LUISS 'Guido Carli'
MSc in Data Science and Management - Data Science in Action Course
May 5, 2024

Crystalball

Understand the past to predict the future

Project Technical Report

Camerlengo, Vincenzo
ID: 773731, email: vincenzo.camerlengo@studenti.luiss.it (Group Delegate)

Paolantoni, Riccardo
ID: 773691, email: riccardo.paolantoni@studenti.luiss.it

Torelli, Raffaele
ID: 775831, email: raffaele.torelli@studenti.luiss.it

Unieuro's Business Case

1 Introduction

The goal of this project is to build a statistical model able to predict the sales units for each product featured in promotional flyers in November 2023 and December 2023. Unieuro provided us six datasets regarding the period January 2018 – October 2023 that contains information about: results and detail of previous promotional flyers, historical trends, products featured in the flyers, stock levels and Key Performance Indicators (KPI) related to the products.

As we will see in the next section, we started our "journey" by studying the data in order to get a clear overview. After the exploration, we focused on preparing the data for the next steps and then we focused on the model implementation. Our "modeling-phase" started by dividing the dataset into 5 subsets, that correspond to different products groups: Smartphones, Washing Machines Freestanding, Mobile Computing, Core Wearables, PTV/FLAT.

The model implementation can be divided in three steps, initially we used the whole dataset as training and tested the performance through a 5-folds cross validation, then we excluded 2020's flyer from the training set using again the 5-fold cross validation to test. Finally, we manually split the data into training and test, selecting August, September and October 2023 as test, leaving the remaining data, 2020 excluded, as training.

For each step we implemented three models: Linear Regression, Random Forest and Extreme Gradient Boosting tree (XGBoost).

Anticipating our results, we choose the best performing model for each product group according to the Root Mean Squared Error (RMSE): Linear Regression for Smartphones, Random Forest for Washing Machines Freestanding and XGBoost for Mobile Computing, Core Wearables, PTV/FLAT.

2 Methods

2.1 Preprocessing

The starting point of our analysis was the 6 datasets provided by Unieuro containing information about the performance of past flyers and the characteristics of the products and their KPIs:

- *Anagrafica_volantini*: This dataset contains information about flyers and details of products featured in each flyer.
- *KPI_prodotti_volantini*: This dataset contains KPIs describing the goodness of products at the time of day when the prediction must be made.
- *Storico_quantità*: This dataset contains the historical sales quantities per product per flyer for the 30 days leading up to the day when the prediction must be made.
- *Gfk_caratteristiche*: This dataset contains the most significant market characteristics provided by GFK for the best-selling products in the market.
- *Icecat_caratteristiche*: This dataset contains all the features for each product provided by Icecat.
- *Risultati_prodotti_volantini*: This dataset contains the real results generated by the products featured in the flyer.

We initially considered 4 out of the 6 datasets: Anagrafica_volantini, KPI_prodotti_volantini, Storico_quantità and Risultati_prodotti_volantini. Anagrafica_volantini represents the backbone of our dataset and consists of 20.983 observations. We decided to temporarily exclude the datasets related to product characteristics as they required further study, and following the advice received, our choice was to start by adopting a simple strategy, and then gradually make it more sophisticated, including further information.

Soon after an exploration of the dataset, to understand its underlying structure, the first step was to perform preprocessing and handle null values within the columns. This was an extremely delicate phase, as the goal was to

retain the maximum number of observations while integrating existing data with accurate estimates of the missing ones to preserve data integrity. To achieve this, it was also necessary to cross-reference information from different datasets.

We initially addressed the dataset concerning flyer product KPIs. Null values were only contained in the columns related to KPIs 2, 3, 4 and 5, which pertain to various product demand characteristics. As stated in the data mapping, provided by the company, missing values correspond to product that did not sell. Thanks to this information, we opted to substitute null values with 0. Moreover, in KPI_3 we encountered negative observations, which we decided to zero since the demand can not be negative.

Subsequently, we analyzed the "Storico_quantità" dataset: we decided to use "CODICE_VOLANTINO" and the "ART_COD" to sum up the daily units sold, obtaining an overall value indicating the quantities sold in the period under consideration, and another for the respective revenue.

In "Risultati_prodotti_volantini", instead, there were missing values only in the columns related to the performance of past flyers, namely QTA and FATTURATO. However, thanks to the data mapping, we were able to replace them with 0. In fact, it was expressly communicated to us that any missing values in these columns were due to the product not being sold or being excluded from the flyer, therefore not sold in any case.

In "Anagrafica_volantini", we initially addressed inconsistencies within the available data. Regarding negative values "SCONTO_PERC", we calculated their absolute value, as we noticed that the effect of the discount on the list price was the same, while occurrences less than one were multiplied by 100 to convert them to percentages and be consistent with other observations. Subsequently, "PREZZO_PROMO" greater than "PREZZO_LISTINO" and "PREZZO_LISTINO" equal to 0 were imputed as null values. As for the warehouse information, "STOCK_PZ", those reporting negative values were imputed as 0, as it is not possible to have negative stock in the warehouse. Instead, we decided to drop the column related to incoming orders, "STOCK_ORD_IN_ARRIVO_PZ", as it was mostly composed of null values. Regarding the null values related to "PREZZO_LISTINO", we computed the average price for the products with the same "ART_COD". While for the imputation of null values in the columns related to promo price, and percentage discount, we leveraged the relationships between these three columns. Namely:

$$SCONTO\_PERC = \left(\frac{PREZZO\_LISTINO - PREZZO\_PROMO}{PREZZO\_LISTINO} \right) \times 100$$

$$PREZZO\_PROMO = PREZZO\_LISTINO \times \left(1 - \frac{SCONTO\_PERC}{100} \right)$$

To manage the remaining missing values in "PREZZO_LISTINO" and "PREZZO_PROMO" we divided "FATTURATO" by "QTA" from Storico_quantità. Finally, we dropped all observations with promo price equal to 0.
From the information contained in this dataset, we defined 3 new variables: "DURATA_VOLANTINO_IN_GIORNI", "SCONTO_PERC_MEDIO_NOME_CAMPAGNA", "SCONTO_PERC_MEDIO_VOLANTINO". The first was calculated as the difference between the start date and end date of the flyer, aiming to capture the effect that a different duration from the default one, on average 20 days, could have on the flyer's performance. Regarding the average discount per flyer, a weighted average was calculated to determine the overall average of discounts contained in each flyer, based on the assumption that flyers with higher discounts could attract a greater number of potential customers and consequently sell higher quantities. Finally, for the average discount per campaign name, it was defined by dividing the campaign names into macro-categories based on common characteristics (examples: blackfriday, scontissimi, tasso 0, ecc..) , and then calculating the average discount. The ratio was similar to the previous variable: campaigns generally associated with more aggressive discounts attract a larger number of people and therefore record higher sales volumes.
After analyzing these 4 datasets, we focused on integrating information related to product characteristics. Our goal was to avoid performing one-hot encoding, which consists in creating a dummy variable for each class contained in a categorical feature, and thus creating a large number of binary variables with almost zero variance. We immediately understood that our choice would be to include only one between "Gfk_caratteristiche" and "Icecat_caratteristiche", as both contained similar information and including them together would have introduced redundant feature. To evaluate which one to consider, we analyzed the structure to understand which one was simpler to integrate. In "Icecat_caratteristiche" each row consists in a single characteristic for a given product. The issue with this dataset

is that columns are not well organized, since they present mix types. This lack of predefined structure makes it very difficult to directly implement it without further manipulations.
On the other hand, "Gfk_caratteristiche" is organized so that each row represents a product and contains a maximum of 8 of its characteristics. At first glance, the characteristics also seemed to be randomly arranged without following any logic among the different observations in the dataset. However, after dividing by product group, it became clear that each column contains the same macro characteristic, and the specific classes, within each column, could be ordered. Thanks to the orderability of such classes, it was possible to perform ordinal-encoding, which consists in assigning an integer to each class based on its rank among the others, thus limiting the sparsity of the columns after the encoding of categorical variables.

The division by product group of the "Gfk_caratteristiche" confirmed what we had previously suspected, defining a single model for all product groups would be very complex. Therefore, our choice was to merge "Anagrafica_volantini", "Risultati_prodotti_volantini", "Storico_quantità", "Gfk_caratteristiche" and "Kpi_prodotti_volantini" based on "art_cod" and "codice_volantino", and then split the merged dataset based on the product group. Therefore, the final result was 5 different datasets named: Smartphones, Washing Machines Freestanding, Mobile Computing, Core Wearables, PTV/FLAT. The number of observations within each dataset depends on the number of products belonging to that group contained in "Anagrafica_volantini". Therefore, the respective size of each dataset is:

- *Smartphones*: 9.169 Observations
- *Washing Machines Freestanding*: 1.730 Observations
- *Mobile Computing*: 1.766 Observations
- *Core Wearables*: 3.162 Observations
- *PTV/FLAT*: 3.030 Observations

After performing the division by product group, we defined a new variable called "Month_Redditivity". This was computed for each product group by summing up the historical sales for each month, and then the months were divided into 3 profitability classes: low, medium, and high. At the conclusion of the dataset preprocessing phase, the variables included are the following:

Table 1: Description of the variables

VARIABLE	**Description**	**VARIABLE**	**Description**
CODICE_VOLANTINO	ID of the flyer	DATA_INIZIO	Flyer starting date
DATA_FINE	Flyer end date	NOME_CAMPAGNA	Flyer name
ART_COD	ID of the product	PREZZO_PROMO	Product price in the flyer
PREZZO_LISTINO	List Price	STOCK_PZ	Number of units of the product in stock
TIPOLOGIA_PRODOTTO	Product type in flyer	SCONTO_PERC	Discount percentage
BRAND	Brand of the product	ITEM_ID	ID of the item
KPI_1	KPI measuring product seasonality.	KPI_2	KPI measuring product demand in consumer electronics market.
KPI_3	KPI measuring product demand in Unieuro.	KPI_4	KPI measuring how much customers are interested in the product features.
KPI_5	KPI measuring if the product is in a price range relevant for the customer.	PRODUCT_GROUP	Group of products to which the item belongs
QTA	Quantity of the product sold in flyer	FATTURATO	Revenue of the product in flyer
QTA_storico	Quantity of product units sold	FATTURATO_storico	Revenue of product sold
DURATA_VOLANTINO_IN_GIORNI	Number of Days the flyer is active	CARATTERISTICA_01	More detail in the following table
CARATTERISTICA_02	More detail in the following table	CARATTERISTICA_03	More detail in the following table

VARIABLE	Description	VARIABLE	Description
CARATTERISTICA_04	More detail in the following table	CARATTERISTICA_05	More detail in the following table
CARATTERISTICA_06	More detail in the following table	CARATTERISTICA_07	More detail in the following table
SCONTO_PERC_MEDIO_VOLANTINO	Average discount per flyer	Month_Redditivity	Level of redditivity of the month the flyer is active
Year	Year the flyer is active	SCONTO_PERC_MEDIO_NOME_CAMPAGNA	Average discount per campaign name

Table 2: Focus on Caratteristiche

Variable	Smartphones	Washing Machines Freestanding	Mobile Computing	Core Wearables	PTV/FLAT
CARATTERISTICA_01	Display Size	Type	GPU Model	Category	Display Size
CARATTERISTICA_02	Operator	Top Loading	Processor	Display Quality	HD Quality
CARATTERISTICA_03	Generation	Depth > 38cm	RAM in GB	Display Size	MFM
CARATTERISTICA_04	Capacity	Energy Class	Storage in GB	Wifi	Smart
CARATTERISTICA_05	Number of SIM	Loading Kg	OS Version	Bluetooth	OLED
CARATTERISTICA_06	Operating System	Smart Connect	Display Size		Wide Color Gamut
CARATTERISTICA_07			Convertible		Miniled

2.2 Modeling

Regarding the division of the dataset into training and test sets, our work can be divided into three distinct phases.

Initially, we considered the entire dataset and performed 5-fold cross-validation to evaluate the model's performance. Initially, the dataset is partitioned into five equal-sized subsets or folds. The cross-validation process then iterates five times, each time utilizing a different fold as the validation set while the remaining four folds serve as the training data. This ensures that every data point is used for validation exactly once across the five iterations, thereby maximizing data utilization. During each iteration, the model is trained on the training folds and evaluated on the validation fold, yielding a performance metric. These performance metrics are recorded for each iteration. Subsequently, the average performance across all five iterations is computed to obtain a single, robust estimate of the model's performance. This approach not only reduces bias in performance estimates but also enhances the reliability and robustness of the model evaluation process. Furthermore, the use of multiple folds allows for a comprehensive comparison of different models or parameter settings, aiding in the selection of the most suitable model configuration for the given dataset. Subsequently, after analyzing the sales distributions during the period under examination, we noticed that 2020 showed anomalous sales volumes due to the Covid-19 pandemic crisis (figure 1).

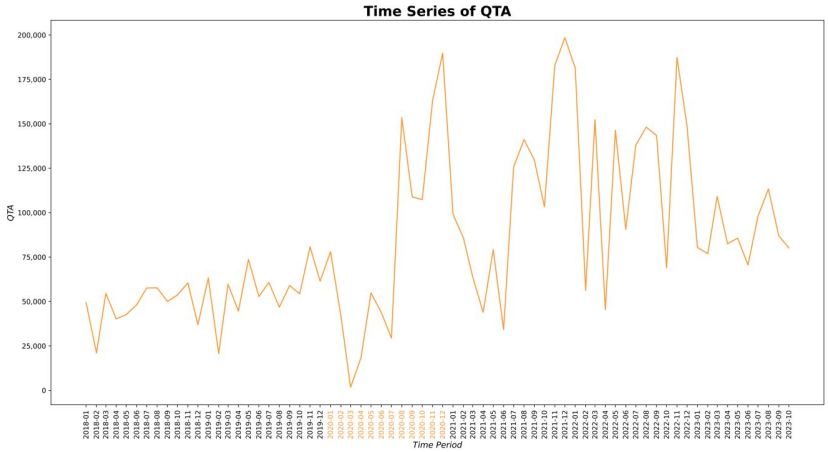

Figure 1

For this reason, given that we had 6 years available, we realized that integrating the flyer data for 2020 could harm the predictive qualities of the model. Therefore, we decided to exclude it. Hence, our second approach involves using all the flyers except those published in 2020, and then performing five-fold cross-validation.

Ultimately, as a final approach, we decided to radically change our method, effectively abandoning the 5-fold cross-validation. Our choice was to manually split the available dataset, using flyers from the beginning of 2018 to July 2023 as the training set, and flyers from August, September, and October 2023 as the test set. This was primarily for two reasons: on one hand, we wanted the flyers used to evaluate the model's performance to best reflect current market conditions, while on the other hand, we aimed to replicate a real-world scenario as closely as possible. To achieve this, we selected the latest period available so that the model would not use data beyond the test set for training.
Regarding the models, we decided to implement three different ones and compare them with a benchmark. Just as we did for integrating the datasets, initially, we adopted a simpler approach, implementing a linear regression, and then used more sophisticated models, such as the random forest and XGBoost.

Linear regression works by establishing a linear relationship between the independent variables and the dependent variable. The goal is to identify the best-fitting line that minimizes the difference between the actual and predicted values; this is achieved by minimizing the sum of squared errors. In summary, linear regression can be represented as the linear combination of the independent variables plus the intercept value. Linear regression represented a starting point for our project; indeed, it was chosen because it is simple to implement and computationally efficient. However, being able to capture only linear relationships, we knew it would not be sufficient for carrying out our task. Before fitting the model, it's essential to standardize the data in linear regression. This process ensures fair comparisons between variables of different scales, improves coefficient interpretability, and enhances model stability during optimization, thereby promoting accurate predictions and reliable model selection.

Random forest, on the other hand, belongs to the ensemble learning category of algorithms. It constructs multiple decision trees during training and combines their predictions to make a final prediction. Each decision tree is built independently using a subset of the training data and a random subset of features. This randomness helps to reduce overfitting and decorrelate the trees. During prediction, each tree in the random forest independently predicts the target variable based on the input features, and the final prediction is obtained by aggregating the predictions of all the trees; such aggregation can take the form of averaging for regression tasks. Random forest is known for its robustness and ability to handle high-dimensional data with complex interactions between variables and doesn't require normalization of the data.

XGBoost, short for Extreme Gradient Boosting tree, is a powerful gradient boosting algorithm that builds an ensemble of decision trees in a sequential manner. Unlike random forest, where each tree is built independently, in XGBoost, each new tree is trained to correct the errors made by the existing ensemble of trees. The training process involves iteratively adding decision trees to the ensemble, with each tree trained on the residuals of the previous predictions. XGBoost optimizes a differentiable loss function by minimizing the gradient of the loss function with respect to the predictions of the ensemble. During prediction, each tree in the ensemble sequentially applies its rules to the input features, and their predictions are aggregated to produce the final prediction. XGBoost is known for its high performance, scalability, and ability to handle complex datasets with nonlinear relationships.

As for the benchmark, following the feedback received, we opted for a solution that did not delve into the structure of the data. This way, we could understand whether the models we implemented were able to capture relationships and patterns within the features present in the dataset. Our benchmark is constructed as follows: we calculated the daily average of products sold for each product group using historical quantity data. Then, to make the estimation, we multiplied this average by the number of days the flyer was online.

Hyperparameter tuning was used to train the model and find the best parameters, and to be precise we decided to implement a random search to train the random forest and the XGBoost.
Random search is a hyperparameter tuning technique used to efficiently search through a predefined hyperparameter space. This method is particularly effective when the hyperparameter space is large and there is limited computational resources, this is also the reason why we chose this technique instead of a grid search, since there was not enough time to implement a proper grid search.

3 Results

3.1 Technical Results

In this section we are going to present the technical and business-oriented results of our analysis. The technical part will discuss the results obtained by our three models Logistic Regression, Random Forest and XGBoost. As already explained in the previous section the modeling phase was characterized by three distinct phases: firstly a 5-fold cross-validation using the entire dataset, secondly a 5-fold cross-validation without the year 2020 and lastly a training-test split where the test is composed by August, September and October of 2023 and as the training the entire dataset without the year 2020. The best performing model was selected according to its Root Mean Squared Error (RMSE), as required by the company, on the different product groups.
The figures related to the RMSE comparison among the models show some interesting insights.
In all different product groups, we can see how removing the year 2020 increased the performance of the models, including the benchmark, by reducing the RMSE on average by 12 points (in the cross validation). This fact underlines how, on one hand, removing this year was the right choice, and on the other, what a particular year 2020 was.
Now focusing on the last modeling phase, addressed in the figures as FINAL MODEL, we can see that the overall RMSE is lower than the one in the cross-validation phases, in some case by a lot, in fact it is not uncommon to observe that cross-validation may yield lower or equal accuracy compared to the train-test split technique, the reason behind this is to be found in how cross validation and train-test split work, on one hand, cross-validation provides a more realistic assessment of the model's ability to generalize but may result in a slight sacrifice in performance, on the other hand, the train-test split technique divides the dataset into two parts, this approach may yield slightly higher accuracy because the model is optimized specifically for the test set, but, potentially leading to overfitting.
Now we are going to present the result for each product group, by looking at their RMSE scores, the predicted VS true values and lastly the feature importance.
The plot for the feature importance show the first fifteen variables for feature importance, the higher the score for a feature, the larger effect it has on the model to predict a certain variable, in our case we decided to highlight the variables created during the feature engineering phase, already explained in section 2, in a different color, we did this in order to understand if the work that we was actually meaningful. The results of the feature importance show that in all the five different product groups at least 2 "custom" variables were present, suggesting that our work was meaningful for the prediction.

3.1.1 Smartphones

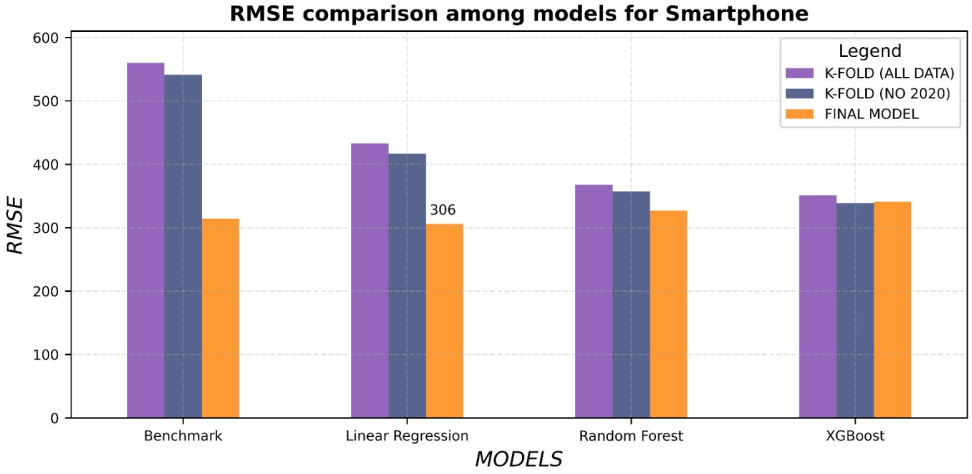

Figure 2

In the case of Smartphone, we have that, in the case of the 5-fold cross-validation cases, all the models perform better than the benchmark, which has a RMSE of 560 in the 5-fold cross-validation with all the data and 541 in the 5-fold cross-validation without 2020. XGBoost is the best model, in sthe 5-fold cross-validation cases, with an improvement in respect to the benchmark of 200 points. If we instead focus on the FINAL MODEL, we will see that the model performance is similar, with random forest and XGBoost performing worse than the benchmark, with a RMSE of 314. Surprisingly the best performing model in this case is the linear regression with a score of 306 (figure 2). The reason why this happens is difficult to find, one reason could be the reduce number of observations, since both random forest and XGBoost tend to not work well with few observations, or maybe this product group, in this particular setting (test on September, October and November

of 2023) show a more linear combination in respect to other product groups.

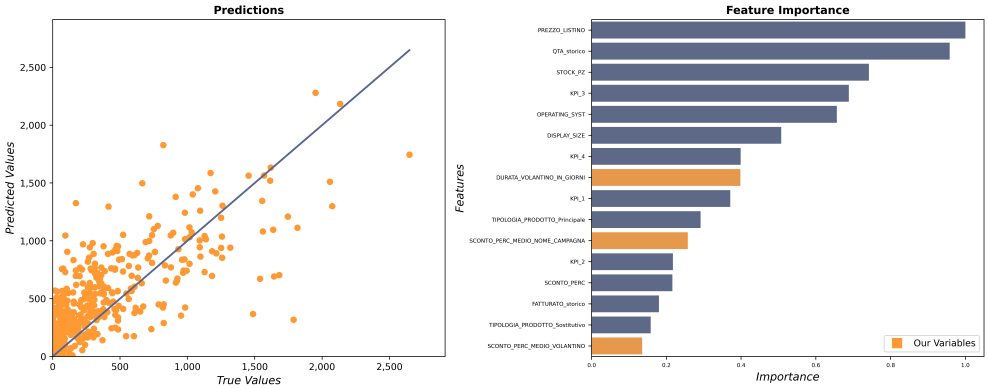

Figure 3

We then plotted the predicted values against the true values of the FINAL MODEL (figure 3, left). Here it is possible to see how we were able to grasp the test's trend. The model, though, shows a bit of bias when treating large numbers of quantities sold; in fact, the model tends to underestimate the values.
As for the feature importance (figure 3, right), we have that "PREZZO_LISTINO", "QTA_storico", and "STOCK_PZ" are the most important variables. Furthermore, three out of four of our "custom" variables ("DURATA_VOLANTINO_IN_GIORNI", "SCONTO_PERC_MEDIO_NOME_CAMPAGNA" and "SCONTO_PERC_MEDIO_NOME_VOLANTINO")appear to be important for the performance of the model.

3.1.2 Mobile Computing

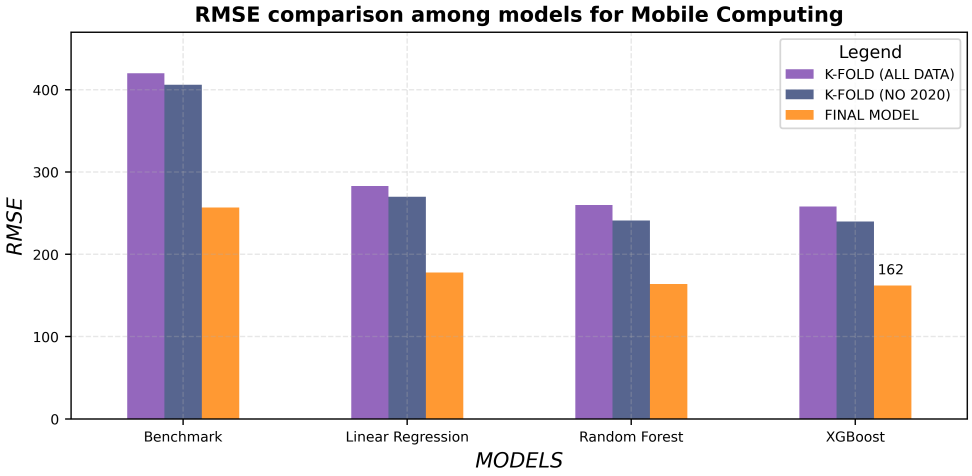

Figure 4

In Mobile Computing we have again the proof that removing the year 2020 led to increasing in performances furthermore, all models, in all the splits of the data, perform better than the benchmark, which has a values of: 420 in the 5-fold cross-validation with all the data, 406 in the 5-fold cross-validation without 2020 and 257 for the FINAL MODEL. In the latter case all models tend to perform similarly but the best performing model is the XGBoost with a RMSE of 162, compared to the linear regression (178) and the random forest (164) (figure 4).

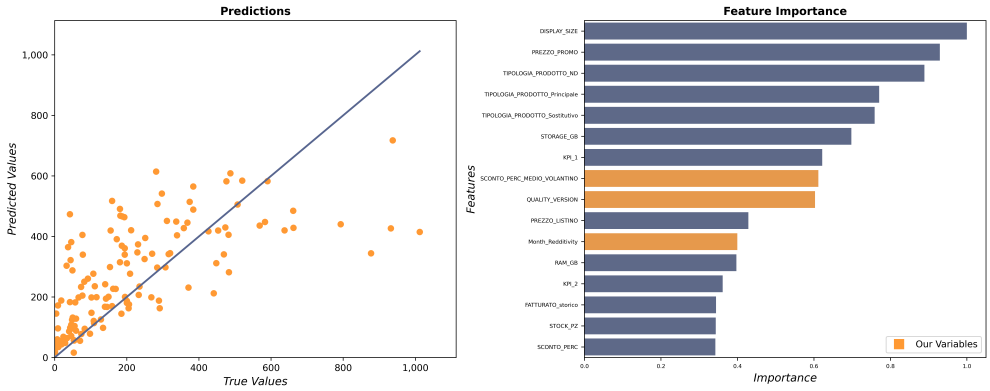

Figure 5

From the plot of predicted against true values (figure 5, left) we can see that the best performing model is able to understand the overall trend, but, in this case, it is even more clear the behavior of the model towards large values. The model tends to not predict above a given threshold, 610, thus fails to predict very high quantities of the response.
"DISPLAY_SIZE"," PREZZO_PROMO" and "TIPOLOGIA_PRODOTTO_ND" are the most important variables (figure 3, right), and that, two out of four of our "custom" variables ("SCONTO_PERC_MEDIO_VOLANTINO" and "Month_Redditivity") appear to be important for the performance of the model.

3.1.3 Washing Machine Free Standing

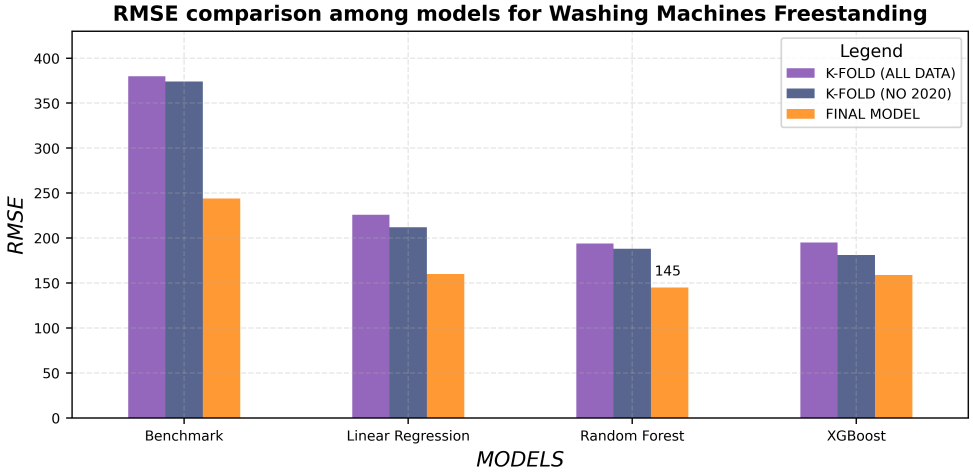

Figure 6

Again, as we can see in the figure above, removing the year 2020 led to increased performances.
All the models outperformed the benchmark, which has values: 380 in the 5-fold cross-validation with all the data, 374 in the 5-fold cross-validation without 2020 and 244 for the FINAL MODEL.
The best performing model is the random forest with a RMSE of 145, followed by the XGBoost (159) and then the linear regression (160) (figure 6).

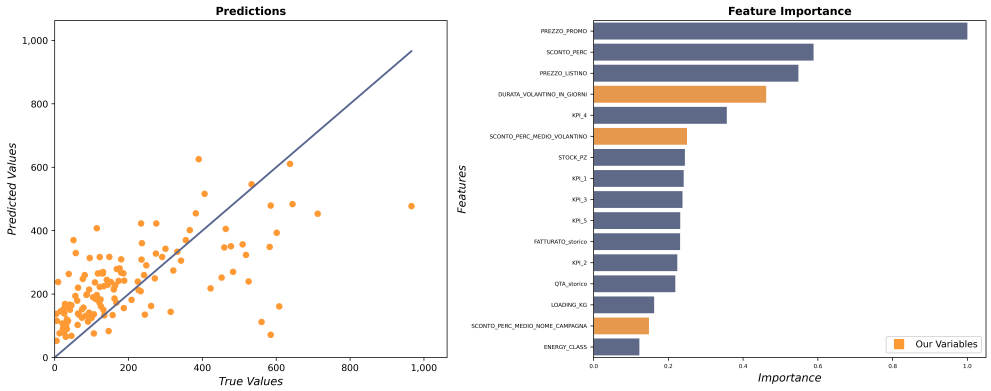

Figure 7

Like for the previous product groups the model under predicts large quantities of the response, but in this case, it is less extreme than in Mobile Computing, that presented more outliers and thus more complicated points to predict (figure 7, left). The feature importance (figure 7, right) shows: "PREZZO_PROMO", "SCONTO_PERC" and "PREZZO_LISTINO", and three out of four of our "custom" variables ("DURATA_VOLANTINO_IN_GIORNI", "SCONTO_PERC_MEDIO_VOLANTINO", "SCONTO_PERC_MEDIO_NOME_CAMPAGNA") appear to be important for the performance of the model.

3.1.4 PTV/FLAT

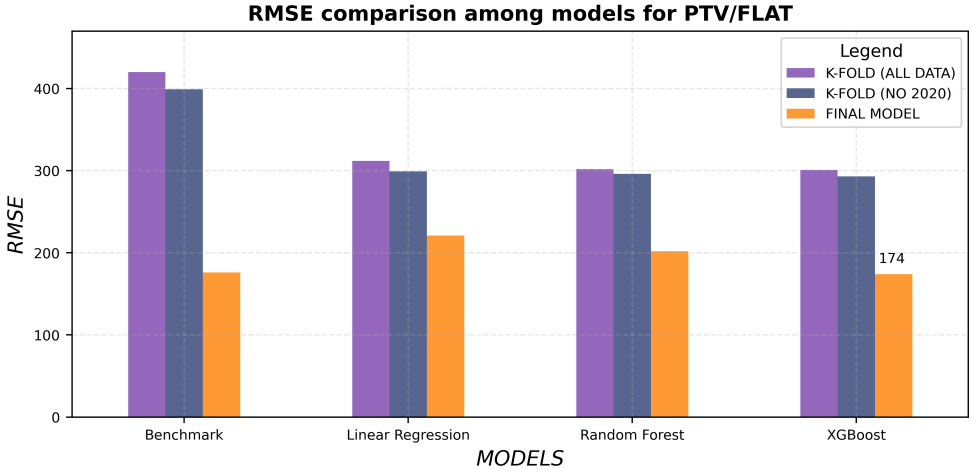

Figure 8

In the case of PTV/FLAT something interesting happens, we have that in the cross-validation attempts all models perform better than the benchmark, which has values of 420 in the 5-fold cross-validation with all the data, 399in the 5-fold cross-validation without 2020, but in the FINAL MODEL it shows to have very low RMSE, 176. This may be due to a fortunate set of observation in the test set that led to this low number.
If we focus on the results for the FINAL MODEL we can see that the benchmark outperforms both the linear regression (221) and the random forest (202), leaving the XGBoost the only model that is able to outperform the benchmark with a RMSE of 174 (figure 8), an improvement of only 2 points. This may suggest that the benchmark could be used as a good approximation of the error, but in our opinion using the benchmark could lead to lower performance with real data, thus we preferred to opt for the XGBoost.

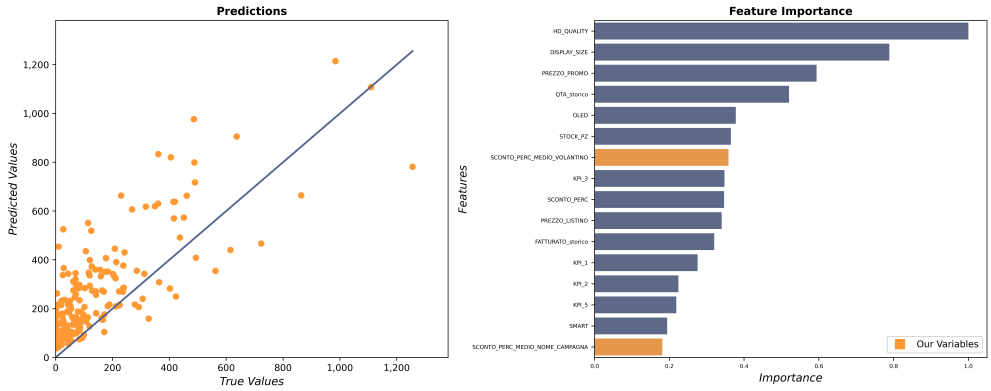

Figure 9

The model shows a tendency to over predict the majority of observations (figure 9, left), unlike the previous cases where there was the opposite problem in fact, in this case the model shows less bias towards larger quantities in respect to other product groups like Washing Machine Free Standing and Mobile Computing.
The feature importance (figure 7, right) shows: "HD_QUALITY", "DISPLAY_SIZE" and "PREZZO_PROMO", and two out of four of our "custom" variables ("SCONTO_PERC_MEDIO_NOME_CAMPAGNA, "SCONTO_PERC_MEDIO_VOLANTINO") appear to be important for the performance of the model.

3.1.5 Core Wearables

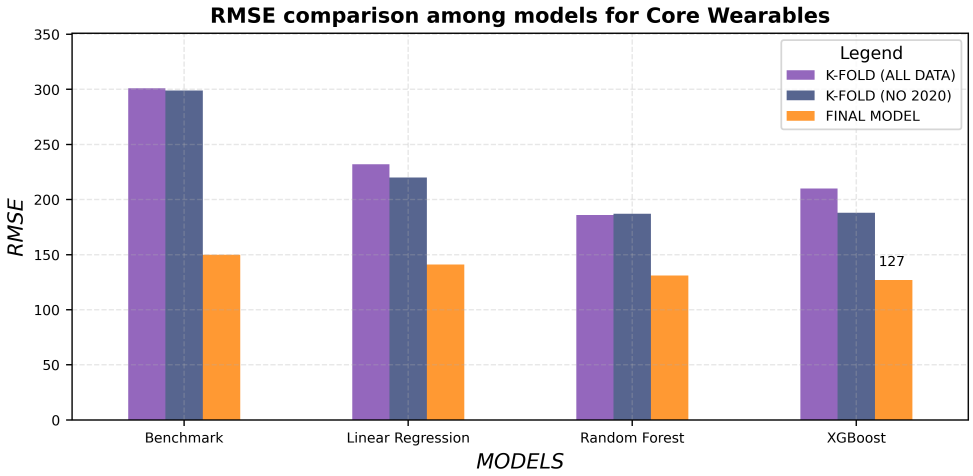

Figure 10

Lastly, we have the product group Core Wearables. This product group has a pretty normal behavior since all models tend to perform better than the benchmark and that dropping the 2020 leads to increase in the performance. Th benchmark has values of: 301 in the 5-fold cross-validation with all the data, 299 in the 5-fold cross-validation without 2020 and 150 for the FINAL MODEL. The best performing model is the XGBoost with an RMSE of 127 (figure 10), followed by the random forest (131) and the linear regression (141).

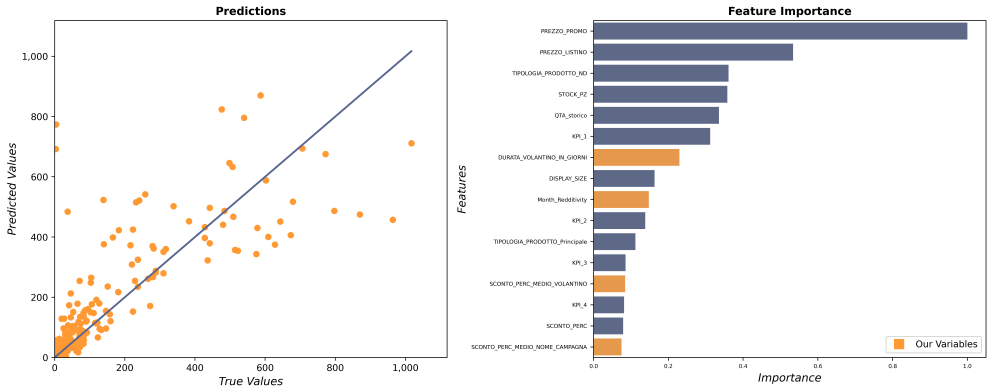

Figure 11

The predicted versus true values show that, for this product group, there is less bias when predicting large quantities of data, showing overall a good performance (figure 11, left). The feature importance (figure 11, right) shows: "PREZZO_PROMO", "PREZZO_LISTINO" and "TIPOLOGIA_PRODOTTO_ND", and three out of four of our "custom" variables ("DURATA_VOLANTINO_IN_GIORNI", "Month_Redditivity", "SCONTO_PERC_MEDIO_NOME_CAMPAGNA") appear to be important for the performance of the model.

In conclusion we showed how our models are able to increase the performance of the benchmark, in some case only slightly, and that the feature engineering work performed previously proved to be effective in impacting the model performance.

3.2 Project Oriented Results

One of the most important choices we had to make during the model-building process was whether to use a single model for all products or to create customized models for specific product categories. This decision basically came down to balancing the benefits and drawbacks of more accurate forecasting versus easier implementation for our company's operations.

Using a consistent model, on the one hand, makes it easier to integrate into our business processes, makes maintenance tasks like upgrades to maintain performance over time easier, and, most importantly, improves usability for non-technical people. On the other hand, a single model method fails to take into account the subtle differences between various product groups, which results in a generalized feature importance across products with diverse features. This greatly reduces performance.

As a result, we decided to divide models based on product groupings in order to give performance priority. This choice was further supported by the realization that high performance results in a multitude of positive effects for our company, as will be discussed in more detail in the parts that follow

Our predictive models provide many advantages to many business operations, mainly to supply chain and inventory management, marketing and financial planning. Through the provision of detailed insights into sales dynamics, these models enable the organization to improve its overall operational efficiency and strategic decision-making procedures.

Starting from marketing benefits, equipped with precise sales projections, the business can enhance its marketing endeavors by more precisely focusing promotions.
The business can better target its marketing campaigns to align with client preferences and increase sales conversion rates by determining which products are most likely to perform well during particular promotional times. This focused approach fosters more customer involvement and brand loyalty in addition to optimizing the return on investment for marketing expenditures.

Our predictive algorithms can help the business optimize its logistics and supply chain management. Accurate sales volume forecasting helps the business better predict changes in demand and adjust its distribution and procurement procedures accordingly. By taking a proactive stance the danger of supply chain interruptions is reduced, and prompt product delivery to satisfy customer demand is guaranteed.

Furthermore, our models can aid the organization in optimizing its inventory storage and warehouse management procedures by detecting trends and patterns in client purchasing behavior, hence augmenting operational efficiency and cost-effectiveness. Indeed, implementing a "just in time" warehouse strategy enhances the efficiency of inventory investment by averting prolonged storage of purchased goods on shelves, thus minimizing opportunity costs.

In general, the application of predictive modeling signifies a paradigm change in the way the business formulates strategies and makes decisions. Leveraging data-driven insights to its full potential can help the business become a market leader by creating new avenues for growth, efficiency, and innovation.

4 Conclusions

This project showed how data can be used in order to enhance business performance.
After merging four out of the six datasets provided by the company we started by exploring and preparing the data.
The preprocessing phase was by far the most important since it enabled us to retain the great majority of observations, by leveraging the relationship between variables, where it was possible, and by encoding missing and wrong data.
Afterward we were able to merge also a fifth dataset, "gfk_caratteristiche", which gave us the possibility to exploit the information related to the different product characteristics. With the preprocessed dataset we then performed some feature engineering by creating 5 new variables that showed significance later during the modelling phase. Then we started the modelling phase where, by utilising different predictive models, we were able to create a model that outperforms the trivial model used as the benchmark in all the different product groups. As further steps it would be interesting to create new variables in order to address the problem of bias in the models, by leveraging both data that we already have and outside data, like for example data regarding competitors, customer behaviour and macroeconomic indicators. Moreover, performe a more intense hyper-parameter-tuning for both random forest and XGBoost, since, due to lack of time it was not feasible, together with some other techniques such as dimensionality reduction, like PCA, and other unsupervised techniques, like a clustering on the different characteristics (trying also to implement the icecat_caratteristiche dataset).

Appendix A: Code Description

As we already saw, since our work is divided in several notebooks, we will briefly explain the main steps of each of them.

- *file_iniziale.ipynb*
  Here we imported and cleaned the data provides us by the company. The most relevant operation was the retrieve of the variables 'PREZZO_LISTINO', 'PREZZO_PROMO' and 'SCONTO_PERC'.

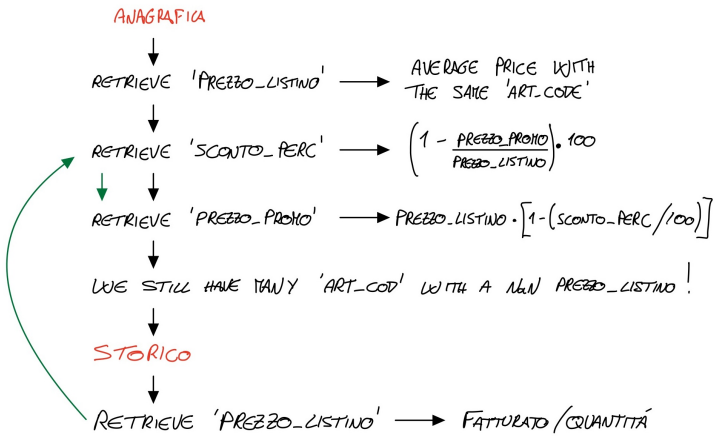

Figure 12

Firstly, we tried to recover the missing values operating in 'Anagrafica' dataset, then we retrieved 'PREZZO_LISTINO' thanks to 'Storico' dataset and repeated the same operations in order to fill all the possible NaN values.

- *gfk_divisione_prod.ipynb*
  In this notebook we divided our final dataset by product and then merged to each of them the dataset 'gfk_caratteristiche". The first problem we encountered was in the in the column 'CARATTERISTICA_03' in the 'smartphones' dataset: in fact, instead of null values, were present the values of the subsequent columns, thus altering the entire rows. So, we did the following operation to fix it.

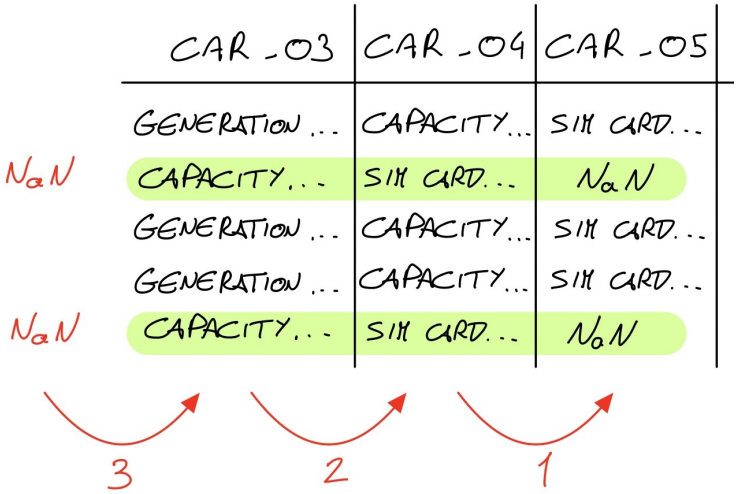

Figure 13

We encountered a similar problem also in the column 'CARATTERISTICA_04' in the 'tv' dataset: in this case, since the information about 'curved' was available just for 167 out of 2906 observations, we decided to lose this information to bring order in the dataset.

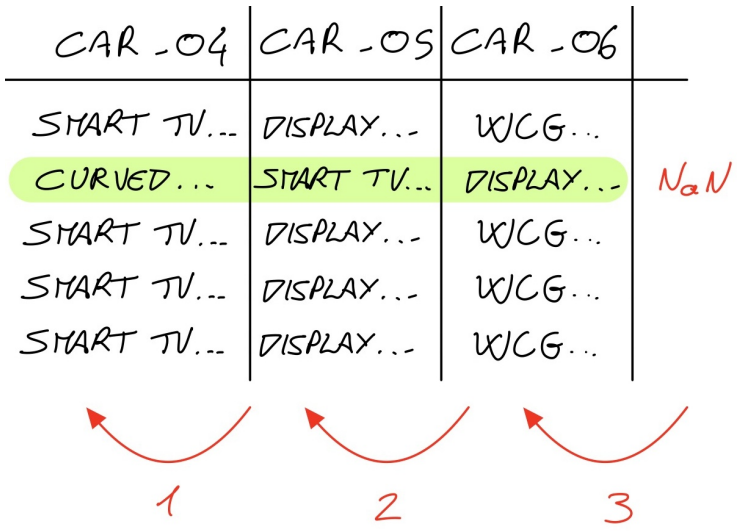

Figure 14

- *manipolazione_prodgroups.ipynb*
  Here we defined a function called 'nomi_campagna()' in order to compute the average discount applied to the products belonging to a campaign name.

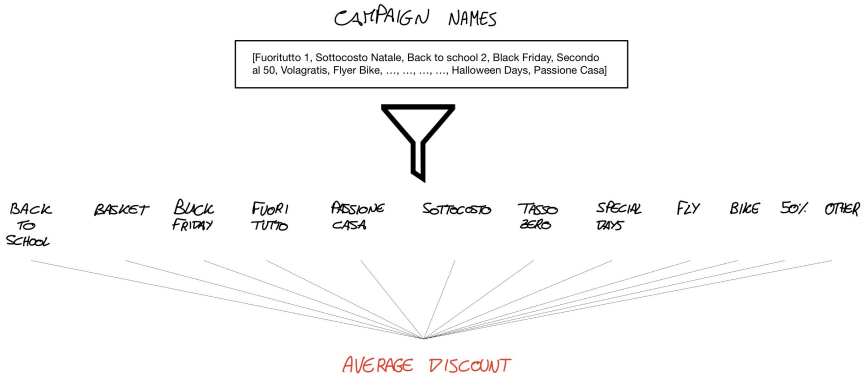

Figure 15

Then, in order to compute the profitability of each month, we plotted the monthly cumulative revenue and then assign to each month a number according to his profitability.

- *modelli_kfold.ipynb*
  Here we compute a Linear Regression, a Random Forest and an XGBoost for each product group using all data available and validating them with a k-Fold cross validation.

- *modelli_nocovid.ipynb*
  Here we compute a Linear Regression, a Random Forest and an XGBoost for each product group excluding 2020's data and validating them with a k-Fold cross validation.

- *modelli_testspec_nocovid.ipynb*
  Here we compute a Linear Regression, a Random Forest and an XGBoost for each product group excluding 2020's data from our training set and used as test set August, September and October 2023.

- *grafici.ipynb*
  This notebook contains the codes we used to create the graphs present in the final pitch. Firstly, we plotted the

performance of each model; secondly, we gave an example of how we calculated the monthly profitability; lastly, we displayed our predictions and the feature importance.

- *forecast.ipynb*
  In the last notebook, we apply the changes we have already seen to the test data in order to calculate the sales forecast.

Appendix B: Author Contribution

Camerlengo Vincenzo: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data Curation, Writing - Original Draft, Writing - Review & Editing, Visualization, Supervision, Project administration.
Paolantoni Riccardo: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data Curation, Writing - Original Draft, Writing - Review & Editing, Visualization, Supervision, Project administration.
Torelli Raffaele: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data Curation, Writing - Original Draft, Writing - Review & Editing, Visualization, Supervision, Project administration.