Samsung Innovation Campus

Universidade Nova de Lisboa
Faculdade de Ciências e Tecnologia

PowerCoding - Group 8

FilterBond Workflow Optimization

Students:

João Jorge
Luís Tripa
Maria Dias
Rúben Inocênci

Adviser:

Miguel Cardoso

5th Edition 2023

Contents

List of Figures

List of Tables

1 Summary

Filterbond is a Portuguese company specialising in the manufacturing of diverse types of air filters. The company receives filter-based solutions inquiries from various clients and generates proposals. During this process, a challenge arises: crafting proposals manually is costly and does not promise the highest possible income due to substantial variations in applied prices. The objective of this study is to implement a prototyping tool that automatically builds proposals to clients' requests based on past proposals using machine learning techniques.

The data provided by Filterbond consists of a set of CSV files, one for each proposal, where the name of the file has the client's name. A proposal consists of three variables: the types of filters requested (including the dimensions), the number of filters from each type and the unit price applied to each type of filter.

During the preprocessing of data, were extracted from the CSV files the following features: Company Name, Quote ID (ID of the proposal), Quantity (number of filters from each type), Unit Price, Filter Efficiency (categorical feature that depends on the quality of filtering), Length, Height, Gutter, Depth, Pockets and Date of the proposal.

After exploratory data analysis, we aimed to identify the most effective predictive model for filter prices, considering the three most purchased filter efficiency types globally (G4, M5, F7). Our selection comprised six models—Linear Regression, Epsilon-Support Vector Regression, Nearest-Neighbors based Regression, Decision Tree Regression, Random Forest Regression, and a neural network-based regression model.

In our quest to identify the optimal regression models, Mean Relative Error (MRE) served as our benchmark for comparison. The evaluation process involved assessing both Mean Relative Error (MRE) and Mean Squared Error (MSE). After a thorough analysis, we arrived at the conclusion that, across the board, including each of the three principal filter efficiency types (G4, M5, F7) and the overall dataset, the neural network emerged as the best-performing model.

To establish a functional front-end, we developed a simple interface using *Gradio* that enables the client to ask for a filter solution and receive an automatic and real-time proposal based on the prediction performed by the model with the best performance.

2 Introduction

2.1 Background

Filterbond is a company that manufactures air filters of various types in Portugal. They need us to create a system that helps in predicting the final product price of a filter given information about it such as its dimensions and efficiency.

2.2 Problem Statement

Currently, the company relies on experienced personnel to more or less 'guess' the final price of the manufactured product, however, such a solution is not scalable, making the company incredibly dependent on a single individual and making it hard for inexperienced employees to create quotes for clients without incurring in losses to the company.

2.3 Dataset

The dataset we're going to use is a cleaned version of the company's quote database. Quotes are provided in many CSV files, one for each quote, containing all the items' descriptions, quantities and prices provided to the client.

Most of the features were extracted from the description field of each item. This field contains valuable information to us, such as the filter's efficiency and its dimensions that can be used to train the predictive models we're trying to achieve.

Filters come in several categories (or efficiencies) represented by the letters G, M, F and H. G represents filters that filter coarse-grained particles, M represents the ones that filter medium-sized particles, F represents those that filter fine-grained particles and H represents filters with special filtering capabilities (usually extremely small particles). Each category also contains sub-categories, for example, an F9 filter filters more particles than an F7 one, even though they belong to the same category F.

There are many categories and sub-categories of filters available in the dataset, however most of them don't have enough items to make informed predictions, some even having less than 10 data points. To that effect, we decided to restrict the dataset to just the 3 filter efficiencies with most items: G4, M5 and F7, exclusively for model training. This seems like over-simplifying the problem, since there are several other filter efficiencies available, however, by covering just these 3 categories, we cover most of the sales of the company. All filter efficiencies will still be used for the exploratory data analysis part and to train a general predictive model.

Also, filters can be of many different sizes. Each filter may have either 3 or 5 dimensions, depending on whether they are or not a pocketed filter. Standard filters have 3 dimensions: the length, height and gutter, while pocketed ones have 5: the length, height and gutter plus the depth and the number of pockets.

Due to several data consistency problems in the data regarding pocketed filters' dimensions, we decided to effectively drop all the data associated with these types of filters. In the following sections, we'll exclusively analyse the data and train models based on the standard filters with 3 dimensions.

3 Data Collection and Preprocessing

3.1 Data Collection

Our team didn't directly take part in the process of data collection, however the information we were able to obtain shows that the company uses Excel files to create quotes for clients. These are then saved to their quote database for future reference. The data we were handed over were these files converted to CSV format, containing 3 columns: the description, the quantity and the price of an item.

3.2 Data Preprocessing

Preprocessing the data was an extremely important and incredibly challenging step in the execution of his project since we were given raw data from a company's quote database in many CSV files, one for each quote, instead of a clean and curated dataset.

Most of the files we were provided with adhered to a well-defined 3 column structure that was easy to parse: a small description of the product that contained general information about it like the dimensions and filter efficiency, the quantity of a specific product and its unitary price.

Some of these files, however, either didn't have the correct structure or they were completely broken, some even having no data at all. In these specific cases, we had to use a manual process to salvage the data, including reformating or restructuring the files in a way that could easily be read by automated processes later on. We weren't able to manually cover all files, however we managed to get enough data to achieve a average-sized dataset.

Next, we used python code to iterate over all files and join them into a single CSV, in order to make the data analysis and model training easier by having all the data in just a single place.

After the initial data processing phase, we still didn't have a dataset we could use, since we only had three columns in the CSV file and none of them were sensible features. However, the description column contained a bunch information that we could extract into independent features, such as the filter efficiency and its dimensions. The last, we also split it into 3 separate features: the length, height and gutter. We managed to do this using regular expressions and standard python code.

Then, we also observed there were some products that didn't contain the prices, most likely due to data conversion problems when converting the quotes from an Excel file to a CSV file. We didn't have any control over this process, so there was nothing we could do besides removing all the null values.

4 Exploratory Data Analysis (EDA)

Given the pre-processing of the collected data and its rearrangement into a *Pandas DataFrame* the next step is to **understand the data and try to identify relations between features in order to learn possible patterns**, this is called an EDA. A good EDA allows the development of further Machine Learning (ML) models that will integrate this knowledge to perform tasks.

By looking at the cleaned dataset, we infer that most of the features are numerical which is optimal for pattern finding and correlation analysis. All the non-numerical features such as the "Date" (which is as a *timestamp* type), the "Company Name" and "Efficiency" (are both categorical features) need to be either erased from the dataset or transformed into a numerical meaningful feature. Although this procedure is not entirely mandatory, it enables data to be represented in a *PairPlot* or in a *Correlation Matrix* .

PairPlot

The first chosen method of exploring our dataset is by building a *PairPlot*, a matrix of scatter plots for exploring relationships between feature pairs, accompanied by diagonal histograms showing individual variable distributions. For this particular analysis, we decided to drop some less relevant features such as the "Company Name" and "Quote ID". To simplify the "Date" data, we extracted only the year when each quote was asked for. We also concluded that "Efficiency" is an important feature so we encoded the type of filtering into a key (table 1), this way we preserved some of the information given by the filter efficiency.

Filter Efficiencies in the Dataset	Description	Numerical Key
G2, G3, G4	Primary filtration (G)	1
M5, M6	Medium filtration (M)	2
F5, F6, F7, F8, F9	Fine filtration (F)	3
H3, H11, H13, H14	Special filtration (H)	4

Table 1: Filter efficiency key for label encoding.

By making these changes we built a *PairPlot* that shows the relation between: the "Ordered Quantity" of each filter in each quote; the "Unit Price" applied to each type of ordered filter; the "Efficiency" of filtering (using the key shown in table 1); the "Length", "Height" and "Gutter" measures of the ordered filters; and the "Year" in which the filters were quoted. Figure 1 shows the *PairPlot*.

Countless conclusions could be achieved through such rich visualisation, below are explained some of the most important and clear findings.

1. Unit Price VS Ordered Quantity (subplots 2 and 7): when a proposal presents a higher quantity of ordered filters, the price per unit applied decreases and vice versa. This evidence is called the principle of Economies of Scale, this manifestation indicates that the company experiences cost advantages as a result of increasing its level of production or output.

2. Gutter, Height and Length: By looking at the diagonal plots concerning the measurement features (Gutter, Height and Length), we conclude that there are indeed standard sizes, which is why the distributions in these graphs are not random. But, since the distributions are Gaussians it shows that Filterbond has some flexibility in producing other sizes.

3. Price Unit VS Efficiency (subplots 3 and 14): The filtering quality means more expensive filters. This is important for model planning because it assures a high correlation between these features, meaning that they cannot be disregarded.

4. Year of purchase: Data does not show a clear correlation between Date and any of the features. Even so, by looking at the diagonal plots, the year 2018 is without a doubt the year when was sold the greatest amount of filters.

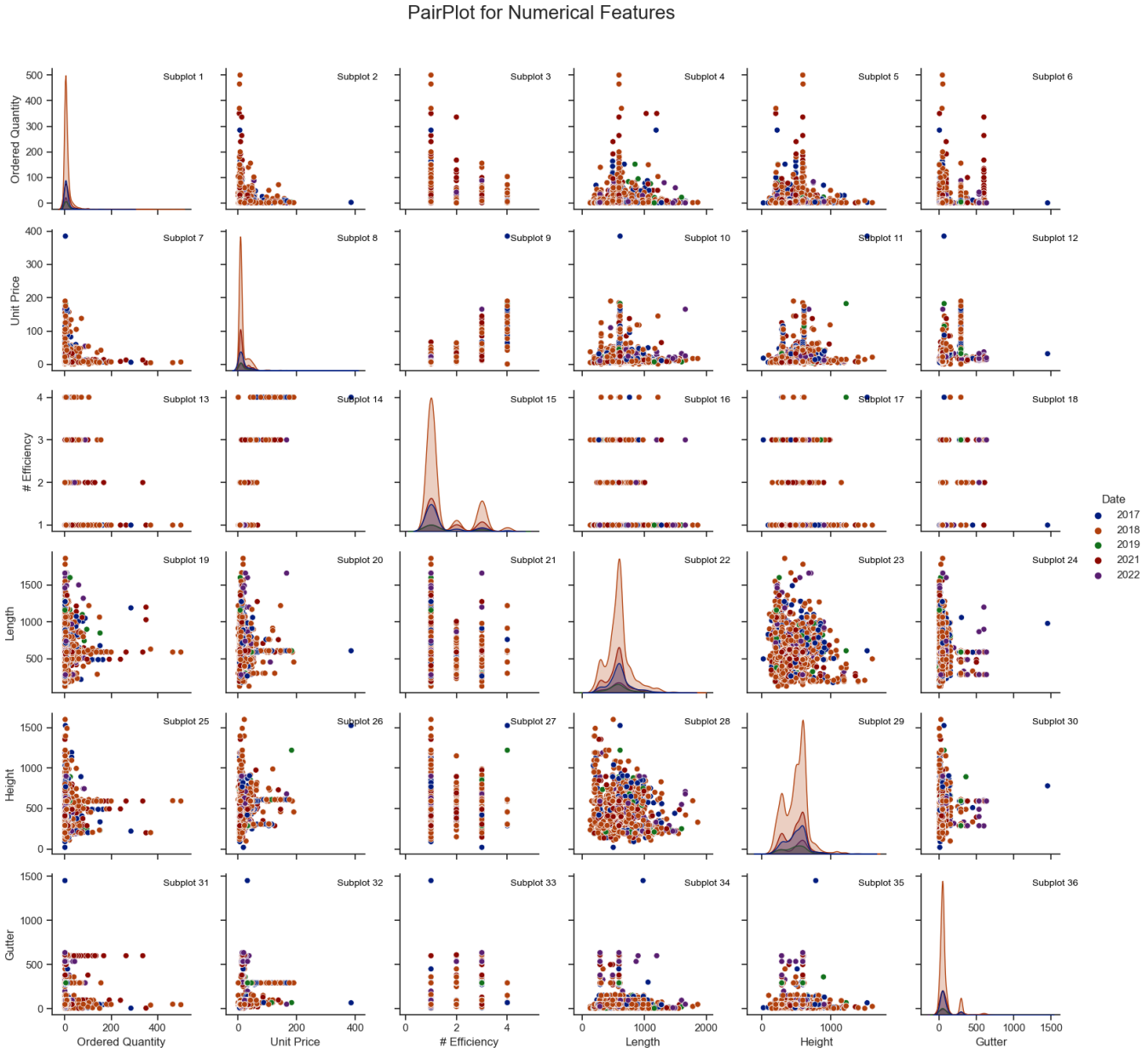


Figure 1: PairPlot containing the relation between the numerical features for all the data in the dataset. For a better understanding of the year in which each proposal was committed, different years are represented with different colors.

Correlation Matrix

The PairPlot-based analyses suggest correlations between some pairs of features. To quantify these correlations, a correlation matrix is presented where the same numerical features as those presented in figure 1 show how related are to each other. In a correlation matrix, the degree of correlation between features is indicated by the correlation index. A correlation index nearing 1 signifies a strong positive correlation, implying that the features move in the same direction (with 1 denoting perfect correlation achieved within the same feature). Conversely, a correlation index approaching -1 indicates a strong negative correlation, suggesting that the features move in opposite directions. When the correlation index is close to 0, it signifies a lack of apparent correlation between the pair of features. Figure 2 presents the correlation matrix for the mentioned features.

From a general perspective, there are very few highly correlated features. This means that there is no need to exclude any particular feature, they all contribute with information that may be useful to teach a predictive model, for example. To understand further the feature correlations we present a short analysis of the most related features:

1. Ordered Quantity and Unit Price (-0.05): Negative correlation indicates that as ordered quantity increases,

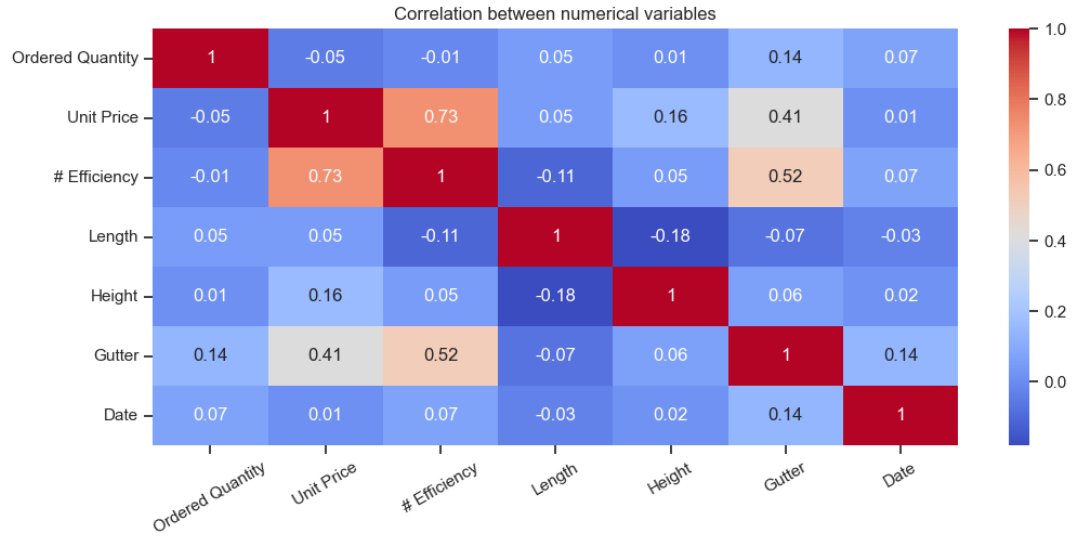


Figure 2: Correlation matrix with the numerical feature extracted from the dataset.

unit price tends to decrease slightly. This aligns with the concept of "quantity discount" or "economies of scale."

2. Unit Price and Efficiency (0.73): A high positive correlation suggests that more efficient filters tend to have higher unit prices. This is likely due to the advanced technologies or materials associated with higher efficiency.
3. Efficiency and Length (-0.11): Negative correlation implies that filters with higher efficiency are shorter. This may be attributed to achieving the desired filtration with a smaller physical size, reducing material costs.
4. Gutter Measure and Unit Price (0.41): Positive correlation indicates that filters with higher unit prices tend to have a higher measure of gutters. Higher gutter measures may contribute to increased production costs.
5. Gutter Measure and Efficiency (0.52): Positive correlation suggests that filters with higher efficiency also tend to have a higher measure of gutters. Advanced filters designed for better efficiency may be associated with higher gutter measures.
6. Date and Gutter Measure (0.14): Positive correlation implies an increase in the measure of gutters in filter designs over time. This may be attributed to technological advancements or shifts in customer preferences affecting the measure of gutters.

Since the relations between the filter dimensions are similar, the idea of merging the dimensions into a "Volume" feature ($V = Length \times Gutter \times Height$) might benefit the interpretation and visualisation of the data. COVID-19 might have led to an increase in thicker filters (higher gutter) due to better filtering performances. In conclusion, there is an overall agreement between the analyses using the PairPlot and the correlation matrix.

Client Analysis

Until this stage, the analyses embraced simply the numerical data disregarding clients and their profile or purchase history. An equally important analysis must be performed to look for tendencies regarding the customer profile. Such analysis might start from various approaches, but the one followed by us tries to sort clients using the total income each one generated. From that, we can infer a whole new variety of curious features, for each client and compare them, such as: the total amount of purchases, total quantity of purchased filters and ratios involving these last features. Such insightful vision is shown in figure 3.

When looking at figure 3 various conclusions can be drawn:

1. By **arranging clients based on their generated income**, it becomes evident that a small proportion of clients contribute significantly, yielding revenues in the tens of thousands of euros. Conversely, almost half of the clients generate less than one thousand euros. This observation suggests that strategically grouping the most valuable clients could prove advantageous for the company, especially during the model planning phase. Creating personalised models for these high-value clients, and predicting prices tailored to their needs, could facilitate exclusive special offers and potentially optimise overall company revenue more efficiently.

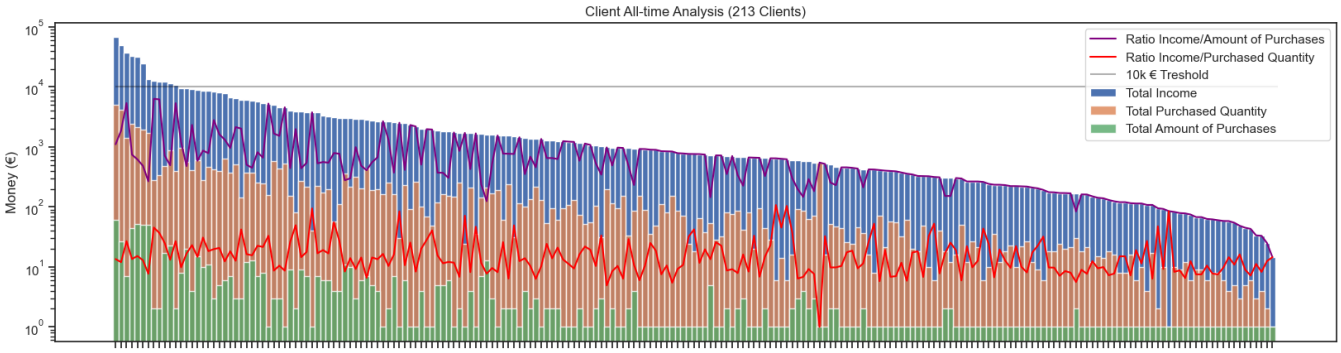


Figure 3: Bar chart representing the total income, the total purchase amount, the total quantity of purchased filters, and the ratios concerning these 3 values for each client. The dependent axis is in a logarithmic scale for better comparison of data. The client's name is anonymized due to privacy issues.

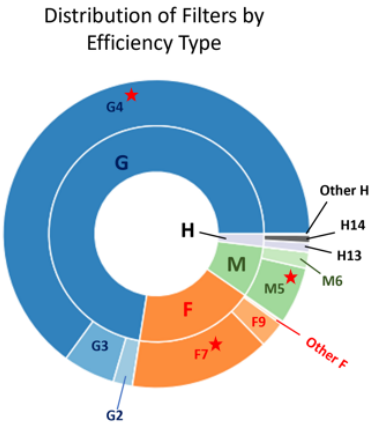
2. **Examining the Income/Amount of Purchases Ratio** reveals clients who allocate a significant portion of their income to just a few purchases. While this ratio is useful for identifying single-purchase clients (when it equals total income), for high-value clients, it indicates a higher probability of spending larger amounts with each purchase. Grouping and modelling algorithms around these clients may enhance generated income in a limited number of purchases.

3. Examining the **Income/Purchased Quantity of Filters Ratio** reveals the clients that tolerate higher unit prices. There are very few clients that present a high value for this ratio, by investing more in these clients, the company may achieve a sustained growth of prices over time, which means sustained higher incomes.

4. Given the widely scattered customer profile, **clustering clients** could emerge as a potent strategy to enhance our model's predictive capabilities when proposing prices. This involves leveraging insights from companies that have demonstrated a propensity to accept higher-priced filter proposals.

Filter Efficiency Distribution

Despite the provided data including a wide variety of filter types and efficiencies, they are not equally represented in the dataset. This means that there might not be enough data to teach predictive models with reasonable quality. To know which filters have more data input, a circular graph along with a table are presented in figure 4 with the relative frequency of each filter type and efficiencies in the dataset.



Filter Groups	Filter Efficiency	Absolute Frequency	Reative Frequency (%)	Group Frequency (%)
G	★ G4	2456	64.97	72.49
	G3	207	5.48	
	G2	77	2.04	
M	★ M5	227	6.01	7.7
	M6	64	1.69	
F	★ F7	560	14.81	17.9
	F9	104	2.75	
	F8	10	0.26	
	F5	2	0.05	
	F6	1	0.03	
H	H13	37	0.98	1.88
	H14	30	0.79	
	H11	4	0.11	

Figure 4: On the left, circular graph with a qualitative representation of the filter types and efficiencies that make the dataset. On the right, is a table expressing the information in the circular graph but with quantitative values as a percentage of the total. A red star marks the most represented filter types, upon which will be created a predictive model.

Figure 4 shows a clear representation of which filters are in conditions to be used in a predictive model, given their high amount considering the total amount of proposals. We then concluded that using filters G4, M5 and F7 will be beneficial for future analysis and the creation of predictive models for each filter is justified.

Dynamic Plot - Unit Price VS Volume

As mentioned earlier, the creation of a new feature merging the dimensional features, the "Volume", would mean an interesting tool to visualize how "Unit Price" would vary with size as a whole. With this idea in mind, a dynamic scatter plot was built, plotting the "Unit Price" as a function of the "Volume" for each filter type as efficiency. This way a clearest analysis can be made to search for possible trends in data that can be taught to a predictive model.

Last analysis indicated that filter of the type G4, F7 and M5 may be ideal for a case study. Using the dynamic plot and choosing to plot only these 3 categories, patterns are emerging as shown in figure **??**.

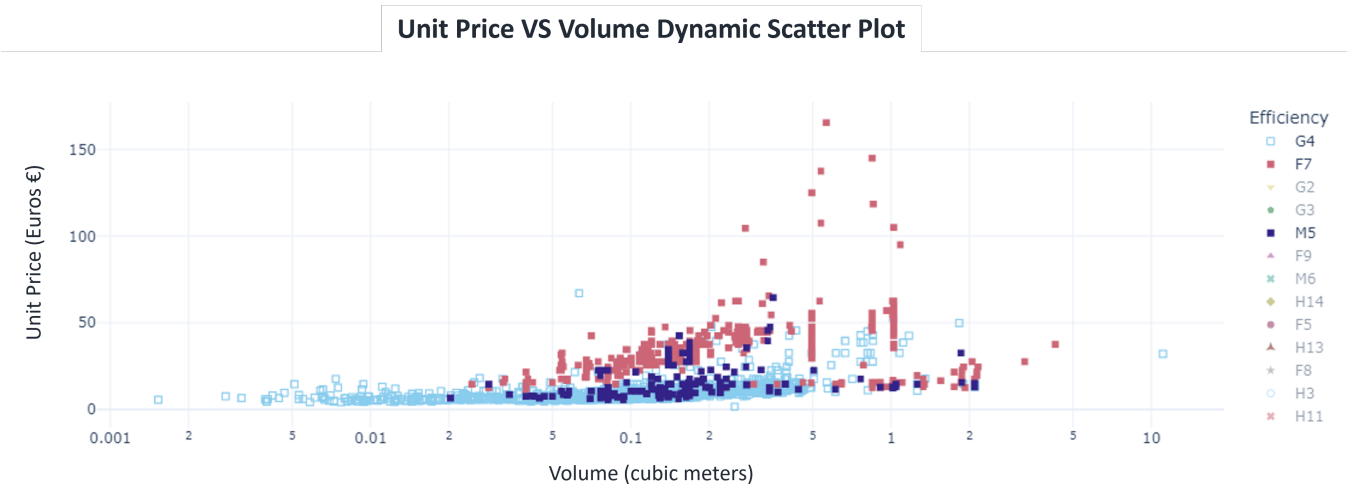


Figure 5: Scatter plot of the "Unit Price" as a function of the "Volume" toy feature for G4, F7 and M5 filters. "Volume" is expressed in a logarithmic scale for better visualisation. The dynamic functions are available on the EDA notebook.

Figure 5 ensures full confidence for us to proceed with the creation of predictive models concerning, at least, these three filter groups. A trend is shown by the data in figure 5, this means that there is a hidden rule or thought that can be taught to a model in order to make unit price prediction.

5 Methodology

Since a there wasn't a "rule" to determine the final price of each product, we took finding a logic behind the reasoning of the attributed price into our own hands. In order to do so, we trained several regression models and used various metrics to evaluate the results.

5.1 Data

In our pursuit of identifying the optimal model for predicting filter prices, the dataset, as detailed in Section 2.3, encompasses various features. These include the company name, quote ID, description, quantity, unit price, filter efficiency, dimensions, item type, length, height, gutter, depth, and pockets. A representative entry from the dataset illustrates the available information:

```
company_name, quote_id, description, qty, unit_price, filter_efficiency,
dimensions, item_type, Length, Height, Gutter, Depth, Pockets
Company&name, 17171.0, Filtros MV/G4.625.500.50, 36.0, 7.5, G4,
625.500.50, Filtro, 625.0, 500.0, 50.0, ,
```

Upon examination, it became evident that the dataset exhibited redundancy. Specifically, information designating an item as a filter was redundantly present in both the description and the item type. Moreover, details regarding dimensions were redundantly conveyed through the description, the 'dimensions' field, and the individual variables such as Length, Height, Gutter, Depth, and Pockets.

Recognizing the need for data refinement before model training, we undertook a two-step preprocessing approach:

1. **Consolidating Dimensional Information:** We collected and organized information related to filter dimensions, consolidating it into appropriate columns. Simultaneously, we pruned obsolete columns, including the 'description' and 'dimensions' columns.

2. **Handling Categorical Features:** To enhance 'filter efficiency,' we chose One Hot Encoding for better insights during model training. However, when dealing with 'company name' and 'quote id,' we decided to exclude them entirely from training. This was to avoid potential issues when predicting values for new companies. We considered one-hot encoding for 'company name,' but dropped the idea due to the risk of a significant increase in columns and longer model training times. Our goal was to strike a balance between comprehensive encoding and practical computational efficiency.

This process resulted in the following refined dataframe structure:

```
qty,unit_price,Length,Height,
Gutter,_F7,_F8,_F9,_G2,_G3,_G4,_H13,_H14,_M5,_M6
36.0,7.5,625.0,500.0,50.0,0,0,0,0,0,0,1,0,0,0,0
```

5.2 Models

From the dataframe structure referred before, we opted to aproach two diferrent ways: including company and quote id, and 'dropping' those features from the cleaned dataset. Furthermore, considering the weight of the 3 most bought filter efficiency type filters in the global dataset, we decided to find the best model for the global one, as well as for the respective datasets of G4, M5 and F7. For predicting the prices of filters, we opted for a selection of six models: Linear Regression, Epsilon-Support Vector Regression, Nearest-Neighbors based Regression, Decision Tree Regression, Random Forest Regression, and a regression model implemented using neural networks.

The choice of models for our project was largely influenced by the concepts introduced in our course. Given our familiarity and hands-on practice with these models, it seemed natural to incorporate them into our project. Notably, the Nearest Neighbors Regressor emerged from the observation that the data inherently forms clusters based on the filter type. Leveraging this characteristic, employing nearest neighbors for prediction appeared logical, as filters of the same type are expected to be proximate in the feature space. The inclusion of a Neural Network as the final model served the purpose of exploring and leveraging the full spectrum of tools and techniques imparted by this course

To train these models effectively, we employed the 'GridSearch' functionality from the 'sklearn' library. This allowed us to systematically explore various parameter combinations for each model, excluding the neural network, with the aim of identifying the optimal set of parameters and scoring metrics that yielded the best predictive performance.

5.3 Metrics

After training all the models with various hyperparameters, a crucial step involved comparing them. To facilitate this comparison, we employed several metrics, namely: Mean Squared Error and Relative Error, as well as Relative Error's Mean, Standard Deviation and Median.

The Mean Squared Error (MSE) comprises the objective of gauging the disparities between predicted and actual values, which is quantified by squaring this difference. Regarding the Mean Relative Error, it demonstrates the ratio of the absolute error to the real value, helping estimate the relative error size in comparison to the actual value. It's Standard Deviation demonstrates how dispersed the Relative Error is from the Mean. Finally, the Median represents the middle value of Relative Errors. These latter metrics related to Relative Error contribute to a more comprehensive understanding of the distribution of errors in our models' predictions. Combining these metrics allowed us to understand how close (or far) the models' predictions are from real value.

6 Model Development

The model development unfolded smoothly for the majority of models, barring Neural Network Regression, which presented unique challenges. We harnessed the capabilities of the scikit-learn (sklearn) library for other models, systematically fine-tuning hyperparameters. Each model was equipped with a dedicated function, overseeing parameter variation, preserving optimal configurations, and conducting evaluations of performance metrics.

Additionally, for every model, a five-fold cross-validation was implemented. In each invocation of the 'GridSearchCV' function from scikit-learn, we varied the scoring metric between MSE, Mean Relative Error and Relative Error's Standard Deviation and Median, ensuring a comprehensive assessment of model performance across different evaluation criteria.

6.1 Linear Regression

In implementing Linear Regression, we opted for the default parameters, as our objective was to commence with the most straightforward model and subsequently explore more sophisticated regression models.

6.2 Epsilon-Support Vector Regression

Subsequently, we proceeded with the implementation of Epsilon-Support Vector Regression, adjusting the regularization parameter (C). This parameter plays a crucial role in determining whether the model prioritizes minimizing insensitive loss. Furthermore, we explored the impact of selecting different kernels, specifically radial basis function (rbf) and polynomial (poly).

6.3 K Neighbors Regressor

For the K Neighbors Regressor, the following code snippet varies the number of neighbors, influencing how the weights of each point in the neighborhood are calculated. Additionally, it adjusts the parameter p, which dictates the method for calculating the distance between two points. When $p = 1$, the code employs the Manhattan distance ($L1$), and for $p = 2$, it utilizes the Euclidean distance ($L2$). For arbitrary p, the Minkowski distance is applied.

6.4 Decision Tree Regressor

In the Decision Tree Regressor, we explored various parameters, including the criterion, which defines the function to measure the quality of the split. The splitting strategy was also adjustable, allowing for the selection of either the best split or a random split. Additionally, the max features parameter determined the number of features considered when identifying the optimal split.

6.5 Random Forest Regressor

For the Random Forest Regressor, we conducted parameter variations encompassing the number of estimators (the quantity of trees in the forest), the criterion (the function measuring split quality), the minimum number of samples needed to split a node, and the number of features considered for identifying the optimal split.

6.6 Neural Network Regressor

As a last try to improve the models we quickly implemented a regression using neural networks. The architecture of the model is structured within the Sequential API, comprising distinct layers for comprehensive feature transformation.

The initial layer, serving as the input layer, consists of 128 neurons employing the rectified linear unit (ReLU) activation function. This facilitates the extraction of nonlinear patterns within the input data. Subsequently, three hidden layers, each comprising 256 neurons with ReLU activation, were integrated to enable the model to capture intricate relationships and intricate dependencies in the dataset.

The final layer, designed as the output layer, incorporates a single neuron with a linear activation function to produce continuous output values. The model optimization was realized through the mean absolute error (MAE) loss function and the Adam optimizer, aligning with the objective of minimizing the absolute differences between predicted and actual values. The incorporation of this neural network architecture and optimization strategy aims to yield a robust regression model for accurate predictions.

Filter Type	Model	MRE Val	MRE Test	Std Dev MRE Test	Median MRE Test	MSE Test
F7	LinearRegression	31,17%	33,69%	34,61%	24,63%	211,53
F7	NeuralNetwork	13,88%	13,84%	15,12%	8,57%	143,87
F7	RandomForestRegressor	18,09%	16,95%	21,45%	11,36%	106,09
F7	KNeighborsRegressor	20,95%	23,93%	37,93%	9,83%	179,64
F7	DecisionTreeRegressor	22,38%	23,02%	45,64%	14,13%	264,64
F7	SVR	31,64%	33,75%	45,70%	19,01%	210,91
G4	LinearRegression	18,84%	21,15%	22,30%	16,68%	19,55
G4	NeuralNetwork	13,54%	14,24%	21,15%	9,75%	22,84
G4	SVR	13,91%	14,31%	13,54%	10,93%	17,01
G4	RandomForestRegressor	14,74%	16,32%	16,54%	11,57%	8,69
G4	KNeighborsRegressor	16,08%	17,71%	33,24%	10,31%	14,26
G4	DecisionTreeRegressor	17,77%	14,43%	15,45%	10,53%	6,39
M5	LinearRegression	44,02%	47,14%	27,66%	47,65%	113,38
M5	NeuralNetwork	26,91%	28,40%	25,97%	19,29%	138,60
M5	SVR	31,16%	29,39%	22,11%	20,68%	127,07
M5	RandomForestRegressor	42,35%	37,94%	27,55%	32,75%	47,41
M5	KNeighborsRegressor	45,15%	47,33%	27,65%	49,48%	90,20
M5	DecisionTreeRegressor	47,97%	33,88%	47,94%	28,31%	71,72
general	LinearRegression	58,55%	57,35%	492,30%	24,83%	227,48
general	NeuralNetwork	26,40%	24,46%	41,01%	13,96%	450,45
general	SVR	40,22%	37,05%	114,96%	17,22%	580,77
general	RandomForestRegressor	46,53%	21,48%	79,21%	11,80%	152,24
general	DecisionTreeRegressor	51,39%	22,05%	123,36%	11,01%	234,43
general	KNeighborsRegressor	57,98%	43,60%	147,98%	23,94%	329,43

Figure 6: The selected best models for the top 3 filters and the general from the grid search based on MRE Val.

7 Results & Discussion

The table below presents details about the trained models, showcasing their best parameters obtained through the 'GridSearch'. The initial entry corresponds to our baseline model, Linear Regression, and the subsequent entries are arranged in descending order based on Mean Relative Error in the validation set.

The performance metrics considered are, as mentioned in Section 5.3, Mean Squared Error (MSE), Mean Relative Error (MRE), Standard Deviation of Relative Error (Std Dev MRE) and Median of Relative Error (Median MRE). The best models' results for each filter efficiency type is highlighted in the image above. Bear in mind that a validation set was used to compare the models before the final evaluation of each model, to avoid bias.

Upon closer look, it's evident that the model that got the best results was the Neural Network, even though grid search wasn't performed nor any other kind optimisation. This is explained by the superior complexity of the Neural Network model in relation to the other models. It is also noticeable the difference between the performances of the best model for each filter efficiency type and the respective baseline model.

In our strategic decision to tailor models for the most frequently sold filters, a natural expectation emerged that the performance of Filter G4, with a substantial dataset of 2456 points, would surpass that of Filters M5 and F7, which recorded 227 and 560 data points, respectively. This anticipation was grounded in the sheer volume of data available for G4, as illustrated in Fig.4, showcasing a significant data advantage over M5 and F7.

Upon examining the outcomes presented in Fig.6, a noteworthy revelation emerges: the optimal model for Filter F7 turned out to be a Neural Network with a 14.88 percent Mean Relative Error (MRE) on the validation set. Similarly, for Filter G4, the most effective model was also a Neural Network, registering a 13.54 percent MRE Val. This outcome starkly contrasts our initial expectations, introducing a surprising twist to our assumptions.

While delving into our data exploration, a noteworthy observation was the heightened variability in prices for the M5 filters, distinct from the others. This increased variability might contribute to the subpar performance of the models. Specifically, the M5 model yielded a 26.91 percent Mean Relative Error (MRE) validation, slightly worse than the general model's score of 26.40 percent. While this discrepancy hints at the potential impact of price variability on model performance, it's crucial to acknowledge that other factors might also be at play. Regrettably, due to time constraints, a more in-depth investigation to pinpoint the exact sources of this behavior was not feasible.

The overall model performance experienced a significant boost when trained with the Neural Network. It transitioned from a 40.22 percent Mean Relative Error (MRE) validation using Support Vector Regression (SVR) to an impressive 26.40 percent with the Neural Network. These models demonstrate exceptional proficiency in capturing intricate non-linear relationships within the data, offering a compelling explanation for the remarkable improvement observed. Despite anticipating enhanced results due to their capacity for handling complex patterns, we were pleasantly surprised by the extent of the performance enhancement.

The overarching objective of training the general model encompassed two key aspects. Firstly, we aimed to enable accurate price predictions for any filter within the dataset, including those with limited datapoints, ensuring reasonable outcomes. Secondly, our goal was to determine the optimal strategy: employing separate models for each filter type or adopting a unified model capable of predicting prices for any filter. Initial expectations leaned towards individual models, evident in the standout performance of models like F7 and G4. However, the general model defied expectations, achieving a remarkable 26.40 Mean Relative Error (MRE) validation. Notably, this result surpassed the performance of the M5 model, prompting speculation that the broader dataset facilitated a more comprehensive understanding and effective mapping of the intricate pricing dynamics across filters. Ideally, a direct comparison between the general model and individual filter models, using exclusive data from each filter, would have provided deeper insights. Unfortunately, time constraints prevented this detailed comparison.

8 Conclusion

In the culminating project of our AI course, we were privileged to dive into the dataset provided by Filterbond. Our exploration not only led to a comprehensive understanding of the data but also unveiled intriguing relationships between various features, shedding light on existing patterns. A focal point of our investigation was the development and training of diverse regression models tailored to predict filter prices for each efficiency type. This hands-on experience not only honed our analytical skills but also underscored the practical application of Machine Learning in unraveling pricing dynamics within the context of Filterbond's products.

This endeavor also marked a distinctive opportunity to engage with raw, real-world data rather than the conventional academic-curated datasets. This practical experience added a layer of complexity and authenticity to our project, emphasizing the significance of bridging theory with the challenges posed by genuine industrial datasets.

9 Future Work

In future opportunities, we believe that a point that could be implemented with a significant impact on the company's future would be the integration of a solution that allows for a workflow of Filterbond integrated with the machine learning methods we have presented. To this end, the creation of a pipeline that uniformly stores and introduces new data into a feedback loop feeding the already created models would be beneficial. This not only easily expands the database with standardized data but also facilitates the company's access to the same information.

Client segmentation has been studied but never materialized, although we believe that a pipeline capable of performing this clustering could provide relevant information about the profiles of customers purchasing Filterbond solutions. After a thorough market segmentation, training optimized models for each customer group could reduce errors in the models while generating more income for Filterbond.

Studying which customers maintain a consistent order frequency over time could also be important, as it indicates possible negative reasons that can be addressed in the proposal creation process.

Regarding the models, we understand that neural networks have more potential than what we have explored. In the future, a more complex model integrating all the variables mentioned above could result in a truly significant improvement over the ones presented.