Business case
Alkemy x LUISS

Fabiana Caccavale, Matteo Gioia, Martina Manno, Vincenzo Junior Striano

Big Data and Smart Data Analytics - A.Y. 2022/2023

Contents

1 Task 0: Getting comfortable with the data

1.1 Introduction & Methodology

The business case is about an e-commerce firm specialized in consumer electronics which sells its products both online and offline. The purpose of the case is to analyse some internal and external dynamics in order to optimize the company pricing strategy. The main goals to be achieved are:

- *Is there a First Mover in setting the price?* Understanding the pricing leaders and followers among the players in the market. This implies a deep analysis on the competitors pricing strategy and the detection of useful historical patterns and correlations.
- *What factors influence the sale of a product or product category?* Understanding whether sales are influenced by seasonality effects and extract the popularity index of products to activate targeted marketing campaigns.

To achieve these results, a database is provided, which will be analyzed below. The datasets available are as follows:

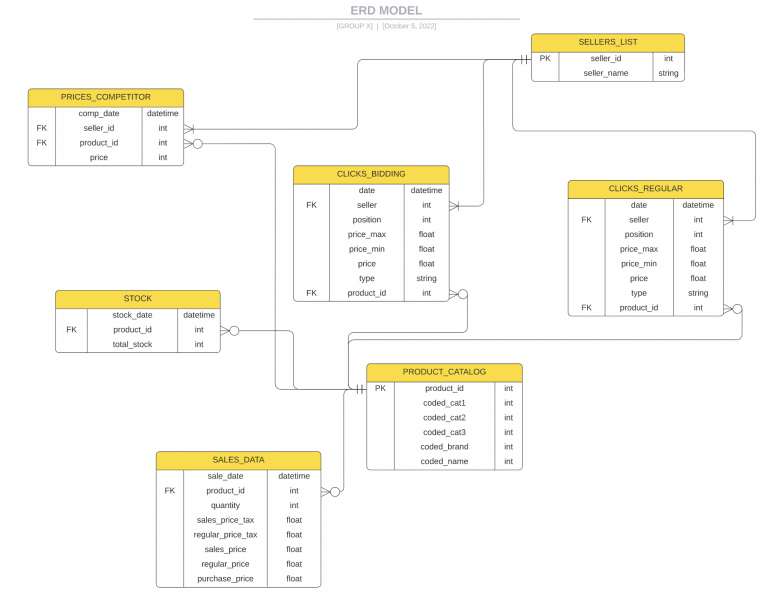
- clicks_bidding.csv
- clicks_regular.csv
- prices_competitor.csv
- product_catalog.csv
- sales_data.csv
- sellers_list.csv
- stock.csv

The aim of *Task 0* is familiarize with the data, merging them when possible, clean them, and evaluate the business requirements.

1.1.1 Data Integration

In order to be able to merge the datasets and gain an in-depth understanding of the relationships between the variables, an ER diagram was designed with the online tool Lucidchart. The ERD model shows what the entities are and the relationships between them. It was the starting point for understanding how the datasets could be merged.
For example, in *product_catalog*, the product_id is the primary key, or the unique identifier for a product. Whereas, in *sales_data* the product_id is a foreign key. What are the relationships?

- How many unique *product_id*'s have been sold? Zero to many
- How many unique *product_id*'s are in a single sales observation? One and only one



The preliminary analysis is realized using SQL implemented in Python, thanks to the *sqlite3* library. That library realizes a database, a collection of the dataset provided, and let interact with them using the SQL queries syntax. An example is provided below:

```
SELECT c.date, c.seller, c.position, c.price_max, c.price_min, c.price, c.type,
 c.product_id, p.price
FROM concatenato AS c
LEFT JOIN prices AS p
ON DATE(c.date) = p.comp_date
AND c.seller = p.seller_id
AND c.product_id = p.product_id
WHERE c.price IS NULL AND p.price IS NOT NULL
```

The datasets, the sqlite3 database creation python script, the data visualization script, and the main queries are stored into a shared, cloud-based virtual environment on replit.com for real-time edits. The next step will be to upload the major releases on a GitHub.

1.1.2 Data Visualization

To gain further insights and find confirmation about the findings, the Python library *plotly.express* was used for plotting the most interesting findings. Plotly was chosen thanks to its highly customizable and interactive plots.

The following sections report the results of the analysis for each single dataset.

1.2 Prices competitors

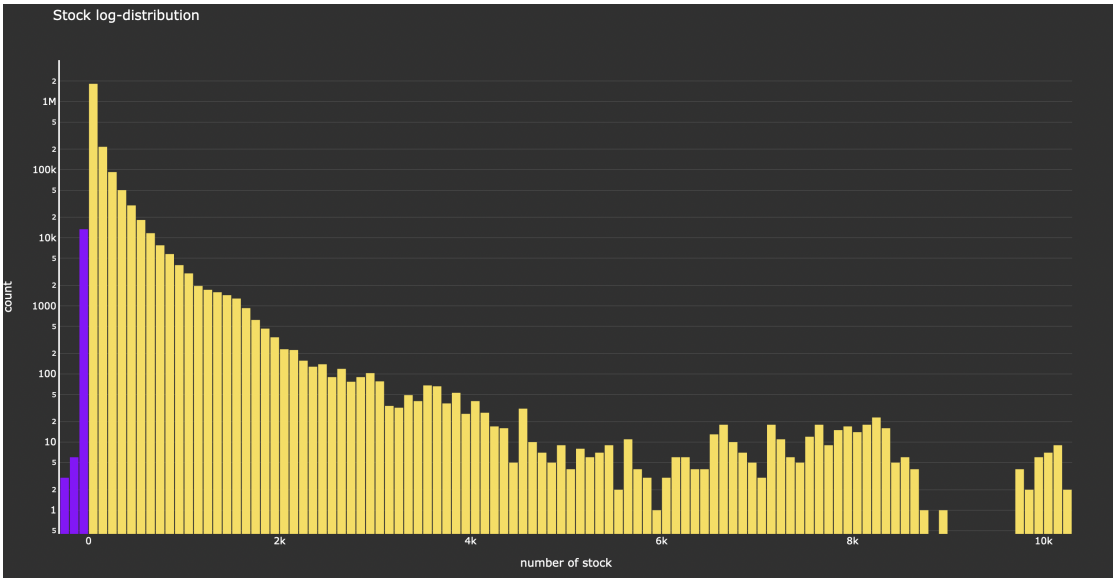
Prices_competitor contains all the prices for a specific product, for a specific seller and a specific date.
The dataset does not contain null values. The unique products_id are 6461 and there is no data on the remaining 1068 which are instead present in the *product_catalog*.

On average, for each day, there are records for 5393 unique products.

Also, there is a match between the number of sellers in the present dataset and the ones contained in the *sellers_list*.
It would be interesting to analyse the prices distribution for each seller and for each product over time in order to identify insightful patterns.

1.3 Stock

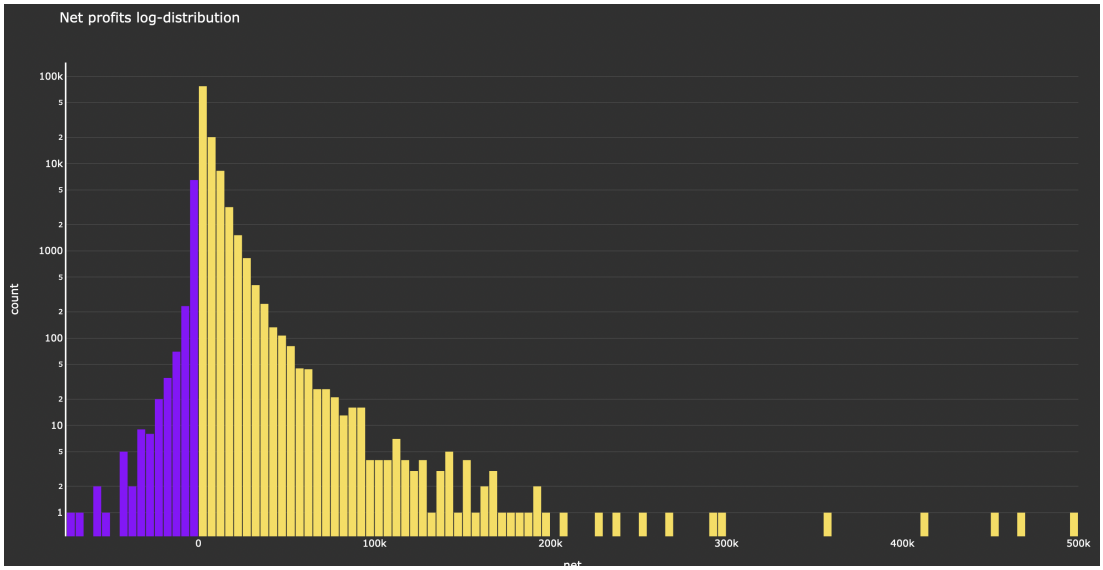
Stock contains the total stock for a specific product and for a specific date.
Stock values relate to a time interval that goes from 1 January 2021 to 31 December 2021.
Also in this dataset, there are no null values. There are stock data for all the products in the catalog.
On average, for each day, there are 6273 unique products whose stock value change.
The 75% of the total_stock is less or equal to 75 (3rd percentile), while the maximum stock value is 10247. The overall distribution of the total stock is right skewed, as can be seen in the plot below.
Additionally, the minimum value of the total_stock is negative, which is unexpected. This leads to a more in-depth analysis in relation to sales.
As a matter of fact, by joining *stock* with *sales_data* on date and product_id, it is possible to notice that there are 3818 cases in which a certain quantity of a product is sold even if, in the date of sale, the stock is less than that quantity.
However, it could be possible that in these cases the product is restocked in the coming days and it would not be a problem for the company to sell and deliver them on time.



1.4 Sales

Sales_data tracks all the purchases made by customers on the e-commerce website.
The dataset is complete, there are no missing values. Each row corresponds to a single transaction and provides information about the date of the purchases, the product id number, the number of products purchased, the sales price with and without taxation, the regular price, and the purchased price.
The time period runs from January 2 to December 31, 2021.
The product_id is a 6-digit unique identifier of the product and there are 7529 unique products that match with the value in the *product_catalog*.
The product quantity range for each transaction goes from 1 to 70.
The sales_price_tax is the final sales price for a single unit of the product which is affected by discounts and taxation (unlike the sales_price). Both the sales prices are at least equal to 1 up to 453303.
The regular_price_tax is the list price for a single unit affected by taxation, and it is at least equal to 1. Whereas the regular_price without tax has a minimum value of 0.833.
The regular price and the sales price are different if there are promotions or marketing campaigns for that product at that time. Hence, there is an inconsistency because the minimum list price

is lower than the minimum price after a promotion is applied. The purchase_price is the cost at which a single unit of product is purchased.
After understanding the data provided, the taxation analysis was performed. The first step is to compute the difference between the price with taxation and without for both the sales and regular prices. Then, the difference between the taxes is computed and points out that the average of both taxes is 20%, with a maximum value of 20% and a standard deviation equal to 0.35. The focus is on the minimum value of taxation that is equal to 0. This data seems to be inconsistent since even primary goods should be taxed.
Later, analyses were computed at the business level. Revenues were calculated by multiplying between sales_price_tax and quantity for each transaction. Then, the profits were calculated as the difference between the revenues and the purchased_price.



1.5 Product catalog

Product_catalog contains all the information related to the features of a product.
The dataset does not contain missing values and shows that there are 7529 unique products, exactly as in the sales dataset.
The attribute product_id is considered a primary key. This implies that the dataset can be joined with all of the others containing product_id in order to add information about the products' categories.
In particular, each product belongs to 3 categories, which correspond to sub-categories of products on the site from the most general to the most detailed one. Instead of having category and brand names, there are codes. Products belonging to the same category 3 are substitute products.
The data are consistent with expectations because there are 12 unique categories in coded_cat1, 60 unique categories in coded_cat2, and 238 unique categories in coded_cat3. Furthermore, there are 292 brands.

1.6 Sellers list

Sellers_list contains the primary key for sellers, namely seller_id.
The dataset does not contain missing values and it is consistent because there are 9 unique sellers, each associated with a letter.
The e-commerce has as seller_id the number 24.

1.7 Clicks

Some data are available from a price comparator website specific for consumer electronics. The customer selects the product that wants to buy and the site shows different sellers of that same product with respect to 2 different sections:

- the bidding section in which there are the 3 sellers who paid to be at the top of the page,
- the regular section in which there are other 10 sellers in decreasing price order.

Clicks_regular and *clicks_bidding* have the same columns but a different time span. Bidding contains observations from 02-04-2021 until 04-01-2022 while *clicks_regular* contains observations from 01-01-2020 until 04-01-2022. These 2 datasets show the position of products shown in the comparator for each date. A concatenation between them allows creating a chronological record of all clicks. Some data cleaning and transformation must be performed before merging.

1.8 Clicks Regular

Clicks_regular contains observations regarding 6457 distinct product_id entries from 9 distinct sellers.
The position contains 51 distinct values from which 2 must be subtracted **(to be confirmed)**, which are position 0.0 and the position of null values. An interesting insight is that the positions clicked range from 1 to 32, the 104-106-109 and then move on to positions 1010 and later. It contains 1093664 null values for 8 out of 9 seller_id's, seller_id 490 has no null positions.
Moreover, there are 545 observations in which the position is 0.0 and 124 of these also show price, price_min and price_max equal to 0. They represent a negligible amount relating to the length of the dataset. We couldn't spot any pattern which could have explained this anomaly, so they may be only random. That said, we deleted them.
The price, instead, has 1093244 null values for 8 out of 9 seller_ids, seller_id 490 has no null prices. As for the position variable, the null values of price appear to be disconnected from the seller_id, to the date, and to the product_id. When price is null, the position is also null but the opposite relationship does not hold (i.e. there could be null positions but with price).
Since the data sources obtained from the comparator are: *prices_competitors*, *clicks_regular*, and *clicks_bidding*, some missing prices are inferred by joining the *price_competitors*. In particular, 87000 prices out of 1093244, 8% of them.
In order to check that the retrieved prices are accurate,the prices column of the *prices_competitor* are merged to the *clicks_regular* using the date, product, and seller as an index. The check is applied only to rows that do not contain null values. The result of this check reports that 90% of the prices have difference equal to 0. As a consequence, deriving the missing prices in *clicks_regular* by taking them from *prices_competitor* is a valid method.

1.9 Clicks Bidding

Clicks_bidding shows up sellers who paid to be at the top 3 position of the page. The position column accordingly contains values ranging from 1 to 3.
The dataset has no null values in any observation. The dataset contains 696999 observations, concerning 5797 distinct product_id inserted by 9 distinct sellers.

2 Task:1 Leaders, Followers and Pricing automation detection

The objective for task 1 was to determine, for each quarter and product sold, who was the market leader, the follower(s), which sellers used automation price algorithms and uncover price strategies for each seller against the competitors.
Alkemy provided the definition either for leaders and followers:
- A **leader** sets a price which then serves as a benchmark for the other market players [...]. It does not necessarily imply setting the best or the highest price but it is rather about setting trends which requires the competitors to react.
- A **follower** adjusts its own prices to the ones set by competitors. It means that a price change by a competitor seller is followed or reacted by corresponding actions
Given a quarter of 90 days, a catalog of 7529 unique products potentially sold by 9 sellers, one would have run into a problem of potentially more than 5 million unique data..[1]
Taking that in consideration, Alkemy suggested grouping the products into the 13 categories into which they are divided, reducing the dimensionality of the problem by 99.8%. [2]. This would have allowed the task to be solved by displaying only 13 graphs, one for each category, each showing, for each day of the quarter, the average price trend per category, set by each seller. From the visualization of the graphs, recurring patterns would have emerged.
According to our opinion, the aggregation of 7529 unique products into 13 categories, and thus the aggregation of their characteristics into 13 average characteristics, such as price, would have enabled the task to be completed in a short time, but neglected critical information, peculiar to each product, and sacrificed the accuracy of the predictions.
The value added of this approach is that the task is solved by working in the detail of each product, without aggregating valuable information. The algorithms are easily applicable for each product and each quarter. To do this, the chosen approach is rule-based and not plot-based. If it turned out to be necessary, aggregation by categories could be implemented at any time. Viceversa, information on individual products could not be derived by drawing conclusions on more general categories.
Therefore, abstract and general rules were setted and included in softcoded algorithms, whose parameters were the reference quarter and the code of the product to be analyzed.

2.1 Leaders and Followers detection

Leader and follower analysis is very important to understand market dynamics, because before taking a price decision, it is important to know which role is playing each competitor in the market, in particular when a seller decides to set the prices based on the competitors ones.

2.1.1 Correlation and Cointegration

The idea is to predict the price movement of followers basing on the price movement of leaders. The assumption is that followers will change the price with a lag of 1 day.
So, for each seller, the correlation is computed between every other seller in an equal dataset, but shifted by one row (since the assumption is that when there is an increase in the prices of seller X, the prices of seller Y will follow on the next day).

Original dataset:

[1] 7529 * 90 * 9.
[2] ((13 * 9 * 90) - 5mln) / 5mln * 100

seller_id	23	24	26	41	48	180	407	490
comp_date								
2021-01-01	27990.0	27990.0	27990.0	27490.0	27990.0	28990.0	28590.0	NaN
2021-01-02	27990.0	27990.0	27990.0	27490.0	27990.0	28990.0	28590.0	NaN
2021-01-03	27990.0	27990.0	27990.0	27490.0	27990.0	28990.0	28590.0	NaN
2021-01-04	27990.0	27990.0	27990.0	27490.0	27990.0	28990.0	28590.0	NaN
2021-01-05	27990.0	27990.0	27990.0	27490.0	27990.0	28990.0	28590.0	NaN
...	...	...	...	...	...	...	...	...
2021-03-27	27990.0	28990.0	25769.0	26388.0	26091.0	27610.0	25989.0	28990.0
2021-03-28	27990.0	28990.0	25769.0	26388.0	26091.0	27610.0	25989.0	28990.0
2021-03-29	27990.0	25769.0	28990.0	26388.0	28990.0	27610.0	25989.0	28990.0
2021-03-30	27990.0	25769.0	28990.0	26388.0	28990.0	27579.5	26299.0	28990.0
2021-03-31	27990.0	25769.0	28990.0	26388.0	28990.0	27549.0	26299.0	28990.0

Shifted dataset:

seller_id	23	24	26	41	48	180	407	490
comp_date								
2021-01-01	27990.0	27990.0	27990.0	27490.0	27990.0	28990.0	28590.0	NaN
2021-01-02	27990.0	27990.0	27990.0	27490.0	27990.0	28990.0	28590.0	NaN
2021-01-03	27990.0	27990.0	27990.0	27490.0	27990.0	28990.0	28590.0	NaN
2021-01-04	27990.0	27990.0	27990.0	27490.0	27990.0	28990.0	28590.0	NaN
2021-01-05	27990.0	27990.0	27990.0	27490.0	27990.0	28990.0	28590.0	NaN
...	...	...	...	...	...	...	...	...
2021-03-27	27990.0	28990.0	25769.0	26388.0	26091.0	27610.0	25989.0	28990.0
2021-03-28	27990.0	25769.0	28990.0	26388.0	28990.0	27610.0	25989.0	28990.0
2021-03-29	27990.0	25769.0	28990.0	26388.0	28990.0	27579.5	26299.0	28990.0
2021-03-30	27990.0	25769.0	28990.0	26388.0	28990.0	27549.0	26299.0	28990.0
2021-03-31	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

If this correlation is higher than a certain threshold (e.g. 0.8), a possible pair of leader and follower is identified. Their relationship is better inspected by plotting the prices they set over time and by visually analyse what is their behavior.
For each leader-follower pair, the respective co-integration test's p-value is then computed. Co-integration test is used to describe some underlying relationship between variables by testing the correlation between two or more non-stationary time series in the long run or for a specified period. The test compares the null hypothesis of no cointegration against the alternative of cointegration. In other words, co-integration tells you that, although two series move independently, the average distance between them remains relatively constant.
The output of this method is a list of leaders and their respective followers for each product id and, for each sellers pair, the correlation score and the co-integration p-value are also shown.

	Leader	Follower	Pearson Correlation	Co-Integration p-value
0	41	180	0.732422	0.168988

However, for certain product ids it happens that for some seller pairs, both are leaders and followers of the other (see figure below). This could be explained by the fact that for some periods it is one seller who changes prices first, while for other periods it is the other.

Therefore, it should be detected who causes the other's prices to change the most. This can be determined by looking at the sellers' time series and comparing them. Obviously, it would not be efficient to plot each seller's prices for each product where this situation occurs, and so there is a need for an algorithm to determine this on its own.

2.1.2 Granger Causality

One way to understand who causes the prices variation is to perform the Granger Causality test. The Granger causality test is a statistical hypothesis test for determining whether one time series is useful in forecasting another and, therefore, in determining if a given time series and its lags is

	Leader	Follower	Pearson Correlation	Co-Integration p-value
0	26	24	0.849637	6.742450e-07
1	48	24	0.875885	8.102065e-03
2	24	26	0.742710	5.973242e-04
3	48	26	0.782319	1.548132e-08
4	24	48	0.710591	1.651746e-03
5	26	48	0.735388	9.017176e-08

helpful in explaining the value of another series. This test uses the following null and alternative hypotheses:

- Null Hypothesis (H0): Time series x does not Granger-cause time series y
- Alternative Hypothesis (HA): Time series x Granger-causes time series y

This test produces an F test statistic with a corresponding p-value. If the p-value is less than a certain significance level (i.e. alpha = .05), then we can reject the null hypothesis and conclude that we have sufficient evidence to say that time series x Granger-causes time series y.
This test is performed for each possible pair of sellers and, when the situation described above occurs, the p-values obtained when testing whether changes in seller A's prices cause changes in seller B's prices are compared with the p-values obtained in the opposite case (i.e., when testing whether it is seller B that influences seller A).
The test to which the lowest p-values correspond to (thus the most significant ones) will be the ones to determine who, in the analyzed sellers pair, is the leader. This will allow to obtain a dataset like the one below, in which there are no reversed pairs of leaders and followers.

	Leader	Follower
0	24	26
1	24	48
2	26	48
3	41	180

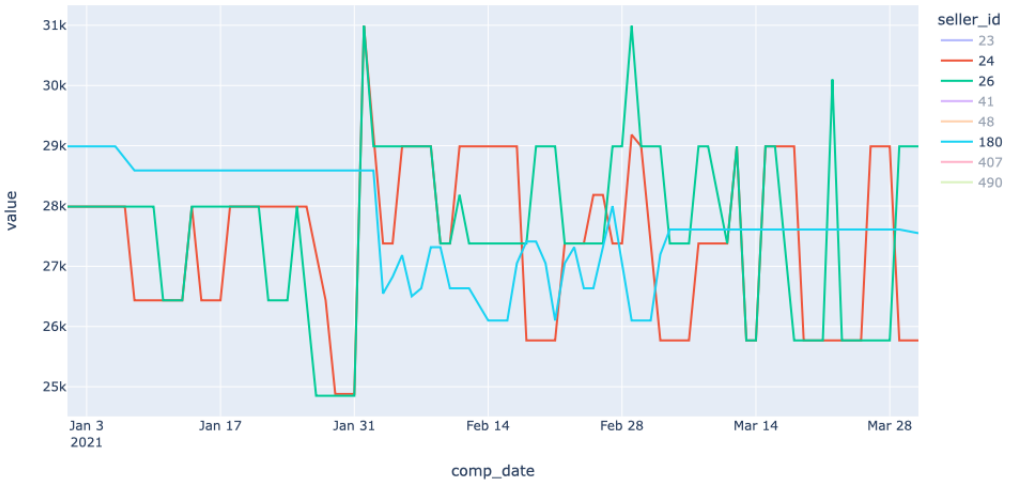
2.2 Pricing automation

The second requirement of Task 1 was to detect whether there is any competitor with an automated pricing system implemented.
With pricing automation, prices are determined automatically using a software system based on a predetermined set of rules or constraints that take into account competitors' pricing strategies and a company's market value.
In order to detect sellers in the market that use an automated pricing strategy, some possible rules that the algorithm they use could follow were identified. The ideas are as follows:
- The algorithm might set up maximum and minimum prices to ensure that the company is not only gaining sales, but also gaining the highest profit margins possible. Once set up, the repricer will continuously study competitors' behaviour to automatically configure prices to allow the best results for the company. This doesn't just mean setting products at the lowest price, but can also include an increase in price when competitors go out of stock, for instance.
- The algorithm might set a price that always deviates by a fixed percentage or value over time (e.g., a seller sets a price that is always 10% lower than the price charged by the market leader).

- The algorithm might set a new price after always the same time lag with respect to another seller (e.g., a seller always changes its price the day after the leader's price change).

2.2.1 Visual Analysis

It is possible to detect a seller's compliance with these rules by visually analysing the respective time series, and thus how the price it sets changes over time in relation to other sellers.



In this case, sellers 24 and 26 might use an automated pricing system since for certain periods they set a price that is not higher/lower than a fixed threshold. Moreover, by comparing their time series with seller 180's price movements, it seems that they change their pricing accordingly, after always the same time lag.
However, manual analysis is not only an impossible task, but also requires a lot of time and resources. That is why two algorithms were created which identify compliance with the last two rules listed above (the one related to the fixed percentage/value and the one related to the fixed time lag). Both follow the same basic logic, but each has its own peculiarity.

2.2.2 Identification of a constant deviation from another seller

In order to identify sellers that possibly use an algorithm that sets a price that always deviates by a fixed percentage or value over time, the price difference with all other sellers,for each seller, were computed and the results were stored in a new dataframe.
If a price set by a seller deviates by the same percentage/value from another every day, the differences of the prices should be constant over time, thus their standard deviations should be low.
Therefore, the thresholds are set and on these it is determined who might follow the percentage/-value rule:
- If the value in a cell is less than a certain threshold, it is assigned the label 'A' and the corresponding seller might use pricing automation systems.
- Otherwhise, it is assigned the label 'B'.
For instance, seller 23 sets prices that always deviate by a fixed percentage or value with respect to seller 41.

2.2.3 Identification of a constant time lag

The second algorithm designed aims at identifying sellers that possibly use an algorithm that sets a new price after always the same time lag with respect to another seller.

	23	24	26	41	48	180	407	490
23	A	B	B	A	B	B	B	A
24	B	A	B	B	B	B	B	B
26	B	B	A	B	B	B	B	B
41	A	B	B	A	B	A	B	A
48	B	B	B	B	A	B	B	B
180	B	B	B	A	B	A	B	A
407	B	B	B	B	B	B	A	A
490	A	B	B	A	B	A	A	A

In order to do that, for each seller, the days after which a certain seller changed price relative to another were computed. Again, to tell whether the time lag is constant over time, its standard deviation is considered.
Also here, the sellers indicated in the column to which label 'A' is assigned are the ones who might use a pricing algorithm that follows the time lag rule.

	24	26	41	48	180	407
24	0	B	B	B	B	B
26	B	0	B	B	B	B
41	B	B	0	B	B	B
48	B	A	B	0	B	B
180	B	B	B	B	0	B
407	B	B	B	B	B	0

For instance, seller 26 changes its prices after approximately the same time lag every time with respect to seller 48.

2.3 Maximum and Minimum Price Strategy

These functions aim to determine the behaviour of each seller with respect to the maximum price and with respect to the minimum price. Only the max_strategy function will be taken as an example, aimed at analyzing the behaviour of each seller with respect to the maximum price. All of the above can be applied specularly to the min_strategy function, whose reference is the minimum price. For each day, the percentage difference between each seller's price *(P of seller i at time t)* and the maximum price of the other sellers is calculated.

- If the difference is positive, then the seller will charge the highest price
- If the difference is zero, then the price charged by the seller is the highest one and there is at least one other seller with the same price.
- If the difference is negative, then the seller will charge a lower price than the maximum price.

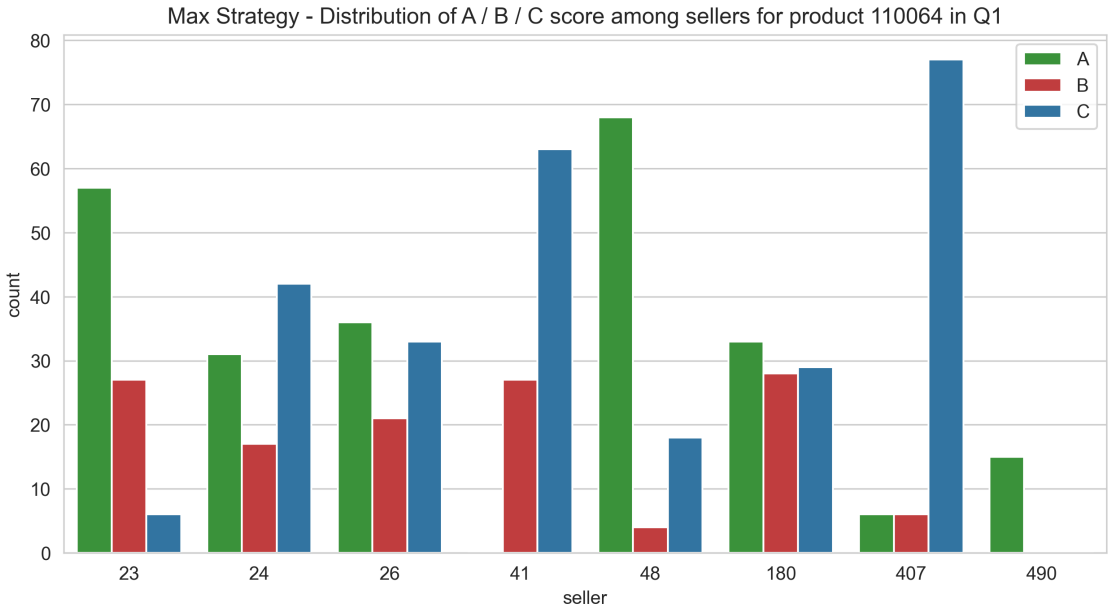
$$difference = \frac{P_{seller(i)t} - max(P_{seller(j \neq i)})}{P_{seller(i)t}}$$

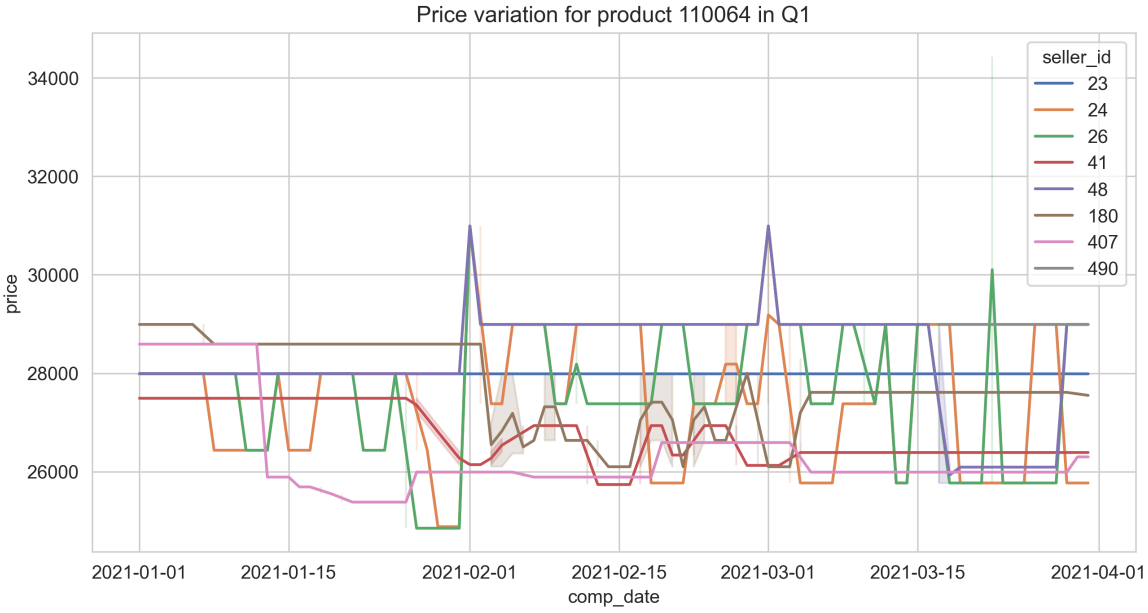
For this function, the focus is on the first scenario. For each day, the array of differences is transformed into ordinal scores, thanks to a rule based on the calculated quantiles of the array.

If the value of the array corresponding to seller Z belongs to the first quantile, then seller Z will score A, otherwise B or C if in the second or third.
By rephrasing, if seller Z charges a higher price than the others on day 1, then it will score A.
The algorithm is iterated for each day and a countplot is rendered showing how many times A, B and C scores are assigned for each seller.

Based on this approach, for the first quarter, for product 110064, the leader 48 is most frequently assigned the A label, as can be seen in the graph below. Therefore, we can conclude that leader 48, for the selected period and product, tends to set the highest price. proof of these conclusions is also found in the graph

The proof of these conclusions is also found from a graphical point of view: in the first quarter, up to 1 February the seller 48 places a price among the highest, from 1 February until mid-March it always places the price highest, and for the last 15 days only, it shows abnormal behaviour.





The min_strategy function is based on the same approach, but takes the minimum price as a reference. If the array value corresponding to seller Z belongs to the first quantile, then seller Z will score A, otherwise B or C if in the second or third. This rule, however, awards the score A if the seller tends to charge the minimum price.

For example, seller 407 tends to score A more frequently than the others, for the product and period selected above. Thus, it is possible to conclude that the leader 407 for the selected period and product, tends to set the minimum price.

Again, there is the visual confirmation in the price trend, as the graph above shows.

3 Task 2: Popularity Index and applications

The objective for task 2 was to determine, for each quarter and for the Black Friday timeframes, an index to give the customer an accurate picture of product popularity. The calculation of the popularity index is useful to help the company making future decisions including budget allocation, trend forecasting, inventory management, and more.

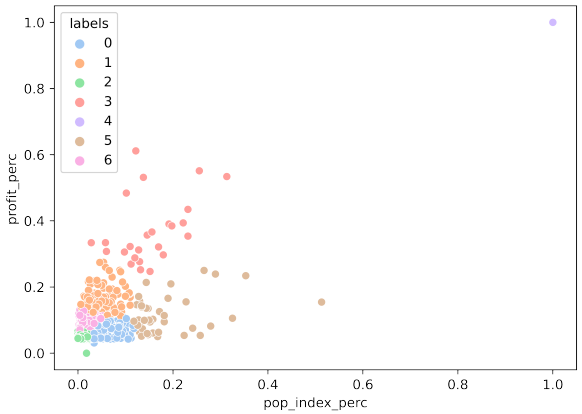
The analysis of this index, along with other measures of profitability, can help the company in evaluating certain products and, possibly, their presence in the catalog.

3.1 Sales Approach

This first approach calculates the popularity index for each quarter by dividing the quantities sold of each individual product against the total amount of products sold in that period. In this way you get a popularity index based on sales frequency.

For each product, it is visually analyzed how much it contributes to the company's profit, so as to target the same strategies to products with similar characteristics in terms of profitability and popularity.

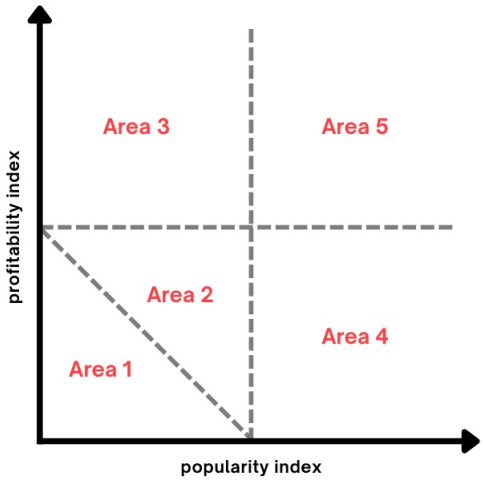
The identification of products with similar features is done through the k-means algorithm. The figure below shows the products sold in the first quarter of the year. Each point in the graph represents a product and the colors change depending on the cluster they belong to.



As it is shown, there are:

- lot of products that have low popularity and low profitability;
- few products with high popularity and high profitability;
- a discrete number of products with medium popularity and low profitability;
- a discrete number of products with low popularity and medium/high profitability.

Obviously, for each quarter the clusters formed are different, but the strategies, for clusters of products with similar characteristics, are the same. For example, assuming that products, based on their profitability and popularity, are divided into 5 categories, such strategies could be defined:



- Area 1: Area where products with poor popularity index and poor profitability fall. Not knowing in detail the state of the art we opt to suggest not to dedicate budget allocation for these products.
- Area 2: Products with an acceptable popularity index and profitability. These products are the ones that will require the most effort by the company in both monetary and management terms. Strategies will have to be adapted depending on the area in which you aim to want to move these products. It will be very difficult to take them directly to zone 5 i.e., the ideal zone. There will have to be an intermediate step where these products will be in zone 3 or 4 and only then in the ideal zone. If there is a lot of competitive supply, the strategy for the product will be to increase its popularity. Whereas in case of little supply, the strategy will be in increasing profitability.
- Area 3: Products with a high profitability but poor popularity. We propose a strategy that attempts to increase popularity by sacrificing some profit. One possible approach could be precisely to decrease the price and perhaps have it end with 9 (psychological pricing strategy). The most obvious consequence is an increase in sales and popularity of that product for our seller.
- Area 4: Products with high popularity but low profitability. The strategy that we think should be pursued is to increase its price with the release of coupons or bonuses for the buyer in return. The price increase could cause a disencent for consumers in proceeding to purchase, which, however, should be rebalanced or even boosted by the coupon campaign. Another possible effect of coupons, expendable only toward products advertised by seller 24, is to indirectly boost sales of other products from our seller.
- Area 5: Products that have a high profitability and popularity. This is the ideal situation that we also want to achieve with products in the other zones so we opt not to start strategies that could affect this already optimal situation.

3.2 Clicks Approach

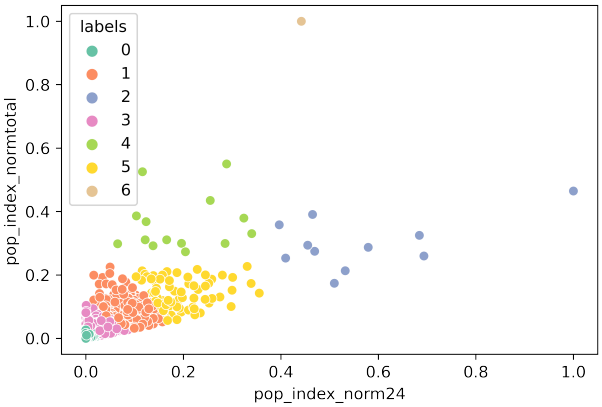
The sales approach might not give a clear perspective on the actual popularity of individual products with respect to the market. Therefore, it would be useful to compare the internal (i.e. that of seller 24) popularity of a certain product with that of the market. The only data available with which the latter can be determined is clicks data. With this approach, a product's popularity is defined by the number of clicks it receives.

Two products' popularity indexes are computed, with respect to:

- Clicks on products sold by seller 24: This index gives an idea of how popular a product is (based on the clicks it receives) compared to those sold exclusively by seller 24.
- Clicks on products sold by all sellers: This index measures the popularity of the product within the market, regardless of who is selling it.

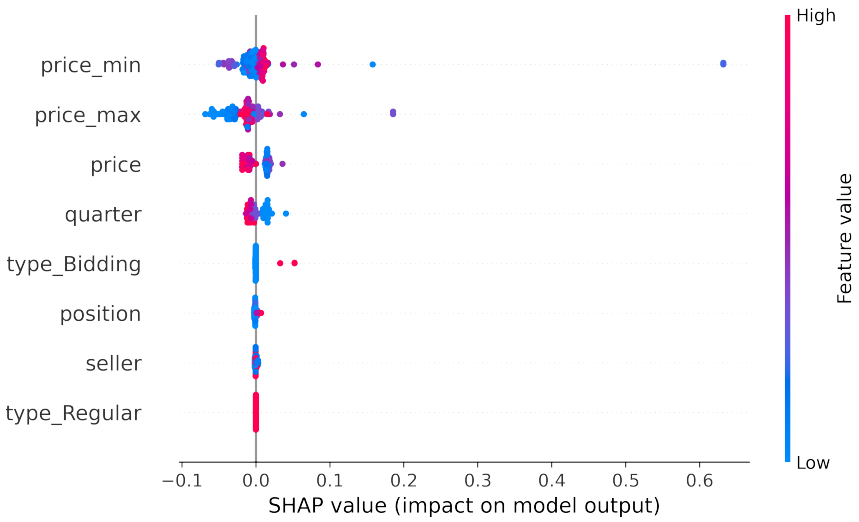
Again, a clustering algorithm is used to identify products with similar characteristics, and the products are analyzed visually.

The figure below takes the first quarter as a reference and shows how products rank with reference to popularity (in terms of click-through rate) in the market and internally.



Additionally, a machine learning model (XGBRegressor), appropriately tuned, was built in order to understand which variables most affect the products' popularity index.

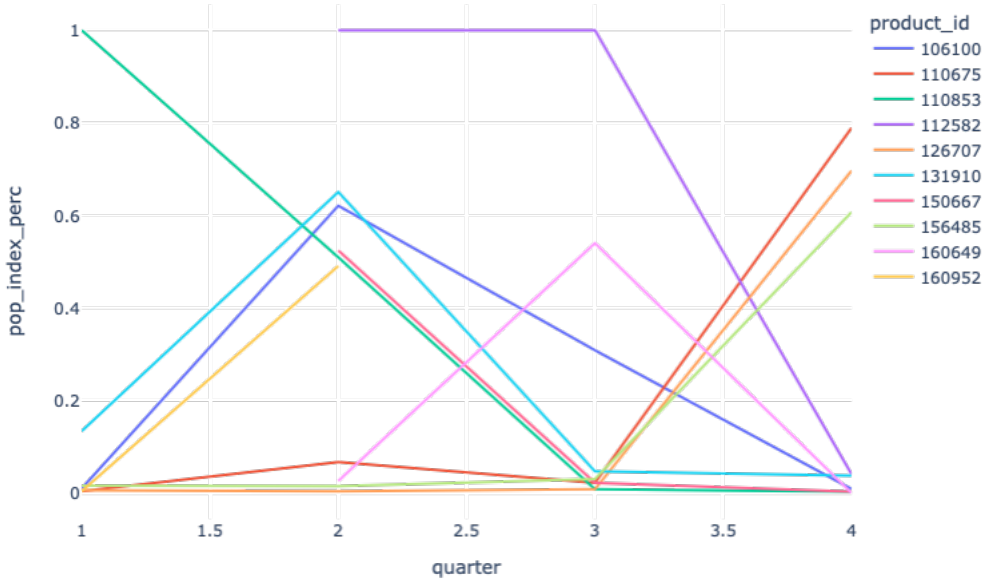
Competitor prices (minimum and maximum price in the market) are the ones that most influence how much a product is clicked. The seller does not turn out to be important in defining the popularity index for a given product. Therefore, it shows up that the lower (higher) the price of competitors, the lower (higher) the popularity index is. Unexpectedly, the position within the site does not turn out to be important in defining a product's popularity either. The quarter, on the other hand, slightly affects the target variable, showing that there are products with seasonality whose popularity changes over time.



Example of interpretation: For low values of *price_min*, the SHAP value (i.e. the popularity index) decreases. For high values it increases.

3.3 Additional insights

By looking at the popularity index detection through the sales approach, it is possible to extrapolate further insights. The popularity index was obtained by dividing the quantities sold of each individual product against the total amount of products sold in that quarter. From that, it is possible to compute the standard deviation of the popularity index of each product over quarters and then rank the top 10 products with the highest standard deviation. This is useful for highlighting products whose popularity index changes the most over time. The plot below shows a visual confirmation of the popularity index changing in the quarters of the year for the 10 products with higher standard deviation:



By looking at this graph, some interesting insights about the products behaviour can be detected: there are some products whose popularity index has more or less the same trend over quarters, such as the products number 131910 and 106100 which have a peak in the second quarter and then decrease. Another example are the products number 110675, 126707 and 156485 which popularity index increases with the same trend after the third quarter. Could they potentially be influenced by seasonality effects?