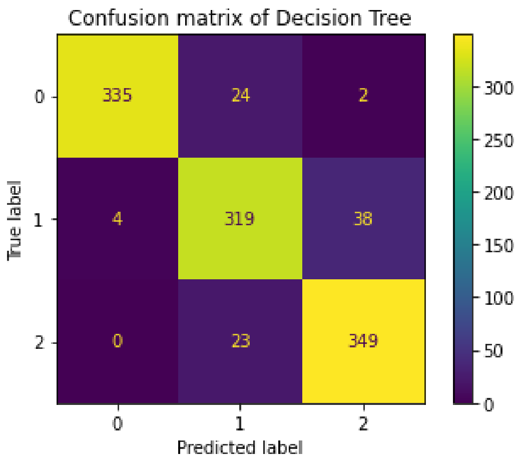
We decided to report here the classification results just for the top-4 models. First, in order to understand which of these are the best, we use the following summarizing table of the accuracy metrics (in descending order). We highlight the best 4 classifiers in green.

Model	**Accuracy**
Decision Tree	0.916
Multi-layer perceptron	0.869
Voting	0.856
K-Nearest Neighbors	0.838
Feed-forward Neural Network	0.783
Random Forest	0.758
SVM	0.756
Radious-Neightbor	0.733
Gaussian Naive Bayes	0.606
Multinomial Naive Bayes	0.595

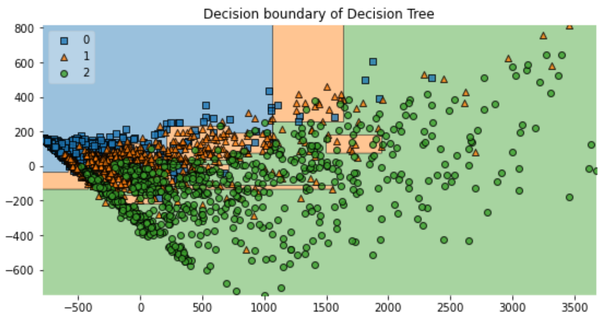
6.1.1 Decision Tree

The accuracy of this model is optimal, both for the training and for the test set. The misclassified points are very few. This leads us to argue that the Decision Tree is one of the best models for our data. There are very few errors in class attribution for this model. The decision boundaries traced by this model for our data are geometric and so not-curvilinear: besides their shape, they seem to be quite precise and accurate.





	0	1	2
0	335	24	2
1	4	319	38
2	0	23	349

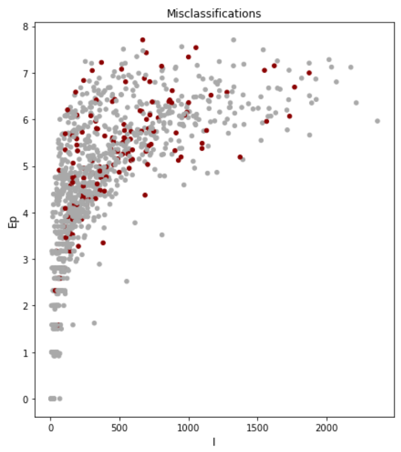


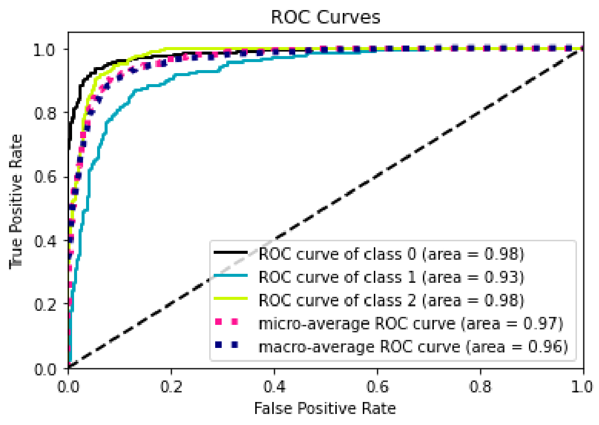
6.1.2 Multi-layer perceptron

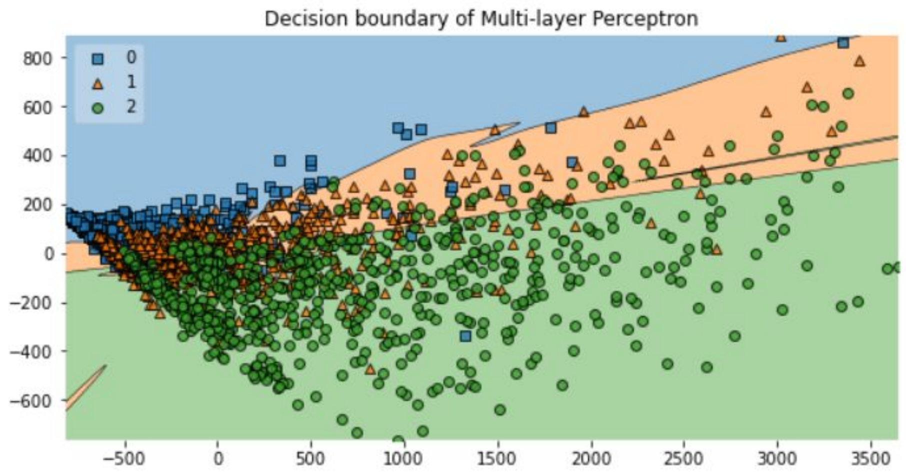
This model needed the normalization of both the training and the test set. For this reason, before running it, we applied a MinMax scaler preprocessing.

The misclassified points are very few: the fact that the prediction is really accurate confirms the power of this model. Also the hyperplane found by the model to divide the data-space into three categories seems quite good.

Finally, the ROC curve confirms the goodness of this classifier: the elbow, infact, is near 1 for the True Positive Rate (y axis) and this means that the False Positives and the False Negatives predicted are very few.



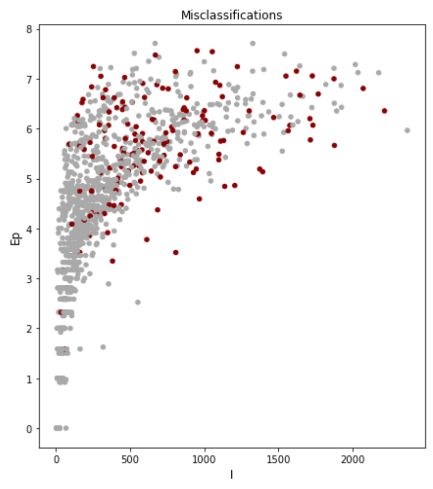


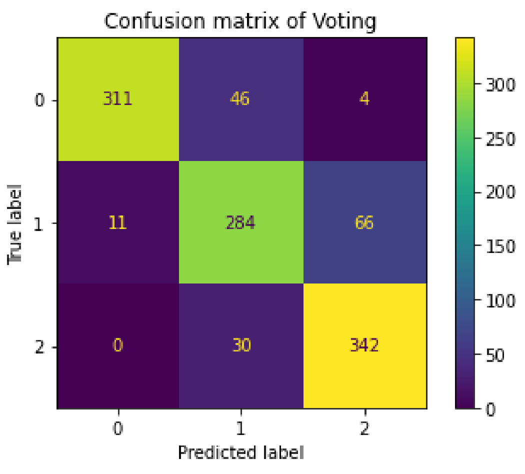


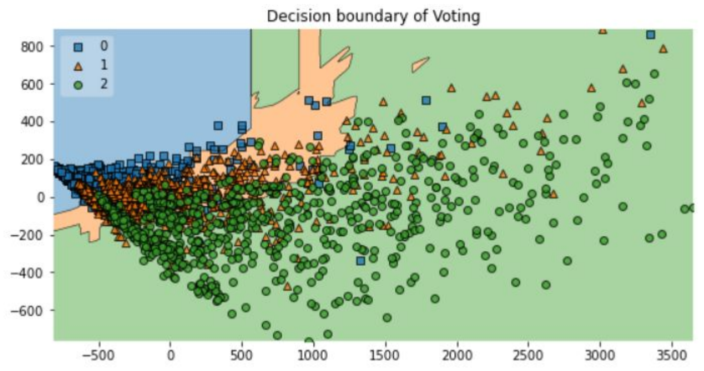
6.1.3 Voting

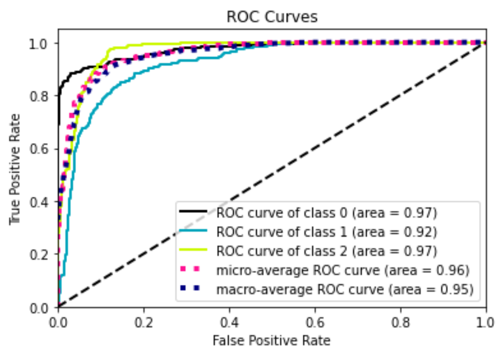
The idea behind the Voting Classifier is to combine conceptually different machine learning classifiers and use the average predicted probabilities (soft vote) to predict the class labels. Such a classifier can be useful for a set of equally well performing models in order to balance out their individual weaknesses.

For this reason we considered this algorithm very interesting in the context of ensemble methods and also new (compared to the algorithms we have seen in class), and so we decided to include it in our analysis.





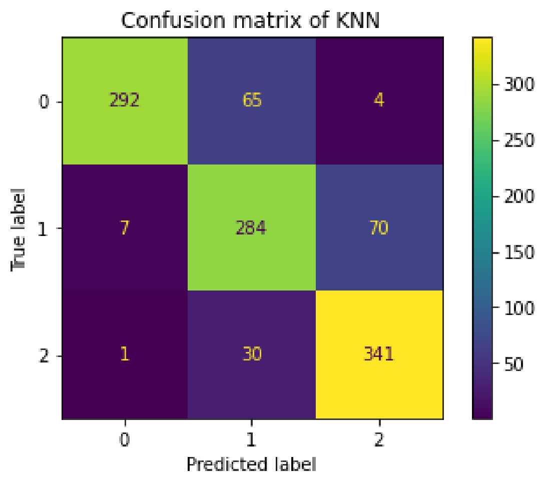


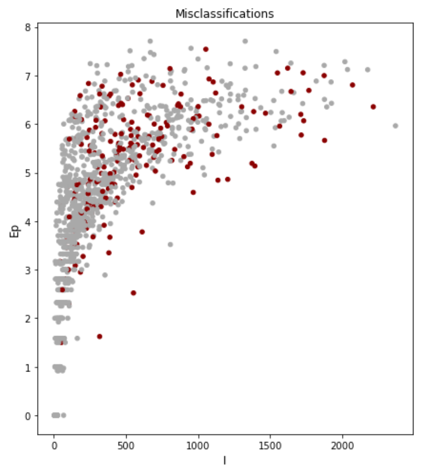


Misclassified points as well as errors in the arribution of the class are very few w.r.t. other classifiers.

The fact that the decision boundaries for this model have both straight and curvilinear lines could result from the fact that it is a combination of other models. However, the hyperplanes seem to accurately divide the data into the three classes.

6.1.4 K-Nearest Neighbors

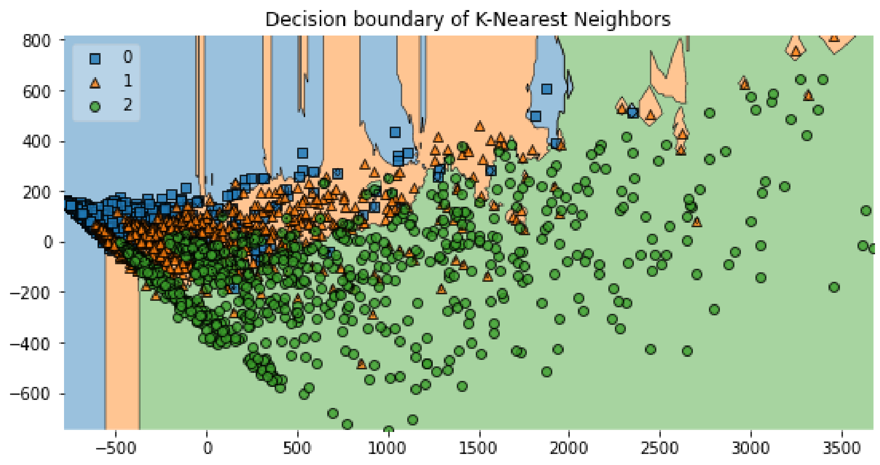




This model is quite accurate as it did not make excessive errors in the prediction.

The class that reports the major number of errors is that of medium spending, erroneously classified most often as high spending.

The decision boundaries drawn by this model for our data are very precise: we will see, by comparing this plot with those of the following models, that the hyperplane found by knn to separate data into the three classes is one of the best.



6.2 Alternative version of Task 3: classification based on clustering results

In the notebook we propose an alternative classification having the label based on the clustering result of with DBSCAN. In fact, given that this algorithm had already identified clusters of clients grouped according to their shopping behavior during Task2, we believe that it may be interesting to present this second analysis and show a comparison between the two approaches. This second analysis has been performed on all the ten proposed classifiers.

We have modeled the labels according to the cluster they belong to, merging together the high-spending cluster and the wholesalers cluster, in order to guarantee three classes as output. We ran all 10 classifiers for this new label.

In short, the results are far from good: too many misclassified points, too many errors in class assignment, and rough decision boundaries suggest that the result of clustering is not sufficiently complete and exhaustive to be able to base on it all the classification task.

Only three models over a total of ten performed well: Decision Tree, Random Forest and Voting (by the way, the more powerful ones). Surprisingly, also the Gaussian Naive Bayes, which is a very simple model, performed quite well.

We also believe that the **unbalanced support** between the three classes **had a huge impact** on these results. In fact, the population of the three classes was not equally distributed, as already observed in the "DBSCAN evaluation" section. In that section, in fact, it is clearly visible that low-spending customers and wholesalers are little populated groups.

So, even if some models performed really well in this second version, the errors of the other models are too high to be able to base the entire classification task on this label deriving from clustering.

This confirms that our choice to base the classification on the Savg attribute was correct.

7 Task 4: Sequential Pattern mining

For this task we extracted from the original dataset a structure containing the list of customers and the sequences of products (represented by their ID) purchased by them. In order to speed up the computation and avoid the analysis of the behaviour of irrelevant customers, we excluded from this structure every customer who had performed only one shopping session.

The a priori function was used for identifying patterns with a minimum support equal to different percentages of the dataset population. Searches for 5, 10, 15 and 20 percent have paid off, returning a substantial series of sequences.

The itemsets obtained were then manipulated through a function built for the occasion. The ProdID of each product has been replaced with the respective product description extracted from the original dataset. This simplified the process of analyzing the sequences in question.

As demonstrated by the very large presence of sequences composed of the same or similar articles , we suspect the correlation of this phenomenon to a habitual purchase of the products in question.

Some possible explanations for this trend are:

- the "disposable" nature of the products in question and therefore the need for customers to repurchase them continuously over time;
- the nature of wholesalers or retailers of the customers themselves who could therefore repeatedly purchase the products in question from the commercial entity under analysis in order to resell them to any third parties.

Some attempts were made in order to tackle the optional task of including time constraints within the identification of general sequential patterns. The previous structure was modified in order to associate each sequence of products with the date of the basket that included them but, due to the proximity of the deadline, no further actions other than these were made.